2012

TR-2012004: Solving Linear Systems of Equations with Randomization, Augmentation and Aggregation

Victor Y. Pan

Guoliang Qian

Solving Linear Systems of Equations with Randomizaion, Augmentation and Aggregation *

Victor Y. Pan[1,2],[a] and Guoliang Qian[2],[b]

[1] Department of Mathematics and Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA
[2] Ph.D. Programs in Mathematics and Computer Science
The Graduate Center of the City University of New York
New York, NY 10036 USA
[a] victor.pan@lehman.cuny.edu
http://comet.lehman.cuny.edu/vpan/
[b] gqian@gc.cuny.edu

Abstract

Seeking a basis for the null space of a rectangular and possibly rank deficient and ill conditioned matrix we apply randomization, augmentation, and aggregation to reduce our task to computations with well conditioned matrices of full rank. Our algorithms avoid pivoting and orthogonalization, preserve matrix structure and sparseness, and in the case of an ill conditioned input perform only a small part of the computations with high accuracy. We extend the algorithms to the solution of nonhomogeneous nonsingular ill conditioned linear systems of equations whose matrices have small numerical nullities. Our estimates and experiments show dramatic progress versus the customary matrix algorithms where the input matrices are rank deficient or ill conditioned. Our study can be of independent technical interest: we extend the known results on conditioning of random matrices to randomized preconditioning, estimate the condition numbers of randomly augmented matrices, and link augmentation to aggregation as well as homogeneous to nonhomogeneous linear systems of equations.

AMS Classification: 15A06, 15A12, 15A52, 65F22, 65F05, 65F10

Key words: Linear systems of equations, Rank, Numerical rank, Randomization, Augmentation, Aggregation

1 Introduction

1.1 Background: computations of vectors and bases in the null space

Solution of a homogeneous linear system of equations $A\mathbf{y} = \mathbf{0}$ is a fundamental problem of matrix computations (see our Sections 9 and 13, [27, Sections 7.2 and 11.1], and [32] on its links to other

*Supported by NSF Grant CCF-1116736 and PSC CUNY Awards 62230–0040, 63153–0041 and 64512–0042. . Some results of this paper have been presented at the Fifth International Computer Science Symposium in Russia (CSR 2010) in Kazan' [28] and at the 16th Conference of the International Linear Algebra Society (ILAS) in Pisa, Italy, both in June 2010.

central subjects of that field). We call the solution vectors $\mathbf{y}$ the *null vectors* of the matrix A. They form the *null space* $\mathcal{N}(A) = \{\mathbf{y} : A\mathbf{y} = \mathbf{0}\}$; if its basis is given by the columns of a matrix B, we call B a *null matrix basis (nmb)* for a matrix A and write $B = \mathrm{nmb}(A)$.

The customary algorithms for computing null vectors and nmbs employ rank revealing LU or QR factorization, with pivoting (see [21] and the references therein) or SVD. The algorithms destroy matrix structure and sparseness and are quite costly even for general matrices. The SVD computation is most costly, but even "pivoting usually degrades the performance" [13, page 119].

1.2 Our contribution

Our present nmb algorithms avoid pivoting and orthogonalization by employing randomized matrix multiplication, augmentation and aggregation. As the result we accelerate the customary algorithms by order of magnitude for a large class of general and structured input matrices according to our estimates in Section 14 and numerical experiments in Section 15 (cf. Table 15.1).

We extend our algorithms to computing an *approximate nmb* or *anmb* of an ill conditioned matrix, that is a nmb of a nearby matrix, as well as to the solution of a nonsingular ill conditioned linear system of equations whose coefficient matrix is given with a small upper bound on its numerical nullity, that is on the number of its singular values that are dramatically smaller than its norm. In this case our *preconditioning* techniques reduce the computations to the case of well conditioned natrices of full rank. Our present techniques much better preserve sparseness and structure of the input matrices than our nmb algorithms of [27], based on randomized additive preprocessing.

Our study can be of independent technical interest, e.g., we estimate the impact of randomized augmentation on the condition number of a matrix, extend the known results on conditioning of random matrices in [4], [7], [9], [2], and [35] to preconditioning by means of randomized augmentation and aggregation, explore and exploit the links between the solution of nonhomogeneous and homogeneous linear systems of equations.

We refer the reader to the papers [24]–[32] on applications of randomized preprocessing to fundamental matrix and polynomial computations. In particular augmentation is closely linked to additive preprocessing of [24, Section 12] and [27, Section 4] but can a little better preserve matrix structure and sparseness (cf. Section 12).

1.3 Organization of the paper

In the next section we recall some definitions and basic facts, including the estimates for the ranks and condition numbers of random matrices and randomized matrix products. In Section 3 we compute a $\mathrm{nmb}(A)$ by combining randomization and nonorthogonal projection; this involves symmetrization and squares the condition number of A. In Section 4 we avoid such shortcoming by applying randomized pre- and post-multiplication provided that the input matrix has full row rank. In Sections 6–12 we present and analyze our alternative techniques of randomized augmentation, block modification and aggregation for computing nmbs and anmbs. These techniques better preserve matrix sparseness and structure than randomized multiplications and the nmb techniques of [27]. In Section 13 we extend our nmb and anmb algorithms to solving a nonhomogeneous nonsingular linear system of equations. In Section 14 we estimate the computational cost of our randomized algorithms. Section 15 covers the results of our numerical tests, which make up the contribution of the second author of this paper. Section 16 concludes it.

Acknowledgement. We thank Marc Van Barel for directing us to his Toeplitz solver in [36] and are grateful to a reviewer and Jesse Wolf for helpful comments.

2 Definitions and basic facts

Hereafter "expected" and "likely" mean "with probability 1 or close to 1".

$\mathbb{R}$ and $\mathbb{C}$ are the fields of real and complex numbers, respectively.

A flop is an arithmetic operation with such numbers.

The concepts "large", "small", "nearby", "approximate", "ill conditioned" and "well conditioned" as well as our notation $\approx$, $\ll$, and $\gg$ are quantified in the context of the computational task and computer environment. For two scalars a and b we write $a \ll b$ and $b \gg a$ if the ratio b/a is large; we write $a \approx b$ if $|a-b| \ll |a|+|b|$.

2.1 General matrices, nmbs and annihilators

We use and extend the customary definitions of matrix computations of [13] and [34].

$(B_1 \mid \ldots \mid B_k) = (B_j)_{j=1}^k$ is a $1 \times k$ block matrix with blocks $B_1, \ldots, B_k$.

$\mathrm{diag}(B_1, \ldots, B_k) = \mathrm{diag}(B_j)_{j=1}^k$ is a $k \times k$ block diagonal matrix with diagonal blocks $B_1, \ldots, B_k$.

I_n denotes the identity matrix $(\mathbf{e}_1 \mid \ldots \mid \mathbf{e}_n)$. $O_{k,l}$ denotes the $k \times l$ matrix filled with zeros. $\mathbf{0_k}$ denotes the vector $O_{k,1}$. We drop the subscripts and write I, O, and $\mathbf{0}$ where the size of a matrix or a vector is not important or is defined by context.

A^T and A^H denote the transpose and the Hermitian transpose of an $m \times n$ matrix A, respectively. $A^H = A^T$ for a real matrix A. A matrix A is Hermitian if $A = A^H$. A matrix $A = B^H B$ is Hermitian positive definite if B is a nonsingular matrix. A matrix U is called unitary, orthogonal and orthonormal if $U^H U = I$ or $UU^H = I$.

A matrix has full row (resp. column) rank if its rows (resp. columns) are linearly independent.

$\mathcal{R}(A)$ denotes the range of the matrix A, that is the linear space $\{\mathbf{y} : \mathbf{y} = A\mathbf{u}\}$ generated by its columns. $\mathcal{N}(A)$ denotes its null space $\{\mathbf{v} : A\mathbf{v} = \mathbf{0}\}$, $\mathrm{rank}\, A = \dim \mathcal{R}(A)$ its rank, and $\mathrm{nul}\, A = \dim \mathcal{N}(A) = n - \mathrm{rank}\, A$ its nullity. $\mathbf{v}$ is its null vector if $A\mathbf{v} = \mathbf{0}$. $\mathrm{nul}(A^T) = m - \mathrm{rank}\, A$ is the left nullity of A. It is equal to $\mathrm{nul}\, A$ if and only if $m = n$.

Fact 2.1. *The $m \times n$ matrices of a rank ρ form an algebraic variety $\mathbb{V}$ of dimension $(m+n-\rho)\rho$.*

Proof. Let M be an $m \times n$ matrix of a rank ρ with a nonsingular $\rho \times \rho$ leading block M_{00} and write $M = \begin{pmatrix} M_{00} & M_{01} \\ M_{10} & M_{11} \end{pmatrix}$. Then the $(m-\rho) \times (n-\rho)$ Schur complement $M_{11} - M_{10}M_{00}^{-1}M_{01}$ must vanish, which imposes $(m-\rho)(n-\rho)$ algebraic equations on the entries of M. Similar argument can be applied where any $\rho \times \rho$ submatrix of the matrix M (among all $\binom{m}{\rho}\binom{n}{\rho}$ such submatrices) is nonsingular. Therefore $\dim \mathbb{V} = mn - (m-\rho)(n-\rho) = (m+n-\rho)\rho$. □

A matrix H is a *complete annihilator* of a matrix A if $\mathcal{R}(H) = \mathcal{N}(A)$. It is a *null matrix basis* if it also has full column rank. We use the acronyms nmb, ca, $\mathrm{nmb}(A)$, and $\mathrm{ca}(A)$. Given a $\mathrm{ca}(A)$ we can compute a $\mathrm{nmb}(A)$ based on the following simple fact.

Fact 2.2. *[27]. Suppose H is a $\mathrm{ca}(A)$. Then*

(a) H is a $\mathrm{nmb}(A)$ if and only if $\mathrm{nul}\, H = 0$ and

(b) HY is a $\mathrm{nmb}(A)$ if X is a $\mathrm{ca}(H)$ and if $(X \mid Y)$ is a nonsingular matrix.

Remark 2.1. *In some algorithms for computing a nmb(A) or a ca(A) we assume that $A \in \mathbb{C}^{m \times n}$ for $m \le n$. This is not a serious restriction because we can change the matrix size by appending new rows or columns filled with zeros, then compute a nmb or a ca of the new matrix and immediately output a nmb or a ca of the original matrix. Alternatively we can handle the case of $m > n$ by applying the techniques of [27] or our Section 8, or we can reduce the task to the case $m \le n$ by employing the equations $\mathcal{N}(A) = \mathcal{N}(A^H A)$ or $\mathcal{N}(A) = \cap_{i=1}^h \mathcal{N}(B_i)$ where $A = \sum_{i=1}^h (O \mid B_i \mid O)^T$, $B_i = (O \mid I_{k_i} \mid O)^T A$ are $k_i \times n$ matrices for $i = 1, \ldots, h$, and $\sum_{i=1}^h k_i = m$. Given $\mathrm{nmb}(B_i) = \mathrm{nmb}((O \mid B_i \mid O)^T)$ for $i = 1, \ldots, h$, one can compute a $\mathrm{nmb}(A)$ based on [13, Theorem 12.4.1].*

2.2 SVD, inverses, norms, condition number, and numerical nullity

$A = S_A \Sigma_A T_A^H$ is *SVD* or *full SVD* of an $m \times n$ matrix A of a rank ρ if $S_A S_A^H = S_A^H S_A = I_m$, $T_A T_A^H = T_A^H T_A = I_n$, $\Sigma_A = \mathrm{diag}(\widehat{\Sigma}_A, O_{m-\rho,n-\rho})$, and $\widehat{\Sigma}_A = \mathrm{diag}(\sigma_j)_{j=1}^{\rho}$. Here $\sigma_j = \sigma_j(A) = \sigma_j(A^H) > 0$ is the jth largest singular value of a matrix A for $j = 1, \ldots, \rho$, and we write $\sigma_j = 0$ for $j > \rho$.

The Courant–Fischer minimax theorem [13, Theorem 8.1.2] implies that σ_j is the distance from the matrix A to a nearest matrix of rank $j-1$ for all j, which in turn implies the following fact.

Fact 2.3. *$\sigma_j(A) \ge \sigma_j(A_0)$ for all j if A_0 is a submatrix of a matrix A.*

$A^+ = T_A \operatorname{diag}((\widehat{\Sigma}_A)^{-1}, O_{n-\rho,m-\rho})^+ S_A^H$ is the Moore–Penrose pseudo-inverse of the matrix A.

A matrix $X = A^{(I)}$ is a left (resp. right) inverse of a matrix A if $XA = I$ (resp. $AX = I$). A^+ is an $A^{(I)}$ if and only if a matrix A has full rank. $A^{(I)}$ is unique and is given by A^{-1} if and only if A is a nonsingular matrix.

$\sigma_1(A) = ||A|| = ||A^H||$ is the 2-norm of a matrix $A = (a_{i,j})_{i,j=1}^{m,n}$.

$$\sigma_\rho(M) = 1/||M^+|| \text{ for a matrix } M \text{ of a rank } \rho. \tag{2.1}$$

$||A||_F = \sqrt{\sum_{i,j=1}^{m,n} |a_{i,j}|^2}$ is its Frobenius norm.

We have $||A||/\sqrt{mn} \le \max_{i,j=1}^{m,n} |a_{i,j}| \le ||A||$, $||A|| \le ||A||_F \le \sqrt{n}||A||$.

A is *normalized* if $||A|| = 1$. We write $A \approx B$ if $||A - B|| \ll ||A|| + ||B||$.

$\kappa(A) = \sigma_1(A)/\sigma_\rho(A) = ||A||\, ||A^+||$ is the condition number of a matrix A of a rank ρ. A matrix A is *ill conditioned* if $\sigma_1(A) \gg \sigma_\rho(A)$, otherwise *well conditioned*. See [3], [13, Sections 2.3.2, 2.3.3, 3.5.4, 12.5], [14, Chapter 15], and [34, Section 5.3] on estimating norms and condition numbers.

For a fixed $q < l = \min\{m, n\}$ we write $\bar{r} = m - q$, $r = n - q$, $S_{A,\bar{r}} = S_A(O_{q,\bar{r}} \mid I_{\bar{r}})^T$, $\mathbb{S}_{A,\bar{r}} = \mathcal{R}(S_{A,\bar{r}})$, $T_{A,r} = T_A(O_{q,r} \mid I_r)^T$, and $\mathbb{T}_{A,r} = \mathcal{R}(T_{A,r})$, that is $S_{A,\bar{r}}$ (resp. $T_{A,r}$) is the block formed by the $\bar{r}$ (resp. r) eastern, that is rightmost, columns of the matrix S_A (resp. T_A).

An $m \times n$ matrix $\tilde{A}$ has *numerical rank* q, *numerical nullity* $r = n - q$ and *left numerical nullity* $m - q$ if it has exactly q singular values that exceed $\epsilon||A||$ for a positive tolerance ϵ (cf. Remark 2.2). By setting to 0 all but the q largest singular values of such a matrix $\tilde{A}$ we obtain a well conditioned matrix A that lies nearby and has rank q; in this case $\mathbb{T}_{\tilde{A},r} \approx \mathcal{N}(A)$ and $\mathbb{S}_{\tilde{A},\bar{r}} \approx \mathcal{N}(A^T)$.

Conversely, suppose A is an $m \times n$ well conditioned matrix, $\operatorname{rank} A = q = l - r$, $0 < r < l = \min\{m, n\}$, E is a random matrix, and $||E|| \ll ||A||$. Then the matrix $\tilde{A} = A + E$ has numerical rank q and is likely to have full rank l (cf. Corollary 2.2).

Remark 2.2. *The choice of the tolerance ϵ above can be a challenge, e.g., for $\tilde{A} = \operatorname{diag}(0.9^j)_{j=0}^{2000}$.*

Remark 2.3. *By virtue of Fact 2.1 the $m \times n$ matrices having a numerical rank ρ lie near an algebraic variety of dimension $(m + n - \rho)\rho$, which is monotone increasing as ρ increases.*

Definition 2.1. *A matrix B is said to be an* approximate nmb *or an* approximate ca *of a matrix A if* $\dim \mathcal{N}(A) > 0$ *and if $B \approx \operatorname{nmb}(A)$ or $B \approx \operatorname{ca}(A)$, respectively, and we also call B an approximate ca or approximate nmb of any matrix $\tilde{A}$ that lie near such a matrix A even if* $\dim \mathcal{N}(\tilde{A}) = 0$. *Hereafter we use the acronyms* aca, anmb, $\operatorname{aca}(A)$ *and* $\operatorname{anmb}(A)$.

Given an $\operatorname{anmb}(A)$ we can readily approximate the matrix A by a matrix of a smaller rank.

Theorem 2.1. *(Cf. [27, Section 7.2].) Suppose $\tilde{A} \in \mathbb{C}^{n \times n}$, A has a numerical nullity $r > 0$, $B \in \mathbb{C}^{n \times r}$, the matrix $B^H B$ is nonsingular, and $||\tilde{A}B|| \ll ||\tilde{A}||\, ||B||$. Write $M = \tilde{A}(I - B(B^H B)^{-1} B^H)$, in particular $M = \tilde{A}(I - BB^H)$ if B is a unitary matrix. Then $M \approx A$ and* $\operatorname{rank} M = n - r$.

2.3 Structured matrices

$J = J_n = (\mathbf{e}_1 \mid \ldots \mid \mathbf{e}_n)$ is the $n \times n$ reflection matrix, $J(v_i)_{i=1}^n = (v_i)_{i=n}^1$; $J^2 = I$.

An $m \times n$ *Toeplitz* matrix $T = (t_{i-j})_{i=0,j=0}^{m-1,n-1}$ (resp. *Hankel* matrix $H = (h_{i+j})_{i=0,j=0}^{m-1,n-1}$) is defined by the $m + n - 1$ entries of its first row and first (resp. last) column. TJ and JT are Hankel matrices for a Toeplitz matrix T; HJ and JH are Toeplitz matrices for a Hankel matrix H.

$Z(\mathbf{v})$ is the lower triangular Toeplitz matrix defined by its first column vector $\mathbf{v} = Z(\mathbf{v})\mathbf{e}_1$. $Z^T(\mathbf{v}) = (Z(\mathbf{v}))^T$ is its transpose.

The following theorem is a less known variation of a similar result of [12].

Theorem 2.2. *[12]. Suppose $K = (t_{i,j})_{i,j=0}^n$ is a nonsingular $(n+1) \times (n+1)$ Toeplitz matrix, write $T = (t_{i,j})_{i,j=0}^{n-1}$, $K^{-1}\mathbf{e}_1 = (v_i)_{i=0}^n$, $\mathbf{v} = (v_i)_{i=0}^{n-1}$, $\mathbf{v}' = (v_i)_{i=1}^n$, $K^{-1}\mathbf{e}_{n+1} = (w_i)_{i=0}^n$, $\mathbf{w} = (w_i)_{i=0}^{n-1}$, and $\mathbf{w}' = (w_i)_{i=1}^n$, and assume that $v_0 \ne 0$. Then the matrix $T = (t_{i,j})_{i,j=0}^{n-1}$ is nonsingular and $v_0 T^{-1} = Z(\mathbf{v})Z^T(J\mathbf{w}') - Z(\mathbf{w})Z^T(J\mathbf{v}')$.*

Remark 2.4. *For any positive integer q we can embed a nonsingular $n \times n$ Toeplitz matrix T into an $(n+q) \times (n+q)$ Toeplitz matrix K with the $n \times n$ leading principal block T and then recursively apply Theorem 2.2 to express the inverse T^{-1} via the column vectors $K^{-1}\mathbf{e}_1$ and $K^{-1}\mathbf{e}_{n+q}$. For larger integers q the alternative expression of part (b) of Theorem 13.2 is superior.*

$n \times n$ structured matrices having a small *displacement rank* d extend the class of Toeplitz and Hankel matrices (for which $d \leq 2$) and can be represented by displacement generators of length d, defined by $2dn$ parameters each. Such matrices can be pairwise multiplied in $O(d^2 n \log n)$ flops and, if nonsingular, inverted in $O(d^2 n \log^2 n)$ flops each, where every output has displacement rank at most $2d$ and is represented with at most $4dn$ parameters [1], [11], [15], [16], [22].

2.4 Random sampling and random matrices

$|\Delta|$ is the cardinality of a set Δ.

Definition 2.2. Random sampling *of elements from a set Δ is their selection from this set at random, under the same probability distribution, and independently of each other. Random sampling is* Gaussian *or* uniform *if it is done under the Gaussian or uniform probability distribution on the set Δ, respectively. A matrix is* random *if its entries have been randomly sampled from a fixed set Δ. Such a matrix is* Gaussian or uniform random *over the set Δ if the random sampling is Gaussian or uniform, respectively.*

Definition 2.3. *A matrix or a vector is a* Gaussian random matrix or vector *with a mean μ and a positive variance σ^2 if it is filled with independent identically distributed Gaussian random variables, all having mean μ and variance σ^2. $\mathcal{G}_{\mu,\sigma}^{m \times n}$ is the set of $m \times n$ Gaussian random matrices. They are* standard *if $\mu = 0$ and $\sigma = 1$; they are N-*standard *for a positive parameter N if $\mu = 0$ and if the ratio σ/N is neither large nor small. $\mathcal{G}_{0,(N)}^{m \times n}$ denotes the set of such matrices of size $m \times n$.*

Definition 2.4. *$F_X(y) =$* Probability$\{X \leq y\}$ *for a real random variable X is the* cumulative distribution function (cdf) *of X evaluated at y. $F_A(y) = F_{\sigma_l(A)}(y)$ for an $m \times n$ matrix A and an integer $l = \min\{m, n\}$.*

2.5 Nonsingularity of random matrices and submatrices

Recall that the total degree of a multivariate monomial is the sum of its degrees in all its variables. The total degree of a polynomial is the maximal total degree of its monomials.

Lemma 2.1. *[5], [33], [40]. For a set Δ of cardinality $|\Delta|$ (in a fixed ring or field, e.g., in $\mathbb{C}$) let a polynomial in m variables have a total degree d, and let it not vanish identically on this set. Then the polynomial vanishes in at most $d|\Delta|^{m-1}$ points.*

Hereafter we always sample the values of Gaussian random variables from infinite sets $\Delta \subseteq \mathbb{R}$.

Corollary 2.1. *Under the assumptions of Lemma 2.1 let the values of the variables of the polynomial be randomly sampled under the Gaussian probability distribution. Then the polynomial vanishes with probability 0.*

Corollary 2.2. *Assume a Gaussian random $m \times n$ matrix A and any $m \times n$ matrix M with entries from $\mathbb{R}$. Then any nonempty square submatrix of the matrix $A + M$ is singular with probability 0.*

Proof. The determinant of a $k \times k$ matrix is a polynomial of total degree k in the entries and does not vanish for generic matrices M. It remains to apply Corollary 2.1. □

Under the uniform sampling from a finite set of large cardinality, the above results are readily extended; the probablility bounds become close to 0 rather than 0.

2.6 The extreme singular values of random matrices, randomized matrix products, and randomized reduction of the computation of nmbs to the case of matrices of full rank

A standard Gaussian random matrix M (cf. Definition 2.3) is well conditioned with a high probability [4], [7], [9], [2], and even adding such a matrix is likely to turn a normalized matrix into a well conditioned matrix [30]. We recall some basic results in this area; in particular we specify the respective estimates in Theorem 2.3, taken from [35] and applied in the proof of our Theorem 2.5, which is the basis of our condition estimates.

For an $m \times n$ matrix M of full rank $l = \min\{m, n\}$ we have $\sigma_l(M) = 1/||M^+||$ and consequently $F_{\sigma_l(M)}(y) = F_{1/||M^+||}(y)$; hereafter we write $F_{1/||M^+||}(y)$ more frequently than $F_{\sigma_l(M)}(y)$. Gaussian random matrices have full rank with probability 1 (see the previous subsection)

The following theorem provides an upper bound on the probability (the cdf) that the smallest singular value of Gaussian random matrix M is at most y (cf. (2.1) and Definition 2.4), whereas the argument y of the cdf is a probabilistic lower bound on the smallest singular value of the matrix M. The bounds can be strengthened by a factor $y^{|m-n|}$ [9], [2].

Theorem 2.3. *(See [35, Theorem 3.3].) Suppose $M \in \mathcal{G}_{\mu,\sigma}^{m\times n}$, $l = \min\{m, n\}$, and $y \geq 0$. Then the matrix M has full rank with probability 1 and $F_{1/||M^+||}(y) \leq 2.35\, y\sqrt{l}/\sigma$.*

The following theorem and corollary supply lower bounds on the probabilities that $||M|| \leq y$ and $\kappa(M) \leq y$ for a scalar y and a Gaussian random matrix M. The arguments y of the cdfs can also be viewed as probabilistic upper bounds on the norm $||M||$ and the condition number $\kappa(M)$, respectively. The corollary shows that the function $1 - F_{\kappa(M)}(y)$ is proportional to $\sqrt{\log y}/y$ as $y \to \infty$. Increasing the value σ increases the lower bound on the cdf of $\kappa(M)$, which yields probabilistic upper bound y on $\kappa(M)$. For small values σy and a fixed n the lower bound becomes negative, in which case the result becomes trivial.

Theorem 2.4. *(See [6, Theorem II.7].) Suppose $M \in \mathcal{G}_{0,\sigma}^{m\times n}$. Then $F_{||M||}(y) \geq 1 - \exp(-x^2/2)$ for $x = y/\sigma - 2\sqrt{n} \geq 0$.*

Corollary 2.3. *(See [35, Theorem 3.1].) Under the assumptions of Theorem 2.3, let $||M|| \leq \sqrt{l}$. Then $F_{\kappa(M)}(y) \geq 1 - (14.1 + 4.7\sqrt{(2\ln y)/n})n/(y\sigma)$ for all $y \geq 1$.*

The following theorem shows that $\sigma_{\operatorname{rank} W} \leq y$ with a probablilty of at most the order y for $W = GM$, $W = MH$, and Gaussian random matrices G and H. Therefore it is unlikely that multiplication by them can dramatically decrease the smallest positive singular value of a matrix, even though $UV = O$ for some pairs of rectangular unitary matrices U and V.

Theorem 2.5. *[29]. Suppose $G \in \mathcal{G}_{\mu,\sigma}^{r\times m}$, $H \in \mathcal{G}_{\mu,\sigma}^{n\times r}$, $M \in \mathbb{R}^{m\times n}$, and $y \geq 0$. Then*
$\max\{F_{1/||(GM)^+||}(y), F_{1/||(MH)^+||}(y)\} \leq 2.35y\sqrt{\widehat{r}}/(\sigma_{\operatorname{rank} M}(M)\sigma)$ *for $\widehat{r} = \min\{r, \operatorname{rank} M\}$.*

Remark 2.5. *Corollary 2.2 can be readily extended to all structured matrices of interest. On the extension of the results of this subsection to the case of matrices with complex entries and Toeplitz matrices see [4], [7], [9], [2] and [29, Sections 3.3 and 3.5]. In short, all natural extensions to complex matrices have been proved and strongly supported empirically; all such extensions to the Toeplitz case also have strong empirical support, but the respective formal results have been limited to the extension of Theorem 2.3 in [29, Section 3.3] so far.*

3 A nmb of a matrix via randomization and projection

Algorithm 3.1. A nmb via randomization and projection.

INPUT: *An $m \times n$ matrix A of full rank m, for $m \leq n$ (cf. Section 5).*

OUTPUT: *FAILURE with probability 0 or a* nmb(A).

COMPUTATIONS:

1. *Generate matrix* $G \in \mathcal{G}_{0,1}^{n\times(n-m)}$.
2. *Compute the matrix* $B = (I_n - A^H(AA^H)^{-1}A)G$. *Output FAILURE and stop if this matrix is rank deficient. Otherwise output it as a* $\mathrm{nmb}(A)$.

Correctness proof. Surely $A(I_n - A^H(AA^H)^{-1}A) = O_{m,n-m}$. So $\mathcal{R}(B) \subseteq \mathcal{N}(A)$, whereas $\dim(B) = n - m = \dim(\mathcal{N}(A))$ because an $n \times (n-m)$ matrix G has full rank with probability 1.

Remark 3.1. *If* $\operatorname{rank} A < m$, *then matrix* AA^T *is singular and Algorithm 3.1 fails.*

The map $A \to AA^H$ squares the condition number, $\kappa(AA^H) = (\kappa(A))^2$, thus complicating numerical inversion of the matrix AA^H. We will avoid such a shortcoming by working with matrices of a little larger size and still using no orthogonalization (see Theorem 4.2 and Remark 6.1).

4 Nmbs of a matrix via randomized post-multiplication

Clearly the null space of a matrix having full column rank consists of the vector $\mathbf{0}$. If $m < n$ and if $A = (A_w \mid A_e)$ is an $m \times n$ matrix with nonsingular $m \times m$ western block A_w, then we can compute a $\mathrm{nmb}(A) = \begin{pmatrix} -A_w^{-1}A_e \\ I_{n-m} \end{pmatrix}$. Next we extend this simple recipe to $m \times n$ matrices A of full rank m.

Algorithm 4.1. A nmb via post-multiplication.

INPUT: *An* $m \times n$ *matrix* A *of full rank* m, *for* $m \le n$ *(cf. Section 5).*

OUTPUT: *FAILURE with probability* 0 *or a* $\mathrm{nmb}(A)$.

COMPUTATIONS:

1. *Generate standard Gaussian random matrix* $(S \mid T) \in \mathcal{G}_{0,1}^{n\times n}$ *where* $S \in \mathcal{G}_{0,1}^{n\times m}$. *Output FAILURE if the matrix* $(S \mid T)$ *is singular (cf. Theorem 4.1).*
2. *Otherwise compute the matrix* AS. *Output FAILURE if it is singular (cf. Theorem 4.1).*
3. *Otherwise output the matrix* $(S \mid T)\begin{pmatrix} -(AS)^{-1}AT \\ I_{n-m} \end{pmatrix}$ *as a* $\mathrm{nmb}(A)$.

Correctness of the algorithm is verified by inspection.

Theorem 4.1. *The matrices* AS *and* $(S \mid T)$ *in Algorithm 4.1 are nonsingular with probability* 1.

Proof. $\det(AS)$ (resp. $\det(S \mid T)$) is a polynomial of a degree at most m (resp. n) in the entries of the matrix S (resp. $(S \mid T)$). The polynomial does not vanish identically in these entries (provided that $\operatorname{rank} A = m$). Now the theorem follows from Corollary 2.1. □

The theorem implies that Algorithm 4.1 is extremely unlikely to fail. In its numerical implementation we should also output FAILURE if the matrices $(S \mid T)$ or AS are ill conditioned. The matrix $(S \mid T)$ is expected to be well conditioned in virtue of Corollary 2.3. Next we probabilistically estimate the values $\sigma_m(AS)$ below and $\kappa(A)$ above.

Theorem 4.2. *For the matrices* A *and* S *of Algorithm 4.1 we have*
(a) $F_{1/||(AS)^{-1}||}(y) \le 2.35y\sqrt{m}/\sigma_m(A)$ *and*
(b) the condition number $\kappa(AS)$ *is expected to have at most the order* $\kappa(A)$.

Proof. Part (a) follows from Theorem 2.5 (for $M = A$, $H = S$, and $\sigma = 1$) and implies part (b) because $||AS|| \le ||A||\,||S||$ and by virtue of Theorem 2.4 applied for $M = S$. □

Remark 4.1. *For* $S = A^H$ *the matrix* AS *is nonsingular, but* $\kappa(AS) = (\kappa(A))^2$. *By virtue of Theorem 4.2 we do not expect to have such a problem where* S *is Gaussian random matrix.*

Remark 4.2. *Our estimates for ranks and condition numbers in this and the next sections can be readily extended from random input matrices to all their leading blocks (see Section 11.3). It follows that Gaussian elimination with no pivoting and block Gaussian elimination are likely to be numerically safe for computing the inverse of the above matrix AS as well as the left inverse of the matrix C in the next section.*

5 Extension of nbm algorithms to the case of rank deficient inputs

If $\operatorname{rank} A < m$ in Algorithms 3.1 or 4.1, then the matrices AA^H and AS are singular and the algorithms fail. If, however, we are given $\rho = \operatorname{rank} A$, then by replacing the input matrix A with $\widehat{A} = GA$ for $G \in \mathcal{G}_{0,1}^{\rho \times m}$ we can expect to fix the rank deficiency.

Indeed apply the techniques of Section 2.5 and deduce that with probability 1 we have $\operatorname{nmb}(A) = \operatorname{nmb}(GA)$ and $\operatorname{rank}(GA) = \rho$. Furthermore $||GA|| \le ||G||\,||A||$, and thus Theorem 2.4 implies that the norm $||GA||$ is expected to have order $||A||$. Moreover, in addition apply Theorem 2.5 for $M = A$ to bound the cdf $F_{1/||(GA)^+||}(y)$ and obtain that the map $A \Longrightarrow GA$ is not expected to blow up the condition number of A.

To sum up, we can expect that randomized premultiplication of A by $G \in \mathcal{G}_{0,1}^{\rho \times m}$ enables us to extend Algorithms 3.1 and 4.1 safely to a rank deficient matrix A of any size if we are given its rank $\rho = \operatorname{rank} A$.

If, however, the matrix A is sparse or structured, these advantages are partly lost in multiplication by random matrices. Our nbm algorithms in the next section better keep matrix sparseness and structure.

6 Randomized northern augmentation

Given an $m \times n$ matrix A and $\rho = \operatorname{rank} A$, we generate matrix $V \in \mathcal{G}_{0,(||A||)}^{r \times n}$ for $r = n - \operatorname{rank} A$. Then with probability 1 we have $\operatorname{rank} C = n$ for $C = \begin{pmatrix} V \\ A \end{pmatrix}$ (cf. Theorem 6.1), the first r columns of a left inverse $C^{(I)}$ form a $\operatorname{nmb}(A)$, and the condition number $\kappa(C)$ is expected to have the order of $\kappa(A)$. Consequently with our randomized augmentation we expect to have no numerical problems unless the condition number $\kappa(A)$ is large. Furthermore, by applying such a randomized northern augmentation to a nearby $m \times n$ ill conditioned matrix $\tilde{A}$ where $m \ge n$, we expect to decrease its condition number to the order $\kappa(A)$ (see Theorem 10.3 and Remark 10.1).

6.1 A randomized nmb algorithm based on northern augmentation

Algorithm 6.1. A nmb via randomized northern augmentation.

INPUT: *An $m \times n$ matrix A and its rank ρ, $0 < \rho < n$.*

OUTPUT: *FAILURE with probability 0 or a matrix $B = \operatorname{nmb}(A)$.*

COMPUTATIONS:

1. *Write $r = n - \rho$ and generate matrix $V \in \mathcal{G}_{0,(||A||)}^{r \times n}$. Output FAILURE if the matrix $C = \begin{pmatrix} V \\ A \end{pmatrix}$ is rank deficient. This occurs with probability 0 (see Theorem 6.1).*
2. *Otherwise compute and output the matrix $B = C^{(I)} \begin{pmatrix} I_r \\ O_{m,r} \end{pmatrix}$.*

Correctness proof. Let $Y = \mathrm{nmb}(A) \in \mathbb{C}^{n\times r}$. Then $CY = \begin{pmatrix} VY \\ O_{m,r} \end{pmatrix}$, $Y = C^{(I)} \begin{pmatrix} VY \\ O_{m,r} \end{pmatrix} = C^{(I)} \begin{pmatrix} I_r \\ O_{m,r} \end{pmatrix} VY$, and so

$$\mathcal{N}(A) = \mathcal{R}(Y) \subseteq \mathcal{R}(B) \text{ for } B = C^{(I)} \begin{pmatrix} I_r \\ O_{m,r} \end{pmatrix}. \tag{6.1}$$

It follows that $\mathcal{R}(B) = \mathcal{N}(A)$ because $\dim(\mathcal{R}(B)) = \operatorname{rank} B \le r = \dim(\mathcal{N}(A))$. Now correctness is implied by the following theorem.

Theorem 6.1. *The matrix C of Algorithm 6.1 has full column rank n with probability 1.*

Proof. Let a $\rho \times n$ submatrix $A_{\rho,n}$ of the matrix A have full rank ρ and write $C_{n,n} = \begin{pmatrix} V \\ A_{\rho,n} \end{pmatrix}$. Clearly, $\det C_{n,n}$ is a polynomial of a degree at most r in the entries of the matrix V and does not vanish identically in these entries because the matrix $A_{\rho,n}$ has full rank. By virtue of Corollary 2.1 $\det C_{n,n}$ vanishes with probability 0 in the case of Gaussian random matrix V. □

6.2 Probabilistic bounds on the condition number

In this subsection we prove that the condition number $\kappa(C)$ is expected to be of at most the order $\frac{\sigma_1(A)}{\sigma_\rho(A)} = \kappa(A)$ provided V is a $||A||$-standard Gaussian random matrix.

Theorem 6.2. *Suppose that $A \in \mathbb{C}^{m\times n}$, $V \in \mathbb{C}^{r\times n}$, $C = \begin{pmatrix} V \\ A \end{pmatrix}$, $\operatorname{rank} C = n$, $\rho = \operatorname{rank} A = n - r$, and $\operatorname{rank} V = r$. Let $A = S_A \Sigma_A T_A^H$ be full SVD of the matrix A, where $\Sigma_A = \operatorname{diag}(\widehat{\Sigma}_A, O_{m-\rho,r})$ and $\widehat{\Sigma}_A$ is a $\rho \times \rho$ diagonal matrix of the positive singular values of A. Write*

$$\operatorname{diag}(I_r, S_A^H) C T_A = \begin{pmatrix} M \\ O_{m-\rho,n} \end{pmatrix}, \ M = \begin{pmatrix} V_0 & V_1 \\ \widehat{\Sigma}_A & O_{\rho,r} \end{pmatrix}. \tag{6.2}$$

Then $\kappa(C) \le (\frac{1}{\sigma_\rho(A)} + \frac{1}{\sigma_r(V_1)} + \frac{||V_0||}{\sigma_\rho(A)\sigma_r(V_1)})||C||$.

Proof. We have $\operatorname{rank} M = \operatorname{rank} C = n$, and so the matrix M is nonsingular. Furthermore $||M^{-1}|| \le ||\widehat{\Sigma}_A^{-1}|| + ||V_1^{-1}|| + ||\widehat{\Sigma}_A^{-1}||\ ||V_1^{-1}||\ ||V_0||$ because $M^{-1} = \begin{pmatrix} O_{\rho,r} & \widehat{\Sigma}_A^{-1} \\ V_1^{-1} & -V_1^{-1} V_0 \widehat{\Sigma}_A^{-1} \end{pmatrix}$. Substitute $||\widehat{\Sigma}_A^{-1}|| = \frac{1}{\sigma_\rho(A)}$, $||V_1^{-1}|| = \frac{1}{\sigma_r(V_1)}$, $||M^{-1}|| = \frac{1}{\sigma_n(M)} = \frac{1}{\sigma_n(C)}$, and $\kappa(C) = \frac{||C||}{\sigma_n(C)}$ and obtain the theorem. □

Corollary 6.1. *Under the assumptions of Theorem 6.2 let $||V||/||A|| = t \ne 0$ and write $\kappa = \kappa(A)$ and $\kappa_1 = \frac{||V||}{\sigma_r(V_1)}$. Then $\kappa(C) \le \sqrt{1+t^2}(\kappa_1/t + \kappa + \kappa\kappa_1)$.*

Proof. We have $||C|| \le \sqrt{||A||^2 + ||V||^2} = \sqrt{1+t^2}||A||$ because $||V|| = t||A||$. Moreover $||V_0|| \le ||VT_A|| = ||V||$ for $(V_0 \mid V_1) = VT_A$. Substitute these bounds into Theorem 6.2. □

Next we estimate $\sigma_r(V_1)$ below provided V is Gaussian random matrix.

Theorem 6.3. *Under the assumptions of Theorem 6.2 suppose $V \in \mathcal{G}_{\mu,\sigma}^{r\times n}$. Then $F_{1/||V_1^{-1}||}(y) \le 2.35y\sqrt{r}/\sigma$.*

Proof. Apply Theorem 2.5 for $G = V$ and the unitary matrix $T_A \begin{pmatrix} O_{n-r,r} \\ I_r \end{pmatrix}$ replacing M. □

Remark 6.1. *Assume that A, V and C are the matrices of Algorithm 6.1 where $V \in \mathcal{G}_{0,(||A||)}^{r\times n}$. Then by virtue of Theorem 6.3 and Corollary 6.1, we can expect that the values $||V||$ and $\kappa(C)$ have the orders $||A||$ and at most $\kappa(A)$, respectively.*

Remark 6.2. *One can readily extend the upper bound on the condition number $\kappa(C)$ to the case where $||A|| \approx 1$, $V = (I_r \mid \bar{V})$, and $\bar{V} \in \mathcal{G}_{0,1}^{r\times(n-r)}$. We have similar extension for northwestern augmentation, linked to additive preprocessing of [27] (see the end of Section 11.2).*

7 The nmb computation where the rank is not known

Given a matrix $A \in \mathbb{R}^{m\times n}$ and a range $\rho_- \le \rho \le \rho_+$ for its unknown rank ρ, e.g., $\rho_- = 0$ and $\rho_+ = \min\{m, n\}$, we can search this range for the rank as follows. For a candidate integer i and $G \in \mathcal{G}_{0,1}^{i\times m}$, we can apply one of our nmb Algorithms 3.1 or 4.1 to the product GA. Both algorithms fail for $i > \rho$, but with probability 1 output a correct $\mathrm{nmb}(A) = \mathrm{nmb}(GA)$ for $i \le \rho$. We can apply these tests at first for $i = \rho_-$ and then recursively for $i = \rho_- + 2^h$, $h = 0, 1, \dots$. If the test fails for some positive h but succeeded for $h-1$, then with probability 1 we have $\rho_- + 2^{h-1} \le \rho < \rho_- + 2^h$ and can compute ρ in $h_+ = \lceil \log_2(\rho - \rho_-) \rceil + 1$ steps of binary search, thus performing at most $1 + 2h_+ \approx 2\log_2(\rho - \rho_-)$ tests overall.

Alternatively we can compute the rank by using about $2\log_2(\rho_+ - \rho)$ tests based on the northern augmentation of Algorithm 6.1. Fix a positive i, generate an $i\times n$ random matrix V, write $C = \begin{pmatrix} V \\ A \end{pmatrix}$, $B = C^{(I)} \begin{pmatrix} I_i \\ O_{m,i} \end{pmatrix}$, and $r = n - \rho$, and compute $\rho = \mathrm{rank}\, A$ based on the following properties.

(a) The matrix C is rank deficient and consequently has no left inverse if $i < r$.

(b) With probability 1 the matrix C has full rank and consequently has a left inverse if $i \ge r$.

(c) Suppose the matrix C has full rank. Then (6.1) implies that $i = r$ if and only if $AB = O_{m,i}$.

We can recursively test whether the matrix C has full rank for the integers $i = \rho_+, \rho_+ - 1, \rho_+ - 2, \rho_+ - 4, \dots$. If we detect rank deficiency for $i = \rho_+ - 2^h$ and some positive h but not for $i = \rho_+ - 2^{h-1}$, then we compute ρ by means of binary search in the range $\rho_+ - 2^h \le \rho \le \rho_+ - 2^{h-1}$. The search relies on properties (a) and (b), which imply correctness of the test with probaility 1. We can use property (c) to verify correctness of the output value.

Finally we can compute both $\mathrm{rank}\, M$ and a $\mathrm{nmb}(M)$ in just two steps by combining northern augmentation with aggregation provided the matrix $C = \begin{pmatrix} V \\ A \end{pmatrix}$ has full rank.

Algorithm 7.1. Randomized northern augmentation with aggregation for a nmb.

INPUT: *An $m\times n$ matrix A having an unknown rank ρ and a $k\times n$ matrix V such that $n - k \le \rho < n$ and the matrix $C = \begin{pmatrix} V \\ A \end{pmatrix}$ has full rank n (this holds with probability 1 if V is a random matrix).*

OUTPUT: *a nmb(A).*

COMPUTATIONS:

1. *Compute the matrix $B = C^{(I)} \begin{pmatrix} I_k \\ O_{m,k} \end{pmatrix}$. Output it as a* $\mathrm{nmb}(A)$ *if $AB = O_{m,k}$.*
2. *Otherwise successively compute the matrices $X = \mathrm{ca}(AB)$, $\mathrm{ca}(A) = BX \in \mathbb{C}_{k,h}$ for an integer $h \ge k - n + \rho$, and a* $\mathrm{nmb}(A)$ *based on Fact 2.2.*

Theorem 7.1. *(Cf. [13, Theorem 12.4.1].) The matrix BX computed in Algorithm 7.1 is a* $\mathrm{ca}(A)$.

Proof. Equation $X = \mathrm{ca}(AB)$ implies that $ABX = O_{m,h}$, that is, $\mathcal{N}(A) \supseteq \mathcal{R}(BX)$. Let us prove that $\mathcal{N}(A) \subseteq \mathcal{R}(BX)$, that is, $\mathbf{y} = BX\mathbf{w}$ for some vector $\mathbf{w}$ as soon as $A\mathbf{y} = \mathbf{0}$. Note that $\mathcal{N}(A) = \mathcal{R}(B)$ because $\mathrm{rank}\, C = n$ by assumption. This implies that $\mathbf{y} = B\mathbf{z}$ for some vector $\mathbf{z}$. Finally $\mathbf{z} = X\mathbf{w}$ for some vector $\mathbf{w}$ because $A\mathbf{y} = AB\mathbf{z} = \mathbf{0}$ and because $X = \mathrm{ca}(AB)$ by assumption. □

Algorithm 7.1 is an *aggregation/disaggregation process* (cf. [18] and our Section 10): we first aggregate an $m \times n$ input matrix A into the matrix AB of the smaller size $m \times k$, then, at Stage 2, compute a ca $X = \mathrm{ca}(AB)$ for such a matrix, and finally disaggregate this ca into BX, which is a ca for the matrix A.

8 Block row modification

Suppose we seek a nmb of a matrix $M = \begin{pmatrix} N \\ A \end{pmatrix}$ where $A \in \mathbb{R}^{m \times n}$, $\operatorname{rank} A = n - r$ (we can compute it by applying our techniques in Section 7), and $N \in \mathbb{R}^{r \times n}$. Assume that $\operatorname{rank} A = \operatorname{rank} M$; otherwise we can replace A with GA for an $m \times m$ random matrix G, and then we would have $\mathrm{nmb}(A) = \mathrm{nmb}(GA)$ and $\operatorname{rank} M = \operatorname{rank}(GA)$ with probability 1.

Now instead of northern augmentation we can apply northern block row modification, that is we can replace the northern block N with $||A||$-standard Gaussian random matrix. This technique is similar to northern augmentation, except that it is restricted to $(m+r) \times n$ matrices A for $m+r \geq n$ but has an advantage of keeping the matrix size intact. Let us extend Algorithm 7.1 respectively.

Algorithm 8.1. A nmb via randomized northern block row modification.

INPUT: *Two matrices $A \in \mathbb{R}^{m \times n}$ and $N \in \mathbb{R}^{r \times n}$ such that the ratio $||N||/||A||$ is neither large nor small and* $\operatorname{rank} A = \operatorname{rank} M = n - r$ *for* $M = \begin{pmatrix} N \\ A \end{pmatrix}$.

OUTPUT: *FAILURE with probability* 0 *or a* $\mathrm{nmb}(M)$.

COMPUTATIONS:

1. *Generate matrix $V \in \mathcal{G}^{r \times n}_{0,(||A||)}$. Output FAILURE if the matrix $C = \begin{pmatrix} V \\ A \end{pmatrix}$ is column rank deficient. (The results of Section 2.5 imply that this occurs with probability* 0.*)*
2. *Otherwise compute and output the matrices $B = C^{(I)} \begin{pmatrix} I_r \\ O_{m,r} \end{pmatrix}$ and MB. Output FAILURE if $MB \neq O_{m+r,r}$; otherwise output B.*

Clearly $\mathrm{nmb}(M) = \mathrm{nmb}(A)$ because $\operatorname{rank} A = \operatorname{rank} M$. Consequently the analysis of Algorithm 7.1, including its correctness proof, is extended to Algorithm 8.1.

9 Computing anmbs

We can readily extend all our algorithms for computing ranks and nmbs to computing numerical ranks and amnb. Here is an extension of Algorithm 6.1 to computing an anmb of a matrix $\tilde{A}$ that has numerical rank $\tilde{\rho}$.

Algorithm 9.1. An anmb via randomized northern augmentation.

INPUT: *an $m \times n$ matrix $\tilde{A}$, with $||A|| \approx 1$, and its numerical rank $\tilde{\rho}$, $0 < \tilde{\rho} < n$.*

OUTPUT: *FAILURE with probability* 0 *or an anmb* $(\tilde{A})$.

COMPUTATIONS:

1. *Write $\tilde{r} = n - \tilde{\rho}$ and generate a $||A||$-standard Gaussian random matrix V of size $\tilde{r} \times n$. Output FAILURE if $\tilde{C} = \begin{pmatrix} V \\ \tilde{A} \end{pmatrix}$ is a rank deficient matrix (this occurs with probability* 0*).*

2. *Otherwise compute and output the matrix* $\tilde{B} = \tilde{C}^+ \begin{pmatrix} I_{\tilde{r}} \\ O_{m,\tilde{r}} \end{pmatrix}$.

Suppose $A \approx \tilde{A}$ and $\operatorname{rank} A = \tilde{\rho}$, write $C = \begin{pmatrix} V \\ A \end{pmatrix} \approx \tilde{C} = \begin{pmatrix} V \\ \tilde{A} \end{pmatrix}$, and assume that the matrix C has full rank. Then $B = C^+ \begin{pmatrix} I_{\tilde{r}} \\ O_{m,\tilde{r}} \end{pmatrix} = \operatorname{nmb}(A)$, whereas $\tilde{B} - B = (\tilde{C}^+ - C^+) \begin{pmatrix} I_{\tilde{r}} \\ O_{m,\tilde{r}} \end{pmatrix}$, $||\tilde{B} - B|| \le ||\tilde{C}^+ - C^+|| \le 2||\tilde{C} - C||_F \max\{||C^+||^2, ||\tilde{C}^+||^2\}$ (see [13, Section 5.5.5]) and consequently $\kappa(C) \approx \kappa(\tilde{C})$. Furthermore these condition numbers are likely to have order $\frac{\sigma_1(\tilde{A})}{\sigma_{\tilde{\rho}}(\tilde{A})} \approx \kappa(A) = \frac{\sigma_1(A)}{\sigma_{\tilde{\rho}}(A)}$ provided V is a $||A||$-standard Gaussian random matrix (cf. Section 6.2).

We can search for the numerical rank of A by extending the recipes of the first part of Section 7 where we should test whether the matrix C is ill conditioned instead of testing whether it is rank deficient. Next we extend Algorithms 7.1 and 9.1 to the computation of an $\operatorname{anmb}(A)$ by using aggregation where the input matrix A has an unknown numerical rank. One can extend the proof of Theorem 7.1 to verify correctness of the resulting algorithm. In our augmentation in Section 13.3 we use upper bounds on numerical rank, but not its exact value.

Algorithm 9.2. Northern augmentation with aggregation for an anmb.

INPUT: *An* $m \times n$ *matrix* $\tilde{A}$ *having an unknown numerical rank* $\tilde{\rho}$ *and a* $k \times n$ *matrix* V *such that the matrix* $\tilde{C} = \tilde{C} = \begin{pmatrix} V \\ \tilde{A} \end{pmatrix}$ *has full rank* n *and* $n - k \le \tilde{\rho} < n$.

OUTPUT: *an anmb*(A).

COMPUTATIONS:

1. *Compute the matrix* $\tilde{B} = \tilde{C}^{(I)} \begin{pmatrix} I_k \\ O_{m,r} \end{pmatrix}$ *and the aggregated matrix* $\tilde{A}\tilde{B}$. *Output* $\tilde{B}$, *being an anmb of* $\tilde{A}$ *if* $||\tilde{A}\tilde{B}|| \ll ||\tilde{A}||\ ||\tilde{B}||$.
2. *Otherwise successively compute the matrices* $\tilde{X}$ *being an aca of* $\tilde{A}\tilde{B}$, $\tilde{B}\tilde{X}$ *being an aca of* $\tilde{A}$, *and an anmb of* $\tilde{A}$ *by extending Fact 2.2.*

Remark 9.1. *The computation or estimation of the numerical rank* $\bar{\rho}$ *of a matrix* A *is simpler where the ratio* $\frac{\sigma_{\bar{\rho}}(A)}{\sigma_{\bar{\rho}+1}(A)}$ *is larger. The power transforms* $A \Longrightarrow B = (AA^H)^i A$ *for* $i = 1, 2, \dots$ *increase every ratio* $\sigma_{j-1}(A)/\sigma_j(A)$ *exceeding 1 because* $\sigma_j(B) = (\sigma_j(A))^{2i+1}$ *for all* i *and* j.

10 Randomized western augmentation

Next we study randomized western augmentation for $m \times n$ matrices with $m \le n$. It relays the task of computing nmbs to other algorithms (such as Algorithm 3.1, 4.1, and 6.1) but prepares their application by controlling the rank and condition number of an input matrix. In particular our algorithms of the previous sections for a $\operatorname{nmb}(A)$ involve matrices having condition numbers of at most the order $\kappa(A)$, but randomized western augmentation is expected to reduce the nmb task to the case of a matrix of full rank that has condition number of a smaller order. Namely suppose $A \in \mathbb{R}^{m \times n}$, $U \in \mathcal{G}^{m \times q}_{0,(||A||)}$,

$$0 < s = m - \operatorname{rank} A \le q < m \le n. \tag{10.1}$$

Then by virtue of Theorem 10.3 in Section 10.2, the matrix $(U \mid A)$ is likely to have full rank and to have condition number $\kappa(U \mid A)$ of at most the order $\frac{\sigma_1(A)\sqrt{q}}{\sigma_{m-q}(A)}$. In the following algorithm we assume that an integer q satisfying (10.1) is available. In Remark 10.3 we discuss its computation.

10.1 A nmb via randomized western augmentation: an algorithm

Algorithm 10.1. A nmb via randomized western augmentation.

INPUT: *Three integers m, n, and q and a matrix $A \in \mathbb{C}^{m\times n}$ satisfying (10.1); a randomized* Subroutine NMB *(e.g., one of Algorithm 3.1, 4.1, and 6.1) that either computes a nmb of its $k\times l$ input matrix for $k \le l$ or outputs FAILURE, definitely so if its input matrix is rank deficient, but only with a low probability otherwise.*

OUTPUT: *FAILURE with probability* 0 *or the nullity $r = \operatorname{nul} A = n - \operatorname{rank} A$ and a* $\operatorname{nmb}(A)$.

COMPUTATIONS:

1. Western augmentation: *Generate matrix $U \in \mathcal{G}^{m\times q}_{0,(||A||)}$ and apply the Subroutine NMB to the matrix $(U \mid A)$. If the subroutine fails, the matrix $(U \mid A)$ is likely to be rank deficient; then output FAILURE. This occurs with probability* 0 *(see Theorem 10.1).*
2. Aggregation*: Otherwise the subroutine computes a matrix $Z = \begin{pmatrix} Z_0 \\ Z_1 \end{pmatrix} = \operatorname{nmb}\ (U \mid A)$ where $Z_0 \in \mathbb{C}^{q\times p}$, $Z_1 \in \mathbb{C}^{n\times p}$, and $q \le p \le q + r$.*
3. *Apply the Subroutine NMB to compute a $p\times r$ matrix $X = \operatorname{nmb}(Z_0)$. Output $r = \operatorname{nul} Z_0$.*
4. Disaggregation: *Compute and output the $n\times r$ matrix $Y = Z_1X = \operatorname{nmb}(A)$.*

Correctness proof. By the definition of the matrices Z and X, we have $UZ_0 + AZ_1 = O_{m,p}$ and $Z_0X = O_{q,r}$. Therefore $AY = AZ_1X = O_{m,r}$, that is $\mathcal{N}(A) \supseteq \mathcal{R}(Y)$. Conversely, if $A\mathbf{y} = \mathbf{0}$, then $(U \mid A)\begin{pmatrix} \mathbf{0} \\ \mathbf{y} \end{pmatrix} = \mathbf{0}$. It follows that $Z\mathbf{x} = \begin{pmatrix} \mathbf{0} \\ \mathbf{y} \end{pmatrix}$ for some vector $\mathbf{x}$ because Z is a nmb $(U \mid A)$. Consequently $Z_0\mathbf{x} = \mathbf{0}$, $Z_1\mathbf{x} = \mathbf{y}$, and so $\mathbf{x} = X\mathbf{v}$ for some vector $\mathbf{v}$ because $X = \operatorname{nmb}(Z_0)$. Consequently $\mathbf{y} = Z_1X\mathbf{v}$. Therefore $\mathcal{N}(A) \subseteq \mathcal{R}(N)$, and so Y is a $\operatorname{ca}(A)$.

It remains to prove that the matrix Y has full rank. Assume the opposit, and then let $Y\mathbf{u} = Z_1X\mathbf{u} = \mathbf{0}$ for a nonzero vector $\mathbf{u}$. In this case $X\mathbf{u} \ne \mathbf{0}$ because the matrix X has full column rank, being a $\operatorname{nmb}(Z_0)$. Furthermore $Z_0X\mathbf{u} = \mathbf{0}$ because $Z_0X = O_{q,r}$. Consequently $ZX\mathbf{u} = \mathbf{0}$, but this is impossible because the matrix Z is a nmb $(U \mid A)$ and thus has full column rank.

Algorithm 10.1 is yet another aggregation process. It first aggregates an input matrix A into the matrix Z_0 of a smaller size, then computes the matrix $X = \operatorname{nmb}(Z_0)$, and finally disaggregates this output to produce the matrix $Y = Z_1X = \operatorname{nmb}(A)$.

10.2 Regularization and preconditioning properties of randomized western augmentation

Theorem 10.1. *Assume that $A \in \mathbb{C}^{m\times n}$, $m \le n$, $s = m - \operatorname{rank} A$, and $U \in \mathcal{G}^{m\times q}_{\mu,\sigma}$. Then (a) the matrix $C = (U \mid A)$ is rank deficient for $q < s$, whereas (b) for $q \ge s$ the matrix $(U \mid A)$ is rank deficient with probability* 0.

Proof. We have $\operatorname{rank}\ (U \mid A) \le \operatorname{rank} U + \operatorname{rank} A \le q + \operatorname{rank} A$. This implies part (a) of the theorem. If $q \ge m - \operatorname{rank} A$ and the entries of the matrix U are indeterminates, then clearly the matrix $(U \mid A)$ has full rank and thus has a nonsingular $m\times m$ block. Now part (b) of the theorem follows from Corollary 2.2. □

Theorem 10.2. *Suppose A, U, and $C = (U \mid A)$ are the matrices of Algorithm 10.1, $||C|| \le 1$, $\operatorname{rank} C = m$, $\operatorname{rank} U = q$, $\operatorname{rank} A \ge m - q > 0$ (cf. (10.1)), and $A = S_A\Sigma_A T_A^H$ is a full SVD of the matrix A. Write $\bar{U} = S_A^H U$ and*

$$S_A^H C \operatorname{diag}(I_q, T_A) = (\bar{U} \mid \Sigma_A). \tag{10.2}$$

Delete the last $n - m + q$ columns of the latter matrix $(\bar{U} \mid \Sigma_A)$ and denote by

$$M = \begin{pmatrix} \bar{U}_0 & \widehat{\Sigma}_A \\ \bar{U}_1 & O_{q,m-q} \end{pmatrix} \tag{10.3}$$

the resulting $m \times m$ matrix where $\widehat{\Sigma}_A$ is the $(m-q) \times (m-q)$ leading principlal (northwestern) submatrix of the matrix Σ_A. Then $\kappa(C) \le (\frac{1}{\sigma_{m-q}(A)} + \frac{1}{\sigma_q(\bar{U}_1)} + \frac{||\bar{U}_0||}{\sigma_{m-q}(A)\sigma_q(\bar{U}_1)})||C||$.

Proof. We have $\operatorname{rank}\Sigma_A = \operatorname{rank} A \ge m-q > 0$, and so $\operatorname{rank}\widehat{\Sigma}_A = m-q$. Consequently the matrix M is nonsingular because $\operatorname{rank} C = \operatorname{rank}(U \mid \Sigma_A) = m$. Now invert equation (10.3) to obtain $M^{-1} = \begin{pmatrix} O_{q,m-q} & \bar{U}_1^{-1} \\ \widehat{\Sigma}_A^{-1} & -\widehat{\Sigma}_A^{-1}\bar{U}_0\bar{U}_1^{-1} \end{pmatrix}$. Deduce that $||M^{-1}|| \le ||\widehat{\Sigma}_A^{-1}|| + ||\bar{U}_1^{-1}|| + ||\widehat{\Sigma}_A^{-1}||\,||\bar{U}_1^{-1}||\,||\bar{U}_0||$. Substitute $||\widehat{\Sigma}_A^{-1}|| = \frac{1}{\sigma_{m-q}(\widehat{\Sigma}_A)} = \frac{1}{\sigma_{m-q}(A)}$, $||\bar{U}_1^{-1}|| = \frac{1}{\sigma_q(\bar{U}_1)}$, and $||M^{-1}|| = \frac{1}{\sigma_m(M)}$, which is not less than $\frac{1}{\sigma_m(C)}$ by virtue of Fact 2.3. Obtain that $\frac{1}{\sigma_m(C)} \le \frac{1}{\sigma_{m-q}(A)} + \frac{1}{\sigma_q(\bar{U}_1)} + \frac{||\bar{U}_0||}{\sigma_{m-q}(A)\sigma_q(\bar{U}_1)}$ and multiply both sides by $||C||$. □

Corollary 10.1. *Under the assumptions of Theorem 10.2 let $t = ||U||/||A|| \ne 0$ and write $\kappa_q(A) = \frac{\sigma_1(A)}{\sigma_{m-q}(A)}$ and $\kappa_q(U) = \frac{||\bar{U}_0||}{\sigma_q(\bar{U}_1)}$. Then $\kappa(C) \le \sqrt{1+t^2}(\kappa_q(U)/t + \kappa_q(A) + \kappa_q(A)\kappa_q(U))$.*

Proof. Note that $||C|| \le \sqrt{||A||^2 + ||U||^2} = \sqrt{1+t^2}||A||$ for $||U|| = t||A||$. Moreover $S_A^H U = \bar{U} = \begin{pmatrix} \bar{U}_0 \\ \bar{U}_1 \end{pmatrix}$, and so $||U|| = ||\bar{U}|| \ge ||\bar{U}_0||$. Substitute these relationships into Theorem 10.2. □

Theorem 10.3. *Under the assumptions of Theorem 10.2 let $U \in \mathcal{G}_{\mu,\sigma}^{m\times q}$. Then*

(a) $F_{1/||\bar{U}_1^+||}(y) \le 2.35y\sqrt{q}/\sigma$,

(b) the matrix $C = (U \mid A)$ is rank deficient with probability 0, and

(c) the condition number $\kappa(C)$ is expected to be of at most the order $||A||/\sigma_{m-q}(A)$ provided $U \in \mathcal{G}_{0,(||A||)}^{m\times q}$.

Proof. To prove part (a) apply Theorem 2.5 for the unitary matrix $M = (O \mid I_q)S_A^H$ and the Gaussian random matrix $H = U$, such that $MH = \bar{U}_1$. Part (a) implies that the matrix $\bar{U}_1$ is singular with probability 0. Now equations (10.2) and (10.3) combined imply part (b), whereas part (a) and Corollaries 2.3 and 10.1 together imply part (c). □

10.3 Remarks

Remark 10.1. *Northern augmentation of a matrix A is equivalent to western augmentation of the transpose A^T, and we can readily extend the results of this section respectively. Likewise block row modification of a matrix M of Section 8 is equivalent to block column modification of the transpose M^T and is similar to western augmentation.*

Remark 10.2. *If the matrix A is ill conditioned, then in numerical implementation of Algorithm 10.1 we must compute a* nmb $(U \mid A)$ *with high accuracy [24, Section 7]; we can, however, apply iterative refinement provided the matrix $(U \mid A)$ is well conditioned. In this way we dramatically decrease the overall cost of computing a nmb(A) versus the customary algorithms wherever $q \ll \min\{m,n\}$ (see our Section 14, [24, Sections 8 and 9] and [30]).*

Remark 10.3. *Our correctness proof for Algorithm 10.1 applies to any integer $q \ge s = m - \operatorname{rank} A$ (cf. (10.1)). The observations below can guide us in choosing the integer parameter q and computing the rank and numerical rank of A (see Remark 2.1 on relaxing the restriction that $m \le n$).*

1. *By virtue of Theorem 10.1,* rank $(U \mid A) < m$ *if $q < s$, but we expect that* rank $(U \mid A) = m$ *if U is a random $m \times q$ matrix and if $q \ge s$. The size $p \times n$ of the matrix Z and the amount of work at Stages 2–4 of Algorithm 10.1 decrease as q decreases toward s (cf. Section 7).*

2. *Suppose $m \le n$ and an $m \times n$ matrix $\tilde{A}$ has numerical rank $\tilde{\rho}$ exceeded by m. Furthermore assume that instead of a randomized Subroutine NMB we are given a randomized Subroutine ANMB (e.g., numerical version of Algorithm 4.1) that either computes an anmb of an input matrix or outputs* FAILURE, *definitely so if the matrix is rank deficient or ill conditioned, but only with a low probability otherwise. Then (cf. Section 9) we can apply Algorithm 10.1 to*

the matrix $\tilde{A}$ instead of A, choose q satisfying $m - \tilde{\rho} \le q < m$, and employ the Subroutine ANMB instead of the Subroutine NMB to compute anmbs instead of nmbs throughout. For $q < m - \tilde{\rho}$ the matrix $\tilde{C} = (U \mid \tilde{A})$ is rank deficient, ill conditioned or both, but by virtue of Theorem 10.3 it is expected to have full rank and to be well conditioned if $q \ge m - \tilde{\rho}$ and if U is a $||\tilde{A}||$-standard Gaussian random matrix. In this case we would expect that the algorithm computes numerical rank $\tilde{\rho}$ of the matrix $\tilde{A}$ and its anmb. The work at Stages 2–4 of this modification of Algorithm 10.1 would be minimized for $q = m - \tilde{\rho}$.

11 Randomized northwestern augmentation

Given a matrix $A \in \mathbb{C}^{m \times n}$, one can compute a $\mathrm{nmb}(A)$ or a $\mathrm{ca}(A)$ by applying Algorithm 10.1 where Algorithm 6.1 replaces the Subroutine NMB. Our techniques can be readily extended. In this section we specify the resulting northwestern augmentation $A \to K = \begin{pmatrix} W & V \\ U & A \end{pmatrix}$ and analyze it by extending our earlier analysis. More precisely we specify just the computation of a $\mathrm{ca}(A)$ and omit extensions to computing $\mathrm{nmb}(A)$, $\mathrm{aca}(A)$ and $\mathrm{anmb}(A)$. In Section 13.3 we apply northwestern augmentation to precondition a nonsingular nonhomogeneous linear system of equations.

11.1 Cas and nmbs via randomized northwestern augmentation: an algorithm

Algorithm 11.1. A ca via randomized northwestern augmentation.

INPUT: *A matrix $A \in \mathbb{C}^{m \times n}$ and its rank $\rho > 0$.*

OUTPUT: *FAILURE with probability 0 or a $\mathrm{ca}(A)$.*

INITIALIZATION: *Fix two nonnegative integers q and r such that*

$$n \ge q \ge n - \rho \text{ and } m + r \ge n + q. \tag{11.1}$$

COMPUTATIONS:

1. *(*Northwestern augmentation.*) Generate three random matrices V in $\mathbb{C}^{r \times n}$, U in $\mathbb{C}^{m \times q}$, and W in $\mathbb{C}^{r \times q}$. If the matrix $K = \begin{pmatrix} W & V \\ U & A \end{pmatrix} \in \mathbb{C}^{(m+r) \times (n+q)}$ is column rank deficient, output FAILURE. This occurs with probability 0 (see part (c) of Theorem 11.1).*
2. *(*Aggregation.*) Otherwise* $\mathrm{rank}\, K = n + q$. *Then compute the matrix*

$$Y = (O_{n,q} \mid I_n) K^{(I)} \begin{pmatrix} O_{r,q} \\ U \end{pmatrix}. \tag{11.2}$$

   Output $Y = \mathrm{ca}(A)$ if $AY = O_{m,q}$.
3. *Otherwise reapply Algorithm 11.1 to the matrix AY to compute a $q \times p$ matrix $Z = \mathrm{ca}(AY)$ for an integer $p \ge n - \rho$.*
4. *(*Disaggregation.*) Compute and output the $n \times p$ matrix $YZ = \mathrm{ca}(A)$.*

Our remarks on Algorithms 6.1 and 10.1 can be readily extended to Algorithm 11.1.

11.2 Analysis of randomized northwestern augmentation

Let us analyze Algorithm 11.1.

Theorem 11.1. *Assume that* $A \in \mathbb{C}^{m\times n}$, $U \in \mathbb{C}^{m\times q}$, $V \in \mathbb{C}^{r\times n}$, $W \in \mathbb{C}^{r\times q}$, *and* $K = \begin{pmatrix} W & V \\ U & A \end{pmatrix} \in \mathbb{C}^{(m+r)\times(n+q)}$ *and write* $\rho = \operatorname{rank} A$ *and* $s = \min\{m+r, n+q, \rho+q+r\}$.

(a) *Then we have* $\operatorname{rank} K \le s$.

(b) *In addition suppose that* U *is Gaussian random matrix and* $q \ge n-\rho$. *Then the matrix* $(U \mid A)$ *has rank* m *with probability* 1.

(c) *In addition suppose that* V *and* W *are Gaussian random matrices and* $r+m \ge n+q$; *so* $s = n+q$. *Then* $\operatorname{rank} K = s$ *with probability* 1.

Proof. Part (a) of the theorem can be immediately verified. Parts (b) and (c) are proved similarly to Theorems 6.1 and 10.1, based on Corollaries 2.1 and 2.2. □

Theorem 11.2. *Suppose that* $A \in \mathbb{C}^{m\times n}$, $U \in \mathbb{C}^{m\times q}$, $V \in \mathbb{C}^{r\times n}$, $W \in \mathbb{C}^{r\times q}$, $K = \begin{pmatrix} W & V \\ U & A \end{pmatrix} \in \mathbb{C}^{(m+r)\times(n+q)}$,

$$K^{(I)}K = I_{n+q} \text{ and } W^{(I)}W = I_q \text{ for some matrices } K^{(I)} \text{ and } W^{(I)}. \tag{11.3}$$

Define the matrix $Y = (O_{n,q} \mid I_n)K^{(I)} \begin{pmatrix} O_{r,q} \\ U \end{pmatrix}$ *of (11.2).*

(a) Then we have

$$\mathcal{N}(A) \subseteq \mathcal{R}(Y). \tag{11.4}$$

(b) Furthermore if $\operatorname{rank} U \le \operatorname{nul} A$, *then*

$$\mathcal{R}(Y) = \mathcal{N}(A). \tag{11.5}$$

Proof. Let $\mathbf{y} \in \mathcal{N}(A)$ and $\mathbf{x} \in \mathbb{C}^q$. Then $K \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} W\mathbf{x}+V\mathbf{y} \\ U\mathbf{x} \end{pmatrix}$. Substitute $\mathbf{x} = -W^{(I)}V\mathbf{y}$ and obtain that $K \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ U\mathbf{x} \end{pmatrix}$. Therefore $\mathbf{y} = (O_{n,q} \mid I_n)K^{(I)} \begin{pmatrix} \mathbf{0} \\ U\mathbf{x} \end{pmatrix}$. This proves claim (11.4), which implies claim (11.5) if $\operatorname{rank} U \le \operatorname{nul} A$ because $\operatorname{rank} Y \le \operatorname{rank} U$. □

Suppose relationships (11.1) hold and U, V and W are Gaussian random matrices. Then with probability 1 all of them as well as the matrix K have full rank (cf. Section 2.5 and part (c) of Theorem 11.1), equations (11.3) hold, and Theorem 11.2 implies correctness of Stage 2 of Algorithm 11.1. Correctness of its remaining stages is implied by part (a) of the following theorem. (We do not use its part (b) and only include it for completeness.)

Theorem 11.3. *Suppose* $A \in \mathbb{C}^{m\times n}$, $U \in \mathbb{C}^{m\times q}$, $V \in \mathbb{C}^{r\times n}$, $W \in \mathbb{C}^{r\times q}$, $K = \begin{pmatrix} W & V \\ U & A \end{pmatrix} \in \mathbb{C}^{(m+r)\times(n+q)}$, $\rho = \operatorname{rank} A$, $s = \min\{m+r, n+q, \rho+q+r\}$ *(cf. part (a) of Theorem 11.1), and relationships (11.3) hold. Let* $Y = (O_{n,q} \mid I_n)K^{(I)} \begin{pmatrix} O_{r,q} \\ U \end{pmatrix}$, *as in Theorem 11.2, and obtain that*
(a) YZ *is a* $\operatorname{ca}(A)$ *if a* $q \times p$ *matrix* Z *is a* $\operatorname{ca}(AY)$ *for some integer* p, *in particular if* $AY = O_{m,q}$, $p = q$, *and* $Z = I_q$, *and furthermore*
(b) Z *is a* $\operatorname{ca}(AY)$ *if* YZ *is a* $\operatorname{ca}(A)$ *and if the matrix* Y *has full column rank* q.

Proof. (a) Clearly $A(YZ) = (AY)Z = O_{m,r}$ if Z is a $\operatorname{ca}(AY)$. Conversely let $A\mathbf{u} = \mathbf{0}$. Then $\mathbf{u} = Y\mathbf{v}$ for some vector $\mathbf{v}$ in virtue of (11.4). Therefore $AY\mathbf{v} = \mathbf{0}$. It follows that $\mathbf{v} = Z\mathbf{z}$ for some vector $\mathbf{z}$ because Z is a $\operatorname{ca}(AY)$. Consequently $\mathbf{u} = Y\mathbf{v} = YZ\mathbf{z}$.

(b) Surely $(AY)Z = A(YZ) = O_{m,r}$ if YZ is a $\operatorname{ca}(A)$. Conversely let $AY\mathbf{u} = A(Y\mathbf{u}) = \mathbf{0}$. Then $Y\mathbf{u} = YZ\mathbf{v}$ for some vector $\mathbf{v}$ because YZ is a $\operatorname{ca}(A)$. Therefore $\mathbf{u} = Z\mathbf{v}$ since $\operatorname{rank} Y = q$. □

Corollary 11.1. *Suppose* $A \in \mathbb{C}^{m\times n}$, $U \in \mathcal{G}^{m\times q}_{0,(||A||)}$, $V \in \mathcal{G}^{r\times n}_{0,(||A||)}$, $W \in \mathcal{G}^{r\times q}_{0,(||A||)}$, $K = \begin{pmatrix} W & V \\ U & A \end{pmatrix} \in \mathbb{C}^{(m+r)\times(n+q)}$, $\rho = \operatorname{rank} A$, *and equations (11.1) hold (cf. part (a) of Theorem 11.1). Then the condition number* $\kappa(K)$ *is expected to have order at most* $\frac{||A||}{\sigma_h(A)}$ *for* $h = \min\{m-q, n-r\}$.

Proof. Combine Corollaries 6.1 and 10.1 and Theorems 6.3 and 10.3. □

One can extend our probabilistic upper bound on $\kappa(K)$ to the case where $q \geq n$, $\mu = 0$, $\sigma = ||A|| = 1$, and the matrix W is replaced by $(I_r \mid \widehat{W})$ for $\widehat{W} \in \mathcal{G}^{r,q-r}_{0,1}$ (cf. Remark 6.2). In this case northwestern augmentation is closely linked to additive preprocessing of [30]. The link implies extension of the preconditioning property of randomized northwestern augmentation to randomized additive preprocessing. The direct proof of this property in [30] is more involved.

11.3 Strong regularization and strong preconditioning

Our results on the regularization and preconditioning power of northern, western, and northwestern augmentation of the input matrix can be immediately extended to all its $i \times i$ leading principal (that is northwestern) submatrices for $i = 1, 2, \ldots$ In particular wherever we deduce that the output matrix is expected to have full rank or to be well conditioned, its leading principal submatrices have the same property. Indeed for every fixed i we can pre-multiply the matrix K of Algorithm 11.1 by $(I_{m+r-i} \mid O_{m+r-i,i})$, post-multiply the product by $(I_{n+q-i} \mid O_{i,n+q-i})^T$, and extend our study of this section to the resulting leading block of K. We can proceed similarly replacing K with the matrices C and $\tilde{C}$ of the previous sections. See [29] on proofs and algorithmic applications of strong regularization and strong preconditioning.

12 Randomized structured augmentation

We can restrict randomness of the matrices U, V and W in the previous sections to preserve any fixed patterns of sparseness and structure of an input matrix A. In the special case of northwestern augmentation of a Toeplitz matrix A we can produce a Toeplitz matrix K; then we can exploit its structure based on Theorem 13.2 in Section 13.3 or on Theorem 2.2 and Remark 2.4. Such randomized Toeplitz augmentation techniques still fix degeneracy with probability 1 by virtue of the results in Section 2.5. We cannot extend Theorem 2.5 to the case of Gaussian random Toeplitz matrices G and H, and thus cannot apply its corollaries, but our tests consistently support such an extension. Similar comments apply to randomized block row and block column modifications of the matrix A (cf. Section 8 and Remark 10.1).

Now assume that a matrix A is given with its displacement generator of a small length d and that U, V, and W are $||A||$-standard Gaussian random matrices of sizes $m \times q$, $n \times r$, and $r \times q$, respectively. Then (cf. [22]) we can represent the matrices C and K with displacement generators of length $d + O(r)$, $d + O(q)$, or $d + O(r + q)$, respectively. If the integers r and q are also small, then we obtain compressed representation of the structured matrices C and K and can dramatically accelerate computations with them (see the end of Section 2.3).

In the case of larger integers r or q we can still accelerate the computations based on endowing the matrices U, V and W with the structures consistent with the structure of the matrix A; then we can bound above the displacement ranks of the matrices C or K by $d+O(1)$ (cf. [20], [22]). Our comments in the previous paragraph on the formal and empirical support of the respective extension of our analysis still apply.

Remark 12.1. *Preconditioning power of randomized augmentation and additive preprocessing* $A \Longrightarrow A + UV^H$ *for random matrices* U *and* V *is similar but not identical. Suppose* $K = \begin{pmatrix} W & V^H \\ V & A \end{pmatrix}$ *is a Hermitian positive definite matrix. Then the Interlacing Property of eigenvalues [13, Theorem 8.1.7] implies that* $\kappa(K) \geq \kappa(A)$. *In contrast Gaussian randomized Hermitian additive preprocessing* $A \Longrightarrow A + VV^H$ *is expected to work as preconditioning for an ill conditioned matrix* A *having small numericall nullity provided the ratio* $||VV^H||/||A||$ *is neither large nor small [39].*

13 Solution of a nonhomogeneous linear system of equations

In the previous sections we reduced a homogeneous linear system $A\mathbf{y} = \mathbf{0}$ to nonhomogeneous ones, with matrices AS, C, and K. Conversely, we can reduce the solution of a nonsingular linear system $A\mathbf{y} = \mathbf{b}$ to computing a null vector of either the matrix $(I_n - \frac{\mathbf{b}\mathbf{b}^H}{\mathbf{b}^H\mathbf{b}})A$ or the matrix $(-\eta\mathbf{b} \mid A)$ for a nonzero scalar η. The latter approach seems to be superior; we analyze it in Section 13.1. In Sections 13.2 and 13.3 we extend this study to precondition a linear system $A\mathbf{y} = \mathbf{b}$ provided a matrix A has a positive numerical nullity. In this case our algorithms rely on computing an anmb(A) and randomized northwestern augmentation.

13.1 Solution with an auxiliary matrix defined by western augmentation

The null vector $\mathbf{z} = \begin{pmatrix} 1/\eta \\ \mathbf{y} \end{pmatrix}$ of the matrix $C = (-\eta\mathbf{b} \mid A)$ for $\eta \neq 0$ contains the vector $\mathbf{y} = A^{-1}\mathbf{b}$ as a subvector. To compute the null vector we can apply the algorithms of the previous sections. E.g., we can obtain a vector $\mathbf{z}$ from the linear system $\begin{pmatrix} \mathbf{v}^T \\ C \end{pmatrix}\mathbf{z} = \mathbf{e}_1$ where the vectors $\mathbf{e}_1 = (1, 0\dots, 0)^T$ and $\mathbf{v}^T$ have dimension $n+1$. For Gaussian random vector $\mathbf{v}$ such that $||\mathbf{v}|| \approx ||C||$ we expect that the ratio $\kappa(C)/\kappa(\begin{pmatrix} \mathbf{v}^T \\ C \end{pmatrix})$ is neither large nor small (see Remark 6.1). Assume that $||\mathbf{b}|| = ||A|| = 1$. Then by virtue of the following theorem the map $A \to C = (-\eta\mathbf{b} \mid A)$ is expected to precondition the matrix A on the average pair of A and $\mathbf{b}$ provided the matrix A has numerical nullity 1.

Theorem 13.1. *Suppose $C = (-\mathbf{b} \mid A)$, $||A|| = ||\mathbf{b}|| = 1$, $A = S_A\Sigma_A T_A^H$ is a full SVD of an $n \times n$ matrix A, $S_A^H S_A = S_A S_A^H = T_A^H T_A = T_A T_A^H = I_n$, $\Sigma_A = \mathrm{diag}(\sigma_i)_{i=1}^n$, $\sigma_i = \sigma_i(A)$ for all i, $\mathbf{f} = (f_i)_{i=1}^n = -S_A^H\mathbf{b}$, $f_n \neq 0$, and $\gamma(\mathbf{f}) = \max_{i=1}^{n-1}|f_i|$. Then $\sigma_n(C) \geq \frac{|f_n|\sigma_{n-1} - (1+|f_n|)\sigma_n}{1+|f_n|}$.*

Proof. Write $\Sigma = \Sigma_A$ and $(\mathbf{f} \mid \Sigma) = S_A^H C\,\mathrm{diag}(1, T_A)$, so that $||\mathbf{f}|| = ||\mathbf{b}|| = 1$ and $\sigma_n(C) = \sigma_n(\mathbf{f} \mid \Sigma)$.

Let G be the $n \times n$ matrix obtained by deleting the last column of the matrix $(\mathbf{f} \mid \Sigma)$. The matrix G is nonsingular for $f_n \neq 0$, and we deduce from the Courant–Fischer minimax theorem that $\sigma_n(\mathbf{f} \mid \Sigma) \geq \sigma_n(G) - \sigma_n$. Therefore

$$\sigma_n(C) \geq \sigma_n(G) - \sigma_n. \tag{13.1}$$

It remains to estimate the values $\sigma_n(G) = \frac{1}{||G^{-1}||}$ below or $||G^{-1}||$ from above. Write

$$g_i = \sigma_{i-1} \text{ and } \widehat{f}_i = f_{i-1} \text{ for } i = 2, \dots, n;\ g_1 = f_n\ , \widehat{f}_1 = 0, \text{ and } \widehat{\mathbf{f}} = (\widehat{f}_i)_{i=1}^n \tag{13.2}$$

and cyclically shift the rows of the matrix G down to arrive at the matrix $\widehat{G} = \mathrm{diag}(g_i)_{i=1}^n + \widehat{\mathbf{f}}\mathbf{e}_1^T$. Clearly $||\widehat{G}^{-1}|| = ||G^{-1}||$.

We have $\widehat{G} = \mathrm{diag}(g_i)_{i=1}^n(I_n + (\frac{\widehat{f}_i}{g_i})_{i=1}^n\mathbf{e}_1)$. Combine this equation and equations (13.2) and deduce that

$$\widehat{G}^{-1} = (I_n - (\frac{\widehat{f}_i}{g_i})_{i=1}^n\mathbf{e}_1^T)\,\mathrm{diag}(\frac{1}{g_i})_{i=1}^n = \mathrm{diag}(0, \mathrm{diag}(\frac{1}{\sigma_i})_{i=1}^{n-1}) - (\frac{\widehat{f}_i}{g_i f_n})_{i=1}^n\mathbf{e}_1^T + \frac{1}{f_n}\mathbf{e}_1\mathbf{e}_1^T.$$

Substitute $\widehat{f}_i = f_{i-1}$ for $i = 2, \dots, n$; $\widehat{f}_1 = 0$ (cf. (13.2)), and $\frac{1}{f_n} = \frac{f_n}{f_n g_1}$ and obtain that

$$||G^{-1}|| = ||\widehat{G}^{-1}|| \leq ||\,\mathrm{diag}(\frac{1}{\sigma_i})_{i=1}^{n-1}|| + ||(\frac{f_i}{g_i f_n})_{i=1}^n\mathbf{e}_n^T||.$$

Since $g_i = \sigma_i \geq \sigma_{n-1}$ for $i < n$ and $||(f_i)_{i=1}^{n-1}|| \leq ||\mathbf{f}|| = 1$, it follows that

$$||G^{-1}|| \leq \frac{1}{\sigma_{n-1}} + \frac{1}{|f_n|\sigma_{n-1}}||(f_i)_{i=1}^n|| \leq \frac{1}{\sigma_{n-1}} + \frac{1}{|f_n|\sigma_{n-1}} = \frac{1+|f_n|}{|f_n|\sigma_{n-1}}.$$

Consequently $\sigma_n(G) \geq \frac{|f_n|\sigma_{n-1}}{1+|f_n|}$ and $\sigma_n(C) \geq \frac{|f_n|\sigma_{n-1}}{1+|f_n|} - \sigma_n = \frac{|f_n|\sigma_{n-1} - (1+|f_n|)\sigma_n}{1+|f_n|}$. ∎

Corollary 13.1. *Under the assumptions of Theorem 13.1 we have* $\kappa(C) \le \frac{(1+|f_n|)\sqrt{2}}{|f_n|\sigma_{n-1}-(1+|f_n|)\sigma_n}$.

Proof. Recall that $\kappa(C) = \frac{||C||}{\sigma_n(C)}$ and $||C|| \le \sqrt{||A|| + ||\mathbf{b}||} = \sqrt{2}$. Substitute these relationships into Theorem 13.1. □

Suppose that under the assumptions of Theorem 13.1 and Corollary 13.1 we have $\sigma_n \ll \sigma_{n-1}$, whereas the ratio σ_1/σ_{n-1} and the value $|f_n|$ are not small. Then the matrix A has numerical nullity 1, we have the bound $|f_n|\sigma_{n-1} \gg (1 + |f_n|)\sigma_n$, $\kappa(C)$ has at most order σ_1/σ_{n-1} (compared to $\kappa(A) = \sigma_1/\sigma_n$), and therefore C is a well conditioned matrix. Note that on the average $|f_n| = \frac{1}{\sqrt{n}}$ on the unit sphere $||\mathbf{f}|| = 1$. Corollary 13.1 is closely linked to Theorem 10.2 for $q = 1$ where $\mathbf{b}$ is an $||A||$-standard Gaussian random vector.

A matrix is ill conditioned if it has numerical nullity 1, and we must perform some stages of our algorithm with a high precision. We can, however, confine the high precision computations to iterative refinement of a null vector of the well conditioned matrix C; overall this takes by a factor n less time than the solution of the system $A\mathbf{y} = \mathbf{b}$ by means of the customary algorithms such as Gaussian elimination (in Section 14 we elaborate upon such estimates for a similar algorithm).

13.2 Solution with auxiliary matrices defined by anmbs

Assume a nonsingular but ill conditioned matrix $A \in \mathbb{R}^{n\times n}$ with $||A|| \approx 1$ and a small numerical nullity r, and suppose some normalized or unitary anmbs M_1 and N_1 in $\mathbb{R}^{n\times r}$ of the matrices A^T and A, respectively, have been computed, e.g., by means of a numerical version of an algorithm of the previous sections. Then we can generate standard Gaussian random matrix $S \in \mathcal{G}_{0,1}^{n\times(n-r)}$ and compute the matrices $M_0 = A^T S$, $N_0 = AS$, and $F = (M_0 \mid M_1)^T A(N_0 \mid N_1) = \begin{pmatrix} F_{00} & F_{01} \\ F_{10} & F_{11} \end{pmatrix}$ where $F_{ij} = M_i^T A N_j$ for $i, j \in \{0, 1\}$ and $F_{00} \in \mathbb{R}^{(n-r)\times(n-r)}$.

The value $\sigma_{n-r}(F_{00})$ is likely to have order $\sigma_{n-r}(A)$ by virtue of Theorem 2.5; consequently the block F_{00} of the 2×2 block matrix F is expected to be nonsingular and well conditioned because the matrix A has numerical nullity r. Furthermore this block is expected to be dominant. Indeed the matrices $M_1^T A$, AN_1, and consequently the blocks $F_{01} \in \mathbb{C}^{n\times r}$, $F_{10} \in \mathbb{C}^{r\times n}$, and $F_{11} \in \mathbb{C}^{r\times r}$ have the norms of order at most $\sigma_{n-r+1}(A) \ll \sigma_{n-r}(A)$. The $O(n^2 r)$ flops involved in the computation of the $(2n - r)r$ entries of these blocks (versus order n^3 flops used overall) must be performed in extended precision to counter the expected cancellation of the leading digits of their entries.

The map $A \Longrightarrow F$ and the block Gaussian elimination reduce the computation of the inverse A^{-1} and the solution of a linear system $A\mathbf{y} = \mathbf{b}$ to the similar operations with the matrices F_{00} and $G = F_{11} - F_{10}F_{00}^{-1}F_{01} \in \mathbb{C}^{n\times r}$ of smaller sizes, expected to be nonsingular and better conditioned. The tests of this technique have confirmed its power [30].

13.3 Solution with preconditioning via northwestern augmentation

Assume an $n \times n$ ill conditioned input matrix A with a small numerical nullity r and northwestern augmentation $K = \begin{pmatrix} W & V \\ U & A \end{pmatrix}$ for three $||A||$-standard Gaussian random matrices $W \in \mathbb{C}^{q\times q}$, U and V. Then the matrix K and its block W are expected to be nonsingular and well conditioned (see Corollary 2.3, part (c) of Theorem 11.1, and the end of Section 11.2). Now we are ready to employ the following theorem.

Theorem 13.2. *Let* $K = \begin{pmatrix} W & V \\ U & A \end{pmatrix}$ *where* A, W *and* K *are nonsingular matrices of sizes* $n \times n$, $q \times q$, *and* $(n + q) \times (n + q)$, *respectively, for* $0 < r \le q < n$. *Write* $S = A - UW^{-1}V$ *and* $R = I_q + VS^{-1}UW^{-1}$; S *is the Schur complement of the block* W *in the matrix* K. *Then*

(a) S^{-1} *is the* $n \times n$ *trailing principal (that is southeastern) block of the matrix* K^{-1} *and*

(b) $A^{-1} = S^{-1} - S^{-1}UW^{-1}R^{-1}VS^{-1}$.

Proof. Part (a) is well known and is readily verified. Part (b) follows from the Sherman–Morrison–Woodbury formula [13, page 50] applied to the matrix $A = S + UW^{-1}V$. □

The theorem implies that $\mathbf{y} = A^{-1}\mathbf{b} = (S^{-1} - S^{-1}UW^{-1}R^{-1}VS^{-1})\mathbf{b}$ for $R = I_q + VS^{-1}UW^{-1}$ and $S^{-1} = (O_{n,q} \mid I_n)K^{-1}\begin{pmatrix} O_{q,n} \\ I_n \end{pmatrix}$ where S^{-1} is the $n \times n$ trailing principal block of the matrix $K^{-1} = \begin{pmatrix} X & Y \\ Z & S^{-1} \end{pmatrix}$. This reduces the solution of the linear system $A\mathbf{y} = \mathbf{b}$ essentially to the computation of the matrices W^{-1}, S, R, R^{-1}, and $S^{-1}(\mathbf{b} \mid U)$. Here W and R are $q \times q$ matrices, $\kappa(W) = \kappa(R) = 1$ for $q = 1$, but the matrix W is expected to be well conditioned for any $q \geq r$; for sufficiently large integers q so are the matrices K and S as well. *The approach works whenever we have a small upper bound q* on r, and becomes more effective where we decrease q toward r.

14 Computational cost estimates

Assume that A is a nonsingular $n \times n$ matrix that has numerical nullity $r = 1$ and apply our algorithm of Section 13.2 to compute the solution $\mathbf{y}$ of the linear system $A\mathbf{y} = \mathbf{b}$ with an output precision p_{out}. The algorithm reduces our task to the solution of two nonsingular and well conditioned linear systems with the matrix $F = F_{00}$ and the right-hand side vectors F_{01} and $\mathbf{b}$. Let $F\mathbf{x} = \mathbf{c}$ denote any of them, and next estimate the cost of its solution.

With Gaussian elimination one needs precision of at least the order $p_+ \approx p_{\text{out}} + \log_2 \kappa(A)$ to compute the vector $\mathbf{y} = A^{-1}\mathbf{b}$ with the precision p_{out}, whereas we solve the same problem by by applying the classical iterative refinement [13], [14], [34] to an auxiliary well conditioned linear system $F\mathbf{x} = \mathbf{c}$. We perform all computations in a fixed lower precision p (e.g., in the standard IEEE single or double precision) such that

$$2\log_2 \kappa(F) \leq p \ll p_+ \approx p_{\text{out}} + \log_2 \kappa(A). \tag{14.1}$$

Flowchart 14.1. Solution of a linear system with iterative refinement.

COMPUTATIONS:

1. *Apply $O(n^3)$ flops of Gaussian elimination in a fixed low precision p satisfying (14.1) to compute an approximate inverse $X \approx F^{-1}$ and an approximate solution $X\mathbf{c}$ to the linear system $F\mathbf{x} = \mathbf{c}$.*
2. *Iteratively refine this solution.*

Every loop of iterative refinement essentially amounts to multiplying each of the matrices F and $X \approx F^{-1}$ by a vector (this takes $4n^2 + O(n)$ flops in the precision p in the case of general matrix F) and contributes about $b = p - \log_2 \kappa(F)$ correct bits per an output entry (cf. [13], [14], [24], [30], [34]); we can assume that $b \geq p/2$ under (14.1). This means that the flowchart involves $O(n^3 + n^2 p_+/p)$ flops in the precision p versus $\frac{2}{3}n^3 + O(n^2)$ flops in the high precision $p_+ \approx p_{\text{out}} + \log_2 \kappa(A)$ used in the customary solution by means of Gaussian elimination. Thus our advance is dramatic where p_+ greatly exceeds p and is greatly exceeded by pn.

If the matrix A has numerical nullity $r > 1$, then the algorithm in Section 13.2 reduces the original linear system $A\mathbf{y} = \mathbf{b}$ to $2r$ well conditioned linear systems, each of $n - r$ equations; the cost estimates increase by a factor r, versus the case of $r = 1$, but we still dramatically advance Gaussian elimination as long as p_+ greatly exceeds p and is greatly exceeded by pn/r. Thus for an important and quite general class of ill conditioned matrices A having a small numerical nullity r (cf. Remark 2.3), our estimates still strongly favor our solution based on Flowchart 14.1 versus the standard algorithms such as Gaussian elimination.

Remark 14.1. *We can obtain the same dramatic progress versus Gaussian elimination by employing the algorithm of Section 13.3 rather than 13.2.*

Remark 14.2. *We refer the reader to [22, Section 6.9] or [23] on the inversion of nonsingular Toeplitz and various other structured matrices in nearly linear arithmetic time based on deterministic homotopy continuation techniques.*

Remark 14.3. *In lieu of iterative refinement one can employ other iterations such as the CG (Conjugate Gradient) and GMRES algorithms (cf. [13, Sections 10.2–10.4]). Their iteration loop also performs $O(n^2)$ flops, but unlike iterative refinement, those algorithms use no approximate inverse. This is a significant advantage in the case of a sparse unstructured linear system as well as a multilevel Toeplitz or Hankel linear system [8], [19]. Decreasing the condition number of an input matrix, however, is more critical (and thus the success of preconditioning is more important) for the convergence of such algorithms versus iterative refinement. In particular every nonsymmetric CG iteration loop amounts essentially to multiplication of an input matrix M and its transpose M^T by two vectors but only ensures the decrease of the error norm by a factor $4(\frac{\sqrt{\kappa(M)}+1}{\sqrt{\kappa(M)}-1})^2$ [13, Theorem 10.2.6] or equivalently ensures about $1/\kappa(M)$ new correct bits per an output value. Thus the algorithm requires stronger bounds on $\kappa(M)$ to guarantee convergence in the presence of rounding errors.*

15 Numerical tests

In a series of numerical experiments performed in the Graduate Center of the City University of New York, we tested our algorithms for computing nmbs and null vectors of general and Toeplitz matrices. We conducted the tests on a Dell server with a dual core 1.86 GHz Xeon processor and 2G memory running Windows Server 2003 R2. The test Fortran code was compiled with the GNU gfortran compiler within the Cygwin environment. We generated random numbers with the random_number intrinsic Fortran function assuming the uniform probability distribution over the range $\{x : \ 0 \le x < 1\}$. To shift to the range $\{y : \ b \le y \le a + b\}$ for fixed real a and b, we applied the linear transform $x \to y = ax + b$. CPU time was measured with the mclock function. We computed QR factorizations and SVDs by applying the LAPACK procedures DGEQRF and DGESVD, respectively. The tests have been designed by the first author and performed by his coauthor.

15.1 Computations with Toeplitz matrices

a) Generation of rank deficient Toeplitz matrices

To generate an $n \times n$ singular Toeplitz matrix, we first sampled $2n-2$ random entries $a_{i,j}$ in the range $[-1, 1)$ for $j = 1$, $i = 1, \dots, n-1$ and for $i = 1$, $j = 2, \dots, n$; then we defined the $(n-1)^2$ entries $a_{i+1,j+1} = a_{i,j}$ for $i, j = 1, \dots, n-1$, and finally set $a_{n,1} = 0$. We arrived at an $n \times n$ Toeplitz matrix $A_0 = (a_{i,j})_{i,j=1}^{n}$, computed the entry $y_{n,1}$ of its inverse $A_0^{-1} = (y_{i,j})_{i,j=0}^{n-1}$, and changed the $(1,n)$th entry of the matrix A_0 into $a_{n,1} = -1/y_{n,1}$. As we expected in view of Lemma 2.1, we always had $y_{n,1} \det A_0 \neq 0$ in our tests. Had we had $y_{n,1} = 0$, we could have regenerated the matrix A_0, whereas had this matrix been singular, we would have output it and stopped the computations.

The resulting matrix $A = (a_{i,j})_{i,j=1}^{n}$ had nullity 1. Indeed it was a rank-one modification of a nonsingular matrix A_0, whereas $A\mathbf{y} = \mathbf{0}$ for $\mathbf{y} = A_0^{-1}\mathbf{e}_1$ because $A_0\mathbf{y} = \mathbf{e}_1$, $A = A_0 - \frac{1}{y_{n,1}}\mathbf{e}_1\mathbf{e}_n^T$, and $\mathbf{e}_n^T\mathbf{y} = y_{n,1}$.

b) Augmentation of singular Toeplitz matrices and the computation of their null vectors

We embedded our $n \times n$ singular Toeplitz matrix $A = (a_{i,j})_{i,j=1}^{n}$ into an $(n+1) \times (n+1)$ Toeplitz matrix $K = (a_{i,j})_{i,j=0}^{n} = \begin{pmatrix} w & \mathbf{v}^T \\ \mathbf{u} & A_0 \end{pmatrix}$ for $w = a_{0,0}$, $\mathbf{u} = (a_{i,0})_{i=1}^{n}$, and $\mathbf{v} = (a_{0,j})_{j=1}^{n}$. We defined the entries $a_{i,0}$ and $a_{0,j}$ for $i, j = 0, 1, \dots, n-1$ by applying the equations $a_{i,j} = a_{i+1,j+1}$

and sampled the two entries $a_{n,0}$ and $a_{0,n}$ at random in the range $[-1, 1)$. For such a matrix K we applied Theorem 11.2 for $r = 1$, to compute a null vector of the matrix A given by the vector $(0, I_n)K^{-1}\begin{pmatrix}0\\ \mathbf{u}\end{pmatrix}$. This amounted to solving a nonsingular Toeplitz linear systems of equations with the matrix K. For that task we applied the code in [36], based on the algorithms in [17], [37], [38]. For comparison we also obtained the null vectors of the same matrices A based on computing their QR factorizations and SVDs. We have a little decreased the CPU time by using QR rather than QRP factorization. The latter one, that is QR factorization with pivoting (performed by LAPACK procedures DGEQPF and DGEQP3) is recommended for dealing with ill conditioned inputs [13, Section 5.5], but we avoided them in our tests.

Remark 15.1. *Instead of augmentation in these tests one could have replaced the values in the corner entries $a_{0,n}$ and $a_{n,0}$ by properly scaled random values.*

c) Output data in the tests with Toeplitz matrices

We use the abbreviations "n.-w.a.", "QR", and "SVD" as our pointers to the northwestern augmentation (based on Algorithm 11.1), QR factorization, and SVD, respectively. Table 15.1 covers our computation of the null vectors for Toeplitz matrices. It shows the CPU time of this computation for each of the three methods as well as the ratios of these data for the QR-based and SVD-based solutions versus northwestern augmentation. The ratios are displayed in the last two columns of the table. The CPU time is measured in terms of the CPU cycles. One can convert them into seconds by dividing them by a constant CLOCKS_PER_SEC, which is 1000 on our platform.

In all our tests the computed approximate null vectors $\mathbf{y}$ had relative residual norms $\frac{||A\mathbf{y}||}{||A||\,||\mathbf{y}||}$ of order 10^{-17}.

All data are average over 100 tests for each input size 2^k from 256 to 8192. The table entries are marked by a hyphen "-" where the tests required too long runtime and were not completed.

Table 15.1: CPU time for computing a null vector of a Toeplitz matrix (in cycles)

dimension	n.-w.a.	QR	SVD	QR/n.-w.a.	SVD/n.-w.a
256	3.8	18.4	317.8	4.8	83.6
512	8.0	148.0	5242.1	18.5	655.3
1024	16.1	1534.2	87371.2	97.0	5522.6
2048	33.6	11750.3	–	357.7	–
4096	79.5	–	–	–	–
8192	169.5	–	–	–	–

15.2 Generation of general nonsingular matrices and preconditioning

We first fixed pairs of n and k for $n = 64, 128$ and $k = 7$. Then for every pair (n, k) we generated 100 instances of matrices A, U, V_0, and V_1 and vectors $\mathbf{b}$. We generated the matrices A as the error-free products $S\Sigma T^H$ where S and T were $n \times n$ random orthonormal matrices (generated with double precision) and $\Sigma = \mathrm{diag}(\sigma_j)_{j=1}^n$, $\sigma_{n-j} = 10^{j-17}$ for $j = 1, \dots, k$, whereas $\sigma_{n-j} = 1/(n-j)$ for $j = k+1, \dots, n-1$ (cf. [14, Section 28.3]), and so $||A|| = 1$, $\kappa(A) = ||A^{-1}|| = 10^{-16}$.

U was random $n \times k$ matrix with $||U|| = ||A|| = 1$.

$V = (V_0 \mid V_1)$ was $k \times (n+k)$ matrix for $V_0 = I_k$ and $V_1 = U^T$.

For every choice of these matrices we computed the ratio $\frac{\kappa(A)}{\kappa(M)}$ for $M = \begin{pmatrix}V_0 & V_1\\ U & A\end{pmatrix}$.

Table 15.2 displays the average (mean), minimum, maximum, and standard deviation for the m ratios for $n = 64$ and $n = 128$.

Table 15.2: ratios $\frac{\kappa(A)}{\kappa(M)}$

matrix size	min	max	mean	std
64×64	3.29×10^{9}	1.65×10^{13}	2.49×10^{12}	2.60×10^{12}
128×128	8.27×10^{8}	2.56×10^{12}	5.51×10^{11}	6.44×10^{11}

15.3 Generation of rectangular matrices and computation of their nmbs

At first we fixed pairs m and n where $m = 64, 128$, $n = m + g$, and $g = 1, 4, 16$; for each pair we generated 100 instances of $m \times n$ pairs of random matrices M and H with entries in the range $[-1, 1)$ and $g \times (m - g)$ matrices F with entries -1 and 1 chosen at random. Then we defined the matrix $G = \begin{pmatrix} I_{m-g} & O_{m-g,g} \\ F & O_{g,g} \end{pmatrix}$ and applied Algorithms 3.1, 4.1, and 6.1 to compute $B = \mathrm{nmb}(A)$ for the following three classes of matrices A:

(a) $A = M$,

(b) $A = GM$ (having rank $m - g$), and

(c) $A = GM + 10^{-10}H$ (having rank m and numerical rank $m - g$).

In the cases (a) and (c) we had $\mathrm{rank}(M) = m$, V was $(n-m) \times n$ random matrix, and Algorithm 6.1 inverted a nonsingular $n \times n$ matrix C such that $C^{(I)} = C^{-1}$. In the case (b) we had $\mathrm{rank}(M) = m - g$, V was $(n - m + g) \times n$ random matrix, Algorithm 6.1 dealt with an $(n + g) \times n$ matrix $C = \begin{pmatrix} C_n \\ C_g \end{pmatrix}$ where C_n was a nonsingular $n \times n$ matrix, and we set $C^{(I)} = (C_n^{-1} \mid O_{n,g})$.

We applied Algorithms 3.1 and 4.1 to the matrices of class (a); we applied Algorithm 6.1 to the matrices of all three classes (a), (b) and (c). Tables 15.3 and 15.4 represent the relative residual norms $r = \frac{||AB||}{||A||\,||A||}$ in our tests.

Table 15.3: relative residual norms in the solution tests with $m \times n$ inputs for $m = 64$

Algorithm/class	min	max	mean	std
3.1/(a) $g = 1$	2.38×10^{-16}	1.19×10^{-13}	2.96×10^{-15}	1.22×10^{-14}
3.1/(a) $g = 4$	2.06×10^{-16}	5.94×10^{-16}	3.59×10^{-16}	8.16×10^{-17}
3.1/(a) $g = 16$	1.40×10^{-16}	2.24×10^{-16}	1.74×10^{-16}	1.40×10^{-17}
4.1/(a) $g = 1$	6.02×10^{-16}	5.20×10^{-13}	2.55×10^{-14}	7.20×10^{-14}
4.1/(a) $g = 4$	4.14×10^{-16}	5.35×10^{-15}	1.41×10^{-15}	7.99×10^{-16}
4.1/(a) $g = 16$	1.49×10^{-16}	7.81×10^{-16}	3.42×10^{-16}	1.14×10^{-16}
6.1/(a) $g = 1$	5.94×10^{-17}	1.75×10^{-16}	1.06×10^{-16}	2.06×10^{-17}
6.1/(a) $g = 4$	7.55×10^{-17}	1.49×10^{-16}	1.05×10^{-16}	1.44×10^{-17}
6.1/(a) $g = 16$	9.65×10^{-17}	1.41×10^{-16}	1.15×10^{-16}	1.12×10^{-17}
6.1/(b) $g = 1$	6.38×10^{-17}	1.29×10^{-16}	9.23×10^{-17}	1.50×10^{-17}
6.1/(b) $g = 4$	5.44×10^{-17}	1.30×10^{-16}	7.88×10^{-17}	1.27×10^{-17}
6.1(b) $g = 16$	5.28×10^{-17}	9.97×10^{-17}	7.11×10^{-17}	9.71×10^{-18}
6.1/(c) $g = 1$	6.17×10^{-17}	2.04×10^{-16}	9.85×10^{-17}	2.28×10^{-17}
6.1/(c) $g = 4$	5.43×10^{-17}	1.31×10^{-16}	8.17×10^{-17}	1.28×10^{-17}
6.1(c) $g = 16$	5.59×10^{-17}	1.56×10^{-16}	7.69×10^{-17}	1.46×10^{-17}

16 Conclusions

The computation of a basis for the null space of a rectangular $m \times n$ matrix A having row rank m can rely on computing rank revealing LU or QR factorization (with pivoting) or SVD. The

Table 15.4: relative residual norms in the solution tests with $m \times n$ inputs for $m = 128$

Algorithm/class	**min**	**max**	**mean**	**std**
3.1/(a) $g = 1$	2.88×10^{-16}	7.25×10^{-14}	3.65×10^{-15}	8.76×10^{-15}
3.1/(a) $g = 4$	3.13×10^{-16}	8.21×10^{-16}	4.45×10^{-16}	9.02×10^{-17}
3.1/(a) $g = 16$	1.94×10^{-16}	2.61×10^{-16}	2.21×10^{-16}	1.30×10^{-17}
4.1/(a) $g = 1$	1.84×10^{-15}	1.79×10^{-12}	1.18×10^{-13}	2.80×10^{-13}
4.1/(a) $g = 4$	9.84×10^{-16}	2.37×10^{-14}	4.14×10^{-15}	2.79×10^{-15}
4.1/(a) $g = 16$	4.98×10^{-16}	1.40×10^{-15}	9.25×10^{-16}	2.22×10^{-16}
6.1/(a) $g = 1$	1.06×10^{-16}	2.28×10^{-16}	1.48×10^{-16}	2.28×10^{-17}
6.1/(a) $g = 4$	1.05×10^{-16}	2.04×10^{-16}	1.51×10^{-16}	1.86×10^{-17}
6.1/(a) $g = 16$	1.26×10^{-16}	1.95×10^{-16}	1.57×10^{-16}	1.33×10^{-17}
6.1/(b) $g = 1$	8.77×10^{-17}	1.88×10^{-16}	1.34×10^{-16}	1.97×10^{-17}
6.1/(b) $g = 4$	7.78×10^{-17}	1.92×10^{-16}	1.12×10^{-16}	1.80×10^{-17}
6.1(b) $g = 16$	6.48×10^{-17}	1.78×10^{-16}	9.34×10^{-17}	1.61×10^{-17}
6.1/(c) $g = 1$	9.31×10^{-17}	2.45×10^{-16}	1.34×10^{-16}	2.72×10^{-17}
6.1/(c) $g = 4$	8.48×10^{-17}	1.48×10^{-16}	1.10×10^{-16}	1.45×10^{-17}
6.1(c) $g = 16$	7.00×10^{-17}	1.95×10^{-16}	9.92×10^{-17}	1.85×10^{-17}

computations are numerically stable, but destroy matrix sparseness and structure and are expensive even for general matrices. To fix these mishaps we proposed alternative algorithms employing randomization.

We first described a simple nmb algorithm based on nonorthogonal projection. It squared the condition number $\kappa(A)$, but we avoided this deficiency in our second nmb algorithm based on pre- and post-multiplication of the input matrix A by random matrices. The algorithm is expected to produce a desired nmb and to be numerically safe in the case of a well conditioned matrix A.

Both projection and multiplication by random matrices still destroy the structure and sparseness of an input matrix, and we described and analyzed some alternative techniques that countered the problem provided the input matrix has a small nullity or a small numerical nullity. The resulting nmb algorithms worked for a rank deficient and ill conditioned matrix A of any size by employing binary search, randomization, augmentation or block row modification, and aggregation. We obtained a desired basis for the null space by performing all computations with well conditioned matrices of full rank. Then we extended our algorithms to preconditioning a nonsingular but ill conditioned linear system of equations $A\mathbf{y} = \mathbf{b}$.

With our augmentation, block modification and aggregation we preserved sparseness and structure of an input matrix, avoided the drawbacks of pivoting and orthogonalization, and according to our formal study and experiments, significantly accelerated the customary algorithms without losing the output accuracy.

Some parts of our analysis, in particular our estimates for the condition numbers of randomly augmented matrices, extension of our nmb algorithms to solving nonsingular linear systems of equations, and the link between augmentation and aggregation can be of independent technical interest.

There are interesting challenges for further research, such as the choice of proper combination of northern and western augmentations, block modification, and other techniques of this paper and [27].

References

[1] D. Bini, V. Y. Pan, *Polynomial and Matrix Computations,* volume 1: Fundamental Algorithms, Birkhäuser, Boston, 1994.

[2] Z. Chen, J. J. Dongarra, Condition Numbers of Gaussian Random Matrices, *SIAM. J. on Matrix Analysis and Applications*, **27**, 603–620, 2005.

[3] J. D. Dixon, Estimating Extremal Eigenvalues and Condition Numbers of Matrices, *SIAM J. on Numerical Analysis*, **20, 4**, 812–814, 1983.

[4] J. Demmel, The Probability That a Numerical Analysis Problem Is Difficult, *Math. of Computation*, **50**, 449–480, 1988.

[5] R. A. Demillo, R. J. Lipton, A Probabilistic Remark on Algebraic Program Testing, *Information Processing Letters*, **7**, **4**, 193–195, 1978.

[6] K. R. Davidson, S. J. Szarek, Local Operator Theory, Random Matrices, and Banach Spaces, in *Handbook on the Geometry of Banach Spaces* (W. B. Johnson and J. Lindenstrauss editors), pages 317–368, North Holland, Amsterdam, 2001.

[7] A. Edelman, Eigenvalues and Condition Numbers of Random Matrices, PhD Thesis (106 pages), Math Dept., MIT, 1989 and *SIAM J. on Matrix Analysis and Applications*, **9**, **4**, 543–560, 1988.

[8] I. Z. Emiris, V. Y. Pan, Symbolic and Numerical Methods for Exploiting Structure in Constructing Resultant Matrices, *J. of Symbolic Computation*, **33**, 393-413, 2002. Proc. Version in *ISSAC 97*.

[9] A. Edelman, B. D. Sutton, Tails of Condition Number Distributions, *SIAM J. on Matrix Analysis and Applications*, **27**, **2**, 547–560, 2005.

[10] I. Gohberg, T. Kailath, V. Olshevsky, Fast Gaussian Elimination with Partial Pivoting for Matrices with Displacement Structure, *Mathematics of Computation*, **64**, 1557–1576, 1995.

[11] I. Gohberg, V. Olshevsky, Complexity of Multiplication with Vectors for Structured Matrices, *Linear Algebra and Its Applications*, **202**, 163–192, 1994.

[12] I. Gohberg, A. Sementsul, On the Inversion of Finite Toeplitz Matrices and Their Continuous Analogs, *Matematicheskiie Issledovaniia* (in Russian), **7**, **2**, 187–224, 1972.

[13] G. H. Golub, C. F. Van Loan, *Matrix Computations*, 3rd edition, The Johns Hopkins University Press, Baltimore, Maryland, 1996.

[14] N. J. Higham, *Accuracy and Stability in Numerical Analysis*, SIAM, Philadelphia, 2002 (second edition).

[15] G. Heinig, K. Rost, *Algebraic Methods for Toeplitz-like Matrices and Operators, Operator Theory*, **13**, Birkhäuser, 1984.

[16] T. Kailath, S. Y. Kung, M. Morf, Displacement Ranks of Matrices and Linear Equations, *Journal Math. Analysis and Appls*, **68(2)**, 395–407, 1979.

[17] P. Kravanja, M. Van Barel, Algorithms for Solving Rational Interpolation Problems Related to Fast and Superfast Solvers for Toeplitz Systems, *SPIE*, 359–370, 1999.

[18] W. L. Miranker, V. Y. Pan, Methods of Aggregations, *Linear Algebra and Its Applications*, **29**, 231–257, 1980.

[19] B. Mourrain, V. Y. Pan, Multivariate Polynomials, Duality and Structured Matrices, *J. of Complexity*, **16**, **1**, 110–180, 2000. (Proceedings version in *STOC'98*).

[20] V. Y. Pan, On Computations with Dense Structured Matrices, *Math. of Computation*, **55, 191**, 179–190, 1990. Proceedings version in *Proceedings of International Symposium on Symbolic and Algebraic Computation (ISSAC'89)*, 34–42, ACM Press, NY, 1989.

[21] C.–T. Pan, On the Existence and Computation of Rank-revealing LU Factorization, *Linear Algebra and Its Applications*, **316**, 199–222, 2000.

[22] V. Y. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms*, Birkhäuser/Springer, Boston/New York, 2001.

[23] V. Y. Pan, Newton's Iteration for Matrix Inversion, Advances and Extensions, pp. 364–381, in *Matrix Methods: Theory, Algorithms and Applications* (dedicated to the Memory of Gene Golub, edited by V. Olshevsky and E. Tyrtyshnikov), World Scientific Publishing, New Jersey, ISBN-13 978-981-283-601-4, ISBN-10-981-283-601-2 (2010).

[24] V. Y. Pan, D. Grady, B. Murphy, G. Qian, R. E. Rosholt, A. Ruslanov, Schur Aggregation for Linear Systems and Determinants, *Theoretical Computer Science*, *Special Issue on Symbolic–Numerical Algorithms* (D.A. Bini, V.Y. Pan, J. Verschelde editors), **409**, **2**, 255–268, 2008.

[25] V. Y. Pan, D. Ivolgin, B. Murphy, R. E. Rosholt, Y. Tang, X. Yan, Additive Preconditioning for Matrix Computations, *Linear Algebra and Its Applications*, **432**, 1070–1089, 2010. Proceedings version in *Proc. of the Third International Computer Science Symposium in Russia (CSR 2008)*, *Lecture Notes in Computer Science (LNCS)*, **5010**, 372–383, 2008.

[26] V. Y. Pan, B. Murphy, G. Qian, R. E. Rosholt, A New Error-free Floating-Point Summation Algorithm, *Computers and Mathematics with Applications*, **57**, 560–564, 2009.

[27] V. Y. Pan, G. Qian, Randomized Preprocessing of Homogeneous Linear Systems, *Linear Algebra and Its Applications*, **432**, 3272–3318, 2010.

[28] V. Y. Pan, G. Qian, A. Zheng, Advancing Matrix Computations with Randomized Preprocessing, in *Proc. of the Fifth International Computer Science Symposium in Russia (CSR 2010)*, Kazan, Russia, June 2010 , Farid Ablaev, Ernst W. Mayr (Eds.), *Lecture Notes in Computer Science (LNCS)*, pages 303-314, Springer, Berlin, 2010.

[29] V. Y. Pan, G. Qian, A. Zheng, Randomized Preconditioning versus Pivoting, *Linear Algebra and Its Applications*, in print, http://dx.doi.org/10.1016/j.laa.2011.02.052

[30] V. Y. Pan, G. Qian, A. Zheng, Randomized Matrix Computations, Tech. Report TR 2012005, *Ph.D. Program in Computer Science*, *Graduate Center, the City University of New York*, 2012. Available at http://www.cs.gc.cuny.edu/tr/techreport.php?id=432

[31] V. Y. Pan, G. Qian, A. Zheng, Z. Chen, Matrix Computations and Polynomial Root-finding with Preprocessing, *Linear Algebra and Its Applications*, **434**, 854–879, 2011.

[32] V. Y. Pan, X. Yan, Additive Preconditioning, Eigenspaces, and the Inverse Iteration, *Linear Algebra and Its Applications*, **430**, 186–203, 2009.

[33] J. T. Schwartz, Fast Probabilistic Algorithms for Verification of Polynomial Identities, *Journal of ACM*, **27**, **4**, 701–717, 1980.

[34] G. W. Stewart, *Matrix Algorithms, Vol I: Basic Decompositions*, SIAM, Philadelphia, 1998.

[35] A. Sankar, D. Spielman, S.-H. Teng, Smoothed Analysis of the Condition Numbers and Growth Factors of Matrices, *SIAM Journal on Matrix Analysis*, **28**, **2**, 446–476, 2006.

[36] M. Van Barel, A Supefast Toeplitz Solver, 1999.
Available at http://www.cs.kuleuven.be/˜ marc/software/index.html

[37] M. Van Barel, G. Heinig, P. Kravanja, A Stabilized Superfast Solver for Nonsymmetric Toeplitz Systems, *SIAM Journal on Matrix Analysis and Applications*, **23**, **2**, 494–510, 2001.

[38] M. Van Barel, P. Kravanja, A Stabilized Superfast Solver for Indefinite Hankel Systems, *Linear Algebra and its Applications*, **284**, **1–3**, 335–355, 1998.

[39] X. Wang, Affect of Small Rank Modification on the Condition Number of a Matrix, *Computer and Math. (with Applications)*, **54**, 819–825, 2007.

[40] R. E. Zippel, Probabilistic Algorithms for Sparse Polynomials, *Proceedings of EUROSAM'79, Lecture Notes in Computer Science*, **72**, 216–226, Springer, Berlin, 1979.