2011

TR-2011009: Solving Linear Systems of Equations with Randomized Augmentation and Aggregation

Victor Y. Pan

Guoliang Qian

Solving Linear Systems of Equations with Randomized 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 A we combine scaled randomized augmentation with aggregation to reduce our task to computations with well conditioned matrices of full rank. Our algorithms avoid pivoting and orthogonalization and preserve matrix structure and sparseness. In the case of ill conditioned inputs we perform a small part of our computations with high accuracy by applying iterative refinement, but overall we still dramatically accelerate the customary algorithms for rank deficient or ill conditioned linear systems with general and Toeplitz matrices, according to both our formal estimates and experiments. Our study can be of independent technical interest, e.g., we extend the Smoothed Analysis of conditioning of random matrices in [42] to preconditioning by means of randomized augmentation and link augmentation and aggregation. Our progress has been extended to various other fundamental matrix computations.

Key words: Linear systems of equations, Randomized augmentation, Conditioning of random matrices, Toeplitz matrices

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 and is closely linked to some other central subjects of that field (see our Sections 5.4 and 7, [33, Sections 7.2 and 11.1], and [38]).

The solution vectors $\mathbf{y}$ are said to be 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, then we call B a *null matrix basis (nmb)* for a matrix A. Hereafter we refer to such a basis as a nmb(A).

*Supported by NSF Grant CCF-1116736 and PSC CUNY Awards 61406–0039 and 62230–0040. Some results of this paper have been presented at the Fifth International Computer Science Symposium in Russia (CSR 2010) in Kazan' [34] and at the 16th Conference of the International Linear Algebra Society (ILAS) in Pisa, Italy, both in June 2010.

The customary algorithms compute null vectors and nmbs of a matrix by employing its LU or QR factorization with pivoting or its SVD. These computations are quite costly, particularly the computation of the SVD, but even "pivoting usually degrades the performance" [19, page 119].

1.2 Our contribution

For $m \times n$ matrices A of full rank m, the solution can be readily reduced to the inversion of the $m \times m$ leading block; with probability one we ensure its nonsingularity by means of randomized premultiplication. Furthermore with a probability near one this does not blow up the condition number of A, as we prove in Section 3.

The approach, however, does not work where $\operatorname{rank} A < m$, and it fails numerically where the input matrix is ill conditioned. To fix the problem we apply *scaled randomized augmentation* of the input matrix and *aggregation*. Our algorithms handle rank deficient and ill conditioned $m \times n$ matrices for any pair of m and n, and we extend our null space computations to the solution of nonsingular but ill conditioned linear systems of equations. We probabilistically reduce our tasks to computations with well conditioned matrices of full rank, use neither pivoting nor orthogonalization, and preserves matrix structure and sparseness.

We perform a small part of our computations with high accuracy by applying iterative refinement. According to both our formal estimates and experimental computations (see the formal estimates in Section 8 and the observed CPU time in Table 9.1), overall we still accelerate the customary algorithms by order of magnitude and nearly reach optimality in the cases of general and Toeplitz ill conditioned inputs.

Part of our formal study can be of independent technical interest, e.g., our extension of the Smoothed Analysis of conditioning of random matrices in [42] to preconditioning by means of randomized augmentation, our combination of augmentation and aggregation, and our linking of the solution of homogeneous and nonhomogeneous linear systems of equations.

We refer the reader to the papers [30]–[38] on various further extensions and applications of our randomized preprocessing to fundamental matrix and polynomial computations. In particular augmentation is closely linked to additive preprocessing in [30, Section 12] and [33, Section 4] but can a little better preserve matrix structure and sparseness.

1.3 Organization of the paper

In the next section we first recall some definitions and basic facts and then estimate the ranks and condition numbers of random matrices and randomized matrix products. In Section 3 we cover randomized post-multiplication and in Sections 4–6 randomized augmentation. In Section 7 we solve a nonhomogeneous linear system of equations. In Section 8 we estimate the computational cost of our randomized algorithms. In Section 9 we present the results of our numerical tests, which are the contribution of the second author of this paper. We conclude the paper in Section 10.

Acknowledgement. We thank Marc Van Barel's for pointing us his Toeplitz solver in [44].

2 Definitions and basic facts

2.1 General matrices, nmbs and annihilators

We use and extend the customary definitions of matrix computations (cf. [19] and [41]).

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

Flop is an arithmetic operation with such numbers.

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.

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

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

I_n denotes the identity matrix $(\mathbf{e}_i)_{i=1}^n = (\mathbf{e}_1, \dots, \mathbf{e}_n)$.

$O_{k,l}$ denotes the $k \times l$ matrix filled with zeros.

$O_{k,1}$ and $\mathbf{0_k}$ denote the vector of a dimension k filled with zeros.

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 matrix U is unitary or orthonormal if $U^H U = I$.

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 generated by its columns, $\mathcal{N}(A)$ its null space $\{\mathbf{v} : A\mathbf{v} = \mathbf{0}\}$, $\operatorname{rank} A = \dim \mathcal{R}(A)$ its rank, and $\operatorname{nul} A = \dim \mathcal{N}(A)$ its nullity. $\mathbf{v}$ is its null vector if $A\mathbf{v} = \mathbf{0}$.

Fact 2.1. *The set $\mathbb{S}$ of $m \times n$ matrices of rank ρ is an algebraic variety 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 is nonsingular. Therefore $\dim \mathbb{S} = mn - (m-\rho)(n-\rho) = (m+n-\rho)\rho$. □

A matrix H is a *complete annihilator* of a matrix A if $\mathcal{R}(H) = \mathcal{N}(A)$. Such an annihilator is a *null matrix basis* if it has full column rank. We use the abbreviations nmb, ca, nmb(A), and ca(A). Given a ca(A), we can compute a nmb(A) by applying its LUP or QR factorization or the following simple fact.

Fact 2.2. *[33]. Suppose H is a* ca(A). *Then*

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

(b) HY is a nmb(A) *if X is a* ca(H) *and if (X, Y) is a nonsingular matrix.*

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 $S_A S_A^H = S_A^H S_A = I_m$, $T_A T_A^H = T_A^H T_A = I_n$, $\Sigma_A = \operatorname{diag}(\widehat{\Sigma}_A, O_{m-\rho,n-\rho})$, and $\widehat{\Sigma}_A = \operatorname{diag}(\sigma_j)_{j=1}^{\rho}$.

Here $\sigma_j = \sigma_j(A) = \sigma_j(A^H)$ is the jth largest singular value of a matrix A for $j = 1, \dots, \rho$.

The Courant–Fischer minimax theorem [19, 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, and so we write $\sigma_j = 0$ for $j > \rho$. The same theorem also implies

Fact 2.3. *If A_0 is a $p \times q$ submatrix of a matrix A, then $\sigma_j(A) \geq \sigma_j(A_0)$ for all j.*

$\Sigma_A^+ = \operatorname{diag}((\widehat{\Sigma}_A)^{-1}, O_{n-\rho,m-\rho})$ and $A^+ = T_A \Sigma_A^+ S_A^H$ are the Moore–Penrose pseudo-inverses of the matrices Σ_A and A, respectively.

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

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

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

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

Hereafter 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.

We say that a matrix A is *mildly normalized* if its norm $||A||$ is neither large nor small.

We write $a \ll b$ and $b \gg a$ if the ratio b/a is large and write $a \approx b$ if the ratio is close to one or if $b = 0$ and $|a|$ is small. For two matrices A and B we write $A \approx B$ if $||A - B|| \ll ||A||$.

$\kappa(A) = \sigma_1(A)/\sigma_\rho(A) = ||A||\ ||A^+||$ is the condition number of a matrix A of a rank ρ. Such a matrix is *ill conditioned* if $\sigma_1(A) \gg \sigma_\rho(A)$ and is *well conditioned* otherwise. See [7], [19, Sections

2.3.2, 2.3.3, 3.5.4, 12.5], [21, Chapter 15], and [41, Section 5.3] on effective estimation of norms and condition numbers.

If $\sigma_q(A) > \sigma_{q+1}(A)$, $\bar{r} = m-q$ and $r = n-q$, then we write $S_{A,\bar{r}} = S_A(O_{q,\bar{r}} \mid I_{\bar{r}})^T$, $\mathbb{S}_{A,\bar{r}} = \mathcal{R}(S_{A,\bar{r}})$, $T_{A,r} = T_A(O_{q,r} \mid I_r)^T$, and $\mathbb{T}_{A,r} = \mathcal{R}(T_{A,r})$.

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 are not small relative to its norm. By setting to zero 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. Conversely adding a random matrix of a small norm to an $m \times n$ well conditioned matrix A of rank $q = l - r$ (for $0 < r < l = \min\{m, n\}$) is likely to produce an ill conditioned matrix $\tilde{A}$ that has full rank l and numerical rank q (cf. Corollary 2.2). The linear space $\mathbb{T}_{\tilde{A},r}$ approximates the null space of the matrix A; likewise the linear space $\mathbb{S}_{\tilde{A},\bar{r}}$ approximates the null space of the matrix A^T.

Remark 2.1. *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.*

Hereafter a matrix B is said to be an *approximate nmb* or an *approximate ca* of a matrix A if $B \approx \mathrm{nmb}(A)$ or $B \approx \mathrm{ca}(A)$, respectively. We employ an approximate nmb in the following simple theorem to approximate the matrix $\tilde{A}$ by a matrix of a smaller rank.

Theorem 2.1. *(See [33, Section 7.2]). Suppose an $n \times n$ matrix $\tilde{A}$ has positive numerical nullity r and suppose B is an $n \times r$ matrix such that the $r \times r$ matrix $B^H B$ is nonsingular and $||\tilde{A}B|| \ll ||\tilde{A}||\ ||B||$. Then the matrix $\tilde{A}(I - B(B^H B)^{-1}B^H)$ (equal to the matrix $\tilde{A}(I - BB^H)$ where B is a unitary matrix) closely approximates the matrix $\tilde{A}$ and has rank $n - r$.*

Remark 2.2. *Unlike the nullity and the rank, numerical nullity and numerical rank are not well defined for a large class of ill conditioned matrices, in particular for all matrices A having nested clusters of small singular values but also for the matrix class exemplified by for a 1000×1000 matrix A with singular values $\sigma_j(A) = 10^{1000-j}$, $j = 1, 2, \ldots, 1000$.*

2.3 Structured matrices

$J = J_n = (\mathbf{e}_1, \ldots, \mathbf{e}_n) = (j_{i,k})_{i,k=0}^{n-1}$ is the $n \times n$ reflection matrix, $j_{i,k} = 1$ if $i + k = n - 1$, $j_{i,k} = 0$ unless $i + k = n - 1$; $J^2 = I$.

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 its $m + n - 1$ entries, e.g., by its first row and first (resp. last) column. TJ and JT are Hankel matrices for a Toeplitz matrix T, whereas HJ and JH are Toeplitz matrices for a Hankel matrix H.

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

The Gohberg–Semencul formula of [18] expresses the inverse of a nonsingular $n \times n$ Toeplitz matrix T via its first column $T^{-1}\mathbf{e}_1$ and its last column $T^{-1}\mathbf{e}_n$ provided $\mathbf{e}_1^T T^{-1}\mathbf{e}_1 \neq 0$. The latter provision has been relaxed in [20], [17] and [43]. The following theorem of [18] (cf. [3, Theorem 7]) expresses the inverse T^{-1} via two columns (the first $\mathbf{v}$ and the last $\mathbf{w}$) of the inverse of an $(n + 1) \times (n + 1)$ Toeplitz matrix with the $n \times n$ leading principal (that is northwestern) block T.

Theorem 2.2. *Suppose $K = (t_{i,j})_{i,j=0}^{n}$ is a nonsingular $(n + 1) \times (n + 1)$ Toeplitz matrix, write $T = (t_{i,j})_{i,j=0}^{n-1}$, $\widehat{\mathbf{v}} = (v_i)_{i=0}^{n} = K^{-1}\mathbf{e}_1$, $\mathbf{v} = (v_i)_{i=0}^{n-1}$, $\mathbf{v}' = (v_i)_{i=1}^{n}$, $\widehat{\mathbf{w}} = (w_i)_{i=0}^{n} = K^{-1}\mathbf{e}_{n+1}$, $\mathbf{w} = (w_i)_{i=0}^{n-1}$, and $\mathbf{w}' = (w_i)_{i=1}^{n}$, and assume that $v_0 \neq 0$. 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 2.3. *For any fixed positive integer q we can embed a nonsingular $n \times n$ Toeplitz matrix T into an $(n + q) \times (n + q)$ Toeplitz matrix K_q that has the $n \times n$ leading principal block T. Then we can recursively apply Theorem 2.2 to express the inverse T^{-1} via the two column vectors $K_q^{-1}\mathbf{e}_1$ and $K_q^{-1}\mathbf{e}_{n+q}$.*

A more general class of structured matrices having small displacement ranks d extends the classes of Toeplitz and Hankel matrices, for which $d \leq 2$. Such an $m \times n$ matrix can be represented by $(m+n)d$ parameters and can be multiplied and inverted fast provided its displacement rank d is small [4], [17], [22], [23], [28].

2.4 Random sampling and random matrices

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

Definition 2.1. Random sampling *of elements from a set Δ is their selection from this set at random 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 *if the random sampling is Gaussian or uniform, respectively.*

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. *[9], [40], [49]. 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.*

Lemma 2.1 implies that a fixed nonvanishing polynomial vanishes with probablity zero or converging to zero if the values of its variables are sampled under any reasonable (e.g. Gaussian or uniform) probability distribution on the set Δ whose cardinality is infinite or converges to infinity. Under the uniform probability distribution the probability that the polynomial vanishes is readily estimated even where Δ is a fixed finite set.

Corollary 2.1. *Under the assumptions of Lemma 2.1 let the values of the variables of the polynomial be randomly and uniformly sampled from the set Δ. Then the polynomial vanishes with a probability of at most $\frac{d}{|\Delta|}$.*

Corollary 2.2. *Let A be an $m \times n$ matrix with random entries uniformly sampled from a finite set Δ of cardinality $|\Delta|$. Let $l = \min\{m, n\}$ and let M be any fixed $m \times n$ matrix. Then any $k \times k$ submatrix of the matrix $A + M$ for $k \leq l$ is singular with a probability at most $k/|\Delta|$.*

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. □

Definition 2.2. *A matrix (resp. vector) is a* Gaussian random matrix *(resp. vector) with a mean μ and a variance σ^2 if it is filled with independent Gaussian random variables, all having the same mean μ and variance σ^2. Gaussian random matrix or vector is* standard *if $\mu = 0$ and $\sigma^2 = 1$.*

Definition 2.3. *$F_X(y) =$* 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 The extreme singular values of randomized matrix products

A standard Gaussian random matrix A (cf. Definition 2.3) is well conditioned with a high probability [8], [11], [13], [5]. Furthermore adding such a matrix is likely to turn a mildly normalized matrix M into a well conditioned matrix. We specify the respective estimates in Theorem 2.4, taken from [42], cited in our Section 8, and applied in the proof of Theorem 2.5 from [36]. We extensively use the latter theorem. It shows that a square or rectangular Gaussian random multiplier is unlikely to increase the smallest positive singular value of a matrix dramatically, even though the product UV^T of two rectangular unitary matrices U and V can vanish.

Theorem 2.3. *(See [10, Theorem II.7]). Suppose $A \in \mathbb{R}^{n \times n}$ is a Gaussian random matrix with mean zero and a variance σ^2. Then $F_{||A||}(y) \geq 1 - \exp(-x^2/2)$ for $x = y/\sigma - 2\sqrt{n} \geq 0$.*

Theorem 2.4. *(See [42, Theorem 3.3].) Suppose $A \in \mathbb{R}^{m\times n}$ is a Gaussian random matrix independent of the matrix M and having mean zero and a variance σ^2, $M \in \mathbb{R}^{m\times n}$, $W = A + M$, $l = \min\{m, n\}$, and $y \geq 0$. Then $F_W(y) \leq 2.35\ y\sqrt{l}/\sigma$.*

Corollary 2.3. *(See [42, Theorem 3.1]). Under the assumptions of Theorem 2.4, let $||M|| \leq \sqrt{l}$. Then $F_{\kappa(W)}(y) \geq 1 - (14.1 + 4.7\sqrt{(2\ln y)/n})n/(y\sigma)$ for all $y \geq 1$.*

This bound has been improved by a factor $\sqrt{\log n}$ in [47] and in the case where $M = O$ by a factor $y^{|m-n|}\sqrt{\ln y}$ in [13] and [5].

Theorem 2.5. *[35]. Suppose $G \in \mathbb{R}^{r(G)\times m}$, $H \in \mathbb{R}^{n\times r(H)}$, $\text{rank}(G) = r(G)$, $\text{rank}(H) = r(H)$, $y \geq 0$, $M \in \mathbb{R}^{m\times n}$, $W \in \mathcal{G}_{\mu,\sigma}^{m\times n}$, $A = M + W$. Write $l(G) = \min\{r(G), n\}$, $l(H) = \min\{m, r(H)\}$, $c(r) = 2.35$ for $r > 1$ and $c(1) = \sqrt{\frac{2}{\pi}}$. Then we have*

(a) $F_{GA}(y) \leq yc(r(G))\sqrt{l(G)}/(\sigma_{r(G)}(G)\sigma)$ and

(b) $F_{AH}(y) \leq yc(r(H))\sqrt{l(H)}/(\sigma_{r(H)}(H)\sigma)$.

Remark 2.4. *The results of this section can be extended to the case of matrices with complex entries and Toeplitz matrices (see [8], [11], [13], [5] and [35, Sections 3.3 and 3.5]).*

3 Nmbs of a matrix of full row rank via post-multiplication

In this section we compute a nmb of an $m \times n$ matrix A that has full rank m. In the next section we still assume that $m \leq n$ but relax the assumption about the full rank. For $m > n$, we can apply the alternative techniques from our Sections 5 and 6 or from [33], or we can shift our study to the case $m \leq n$ based on the equations $\mathcal{N}(A) = \mathcal{N}(A^HA)$ 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 are $k_i \times n$ matrices for $i = 1, \dots, h$. [19, Theorem 12.4.1] enables us to simplify the computation of a nmb(A) from nmbs of the h matrices $(O \mid B_i \mid O)^T$ for $i = 1, \dots, h$.

Clearly a nonsingular matrix has empty null space. 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 immediately compute a nmb$(A) = \begin{pmatrix} -A_w^{-1}A_e \\ I_{n-m} \end{pmatrix}$. The following algorithm extends this recipe to $m \times n$ matrices A of full rank m, although it does not apply for $m < n$, fails if $\text{rank}\, A < m$, and is prone to numerical problems if the matrix A is ill conditioned.

Algorithm 3.1. A nmb via post-multiplication.

INPUT: *Three positive integers ρ, m, and n, an $m \times n$ matrix A of a rank m, and a random number generator.*

OUTPUT: *FAILURE or a matrix $B = \text{nmb}(A)$.*

COMPUTATIONS:

1. *Generate a nonsingular $n \times n$ matrix $W = (S \mid T)$ where $S \in \mathbb{C}^{n\times m}$.*
2. *Compute the $m \times m$ matrix AS.*
3. *Compute and output the matrix $W \begin{pmatrix} -(AS)^{-1}AT \\ I_{n-m} \end{pmatrix}$, a nmb$(A)$. This computation fails only if the matrix AS is singular. In this case output FAILURE.*

Correctness of the algorithm is readily verified by inspection.

Here are some relevant choices of the matrix S.

Theorem 3.1. *[19]. Let an $m \times n$ matrix A have full rank m and write $S = A^H$. Then the matrix AS is nonsingular and $\kappa(AS) = (\kappa(A))^2$.*

Theorem 3.2. *Assume that $m \le n$, an $m \times n$ matrix A has full rank, and S is an $n \times m$ Toeplitz (resp. general) matrix with $m+n-1$ (resp. mn) random entries uniformly sampled from a finite set $\Delta \in \mathbb{C}$ of cardinality $|\Delta|$. Then the matrix AS is nonsingular (in which case the matrix S has full rank m) with a probability at least $1 - m/|\Delta|$.*

Proof. $\det(AS)$ is a polynomial of a degree at most m in the entries of the matrix S. The polynomial does not vanish identically in these entries because the matrix A has full rank. Now the theorem follows from Corollary 2.1. □

The theorem implies that Algorithm 3.1 is unlikely to fail. In its numerical implementation we would output FAILURE if the matrix AS is ill conditioned.

Next we probabilistically estimate the value $\sigma_m(AS)$ from below for properly scaled random Gaussian matrix S.

Theorem 3.3. *Assume that an $m \times n$ matrix A has full rank (so that $m \le n$), the $m \times m$ matrix AS is nonsingular, and $S \in \mathbb{R}^{n \times m}$ is a Gaussian random matrix with a mean μ and a variance σ^2. Then $F_{AS}(y) \le cy\sqrt{m}/(\sigma_m(A)\sigma)$.*

Proof. The theorem follows from part (a) of Theorem 2.5 for $M = O_{m,n}$, G replaced by A, $W = M + W$ by S, m by n, and r_G and l by m. □

Corollary 3.1. *Under the assumptions of Theorem 3.3 the matrix AS is singular with probability zero. Furthermore $\kappa(AS)$ is expected to have order at most $\kappa(A)\sqrt{m}$ unless $|\mu| \gg \sigma$.*

Remark 3.1. *For $|\mu| \gg \sigma$ the matrix S is likely to lie near a rank-one matrix.*

Remark 3.2. *Our estimates for ranks and condition numbers in this and the next three sections can be readily extended from the respective matrices to all their leading blocks (see Section 6.3).*

4 Scaled randomized western augmentation

We append q scaled random columns on the left of a matrix $A \in \mathbb{C}^{m \times n}$ such that $0 < \operatorname{rank} A \le m \le n$, $m - \operatorname{rank} A \le q < m$ and prove that the resulting matrix $(B \mid A)$ has full rank with probability one or near one. Furthermore if B is a Gaussian random matrix with mean μ where $|\mu|$ has at most order $||A||$ and a variance of order $||A||^2$, then $\kappa(B \mid A)$ is likely to have at most the order $\frac{\sigma_1(A)\sqrt{q}}{\sigma_{m-q}(A)}$. In this case we can effectively compute a nmb by means of aggregation combined with Algorithm 3.1 (cf. Remark 4.2).

4.1 Cas and nmbs via randomized western augmentation: an algorithm

Algorithm 4.1. A ca via randomized western augmentation.

INPUT: *Three positive integers m, n, and q and a matrix $A \in \mathbb{C}^{m \times n}$ such that $n \ge m$ and $q \ge m - \operatorname{rank} A$, a random number generator, and a randomized Subroutine* CA *(e.g., Algorithm 3.1) that either computes a ca of its $k \times l$ input matrix for $k \le l$ or outputs FAILURE. (We assume that the subroutine definitely outputs FAILURE if its input matrix is rank deficient, but only with a low probability otherwise).*

OUTPUT: *FAILURE or the nullity $r = \operatorname{nul} A$ and* $\operatorname{ca}(A)$.

COMPUTATIONS:

1. Western augmentation: *Generate a random $m \times q$ matrix B and apply the Subroutine* CA *to the matrix $(B \mid A)$. If the subroutine fails, conclude that the matrix $(B \mid A)$ is probably rank deficient and output FAILURE.*

2. Aggregation *(cf. Remark 4.2): Otherwise the subroutine computes a matrix* $Z = \begin{pmatrix} Z_0 \\ Z_1 \end{pmatrix} = \mathrm{ca}(B \mid A)$ *where* $Z_0 \in \mathbb{C}^{q\times p}$, $Z_1 \in \mathbb{C}^{n\times p}$, *and* $q \le p \le q + r$. *Compute and output* $r = \mathrm{nul}\, Z_0$.

3. *Then apply Algorithm 4.1 to the matrix* Z_0 *to compute a* $p \times r$ *matrix* $X = \mathrm{ca}(Z_0)$.

4. Disaggregation: *Compute and output the* $n \times r$ *matrix* $Y = Z_1X = \mathrm{ca}(A)$.

Correctness proof. By the definition of the matrices Z and X, we have $BZ_0 + AZ_1 = O$ and $BZ_0X = O$. Therefore $AY = AZ_1X = O$. Conversely, if $A\widehat{\mathbf{y}} = \mathbf{0}$, then $(B \mid A)\begin{pmatrix} \mathbf{0} \\ \widehat{\mathbf{y}} \end{pmatrix} = \mathbf{0}$. It follows that $Z\widehat{\mathbf{x}} = \begin{pmatrix} \mathbf{0} \\ \widehat{\mathbf{y}} \end{pmatrix}$ for some vector $\widehat{\mathbf{x}}$ because $Z = \mathrm{ca}\ (B \mid A)$. Consequently $Z_1\widehat{\mathbf{x}} = \widehat{\mathbf{y}}$.

Remark 4.1. *Having a* $\mathrm{ca}(A)$ *available, we can obtain a* $\mathrm{nmb}(A)$, *e.g., based on Fact 2.2.*

Remark 4.2. *Algorithm 4.1 is an aggregation process that first aggregates an input matrix* A *into the matrix* Z_0 *of a smaller size, then reapplies itself to this matrix to compute the matrix* $X = \mathrm{ca}(Z_0)$, *and finally disaggregates this output to produce the solution* $Y = Z_1X = \mathrm{ca}(A)$ *(cf. [25], [30]). We employ aggregation processes also in Sections 5 and 6.*

4.2 Regularization and preconditioning by means of randomized western augmentation

Theorem 4.1. *Assume that* $A \in \mathbb{C}^{m\times n}$, $m \le n$ *and* $s = m - \mathrm{rank}\, A$. *Then (a) the matrices* $C = (B \mid A)$ *in Algorithm 4.1 are rank deficient for* $q < s$, *whereas (b) for* $q \ge s$ *the matrices* B *and* $(B \mid A)$ *are rank deficient with a probability at most* $\frac{s}{|\Delta|}$ *provided that the entries of the matrix* B *are randomly and uniformly sampled from a finite set* $\Delta \in \mathbb{C}$ *of cardinality* $|\Delta|$ *and that the Subroutine* CA *fails with probability zero if its input matrix has full rank.*

Proof. $\mathrm{rank}(B \mid A) \le \mathrm{rank}\, B + \mathrm{rank}\, A \le q + \mathrm{rank}\, A = q + m - s$. This implies part (a). If $q \ge m - \mathrm{rank}\, A$ and the entries of the matrix B are indeterminates, then clearly the matrices B and $(B \mid A)$ have full rank, and with no loss of generality we can assume that the $m \times m$ western block of the matrix $(B \mid A)$ is nonsingular. Now part (b) of the theorem follows from Corollary 2.1. □

Our western augmentation is also likely to decrease the condition number of a matrix A of rank ρ to the level of $\frac{\sigma_1(A)}{\sigma_{m-q}(A)}$ provided that $s \le q < m$ and B is a Gaussian random matrix with mean zero and a variance of order $||A||^2$. This is implied by the following two theorems.

Theorem 4.2. *Suppose the matrix* $C = (B \mid A)$ *in Algorithm 4.1 has been scaled so that* $||C|| \le 1$. *Let* $\mathrm{rank}\, C = m$, $\mathrm{rank}\, B = q$, *and* $\mathrm{rank}\, A \ge m - q > 0$. *Let* $A = S_A\Sigma_A T_A^H$ *be a full SVD of the matrix* A *and write*

$$S_A^H C\, \mathrm{diag}(I_q, T_A) = (\bar{B} \mid \Sigma_A). \tag{4.1}$$

Delete the last $n - m + q$ *columns of the matrix* $(\bar{B} \mid \Sigma_A)$ *in (4.1) denote by*

$$M = \begin{pmatrix} \bar{B}_0 & \widehat{\Sigma}_A \\ \bar{B}_1 & O \end{pmatrix} \tag{4.2}$$

the resulting nonsingular $m \times m$ *matrix where* $\widehat{\Sigma}_A$ *is the* $(m - q) \times (m - q)$ *leading principlal (northwestern) submatrix of the matrix* Σ_A; *it is nonsingular because* $\mathrm{rank}\, \Sigma_A = \mathrm{rank}\, A \ge m - q > 0$. *Then* $\kappa(C) \le (\frac{1}{\sigma_{m-q}(A)} + \frac{1}{\sigma_q(\bar{B}_1)} + \frac{||\bar{B}_0||}{\sigma_{m-q}(A)\sigma_q(\bar{B}_1)})||C||$.

Proof. Invert equation (4.2) to obtain $M^{-1} = \begin{pmatrix} O & \bar{B}_1^{-1} \\ \widehat{\Sigma}_A^{-1} & -\widehat{\Sigma}_A^{-1}\bar{B}_0\bar{B}_1^{-1} \end{pmatrix}$. Deduce that $||M^{-1}|| \le ||\widehat{\Sigma}_A^{-1}|| + ||\bar{B}_1^{-1}|| + ||\widehat{\Sigma}_A^{-1}||\ ||\bar{B}_1^{-1}||\ ||\bar{B}_0||$. Substitute $||\widehat{\Sigma}_A^{-1}|| = \frac{1}{\sigma_{m-q}(\widehat{\Sigma}_A)} = \frac{1}{\sigma_{m-q}(A)}$, $||\bar{B}_1^{-1}|| = \frac{1}{\sigma_q(B_1)}$, 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 $\frac{1}{\sigma_m(C)} \le \frac{1}{\sigma_{m-q}(A)} + \frac{1}{\sigma_q(\bar{B}_1)} + \frac{||\bar{B}_0||}{\sigma_{m-q}(A)\sigma_q(\bar{B}_1)}$ and multiply both sides by $||C||$. □

Corollary 4.1. *Under the assumptions of Theorem 4.2 suppose* $||B|| = ||A||$ *and write* $\kappa_q(A) = \sigma_1(A)/\sigma_{m-q}(A)$ *and* $\kappa_q(B) = \frac{||\bar{B}_0||}{\sigma_q(\bar{B}_1)}$. *Then* $\kappa(C) \le \sqrt{2}(\kappa_q(A) + \kappa_q(B) + \kappa_q(A)\kappa_q(B))$.

Proof. Note that $||C|| \le \sqrt{||A||^2 + ||B||^2} = \sqrt{2}||A|| = \sqrt{2}||B||$ for $||A|| = ||B||$. Moreover $S_A^H B = \bar{B} = \begin{pmatrix} \bar{B}_0 \\ \bar{B}_1 \end{pmatrix}$, and so $||B|| = ||\bar{B}|| \ge ||\bar{B}_0||$. Substitute these relationships into Theorem 4.2. □

Next assume that $B \in \mathbb{R}^{m\times q}$ is Gaussian random matrix and estimate the value $\sigma_q(\bar{B}_1)$.

Theorem 4.3. *Under the assumptions of Theorem 4.2 suppose* $B \in \mathbb{R}^{m\times q}$ *is a Gaussian random matrix with a mean* μ *and a variance* σ^2. *Then* $F_{\bar{B}_1}(y) \le cy\sqrt{q}/\sigma$.

Proof. Apply part (a) of Theorem 2.5 for $M = O_{m,q}$, $G = (O \mid I_q)S_A^H$, $W = M + W = B$, $r_G = l = q \le m$, $GW = \bar{B}_1$, and $\sigma_{r_G}(G) = 1$. □

Corollary 4.1 and Theorem 4.3 together imply a randomized upper bound of order $\frac{||A||\sqrt{q}}{\sigma_{m-q}(A)}$ on the condition number $\kappa(C)$ provided that σ has order $||A||$ and $|\mu|$ has at most order $||A||$ in Theorem 4.3 (without such restrictions we could have expected to have $C \approx (O \mid A)$ or $C \approx (B \mid O)$ for some pairs of $|\mu|$ and σ).

Corollary 4.2. *Under the assumptions of Theorem 4.3 the matrix* $C = (B \mid A)$ *is rank deficient with probability zero.*

Proof. Theorem 4.3 implies that the matrix B_1 is singular with probability zero. Therefore the corollary follows from equations (4.1) and (4.2). □

4.3 Further remarks

Remark 4.3. *Our correctness proof for Algorithm 4.1 applies to any integer* $q \ge s = m - \operatorname{rank} A$. *The following observations can guide us in choosing the integer parameter* q.

1. *By virtue of Theorem 4.1* $\operatorname{rank}(B \mid A) < m$ *if* $q < s$, *but we expect to have* $\operatorname{rank}(B \mid A) = m$ *if* B *is a random* $m \times q$ *matrix and if* $q \ge s$.

2. *If the matrix* A *has numerical rank* $\bar{\rho}$, *then for* $q < m - \bar{\rho}$ *the matrix* C *is ill conditioned but is expected to be well conditioned provided* $q \ge m - \bar{\rho}$, σ *has order* $||A||$ *and* $|\mu|$ *has at most order* $||A||$ *in Theorem 4.3. These observations suggest the choice of* $q = m - \bar{\rho}$ *and can be the basis for the search of numerical rank. The search is simplified as the ratio* $\frac{\sigma_\rho(A)}{\sigma_{\rho+1}(A)}$ *increases. We can apply the power transforms* $A \Longrightarrow B = (AA^T)^h A$ *for positive integers* h. *They increase the ratios of the consecutive singular values of* A *because* $\sigma_j(B) = (\sigma_j(A))^{2h+1}$.

3. *One could extend Algorithm 4.1 to computing approximate cas and numerical nullity instead of cas and nullity. Instead of the Subroutine CA employ a randomized Subroutine* APPROXIMATE CA *(e.g., numerical version of Algorithm 5.1 or 3.1) that either computes an approximate ca 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.*

Remark 4.4. *If the matrix* A *is ill conditioned, then in numerical implementation of Algorithm 4.1 one must compute a* $\operatorname{ca}(B \mid A)$ *with high accuracy [30, Section 7]; by applying iterative refinement one would still decrease the overall computational cost wherever* $q \ll \min\{m, n\}$ *provided the matrix* $(B \mid A)$ *is well conditioned (see our Section 8 as well as [30, Sections 8 and 9] and [36]).*

Remark 4.5. *Assume that Algorithm 4.1 produces a matrix* $(B \mid A)$ *at its stage of western augmentation and that Algorithm 3.1 has been applied as the Subroutine* CA. *Then we can save some flops in Algorithm 4.1 by choosing* $W = \operatorname{diag}(I_q, T)$ *where either* $T = A^H$ *or* T *is a random* $n \times m$ *matrix scaled so that* $||T|| \approx 1$. *One can readily modify Theorems 3.1–3.3 to cover this case.*

5 Randomized northern augmentation

Our next randomized nmb algorithm involves computation of a left inverse, rather than just the inverse as in Algorithm 3.1, but removes the restrictions of that algorithm on the input: now we compute a nmb allowing rank deficient and ill conditioned $m \times n$ matrices for any pair of positive integers m and n. Furthermore the algorithm can be readily extended to computing an approximate nmb; this extension is applied to matrix factorization in Section 7.2.

We first append new scaled Gaussian random rows at the top of an $m \times n$ matrix A of rank $\rho < n$ such that the matrix $C = \begin{pmatrix} V \\ A \end{pmatrix}$ is expected to have full column rank (cf. Theorem 5.1). Then the algorithm computes the first $r = n - \rho$ columns of the left inverse $C^{(I)}$ and outputs them as a $\mathrm{nmb}(A)$. The matrix C is expected to have condition number of order $\kappa(A)$, and so our randomized algorithm is expected to create no new numerical problems.

We refer the reader to Section 5.4 on our treatment of ill conditioned inputs.

5.1 A randomized nmb algorithm

Algorithm 5.1. A nmb via randomized northern augmentation.

INPUT: *Three positive integers ρ, m, and n, an $m \times n$ matrix A of a rank ρ, and a random number generator.*

OUTPUT: *FAILURE or a matrix $B = \mathrm{nmb}(A)$.*

COMPUTATIONS:

1. *Write $r = n - \rho$. Output that $\mathrm{nmb}(A)$ is empty if $r = 0$. Otherwise generate a $r \times n$ random matrix V.*
2. *Output FAILURE if the matrix $C = \begin{pmatrix} V \\ A \end{pmatrix}$ is column rank deficient.*
3. *Otherwise compute and output the matrix $B = C^{(I)} \begin{pmatrix} I_r \\ O \end{pmatrix}$.*

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

$$\mathcal{N}(A) = \mathcal{R}(Y) \subseteq \mathcal{R}(B). \tag{5.1}$$

It follows that $\mathcal{R}(B) = \mathcal{N}(A)$ because $\dim(\mathcal{R}(B)) = \operatorname{rank} B = r = \dim(\mathcal{N}(A))$.

5.2 Impact on rank and conditioning

Our next theorem and Corollary 5.2 show that our randomized augmentation is likely to fix the input degeneracy, and so Algorithm 5.1 is unlikely to fail.

Theorem 5.1. *Suppose in Algorithm 5.1 we generate the matrices U and V by uniformly sampling their entries from a set $\Delta \in \mathbb{C}$ of a cardinality $|\Delta|$. Then the matrix C has full column rank n with a probability at least $1 - r/|\Delta|$.*

Proof. Let a $\rho \times n$ submatrix $A_{\rho,n}$ of the matrix A have full rank ρ and let $C_{n,n} = \begin{pmatrix} V \\ A_{\rho,n} \end{pmatrix}$. 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. In virtue of Corollary 2.1 $\det C_{n,n}$ vanishes with a probability at most $r/|\Delta|$ in the case of random matrix V. □

In the rest of this subsection we prove that the condition number $\kappa(C)$ tends to be of order $\frac{\sigma_1(A)}{\sigma_\rho(A)} = \kappa(A)$ where V is Gaussian random matrix with a mean μ and a variance σ^2 such that σ has order $||A||$, whereas $|\mu|$ has at most order $||A||$, e.g., where the matrix A is mildly normalized, whereas V is standard Gaussian random matrix.

Theorem 5.2. *Suppose that $A \in \mathbb{C}^{m\times n}$, $V \in \mathbb{C}^{r\times n}$, $C = \begin{pmatrix} V \\ A \end{pmatrix}$, $\operatorname{rank} C = n$, $\operatorname{rank} A = n - r$, and $\operatorname{rank} V = r$. Let $A = S_A \Sigma_A T_A^H$ be full SVD of the matrix A (where $\Sigma_A = \operatorname{diag}(\widehat{\Sigma}_A, O)$ and $\widehat{\Sigma}_A$ is a $\rho \times \rho$ diagonal matrix of the singular values). Write*

$$\operatorname{diag}(I_r, S_A^H) C T_A = \begin{pmatrix} M \\ O \end{pmatrix}, \ M = \begin{pmatrix} V_0 & V_1 \\ \widehat{\Sigma}_A & O \end{pmatrix}. \tag{5.2}$$

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. $||M^{-1}|| \le ||\widehat{\Sigma}_A^{-1}|| + ||V_1^{-1}|| + ||\widehat{\Sigma}_A^{-1}||\ ||V_1^{-1}||\ ||V_0||$ because $M^{-1} = \begin{pmatrix} O & \widehat{\Sigma}_A^{-1} \\ V_1^{-1} & -V_1^{-1} V_0 \widehat{\Sigma}_A^{-1} \end{pmatrix}$. 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 5.1. *Under the assumptions of Theorem 5.2 suppose the matrices A and V have been scaled so that $||A|| = ||V||$ and write $\kappa = \kappa(A)$ and $\kappa_1 = \frac{||V||}{\sigma_r(V_1)}$. Then $\kappa(C) \le \sqrt{2}(\kappa + \kappa_1 + \kappa\kappa_1)$.*

Proof. We have $||C|| \le \sqrt{||A||^2 + ||V||^2} = \sqrt{2}||A|| = \sqrt{2}||V||$ because $||A|| = ||V||$. Moreover $||V_0|| \le ||VT_A|| = ||V||$ for $(V_0 \mid V_1) = VT_A$. Substitute these bounds into Theorem 5.2. □

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

Theorem 5.3. *Under the assumptions of Theorem 5.2 suppose $V \in \mathbb{R}^{r\times n}$ is a Gaussian random matrix with a mean μ and variance σ^2, $c(1) = 1$ and $c(r) = 2.35$ for $r > 1$ as in Theorem 2.5. Then $F_{V_1}(y) \le c(r) y\sqrt{r}/\sigma$.*

Proof. Apply part (b) of Theorem 2.5 for $M = O_{m,n}$, $W = W + M = V$, $H = T_A \begin{pmatrix} O \\ I_r \end{pmatrix}$, $l(H) = r_H = r$, $V_1 = WH$, and $\sigma_{r_H}(H) = 1$. □

Corollary 5.1 and Theorem 5.3 together imply a randomized upper bound of order $\kappa(A)\sqrt{r}$ on the condition number $\kappa(C)$ provided that we have chosen σ of order $||A||$ and $|\mu|$ at most of order $||A||$ in Theorem 5.3 (otherwise for some pairs of μ and σ we could have expected to have $C^T \approx (O \mid A^T)^T$, $C^T \approx (V^T \mid O)^T$), or the matrix V lying close to a rank-one matrix).

Corollary 5.2. *Under the assumptions of Theorem 5.3 the matrix C is column rank deficient with probability zero.*

Proof. Theorem 5.3 implies that the matrix V_1 is singular with probability zero. Therefore the corollary follows from equation (5.2). □

Corollary 5.3. *Under the assumptions of Theorem 5.3 the condition number $\kappa(C)$ is expected to have order $\kappa(A)$ provided σ has order $||A||$, whereas $|\mu|$ has at most order $||A||$.*

5.3 The choice of the augmentation size via binary search or aggregation

In Algorithm 5.1 we assume that $\operatorname{rank} A$ is a part of the input. Otherwise we can compute it assuming that V is a $k \times n$ random matrix, $C = \begin{pmatrix} V \\ A \end{pmatrix}$, and $\rho = n - r = \operatorname{rank} A$ and based on the following properties.

(a) $k \ge r$ if the matrix C has full rank.

(b) If $k \geq r$, then the matrix C has full rank with probability one.
(c) Suppose the matrix C has full rank. Then $k = r$ if $AB = O$.
Here is an alternative to employing property (c).

Algorithm 5.2. *Assume matrices A of size $m \times n$ and V of size $k \times n$. Suppose the matrix $C = C^{(k)} = \begin{pmatrix} V \\ A \end{pmatrix}$ has full rank n.*

Compute the matrices $B^{(k)} = C^{(I)} \begin{pmatrix} I_k \\ O \end{pmatrix}$ and $AB^{(k)}$.

If $AB^{(k)} = O$, then output $B^{(k)}$, being a nmb(A).
Otherwise compute the matrices

$$X = \mathrm{ca}(AB^{(k)}), \tag{5.3}$$

ca$(A) = B^{(k)}X$ *and then a* nmb(A), *e.g., based on Fact 2.2.*

One can restrict this algorithm to computing just the nullity $\mathrm{nul}\, A = \mathrm{nul}\, B^{(k)}$ and then obtain a nmb(A) by applying Algorithm 5.1.

Theorem 5.4. *The matrix $B^{(k)}X$ computed in Algorithm 5.2 is a* ca(A).

Proof. Equation (5.3) implies that $AB^{(k)}X = O$, that is, $\mathcal{N}(A) \supseteq \mathcal{R}(B^{(k)}X)$. Let us prove that $\mathcal{N}(A) \subseteq \mathcal{R}(B^{(k)}X)$, that is, $\mathbf{y} = B^{(k)}X\mathbf{w}$ for some vector $\mathbf{w}$ as soon as $A\mathbf{y} = \mathbf{0}$. (5.1) holds for $B = B^{(k)}$ since $\mathrm{rank}\, C = n$. This implies that $\mathbf{y} = B^{(k)}\mathbf{z}$ for some vector $\mathbf{z}$. It remains to prove that $\mathbf{z} = X\mathbf{w}$ for some vector $\mathbf{w}$. This equation follows because $A\mathbf{y} = AB^{(k)}\mathbf{z} = \mathbf{0}$ and because $X = \mathrm{ca}(AB^{(k)})$ by assumption. □

Search for a ca$(AB^{(k)})$ is simpler than for a ca(A) because the input size decreases from $m \times n$ to $m \times k$. This is another example of an aggregation process (cf. Section 4): we first aggregate an input matrix A into the matrix $AB^{(k)}$ of a smaller size, then compute a ca for such a matrix defined by (5.3), and finally disaggregate this ca into $B^{(k)}X$, which is a ca for the matrix A.

5.4 Computing approximate nmbs

In numerical implementation of Algorithm 5.1, the input set consists of three integers m, n, and ρ, an $m \times n$ matrix A that has a numerical rank ρ, and a random number generator that generates a Gaussian random matrix V with mean zero and a variance of order $||A||^2$, e.g., a standard Gaussain random matrix V where the matrix A is mildly normalized. Then we can compute an approximate nmb of A by modifying Stages 2 and 3 of the algorithm as follows:

2. Output FAILURE if the matrix C is rank deficient or ill conditioned.
3. Otherwise compute and output the matrix $B = C^{+} \begin{pmatrix} I_r \\ O \end{pmatrix}$.

Hereafter we refer to this algorithm as **Algorithm 5.3**.

Suppose $\mathrm{rank}\, \tilde{A} = \rho$ and $\tilde{A} \approx A$, so that $C = \begin{pmatrix} V \\ A \end{pmatrix} \approx \tilde{C} = \begin{pmatrix} V \\ \tilde{A} \end{pmatrix}$, $B = C^{(I)} \begin{pmatrix} I_r \\ O \end{pmatrix} \approx \tilde{B} = \tilde{C}^{(I)} \begin{pmatrix} I_r \\ O \end{pmatrix}$, $\tilde{B} - B = (\tilde{C}^{(I)} - C^{(I)}) \begin{pmatrix} I_r \\ 0 \end{pmatrix}$, and therefore $||\tilde{B} - B|| \leq ||\tilde{C}^{(I)} - C^{(I)}||$.

Set $C^{(I)} = C^{+}$, $\tilde{C}^{(I)} = \tilde{C}^{+}$ and obtain $||\tilde{B} - B|| \leq ||\tilde{C}^{+} - C^{+}|| \leq 2||\tilde{C} - C||_F \max\{||C^{+}||^2, ||\tilde{C}^{+}||^2\}$ (see [19, Section 5.5.5]) and consequently $\kappa(C) \approx \kappa(\tilde{C})$. Furthermore these two condition numbers are likely to have order $\kappa(\tilde{A}) = \frac{\sigma_1(\tilde{A})}{\sigma_\rho(\tilde{A})} \approx \kappa(A) = \frac{\sigma_1(A)}{\sigma_\rho(A)}$ provided V is a Gaussian random matrix with mean zero and a variance of order $||A||^2$ (cf. Section 5.2). In this case our bound on the relative error norm $\frac{||\tilde{B} - B||}{||\tilde{B}||}$ depends on the condition number $\kappa(A)$.

By extending our recipes from the previous subsection we can extend Algorithm 5.3 to the case where the input matrix A has an unknown positive numerical nullity.

6 Randomized northwestern augmentation

Given two positive integers m and n and a matrix $A \in \mathbb{C}^{m\times n}$, one can compute a $\mathrm{ca}(A)$ by applying Algorithm 4.1 where Algorithm 5.1 replaces the Subroutine *CA*. In this section we specify the resulting northwestern augmentation $A \to K = \begin{pmatrix} W & V \\ B & A \end{pmatrix}$, analyze it by combining the analysis in Sections 4 and 5, and supply some additional comments. In Section 7.3 we apply northwestern augmentation to precondition a nonsingular nonhomogeneous linear system of equations.

6.1 Cas and nmbs via randomized northwestern augmentation: an algorithm

Algorithm 6.1. A ca via randomized northwestern augmentation.

INPUT: *Three positive integers m, n, and ρ, a matrix $A \in \mathbb{C}^{m\times n}$ of rank ρ, and a random number generator.*

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

INITIALIZATION: *Fix two nonnegative integers $q \geq n - \rho$ and $r \geq n + q - m$.*

COMPUTATIONS:

1. *(*Northwestern augmentation.*) Generate three random matrices V in $\mathbb{C}^{r\times n}$, B in $\mathbb{C}^{m\times q}$, and W in $\mathbb{C}^{r\times q}$. If the matrix $K = \begin{pmatrix} W & V \\ B & A \end{pmatrix} \in \mathbb{C}^{(m+r)\times(n+q)}$ is column rank deficient, output FAILURE.*
2. *(*Aggregation.*) Otherwise compute the matrices*

$$Y = (O \mid I_n)K^{(I)} \begin{pmatrix} O \\ B \end{pmatrix} \tag{6.1}$$

and AY. IF $AY = O$, output $Y = \mathrm{ca}(A)$.

3. *Otherwise apply Algorithm 6.1 to the matrix AY to compute a $q \times s$ matrix $Z = \mathrm{ca}(AY)$.*
4. *(*Disaggregation.*) If the matrix Z has rank q, then compute and output the $n \times r$ matrix $YZ = \mathrm{ca}(A)$. Otherwise output FAILURE.*

We can unify Stages 3–5 because $Y = I$ and Z is a $\mathrm{ca}(A)$ if $AY = O$.
Our remarks about Algorithms 5.1 and 4.1 can be readily extended to Algorithm 6.1.

6.2 Analysis of randomized northwestern augmentation

Let us verify correctness of Algorithm 6.1 by combining our analysis of northern and western augmentation.

Theorem 6.1. (a) *Assume six positive integers m, n, q, r, s, and ρ such that $\rho \leq \min\{m, n\}$ and $s = \min\{m + r, n + q, \rho + q + r\}$, and five matrices $A \in \mathbb{C}^{m\times n}$ of rank ρ, V in $\mathbb{C}^{r\times n}$, B in $\mathbb{C}^{m\times q}$, W in $\mathbb{C}^{r\times q}$, and $K = \begin{pmatrix} W & V \\ B & A \end{pmatrix} \in \mathbb{C}^{(m+r)\times(n+q)}$. Then we have* $\mathrm{rank}\, K \leq s$.

(b) *Suppose under the assumptions of part (a) that either the entries of the matrices B, V, and W have been randomly and uniformly sampled from a set $\Delta \in \mathbb{C}$ of cardinality $|\Delta|$ or the entries of the matrices V and W have been randomly and uniformly sampled from such a set and $B = V$. Furthermore let $m \leq \min\{n, \rho + q\}$. Then the matrix $(B \mid A)$ has full rank m with a probability at least $1 - \frac{q}{|\Delta|}$. If in addition $r \geq n + q - m$, then* $\mathrm{rank}\, K = n + q = s$ *with a probability at least $1 - \frac{q+r}{|\Delta|}$.*

Proof. Part (a) of the theorem can be immediately verified. Part (b) is proved similarly to Theorems 5.1 and 4.1, based on Corollary 2.1. □

Theorem 6.2. *Under the assumptions of part (a) of Theorem 6.1, suppose that*

$$n+q \le m+r,\ q \le r,\ K^{(I)}K = I_{n+q} \text{ and } W^{(I)}W = I_q \tag{6.2}$$

for some matrices $K^{(I)}$ *and* $W^{(I)}$*. Then*

$$\mathcal{N}(A) \subseteq \mathcal{R}(Y) \text{ for } Y = (O \mid I_n)K^{(I)}\begin{pmatrix} O \\ B \end{pmatrix}. \tag{6.3}$$

Furthermore if $\operatorname{rank} B \le \operatorname{nul} A$*, then*

$$\mathcal{R}(Y) = \mathcal{N}(A). \tag{6.4}$$

Proof. Let $\mathbf{y} \in \mathcal{N}(A)$ and $\mathbf{x} \in \mathbb{C}^q$. Then $K\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} W\mathbf{x}+V\mathbf{y} \\ B\mathbf{x} \end{pmatrix}$. Substitute $\mathbf{x} = -W^{(I)}V\mathbf{y}$ and obtain that $K\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ B\mathbf{x} \end{pmatrix}$. Therefore $\mathbf{y} = (O \mid I_n)K^{(I)}\begin{pmatrix} \mathbf{0} \\ B\mathbf{x} \end{pmatrix}$. This proves claim (6.3), which implies claim (6.4) if $\operatorname{rank} B \le \operatorname{nul} A$ because $\operatorname{rank} Y \le \operatorname{rank} B$. □

Theorem 6.3. *Under the assumptions of part (a) of Theorem 6.1, suppose equations (6.2) hold and write* $Y = (O \mid I_n)K^{(I)}\begin{pmatrix} O \\ B \end{pmatrix}$*. Then*
(a) YZ *is a* $\operatorname{ca}(A)$ *if* Z *is a* $\operatorname{ca}(AY)$*, in particular if* $AY = O$ *and* $Z = I$*, 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 $A(YZ) = (AY)Z = O$ 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 (6.3). 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 $(AY)Z = A(YZ) = O$ 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 $\operatorname{rank} Y = q$. □

Finally note that Corollaries 4.1 and 5.1 and Theorems 4.3 and 5.3 together imply a randomized upper bound of order $\sqrt{rq}\frac{||A||}{\sigma_{m-q}(A)}$ on the condition number $\kappa(K)$ provided that σ has been chosen of order $||A||$ and $|\mu|$ at most of order $||A||$ in Theorems 4.3 and 5.3.

6.3 Strong regularization and strong preconditioning

Our results on the regularization and preconditioning power of northern, western, and northwestern augmentations and post-multiplication of the input matrix can be immediately extended to all its $k \times k$ leading principal (that is northwestern) submatrices for $k = 1, 2, \ldots$; 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 k we can pre-multiply the matrix K by $\operatorname{diag}(I_{m+r-k}, O_{k,k})$ and post-multiply the product by $\operatorname{diag}(I_{n+q-k}, O_{k,k})$.

We refer the reader to [35] on some algorithmic applications of these properties.

6.4 Saving random parameters, a pitfall and a remedy

By using structured, e.g., Toeplitz augmentations we can save random parameters (see [36, Sections 8.2, 8.3]), but saving by means of symmetrization leads to a pitfalls where $K = \begin{pmatrix} W & V^H \\ V & A \end{pmatrix}$ is a Hermitian positive definite matrix; in this case $\kappa(K) \ge \kappa(A)$ because of the Interlacing Property of the eigenvalues of Hermitian matrices [19, Theorem 8.1.7]. In contrast scaled randomized Hermitian additive preprocessing $A \Longrightarrow C = A + VV^H$ is expected to work as preconditioning for an ill conditioned matrix A having small numericall nullity [48].

7 Solving 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, or 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 $(-\eta\mathbf{b} \mid A)$ for a nonzero scalar η or the matrix $(I_n - \frac{\mathbf{b}\mathbf{b}^H}{\mathbf{b}^H\mathbf{b}})A$. The former approach seems to be more effective; we analyze it in Section 7.1. In Sections 7.2 and 7.3 we extend our earlier study to precondition a linear system $A\mathbf{y} = \mathbf{b}$ provided a matrix A has a positive numerical nullity: we employ the computation of an approximate $\mathrm{nmb}(A)$ in Section 7.2 and randomized scaled northwestern augmentation in Section 7.3.

7.1 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 a nonzero scalar η contains the vector $\mathbf{y}$ as a subvector. To compute the null vector we can apply the algorithms in the previous sections.

One can scale the matrix A to ensure that $||A|| = 1$ and can select the scalar η such that $||\eta\mathbf{b}|| = ||A||$. So we assume that $||\mathbf{b}|| = ||A|| = 1$ and then show that 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 one.

Theorem 7.1. *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

$$\sigma_n(C) \geq \sigma_n(G) - \sigma_n. \tag{7.1}$$

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

$$g_i = \sigma_{i-1} \text{ and } \widehat{f}_i = f_{i-1} \text{ for } i = 2, \dots, n;\ g_1 = f_n\ , \widehat{f}_1 = 0, \text{ and } \widehat{\mathbf{f}} = (\widehat{f}_i)_{i=1}^n \tag{7.2}$$

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 (7.2) 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. (7.2)), 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 7.1. *Under the assumptions of Theorem 7.1 we have $\kappa(C) \leq \frac{(1+|f_n|)\sqrt{2}}{|f_n|\sigma_{n-1} - (1+|f_n|)\sigma_n}$.*

Proof. Recall that $\kappa(C) = \frac{||C||}{\sigma_n(C)}$ and $||C|| \le \sqrt{||A|| + ||\mathbf{b}||} = \sqrt{2}$. Substitute these relationships into Theorem 7.1. □

Suppose the value $|f_n|$ is small. Then under the assumptions of the corollary, $\kappa(C)$ has at most the order σ_1/σ_{n-1} (compared to $\kappa(A) = \sigma_1/\sigma_n$) and therefore C is a well conditioned matrix provided the input matrix A has numerical nullity one. Note that on the average $|f_n| = \frac{1}{\sqrt{n}}$ on the unit sphere $||\mathbf{f}|| = 1$. Corollary 7.1 would turn into Theorem 4.2 for $q = 1$ if $\mathbf{b}$ were a scaled Gaussian random vector.

We still need to perform some stages of our solution algorithm with a high precision, but we decrease the overall computational cost by confining the high precision computations to iterative refinement of a null vector of the well conditioned matrix C, which makes up about $1/n$ fraction of the solution time of the system $A\mathbf{y} = \mathbf{b}$ by means of standard algorithms such as Gaussian elimination (cf. Section 8). At that stage we can apply fast advanced algorithms for accurate sums and products [30], [36].

7.2 Solution with auxiliary matrices defined by approximate nmbs

Suppose a nonsingular matrix $A \in \mathbb{R}^{n\times n}$ with $||A|| \approx 1$ has small positive numerical nullity r and suppose approximate nmbs M_1 and N_1 in $\mathbb{R}^{n\times r}$ of the matrices A^H and A, respectively, are available, possibly computed by means of a numerical version of an algorithm in the previous sections.

Then we can generate two standard Gaussian random matrices S and T in $\mathbb{R}^{n\times(n-r)}$ and compute the matrices $M_0 = A^H S$, $N_0 = AT$, and $F = (M_0 \mid M_1)^H A(N_0 \mid N_1) = \begin{pmatrix} F_{00} & F_{01} \\ F_{10} & F_{11} \end{pmatrix}$ where $F_{ij} = M_i^H A N_j$ 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 $\sigma_{n-r}(A)$ in virtue of Theorem 2.5, and then the block F_{00} in the 2×2 block matrix F is nonsingular and well conditioned because the matrix A has numerical nullity r by assumption. Furthermore this block is expected to be dominant because the matrices $A^H M_1$, AN_1, and consequently their 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 in [36] have confirmed its strong preconditioning power.

7.3 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 $K = \begin{pmatrix} W & V \\ B & A \end{pmatrix}$ for Gaussian random matrices $W \in \mathbb{C}^{r\times r}$, V, and B with mean zero and a variance of order $||A||^2$. Then according to Section 6.2, the matrix K and its block W are expected to be nonsingular and well conditioned. Our algorithms employ the following theorem.

Theorem 7.2. *Let $K = \begin{pmatrix} W & V \\ B & A \end{pmatrix}$ where A, W and K are nonsingular matrices of sizes $n\times n$, $r\times r$, and $(n + r) \times (n + r)$, respectively, for $0 < r < n$. Write $R = I + VBW^{-1}$ and $S = A - BW^{-1}V$; 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) $A^{-1} = S^{-1} - S^{-1}BW^{-1}R^{-1}VS^{-1}$.

Part (a) is well known and is readily verified. Part (b) follows from the Sherman–Morrison–-Woodbury formula [19, page 50] applied to the matrix $A = S + BW^{-1}V$.

The theorem implies that $\mathbf{y} = A^{-1}\mathbf{b} = (S^{-1} - S^{-1}BW^{-1}R^{-1}VS^{-1})\mathbf{b}$ for $R = I + VBW^{-1}$ and $S^{-1} = (O_{n,r} \mid I_n)K^{-1}\begin{pmatrix} O_{r,n} \\ I_n \end{pmatrix}$ 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 inversion of the matrices W and R and the computation of the products $S^{-1}\mathbf{b}$ and $S^{-1}B$. Furthermore the matrix equation $K^{-1}K = I_{n+r}$ implies that $ZW + S^{-1}B = O_{n,r}$ and consequently $ZW = -S^{-1}B$. Therefore we can compute the product $ZW = (O_{n,r} \mid I_n)K^{-1}\begin{pmatrix} W \\ O_{r,n} \end{pmatrix}$ instead of the product $-S^{-1}B$. By scaling the matrices B, V and W we can ensure that $R \approx I$.

If the matrices R, W and K are well conditioned, as can be expected, then we can decrease the overall cost of computing the solution (see Section 8).

If the matrix A is given with its displacement generator of a small length d, we are motivated to choose scaled random matrices W, B, and V with consistent structure, representing them as well as the matrix K with displacement generators of length in $O(d)$. By employing this structure we can accelerate our computations with these matrices (cf. [28]). In the special case of a Toeplitz matrix A we can choose augmentation that produces a Toeplitz matrix K, and then we can exploit this structure based either on Theorem 7.2 or alternatively on Theorem 2.2 and Remark 2.3. Such a randomized structured augmentation techniques is still likely to achieve regularization and preconditioning [35, Section 3.3].

8 Computational cost estimates

Assume that we seek the solution $\mathbf{y}$ with output precision p_{out} to a nonsingular linear system $A\mathbf{y} = \mathbf{b}$ of n equations whose matrix A has numerical nullity one. Gaussian elimination uses $\frac{2}{3} + O(n^2)$ flops in high precision $p_+ \approx p_{\text{out}} + \log_2 \kappa(A)$. By applying the algorithms of the previous section we reduce our task to the solution of a nonsingular and well conditioned linear system $F\mathbf{x} = \mathbf{c}$ of n or $n+1$ equations. We still need to solve it with the high precision p_+ but can apply iterative refinement in a fixed lower precision p (e.g., in the standard IEEE single or double precision) such that

$$2\log_2 \kappa(F) \le p \ll p_+ \approx p_{\text{out}} + \log_2 \kappa(A). \tag{8.1}$$

These bounds can be satisfied because the matrix F is well conditioned but A is ill conditioned.

Here is a flowchart of our computations.

Flowchart 8.1. Solution of a linear system with iterative refeinement.

COMPUTATIONS:

1. *Fix a precision p satisfying (8.1).*
2. *Apply $O(n^3)$ flops of Gaussian elimination in the low precision p to compute an approximate inverse $X \approx F^{-1}$ and an initial approximate solution $X\mathbf{c}$ to the linear system $F\mathbf{x} = \mathbf{c}$ (cf. [30], [36]).*
3. *Iteratively refine this solution. Every loop of iterative refinement amounts essentially to multiplying each of the matrices F and $X \approx F^{-1}$ by a vector (which takes $4n^2 + O(n)$ flops in the precision p) and contributes about $b = p - \log_2 \kappa(F)$ correct bits per an output entry; we can assume that $b \ge p/2$ under (8.1) (cf. [19], [21], [30], [41]).*

Now let $\mu(l)$ denote the time-complexity of a flop performed in a precision l. In terms of the number of bitwise operations involved, $\mu(l)$ ranges between orders $(l \log l) \log\log l$ and l^2, depending on the magnitude of l and computer environement [2], [14]. Our computations in the flowchart involve

$$b_f = O(n^2(n + p_+/b)\mu(p))$$

bitwise operations versus order of $b_g = n^3\mu(p_+)$ in Gaussian elimimnation. Clearly

$$b_f/b_g = O((1 + p_+/(nb))\mu(p)/\mu(p_+)); \tag{8.2}$$

the ratio b_f/b_g is small under the bounds $p_+ = o(nb)$ and (8.1).

If the matrix A has numerical nullity $r > 1$, then the algorithm in Section 7.2 reduces the original linear system $A\mathbf{y} = \mathbf{b}$ to $2r$ well conditioned linear systems, each of $n - r$ equations, and the cost estimates increase by a factor r versus the case of $r = 1$. The estimates are still quite favorable to Flowchart 8.1 versus Gaussian elimimnation as long as the ratio r/n is small, which is an important and quite general case in view of Remark 2.1. The cost of performing the algorithm of Section 7.3 is estimated similarly, and we still yield the ratio (8.2) within a factor r.

In the case of nonsingular $n \times n$ input matrices A having Toeplitz or Toeplitz-like structure and having a displacement rank d and numerical nullity one, the customary GKO type algorithms use order dn^2 flops to solve a linear system $A\mathbf{y} = \mathbf{b}$ or $F\mathbf{x} = \mathbf{c}$ [16], [27], [29], [39], whereas one can multiply the matrices F by a vector by using $O(dn \log n)$ flops. This leads to bound (8.2) up to a factor $\log n$. For matrices A having numerical nullity r we increase the cost estimate by a factor $\min\{r, d\}$ (rather than r) because the inverse A^{-1} is expressed via the solution of $2d$ linear systems with the matrices A and A^T.

Let us compare our estimates with an information lower bound. We must process n^2 input entries (or $2dn$ input parameters in case of Toeplitz-like inputs defined by displacement generators of length d [17], [28]) by using the precision of about p_+ bits to obtain the output with the precision p_{out}. b bitwise operations involve at most $2b$ input bits, and so we arrive at the information lower bounds of $0.5n^2p_+$ bitwise operations or dnp_+ in the case of Toeplitz-like inputs. In both cases of general and Toeplitz-like inputs our algorithms reach these bounds within polylogarithmic factors in n and p_+ if we assume (8.1) and $\mu(l)$ of order $(l \log l) \log\log l$.

Remark 8.1. *In lieu of iterative refinement one can employ other iterations such as the CG (Conjugate Gradient) and GMRES algorithms (cf. [1], [15], [19, Sections 10.2–10.4]). Like iterative refinement, they perform $O(M_A)$ flops per iteration loop, but unlike iterative refinement, they use no approximate inverse and thus save flops required for its computation. This is a significant advantage in the case of a sparse unstructured linear system as well as a multilevel Toeplitz or Hankel linear system [12], [26]. Decreasing the condition number of an input matrix is more critical (and thus preconditioning is more important) for the convergence of such algorithms versus iterative refinement, however. In particular similarly to iterative refinement, every nonsymmetric CG iteration loop amounts essentially to multiplication of an input matrix M and its transpose M^T by two pairs of vectors but only the decrease of the error norm by a factor $4(\frac{\sqrt{\kappa(M)}+1}{\sqrt{\kappa(M)}-1})^2$ [19, Theorem 10.2.6] or equivalently ensures $1/\kappa(M)$ 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.*

9 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. Random numbers were generated 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.

9.1 Computations with Toeplitz matrices

a) Generation of singular Toeplitz matrices

To generate an $n \times n$ singular Toeplitz matrix, we first sampled $2n - 2$ random entries $a_{i,j}$ for $j = 1$, $i = 1, \dots, n - 1$ and for $i = 1$, $j = 2, \dots, n$ in the range $[-1, 1)$, then 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 virtue 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 it been singular, we would have output it and stopped computations.

The resulting matrix $A = (a_{i,j})_{i,j=1}^{n}$ had nullity one. Indeed it was a rank-one A-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 $K = (a_{i,j})_{i,j=0}^{n} = \begin{pmatrix} w & \mathbf{v}^T \\ \mathbf{b} & A_0 \end{pmatrix}$ for $w = a_{0,0}$, $\mathbf{b} = (a_{i,0})_{i=1}^{n}$, 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 6.2 for $r = 1$, to compute a null vector of the matrix A given by the vector $(0, I_n)K^{-1}\begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$. This amounted to solving a nonsingular Toeplitz linear systems of equations with the matrix K. For that task we applied the code in [44], based on the algorithms in [24], [45], [46]. 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 [19, Section 5.5], but we avoided them in our tests.

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 6.1), QR factorization, and SVD, respectively. Table 9.1 covers our computation of 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 9.1: 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	–	–	–	–

9.2 Computations with unstructured matrices

a) Generation of input matrices

We first fixed pairs of n and k for $n = 64, 128$ and $k = 7$. Then for every pair (n, k) we generated $m = 100$ instances of matrices A, B, V_0, and V_1 and vectors $\mathbf{b}$ as follows.

The matrices A have been generated 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 $j = 1, \ldots, k$, and $\sigma_{n-j} = 1/(n-j)$ for $j = k+1, \ldots, n-1$ (cf. [21, Section 28.3]).

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

V was $k \times (n+k)$ matrix $V = (V_0 \mid V_1)$ where and V_0 was the $k \times k$ identity matrix I_k and $V_1 = B^T$.

For every choice of these matrices we performed preconditioning tests and the solution tests as follows.

b) Preconditioning tests

We computed m ratios $\frac{\kappa(A)}{\kappa(M)}$ for $M = \begin{pmatrix} V_0 & V_1 \\ B & A \end{pmatrix}$.

Table 9.2 displays the average (mean), minimum, maximum, and standard deviation for the m ratios for $n = 64$ and $n = 128$.

Table 9.2: 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}

c) The solution tests

We solved nonsingular linear systems $A\mathbf{y} = \mathbf{b}$ where A was the matrix generated above, $\mathbf{b}$ was a random vector, and we scaled them to have $||\mathbf{b}|| = ||A|| = 1$. We first computed the null vector $\mathbf{z}$ of the matrix $(-\mathbf{b}, A)$, then scaled it to obtain the vector $(1, \mathbf{y})^H$, and finally output the solution vector $\mathbf{y}$.

Tables 9.3 and 9.4 display the average (mean), minimum, maximum, and standard deviation for the relative residual norms $\frac{||A\mathbf{y}-\mathbf{b}||}{||\mathbf{y}||}$ in our tests for $n = 64$ and $n = 128$, respectively. For each input instance we computed the solution in two ways, that is by performing two iterations of the extended iterative refinement and with no such iteratiion.

Table 9.3: relative residual norms in the solution tests with 64×64 inputs

refinement	**min**	**max**	**mean**	**std**
2 iterations	7.89×10^{-48}	8.26×10^{-44}	1.40×10^{-45}	8.47×10^{-45}
no iteration	1.43×10^{-31}	7.30×10^{-28}	1.69×10^{-29}	9.12×10^{-29}

Table 9.4: relative residual norms in the solution tests with 128×128 inputs

refinement	**min**	**max**	**mean**	**std**
2 iterations	1.31×10^{-46}	1.37×10^{-43}	4.11×10^{-45}	1.67×10^{-44}
no iteration	8.57×10^{-31}	1.92×10^{-27}	5.12×10^{-29}	2.55×10^{-28}

10 Conclusion

The computation of a basis for the null space of a rectangular $m \times n$ matrix A having full row rank m is immediately reduced to inverting its $m \times m$ nonsingular submatrix. We prove that random premultipliers make the $m \times m$ leading block nonsingular with probability one and are unlikely to blow up the condition number of A; thus the algorithm is expected to be numerically safe in the case of a well conditioned matrix A of full row rank m.

To extend the algorithm to a rank deficient and ill conditioned matrix A of any size, we combine scaled randomized western, northern and northwestern augmentations with aggregation. We obtain a desired basis for the null space by performing all our computations with well conditioned matrices of full rank. We also extend our algorithms to preconditioning a nonsingular but ill conditioned linear system of equations.

We avoid the drawbacks of pivoting and orthogonalization required in the customary algorithms, which we significantly accelerate, according to both our formal study and experiments, in the case of both general and Toeplitz inputs.

Some parts of our analysis, in particular our estimates for the condition numbers of randomly updated matrices and link between augmentation and aggregation can be of independent technical interest.

References

[1] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, England, 1994.

[2] A. V. Aho, J. E. Hopcroft, J. D. Ullman, *The Design and Analysis of Algorithms.* Addison-Wesley, Reading, MA, 1974.

[3] R. P. Brent, F. G. Gustavson, D. Y. Y. Yun, Fast Solution of Toeplitz Systems of Equations and Computation of Padé Approximations, *J. Algorithms*, **1**, 259–295, 1980.

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

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

[6] R.E. Cline, R.J. Plemmons, and G. Worm, Generalized inverses of certain Toeplitz matrices. *Linear Algebra and Its Applications,* **8**, 25–33, 1974.

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

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

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

[10] 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.

[11] 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.

[12] 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*.

[13] A. Edelman, B. D. Sutton, Tails of Condition Number Distributions, *SIAM J. on Matrix Analysis and Applications*, **27**, **2**, 547–560, 1988.

[14] M. Fürer, Faster Integer Multiplication, *Proceedings of 39th Annual Symposium on Theory of Computing (STOC 2007)*, 57–66, ACM Press, New York, 2007.

[15] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, SIAM, Philadelphia, 1997.

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

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

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

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

[20] G. Heinig, Beitrage zur spektraltheorie von Operatorbuschen und zur algebraischen Theorie von Toeplitzmatrizen, Dissertation **B**, *TH Karl-Marx-Stadt*, 1979.

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

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

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

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

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

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

[27] 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.

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

[29] F. Poloni, A Note on the O(n)-Storage Implementation of the GKO Algorithm, *Numerical Algorithms*, **55**, 115–139, 2010.

[30] 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.

[31] 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.

[32] 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.

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

[34] 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.

[35] V. Y. Pan, G. Qian, A. Zheng, Randomized Preconditioning versus Pivoting, *Linear Algebra and Its Applications*, in print.

[36] V. Y. Pan, G. Qian, A. Zheng, Randomized and Derandomized Matrix Computations, Tech. Report TR 2011010, *Ph.D. Program in Computer Science*, *Graduate Center, the City University of New York*, 2011.

Available at http://www.cs.gc.cuny.edu/tr/techreport.php?id=408

[37] 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.

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

[39] G. Rodriguez, Fast Solution of Toeplitz- and Cauchy-like Least Squares Problems, *SIAM J. Matrix Analysis and Applications*, **28, 3**, 724–748, 2006.

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

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

[42] 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.

[43] W. F. Trench, A Note on a Toeplitz Inversion Formula, *Linear Algebra and Its Applications*, **29**, 55–61, 1990.

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

[45] 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.

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

[47] M. Wschebor, Smoothed Analysis of $\kappa(a)$, *J. of Complexity*, **20**, 97–107, 2004.

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

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