2010

TR-2010011: Randomized Preprocessing versus Pivoting II

Victor Y. Pan

Guoliang Qian

Ai-Long Zheng

Randomized Preprocessing versus Pivoting II *

Victor Y. Pan[1,2],[a], Guoliang Qian[2],[b], and Ai-Long Zheng[2],[c]

[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
[c] azheng-1999@yahoo.com

Abstract

It is known that without pivoting Gaussian elimination can run significantly faster, particularly for matrices that have structure of Toeplitz or Hankel types, but becomes numerically unsafe. The known remedies take their toll, e.g., symmetrization squares the condition number of the input matrix. Can we fix the problem without such a punishment? Taking this challenge we combine randomized preconditioning techniques with iterative refinement and prove that this combination is expected to make pivoting-free Gaussian elimination numerically safe while keeping it fast. For matrices having structures of Toeplitz or Hankel types a cubic arithmetic time bound for Gaussian elimination with pivoting decreases to a nearly linear time bound, and our tests show dramatic decrease of the CPU time as well.

2000 Math. Subject Classification: 65F05, 65F22, 65F35

Key Words: Gaussian elimination, Pivoting, Randomized preconditioning, Hankel matrix computations

1 Gaussian Elimination with No Pivoting

Gaussian elimination with no pivoting (hereafter we refer to it as GENP) generally fails to produce an uncorrupted numerical solution to a linear system of equations because of the propagation of rounding errors. Pivoting, that is row or column interchange, however, takes its toll. It "usually degrades the performance" [GL96, page 119] by interrupting the string of arithmetic computations with the foreign operations of comparisons. Furthermore pivoting is not friendly to updating input matrices, hinders parallel processing, threatens or undermines application of block matrix algorithms, and rapidly destroys sparseness and matrix structure of Toeplitz and Hankel types, thus increasing the running time of the solution dramatically, from nearly linear to cubic.

Of course one can support GENP by applying symmetrization instead of pivoting, but at the expense of squaring the condition number of an input matrix. We refer the reader to [VBHK01,

*Supported by PSC CUNY Awards 61406–0039 and 62230–0040

Introduction] and the references therein on various other methods of alleviating this problem in the case of input matrices with the structure of Toeplitz and Hankel types. In particular one can decrease the estimated computational time from cubic to quadratic, although with a larger overhead constant. These methods employ normal equations, augmentation, look ahead techniques, and displacement transformation. See [P90] and [P01, Sections 4.8, 4.9, and 5.6] on the original source of the latter method, [GKO95] on its most celebrated application, and [CGLX], [CGSXZ], [P10a], and [R06] on some more recent advances.

It is easy to prove that GENP is numerically safe where the input matrix is diagonally-dominant, positive definite, or more generally, strongly nonsingular and strongly well conditioned, that is, nonsingular and well conditioned together with all its leading principal submatrices. According to [D88], [E88], [SST06], and [ST02], the latter property is expected to hold for random matrices, for which GENP is therefore expected to be safe.

In computational practice we can rarely consider the input matrices random, but we prove that pre- and post-multiplication of a well conditioned matrix by random multipliers is likely to turn it into a strongly well conditioned matrix, thus allowing numerically safe application of GENP. Moreover according to our extensive tests it is sufficient to use circulant multipliers filled with the integers $+1$ and -1 (with the signs $\pm$ chosen at random for the entries of the first columns) or to replace randomized multiplication with randomized augmentation, which a little increases the input size but perfectly preserves matrix structure and sparseness. Furthermore, according to both our formal analysis and experiments randomized augmentation under proper scaling is also expected to work as preconditioning, and randomized additive preprocessing has similar properties, although it slightly weaker preserves matrix structure.

We demonstrate the power of our approach by applying it to recursive block GENP and block Gauss–Jordan elimination with no pivoting (hereafter referred to as *GJENP*). In the highly important case of linear systems of equations with displacement structures of Toeplitz and Hankel types these algorithms are superfast, running in nearly linear arithmetic time, versus at best quadratic time required in computations with pivoting. We provide some pointers to the extensions of these techniques to numerical computation of determinants in Section 6.4.

In our tests the relative residual norms of the outputs were a little greater in the case of GENP with preprocessing than in the case of Gaussian elimination with partial pivoting (hereafter we refer to it as GEPP), but remained small enough to support fast and inexpensive iterative refinement of the resulting approximate solutions, so that overall we dramatically accelerated GEPP with no sacrifice in the output accuracy. Table 7.3 displays the observed CPU time of the solution of a nonsingular Hankel linear systems of n equations. These data show that our algorithms are faster by the factor $a(n)$ than the QR-based solution where $a(512) \approx 5$, $a(1024) \approx 20$, and $a(2048) \approx 70$.

Our present demonstration of the power of randomized preconditioning shows just the tip of an iceberg. Various randomizations in [PQ10], [PQa], [PQZb], [PQZC], and [PZa] facilitate the solution of linear systems of equations, matrix inversion, the computation of the numerical rank of a matrix, approximation of a nearly rank deficient matrix with a nearby matrix of a smaller rank, approximation of a matrix by a nearby Toeplitz-like or Hankel-like matrix, and root-finding for polynomial and secular equations. Preconditioning via randomized augmentation seems to be a natural effective means at the critical stage of the initialization of Newton's iteration for the inversion and generalized inversion of general and structured matrices (cf. [P92], [P93a], [P93b], [PBRZ99], [P01, Chapter 6], [PRW02], [PVWC04], [CPV04], [P10]).

We organize our paper as follows. We devote the next section to the definitions and auxiliary results. Section 3 covers the basic 2×2 block triangular factorizations, Section 4 their recursive extensions (which include GENP and GJENP as special cases), and Section 5 the respective numerical properties and randomized preconditioning. In Section 6 we discuss preconditioning power of our randomized multiplicative and additive preprocessing and augmentation, and in Section 7 we cover our numerical experiments, which are the contribution of the second and the third authors.

2 Definitions and basic facts

2.1 General matrices

We use and extend the customary definitions in [GL96]).

"GENP" and "GJENP" (resp. "GEPP" and "GJEPP") stand for "Gaussian Elimination with No Pivoting" and "Gauss-Jordan Elimination with No Pivoting" (resp. "with Partial Pivoting").

$\mathbb{C}$ (resp. $\mathbb{R}$) is the field of complex (resp. real) 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), $\rho = \operatorname{rank} A$ denotes its rank, $\operatorname{nul} A = n - \rho$ its nullity, $A^{(k\times l)}$ its $k \times l$ leading (that is northwestern) block submatrix, so that $A = A^{(m\times n)}$. Hereater we write $A^{(k)}$ for $A^{(k\times k)}$.

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 A of a rank ρ has *generic rank profile* if all its leading blocks $A^{(k)}$ of size $k \times k$ for $k \le \rho$ are nonsingular. If in addition $\rho = \min\{m, n\}$, then the matrix is *strongly nonsingular*.

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

I_n or just I denote the $n \times n$ identity matrix $(\mathbf{e}_1, \dots, \mathbf{e}_n)$, with the columns $\mathbf{e}_1, \dots, \mathbf{e}_n$.

$J = J_n = (\mathbf{e}_n, \dots, \mathbf{e}_1)$ is the $n \times n$ reflection matrix. ($J^2 = I$.)

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

$O_{k,l}$ or just O and $\mathbf{0}_k$ or just $\mathbf{0}$ denote a $k \times l$ matrix and a vector of a dimension k filled with zeros, respectively.

$\mathcal{N}(A)$ is the null space of a matrix A, made up of its *null vectors* $\mathbf{y}$ such that $A\mathbf{y} = \mathbf{0}$, so that $\operatorname{nul}(A) = \dim(\mathcal{N}(A))$.

If the columns of a matrix B of full column rank span the null space $\mathcal{N}(A)$, then B is a *null matrix basis* (hereafter we write *nmb*) for a matrix A.

$M(n)$ flops suffice for multiplying a pair of $n \times n$ matrices, $M(n) \le (2n-1)n^2$, $M(n) \le Cn^{2.376}$, $M(n) \le cn^{2.776}$, for an immense constant C [CW90] and a moderate constant c [LPS92].

2.2 Matrices with the structures of Toeplitz, Hankel, Vandermonde, and Cauchy types

The classes of Toeplitz matrices $\mathbb{T} = \{T = (t_{i-j})_{i,j=1}^n\}$ and Hankel matrices $\mathbb{H} = \{H = (h_{i+j})_{i,j=1}^n\}$ are linked to one another via the reflection matrix J as follows, $\mathbb{H} = J\mathbb{T} = \mathbb{T}J$. Note that J (resp. I) is a Hankel (resp. Toeplitz) matrix. We refer the reader to the book [P01] and the bibliography therein on these and some other popular classes of structured matrices, such as Vandermonde matrices $V = (t_i^j)_{i,j=1}^n$, Cauchy matrices $C = (\frac{1}{s_i - t_j})_{i,j=1}^n$, and Toeplitz-like, Hankel-like, Vandermonde-like, and Cauchy-like matrices, having structures similar to the structures of Toeplitz, Hankel, Vandermonde, and Cauchy matrices, respectively. We refer to these four classes as the *THVC matrices*.

Such a matrix M can be readily expressed via its displacement $L(M) = AM - MB = GH^T$ (or $L(M) = M - AMB = GH^T$) having a small rank d, called the *displacement rank* of the matrix M. Here G and H are $n \times d$ matrices, A and B are operator matrices defining the displacement operator L. The displacement ranks d of THVC matrices are small under appropriate choices of operator matrices of shift and scaling. (E.g., $d \le 2$ for Toeplitz and Hankel matrices M, whereas $d = 1$ for Vandermonde and Cauchy matrices M.) Some effective algorithms (see, e.g., [M80], [BA80]) involve representations $L(M) = GH^T$ of length $l > d$ (that is with $G, H \in \mathbb{C}^{n\times l}$ for $l > d$) for structured matrices M having a displacement rank d, but one can compress such representations to the minimal length d by using $O(nl^2)$ flops [P01, Section 4.6.2].

The $n \times n$ matrix $M = L^{-1}(L(M))$ itself can be readily expressed via $2ln$ entries of its displacement generator $\{G, H\}$ of a length l provided $L(M) = GH^T$ and the linear operator L is nonsingular. Simple explicit expressions for the structured matrices M of the above classes via their

displacement generators can be found in [P01, Section 4.4]. Such a displacement representation enables multiplication of the matrix M by a vector in $O(ln \log^h n)$ flops (where $h = 1$ for Toeplitz-like and Hankel-like matrices M and $h \le 2$ for Vandermonde-like and Cauchy-like matrices M).

f-circulant matrices $Z_f(\mathbf{v}) = (z_{i,j}^{(f)})_{i,j=1}^n$ for a scalar f are Toeplitz matrices such that $z_{1,j}^{(f)} = fz_{n+1-j,1}^{(f)}$, $j = 2, \dots, n$ (cf., e.g., [P01, Section 2.6]). They form an algebra and can be pairwise multiplied and inverted by using $O(n \log n)$ flops [CPW74]. Such a matrix $Z_f(\mathbf{v})$ is defined by its first column $\mathbf{v} = Z_f(\mathbf{v})\mathbf{e}_1$. f-circulant matrices are circulant for $f = 1$, skew-circulant for $f = -1$, and turn into lower triangular Toeplitz matrices for $f = 0$.

2.3 Matrix norms, SVDs, inverses, and condition numbers

$||A||_h$ denotes the h-norm of a matrix A, $h = 1, 2, \infty$. We write $||A||_2 = ||A||$. For $A = (a_{i,j})_{i,j=1}^{m,n}$ we have

$$\max_{i,j=1}^{m,n} |a_{i,j}| \le ||A|| = ||A^H|| \le \sqrt{mn} \max_{i,j=1}^{m,n} |a_{i,j}|. \tag{2.1}$$

$A = S_A \Sigma_A T_A^H$ is a *full SVD* of an $m \times n$ matrix A of a rank ρ provided $S_A S_A^H = S_A^H S_A = I_m$, $T_A T_A^H = T_A^H T_A = I_n$, $\Sigma_A = \mathrm{diag}(\widehat{\Sigma}_A, O_{m-\rho,n-\rho})$, $\widehat{\Sigma}_A = \mathrm{diag}(\sigma_j(A))_{j=1}^{\rho}$, $\sigma_j = \sigma_j(A) = \sigma_j(A^H)$ is the jth largest singular value of a matrix A, having its minimax characterization

$$\sigma_j = \max_{\dim(\mathbb{S})=j} \min_{\mathbf{x} \in \mathbb{S}, ||\mathbf{x}||=1} ||A\mathbf{x}|| \tag{2.2}$$

where $\mathbb{S}$ denotes linear spaces [GL96, Theorem 8.6.1] and $j = 1, \dots, \rho$. It follows that

$$\sigma_1 = \max_{||\mathbf{x}||=1} ||A\mathbf{x}|| = ||A||, \quad \sigma_n = \min_{||\mathbf{x}||=1} ||A\mathbf{x}||. \tag{2.3}$$

An $m \times n$ matrix M for $m \ge n$ has numerical nullity r (and then we write $\mathrm{nnul}\, M = r$) if the ratio $\frac{\sigma_{n-r+1}}{\sigma_1}$ is small or vanishes, but the ratio $\frac{\sigma_{n-r}}{\sigma_1}$ is not small.

The matrix $X = A^{(I)}$ is a left (resp. right) inverse of a matrix A if $XA = I$ (resp. $AX = I$). $A^{(I)} = A^{-1}$ for a nonsingular matrix A.

$\mathrm{cond}(A) = \sigma_1(A)/\sigma_\rho(A)$ is the condition number of a matrix A of a rank ρ. Such a matrix is *ill conditioned* if the ratio $\sigma_1(A/\sigma_\rho(A)$ is large and is *well conditioned* otherwise. The concepts "large", "small", "ill conditioned" and "well conditioned" are quantified in the context of the computational task and computer environment.

Our next definition extends the class of strongly nonsingular matrices.

Definition 2.1. *A matrix A is* strongly well conditioned *if all its leading submatrices $A^{(k)}$ are well conditioned.*

2.4 Random sampling and random matrices

$|\Delta|$ is the cardinality of a set Δ in any fixed ring. *Random sampling* of elements from a set Δ is their selection from this set at random and independently of each other. A matrix is *random* if its entries are randomly sampled from a fixed set Δ. Random sampling is *uniform* if it is done under the uniform probability distribution on the set Δ.

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 defree of its monomials.

Lemma 2.1. *[DL78], [S80], [Z79]. For a set Δ of cardinality $|\Delta|$ (in a fixed ring), 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 a probablity converging to zero if the values of its variables are sampled under any reasonable probability distribution on the set Δ whose cardinality converges to the infinity. Under the uniform probability distribution the probability is estimated most readily.

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 at most $\frac{d}{|\Delta|}$.*

Corollary 2.2. *Let the entries of an $m \times n$ matrix have been randomly and uniformly sampled from a finite set Δ of cardinality $|\Delta|$ (in any fixed ring). Let $l = \min\{m, n\}$. Then (a) every $k \times k$ submatrix for $k \le l$ is singular with a probability at most $k/|\Delta|$ and (b) is strongly nonsingular with a probability at least $1 - \sum_{i=1}^{k} i/|\Delta| = 1 - (k+1)k/|\Delta|$.*

Proof. The claimed bound hold for generic matrices. The singularity of a $k \times k$ matrix means that its determinant vanishes, but the determinant is a polynomial of the total degree k in the entries. Now Corollary 2.1 implies parts (a) and (b). □

Definition 2.2. *$F_X(y) = Probability\{X \le y\}$ 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 $l = \min\{m, n\}$. 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. If $\mu = 0$ and $\sigma^2 = 1$, this is a* standard Gaussian random matrix *(resp. vector). $F_{\mu,\sigma}(y) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{y} \exp(-\frac{(x-\mu)^2}{2\sigma^2})dx$ is the CDF for a Gaussian random variable with a mean μ and a variance σ^2.*

2.5 Ranks and condition numbers of randomized matrix products

Gaussian random matrices (cf. Definition 2.2) tend to be well conditioned [D88], [E88], and even perturbations by such a matrix A is expected to make a matrix M well conditioned if the norms $||A||$ and $||M||$ have the same order [SST06]. Next we recall the respective estimates from [SST06] and their nontrivial extension from [PQZb], which supports application of our randomized preprocessing as a substitution for pivoting. Namely our results from [PQZb] show that the products of a well conditioned matrix of full rank and a Gaussian random matrix are strongly nonsingular and strongly well conditioned with a probability one or close to one.

Theorem 2.1. *(See [DS01, 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 $F_{||A||}(y) \ge 1 - \exp(-x^2/2)$ for $x = y/\sigma - 2\sqrt{n} \ge 0$.*

Theorem 2.2. *(See [SST06, Theorem 3.3].) Suppose $M \in \mathbb{R}^{m \times n}$, $\bar{U} \in \mathbb{R}^{m \times m}$, and $\bar{V} \in \mathbb{R}^{n \times n}$ are three fixed matrices, $\bar{U}$ and $\bar{V}$ are unitary matrices, $A \in \mathbb{R}^{m \times n}$ is a Gaussian random matrix independent of the matrix M and having mean zero and a variance σ^2, $W = \bar{U}(A + M)\bar{V}$, $l = \min\{m, n\}$, and $y \ge 0$. Then $F_W(y) \le 2.35y\sqrt{l}/\sigma$.*

Theorem 2.3. *(See [SST06, Theorem 3.1].) Under the assumptions of Theorem 2.2, let $||M|| \le \sqrt{l}$. Then $F_{\mathrm{cond}(W)}(y) \ge 1 - \frac{n}{y\sigma}(14.1 + 4.7\sqrt{\frac{2}{n}\ln y})$ for all $y \ge 1$.*

On further improvement of this bound by the factor of $\sqrt{\log n}$, see [W04].

Theorem 2.4. *(See [PQZb, Theorem 4.5].) Suppose $G \in \mathbb{R}^{q \times m}$ and $H \in \mathbb{R}^{n \times r}$ are two fixed matrices, $r_G = \mathrm{rank}\, G = m$, $r_H = \mathrm{rank}\, H = n$, a random matrix $W \in \mathbb{R}^{m \times n}$ has full rank with probability one, and $y \ge 0$. Then (a) $F_{GW}(y) \le F_W(y/\sigma_{r_G}(G))$, whereas (b) $F_{WH}(y) \le F_W(y/\sigma_{r_G}(H))$.*

Theorem 2.5. *(See [PQZb, Theorem 4.6].) Suppose $G \in \mathbb{R}^{r_G \times m}$, $H \in \mathbb{R}^{n \times r_H}$, $X \in \mathbb{R}^{m \times n}$, $\mathrm{rank}\, G = r_G < m$, $\mathrm{rank}\, H = r_H < n$, $y \ge 0$, and the assumptions of Theorem 2.2 hold for the matrix X replacing W. Then $F_{GX}(y) \le 2.35y\sqrt{l}/(\sigma_{r_G}(G)\sigma)$ and $F_{XH}(y) \le 2.35y\sqrt{l}/(\sigma_{r_H}(H)\sigma)$.*

Corollary 2.3. *(See [PQZb, Corollary 4.4].) Suppose k, m, and n are integers, $1 \le k \le n \le m$, G, $H^T \in \mathbb{R}^{m \times n}$, $\mathrm{rank}\, G = \mathrm{rank}\, H = n$, $X \in \mathbb{R}^{n \times n}$ is a standard Gaussian random matrix with a mean μ and a variance σ, and $y \ge 0$. Then we have $F_{(GX)^{(k)}}(y) \le 2.35y\sqrt{k}/(\sigma_n(G)\sigma)$ and $F_{(XH)^{(k)}}(y) \le 2.35y\sqrt{k}/(\sigma_n(H)\sigma)$.*

Corollary 2.4. *(See [PQZb, Corollary 4.5].) Under the assumptions of Corollary 2.3 choose a scalar* $z \geq 2\sigma\sqrt{n}$. *Then* $F_{\mathrm{cond}(GX)^{(k)}}(yz||G||) \geq 1 - \exp(\frac{(z-2\sigma\sqrt{n})^2}{2\sigma^2}) - 2.35y\sqrt{k}/(\sigma_n(G)\sigma)$ *and* $F_{\mathrm{cond}(XH)^{(k)}}(yz||H||) \geq 1 - \exp(\frac{(z-2\sigma\sqrt{n})^2}{2\sigma^2}) - 2.35y\sqrt{k}/(\sigma_n(H)\sigma)$.

Corollary 2.5. *Under the assumptions of Corollary 2.3 let* $\sigma \neq 0$. *Then with probability one the matrices* GX *and* XH *are strongly nonsingular.*

Similarly to Corollary 2.2 we can prove that such a property is also expected to hold under the uniform random sampling of the entries of matrix X from a finite set Δ in any fixed ring.

Corollary 2.6. *Let the assumptions of Corollary 2.3 hold except that the entries of the matrix* X *have been randomly and uniformly sampled from a finite set* Δ *of a cardinality* $|\Delta|$ *in any fixed ring. Then with a probability at least* $1-(n+1)n/|\Delta|$ *the matrices* GX *and* XH *are strongly nonsingular.*

3 Basic block factorizations

Let n, k, and $r = n - k$ be positive integers and let $M_{00} = M^{(k)}$ be the nonsingular $k \times k$ leading block of an $n \times n$ matrix $M = \begin{pmatrix} M_{00} & M_{01} \\ M_{10} & M_{11} \end{pmatrix}$, represented as a 2×2 block matrix. Then a single step of block GENP and block GJENP outputs the block factorizations

$$M = \begin{pmatrix} I_k & O_{k,r} \\ M_{10}M_{00}^{-1} & I_r \end{pmatrix} \begin{pmatrix} M_{00} & M_{01} \\ O_{r,k} & S \end{pmatrix} \tag{3.1}$$

and

$$M = \begin{pmatrix} I_k & O_{k,r} \\ M_{10}M_{00}^{-1} & I_r \end{pmatrix} \begin{pmatrix} M_{00} & O_{k,r} \\ O_{r,k} & S \end{pmatrix} \begin{pmatrix} I_k & M_{00}^{-1}M_{01} \\ O_{k,r} & I_r \end{pmatrix} \tag{3.2}$$

where

$$S = S(M^{(k)}, M) = M_{11} - M_{01}M_{00}^{-1}M_{01} \tag{3.3}$$

denotes the Schur complement of $M_{00} = M^{(k)}$ in M. These factorizations are infeasible (resp. numerically unstable) if and only if the block M_{00} is singular (resp. ill conditioned) and vice versa. If the matrices M_{00}, M, and S are nonsingular, then we have

$$M^{-1} = \begin{pmatrix} M_{00}^{-1} & -M_{00}^{-1}M_{01}S^{-1} \\ O_{r,k} & S^{-1} \end{pmatrix} \begin{pmatrix} I_k & O_{k,r} \\ -M_{10}M_{00}^{-1} & I_r \end{pmatrix} \tag{3.4}$$

and

$$M^{-1} = \begin{pmatrix} I_k & -M_{00}^{-1}M_{01} \\ O_{r,k} & I_r \end{pmatrix} \begin{pmatrix} M_{00}^{-1} & O_{k,r} \\ O_{r,k} & S^{-1} \end{pmatrix} \begin{pmatrix} I_k & O_{k,r} \\ -M_{10}M_{00}^{-1} & I_r \end{pmatrix}. \tag{3.5}$$

Note that S^{-1} is the trailing (southeastern) block of M^{-1}, and so the matrix S is nonsingular if so is the matrix M.

4 Recursive (block) factorizations, GENP and GJENP

We can recursively extend factorizations (3.1)–(3.5) to the diagonal blocks M_{00}, S, M_{00}^{-1} and S^{-1} as long as these blocks are not scalars, not singular, and have nonsingular leading blocks. When all diagonal blocks finally turn into scalars, we can multiply together all lower triangular factors in factorization (3.1) (resp. all lower triangular factors as well as all upper triangular factors in factorization (3.2)) extended recursively to the unique LU (resp. LDU_1) factorization of the matrix M, where the factor D is diagonal, the factor U is upper triangular, and the factor L is (resp. the factors L and U_1^T are) unit lower triangular.

The computation of recursive factorizations (3.1) and (3.2) (that is block GENP and block GJENP, respectively) is completely defined by the choices of the dimensions k and r in all steps. In

particular Theorem 4.2 below implies that the output LU and LDU_1 factorizations do not depend on the order of decreasing the block sizes to 1×1.

These processes turn into GENP and GJENP, respectively, if we always choose $k = 1$ in all steps, that is choose the size 1×1 for all the leading blocks involved. We call the recursive process *balanced* if $|k - r| \le 1$ in all its steps.

Due to equations (3.4) and (3.5) the same processes also define recursive factorizations of the inverse M^{-1} if M is a nonsingular matrix.

Theorem 4.1. *Computing the recursive block factorization based on factorizations (3.1) or (3.2) for an $n \times n$ input matrix M takes (a) $O(n^3)$ flops for the GENP process and (b) $O(M(n))$ flops for a balanced process and for $M(n)$ defined in Section 2.1. If M is a Toeplitz-like or Hankel-like matrix having a displacement rank d and a displacement representation of length $O(d)$, then displacement representations of lengths $O(d)$ for all auxiliary matrices involved into the recursive block factorization can be computed by using (c) $O(dn^2)$ flops in the case of the GENP process and (d) $O(d^2 n \log^2 n)$ flops in the case of a balanced process. (e) Given a vector $\mathbf{b}$ of dimension n and a complete recursive factorization of a matrix M based on equations (3.1)–(3.3), one can compute the value $\det M$ in $n - 1$ flops, the vector $M^{-1}\mathbf{b}$ for general nonsingular matrix M in $O(n^2)$ flops, and the same vector but for a Toeplitz-like or Hankel-like matrix M having a displacement rank d in $O(nd \log n)$ flops.*

Proof. See, e.g., [GL96] on part (a), [AHU74] on part (b), [GKO95] on part (c), and [M80], [BA80], and [P01, Chapter 5] on parts (d) and (e). □

The following theorem expresses the auxiliary Schur complements in these factorizations as Schur complements in the input matrix and its submatrices.

Theorem 4.2. *In every step of the recursive block factorization process based on (3.1) or (3.2) every diagonal block of a block diagonal factor is either a leading block of the input matrix M or the Schur complement*

$$S(M^{(h)}, M^{(k)}) = (S(M^{(h)}, M))^{(h)} \tag{4.1}$$

for some integers h and k, $0 < h < k \le n$.

Corollary 4.1. *a) The recursive block factorization process based on equations (3.1) or (3.2) can be completed if and only if the input matrix M is strongly nonsingular.*

b) Generally, for any input matrix M, this process completes the recursive block factorization of the strongly nonsingular leading block of this matrix that has the largest size $\rho \times \rho$. The flop estimates in Theorem 4.1 with ρ replacing n apply to these computations.

c) For a matrix M of a rank ρ having generic rank profile, we have $S(M^{(\rho)}, M) = O$, and a $nmb(M)$ is given by the matrix

$$F = \begin{pmatrix} -M_{00}^{-1} M_{01} \\ I_r \end{pmatrix} \text{ for } M_{00} = M^{(\rho)}. \tag{4.2}$$

5 Numerical behavior of recursive block GENP and GJENP

Theorem 5.1. *Write $N = ||M||$ and $N_- = \max_{k=1}^{n} ||(M^{(k)})^{-1}||$. Then $\mathrm{cond}(D) \le (1 + N_- N) N N_-$ for every diagonal block D in the recursive block factorization of a strongly well conditioned matrix M in Section 4.*

Proof. For the blocks M_{00} of M and S^{-1} of M^{-1} we surely have $||M_{00}|| \le N$, $||M_{00}^{-1}|| \le N_-$, and $||S^{-1}|| \le ||M^{-1}|| \le N_-$. We also have $||S|| \le N + N_- N^2$, due to (3.3). Now the claimed bound follows from Theorem 4.2. □

Clearly the bound $(1 + N_- N) N N_-$ is not large if and only if the strongly well conditioned matrix M is strongly well conditioned, and if so, recursive GENP and GJENP for this matrix are numerically stable.

Our study implies that GENP and GJENP (as well as recursive GENP and GJENP) for $\rho = \operatorname{rank} M$ and a matrix $M^{(\rho)}$ are feasible and well conditioned if and only if the matrix $M^{(\rho)}$ is strongly nonsingular and strongly well conditioned. Furthermore these properties are expected to hold for the matrices $XM^{(\rho)}Y$ if the matrix $M^{(\rho)}$ is nonsingular and well conditioned and if X and Y are standard Gaussian $n \times n$ random matrices (cf. Corollary 2.4 and 2.5).

6 Regularization and preconditioning power of randomized preprocessing

6.1 Randomized preprocessing of well conditioned matrices

In the case of structured input matrices the recursive balanced block GENP and GJENP can be computed in nearly linear arithmetic time (see Theorem 4.1), but these superfast algorithms are prone to the overflow and numerical stability problems [B85] even where the input matrix is nonsingular and well conditioned unless it is strongly nonsingular and strongly well conditioned. In Section 2.5 we proved that random multipliers are expected to fix this problem, but can these results, and in particular Theorems 2.2 and 2.5, be extended to the case of random structured multipliers?

Strong regularization property can be extended based on Lemma 2.1 (see [PW08]), but the extension of strong preconditioning remains an open problem, although in the case of Toeplitz matrices we support this extension empirically (cf. [PQa, Remark 5.4] and [PQZb, Remark 5.1]). The extension would immediately follow from our proofs in Section 2.5 if Theorem 2.2 is extended respectively. Then again in the case of Toeplitz matrices we support the latter extension empirically in [PQZb] but cannot prove its validity formally.

Multiplication by a Toeplitz matrix does not destroy Toeplitz structure but a little spoils it by increasing the displacement rank, at most by two [P01, Theorem 1.5.4 and equation (4.2.1)]. Pre-multiplication by f-circulant matrices and post-multiplication by e-circulant matrices for two nonzero scalars e and f, however, preserve Toeplitz structure and the displacement rank [P01, Sections 2.6 and 4.4].

Alternatively consider the flop-free and error-free augmentation

$$M \Longrightarrow K = \begin{pmatrix} W & V^H \\ U & M \end{pmatrix} \tag{6.1}$$

for a positive integer r and nonsingular matrices $M \in \mathbb{C}^{n\times n}$, $W \in \mathbb{C}^{r\times r}$, and $K \in \mathbb{C}^{(n+r)\times(n+r)}$. In this map we can completely preserve the input matrix structure.

We refer the reader to [PQa] on a further study of augmentation. In particular according to this study, we can expect that $\operatorname{cond}(K^{(h+r)})$ has at most the order $\frac{\sigma_1(M^{(h)})}{\sigma_{h-r}(M^{(h)})}$ for $h = r+1, r+2, \dots, n$ in the case of standard Gaussian random matrices U, V, and W and of the matrix M scaled so that the norm $||M||$ is neither large nor small. We can even choose $V = aU$ where a is a scalar whose absolute value $|a|$ is neither large nor small. The preconditioning power of the augmentation is expected to be preserved where the matrices M, U, $V = aU$, and W have norms of the same order except for the case where $a = 1$ and the matrix K is Hermitian and positive definite. Indeed in the latter case augmentation cannot decrease the condition number due to the Interlacing Property of the eigenvalues of Hermitian matrices [GL96, Theorem 8.1.7].

Additive preprocessing

$$M \Longrightarrow C = M + UV^H \tag{6.2}$$

for random scaled matrices $U, V \in \mathbb{C}^{n\times r}$ as well as for random scaled matrix $U \in \mathbb{C}^{n\times r}$ and for $V = U$ has the same regularization and preconditioning power as randomized augmentation in (6.1). In particular we can can expect that $\operatorname{cond}(C^{(h+r)})$ has at most the order of $\frac{\sigma_1(M^{(h)})}{\sigma_{h-r}(M^{(h)})}$ for $h = r+1, r+2, \dots, n$ in the case of standard Gaussian random matrices U and V and of the matrix M scaled so that the norm $||M||$ is neither large nor small [PQZb]. Unlike the case of augmentation, preconditioning power of randomized additive preprocessing (6.2) is preserved and even accentuated in the case where M is a Hermitian positive definite matrix and $U = V$ (cf. [W07]).

In augmentation $M \Longrightarrow K$ and additive preprocessing $M \Longrightarrow C$ the lengths of displacement generators grow by integers in $O(r)$. Therefore for a matrix M represented with a displacement generator of a length d the matrices K and C are represented with displacement generators of lengths in $d + O(r)$, and so our computations are fast where r is a small integer.

According to the analysis in [PQa] and [PQZb], in this case maps (6.1) and (6.2) are expected to yield a strongly nonsingular and strongly well conditioned matrices K and C, respectively, provided that the matrix M is nonsingular and well conditioned (but not necessarily strongly nonsingular and strongly well conditioned) and that its every leading submatrix has at most r singular values that are small relatively to the norm $||M||$.

If the integer r is large, we can preserve the structure by choosing the matrices U, V, and W with consistent structure, but then we would use only $O(dr)$ random parameters. Our randomized preprocessing is still expected to have regularization power [PW08], but proving its preconditioning power in the case where $d = o(n)$ is an open problem, although we consistently observed this power in our tests as well as in the experiments in [PQa] where K, M, U, V, and W were Toeplitz matrices with a total of $2r$ distinct random entries uniformly sampled in the range $[-||M||, ||M||]$.

Remark 6.1. *Lemma 2.1 and Corollary 2.6 can be applied to support randomized symbolic computations in any field.*

6.2 Recovery of the inverse after preprocessing

Suppose that our preprocessing has produced strongly nonsingular and strongly well conditioned matrix C in (6.2) or K in (6.1). Then we can safely perform its recursive factorization and inversion (see Theorems 4.1 and 5.1).

The inverse M^{-1} can be recovered from the inverse C^{-1} based on the Sherman–Morrison–Woodbury formula, $M^{-1} = C^{-1} + C^{-1}U(I_r - V^H C^{-1}U)^{-1}V^H C^{-1}$ (cf. [GL96, page 50]).

Next we specify the recovery of the inverse M^{-1} from the inverse $K^{-1} = \begin{pmatrix} X & Y \\ Z & F \end{pmatrix}$ where $F^{-1} = M - UW^{-1}V^H$ is the Schur complement of the block W in the matrix K. Apply the Sherman–Morrison–Woodbury formula and obtain that $M^{-1} = F + F\bar{U}\bar{G}^{-1}V^H F$ for $\bar{U} = UW^{-1}$ and $\bar{G} = I_r - V^H F\bar{U}$. This reduces the original inversion problem to the inversion of the matrices K, W and $\bar{G}$. Recall that the matrices K and W are expected to be strongly nonsingular and strongly well conditioned assuming scaled random matrices U, V and W. The matrices W and $\bar{G}$ have size $r \times r$. The value $\operatorname{cond} \bar{G}$ has been estimated in [PGMQ] and [PQZb]. It is expected to have the order not greater than $\frac{\sigma_{n-r+1}(M)}{\sigma_n(M)}$, and it equals one where $r = 1$.

One can narrow the inversion algorithm to computing the solution $\mathbf{y} = M^{-1}\mathbf{b}$ of a linear system $M\mathbf{y} = \mathbf{b}$. Due to the equation $M^{-1} = F + F\bar{U}\bar{G}^{-1}V^H F$ the task can be reduced essentially to multiplication of the matrix F (which is a block submatrix of K^{-1}) by the $(n + r) \times (r + 1)$ matrix $(\bar{U}, \mathbf{b})$. This is in turn reduced to solving $r + 1$ linear systems with the matrix K, expected to be strongly nonsingular and strongly well conditioned.

6.3 Extension to preconditioning of ill conditioned inputs

The same augmentation techniques and their analysis can be applied to $n \times n$ ill conditioned matrices M (where multiplication by random matrices is unlikely to help). The condition numbers of the matrix K and its leading block submatrices are expected to have at most the order of $\frac{\sigma_1(M)}{\sigma_{n-r}(M)}$. This ratio is not large (implying that the GENP is expected to be safe for the matrix K) where the matrix M and its every leading block submatrix have at most r singular values that are small relatively to the norm $||M||$. In this case if the ratio $\frac{\sigma_1(M)}{\sigma_{n-r}(M)}$ is large (that is if the matrix M is ill conditioned), then we should compute the matrices F and G with high accuracy to compensate for the expected magnification of the input errors due to ill conditioning of the input matrix M. We can achieve this by applying iterative refinement in computing the matrix F and incorporating the known advanced algorithms for accurate computation of sums and products (cf. [PGMQ], [PQZb]).

The resulting algorithm outputs the entries of the matrices F and G as the sums represented with a high precision implicitly by sufficiently many summands, each computed with double precision.

6.4 Application to numerical computation of determinants

We specified application of our randomized preprocessing to solving linear systems and inversion, but we can readily apply it to numerical computation of determinants and resultatnts (see [BEPP97/99], [EP03/05], [PY99/01], and references therein on important applications to algebraic and geometric computations). In particular we can compute the determinant $\det M$ based on recursive (block) factorization of the matrix M, and we also have $\det K = (\det M)\det(W - V^H M^{-1} U)$ for K in (6.1) and $\det C = (\det M)\det(I_r - V^H C^{-1} U)$ for C in (6.2).

7 Numerical Experiments

Our numerical experiments with random general, Hankel, Toeplitz and circulant matrices have been performed in the Graduate Center of the City University of New York 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 $\{x : -1 \le x < 1\}$.

7.1 Solution of linear systems of equations with multiplicative preconditioning

We solved 1000 linear systems of equations $M\mathbf{x} = \mathbf{b}$ for each input class with vectors $\mathbf{b}$ having random coordinates from the range $[-1, 1)$ and matrices M specified below.

Input Matrices:

$M = \begin{pmatrix} M_k & A \\ B & C \end{pmatrix}$ is an $n \times n$ matrix, M_k is a $k \times k$ matrix, A, B, and C are random Toeplitz matrices such that $||A|| \approx ||B|| \approx ||C|| \approx ||M_k|| \approx 1$, $n = 2^s$, $s = 5, 6, 7, 8, 10$, and $k = n/2$. We have chosen the matrices M_k as follows.

1) *General matrices* $M_k = U\Sigma V^H$ where $\Sigma = \mathrm{diag}(\sigma_i)_{i=1}^k$, $\sigma_i = 1$ for $i = 1, \dots, k-h$, $\sigma_i = 0$ for $i = k-h+1, \dots, k$, $h = 4$, and U and V are $k \times k$ random orthonormal matrices, computed as the $k \times k$ factors Q in the QR factorization of $k \times k$ random matrices where the factor R has positive diagonal entries (cf. [H02, Section 28.3]).

2) *Toeplitz-like matrices* M_k *with nullity* $h = 4$. $M_k = c(T, TS)$ for random Toeplitz matrices T of size $k \times (k-h)$ and S of size $(k-h) \times h$, for $h = 4$, and for a positive scalar c such that $||M_k|| \approx 1$.

Multiplicative preconditioners:

(a) $n \times n$ circulant multipliers, each defined by its first column with the n entries $+1$ and -1 chosen at random,

(b) Householder multipliers $\prod_{i=1}^h (I - 2\mathbf{v}_i \mathbf{v}_i^T / \mathbf{v}_i^T \mathbf{v}_i)$ [GL96, Section 5.1], [S98, Section 4.1.2] with the vectors $\mathbf{v}_i$ filled with the integers $+1$ and -1 chosen at random for all i.

Tables 7.1 and 7.2 show the test results for the solution of linear systems of equations where we apply GENP with randomized structured preconditioning. For all tests the tables display min, max, and average values of the relative residual norm $||M\mathbf{x} - \mathbf{b}||/||\mathbf{b}||$ and the standard deviations. For GENP with preconditioning we show these data obtained before we performed iterative refinement and after the first and sometimes also the third step of it. We continued iterative refinement until we decreased the output residual norms to the level of 10^{-14} (achieved by GEPP). The columns **iterations** in our tables show the respective numbers of steps of iterative refinement.

Due to the singularity of the leading block M_k, the relative residual norms in GENP without preconditioning stayed in the range $[10, 10^8]$ and were too large to allow iterative refinement of the

computed solution. With our randomized structured preconditioning, however, these norms were always small enough to allow rapid iterative refinement to the level achieved in GEPP.

In our tests the growth of the relative residual norm of the output was limited as the input size grew.

We represented the input matrices M and performed our tests with double precision, but then repeated the computations for the entries of the matrix M truncated to the single precision. In this case double-precision multipication by our multipliers was error-free. The residual norms remained essentially at the same level as for the double precision input, so that the impact of rounding errors at the elimination stage dominated their impact at the stage of multiplication by preconditioners.

7.2 Solution of Hankel linear systems of equations with randomized augmentation

We solved 100 linear systems of equations $M\mathbf{x} = \mathbf{b}$ for vectors $\mathbf{b}$ with random coordinates from the range $[-1, 1)$ and nearly singular Hankel matrices M specified below. This also covered the solution of Toeplitz linear systems of equations due to the equations $\mathbb{H} = J\mathbb{T} = \mathbb{T}J$. We presented the test results in Table 7.3. The column marked **AUGM** shows the CPU time of the solution by an algorithm based on randomized augmentation and specified below. For comparison the columns marked **SVD** and **QR** display the CPU time of the solution based on the SVD and QR factorization. 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. All three algorithms produced the solutions with similar average error bounds of about 10^{-15}.

We performed the tests for the input matrices $M = H + 10^{-9}J$ where $H = JT$ for the $n \times n$ real symmetric Toeplitz matrices T of rank $n-1$ from [PQa, Section 10.1b].

For every input pair M and $\mathbf{b}$ we computed the solution vectors $\mathbf{x} = M^{-1}\mathbf{b}$ by applying the augmentation techniques in Section 6 (with $U = V = \mathbf{v}$). Let us specify these computations first assuming a generic (possibly unstructured) matrix M. Embed it into the $(n+1) \times (n+1)$ matrix $K = \begin{pmatrix} w & \mathbf{v}^T \\ \mathbf{v} & M \end{pmatrix}$ for a scalar w, two random values v_0 and w from the range $[-1, 1)$, and a vector $\mathbf{v} = (v_i)_{i=1}^n$, where $v_i = h_{i-1}$ for $i = 1, \dots, n-1$. Then for the $n \times n$ trailing principal (that is southeastern) submatrix $F = (\mathbf{0}_n, I_n)K^{-1}\begin{pmatrix} \mathbf{0}_n^T \\ I_n \end{pmatrix}$ of the inverse matrix K^{-1} we have $F = (M - \frac{1}{w}\mathbf{v}\mathbf{v}^T)^{-1}$, so that $M = F^{-1} + \frac{1}{w}\mathbf{v}\mathbf{v}^T$. The Sherman–Morrison–Woodbury formula implies that $M^{-1} = F - \frac{1}{g}F\mathbf{v}\mathbf{v}^T F$ for $g = 1 + \frac{1}{w}\mathbf{v}^T F\mathbf{v}$. Therefore $\mathbf{x} = F\mathbf{b} - \frac{1}{g}F\mathbf{v}\mathbf{v}^T F\mathbf{b} = \mathbf{z} - \frac{1}{g}\mathbf{y}\mathbf{v}^T\mathbf{z}$ for $\mathbf{y} = F\mathbf{v}$, $\mathbf{z} = F\mathbf{b}$, $g = 1 + \frac{1}{w}\mathbf{v}^T\mathbf{y}$.

This reduces the solution of the linear system $M\mathbf{x} = \mathbf{b}$ to computing the vectors $\mathbf{y} = F\mathbf{v}$ and $\mathbf{z} = F\mathbf{b}$ and performing $O(n)$ additional flops. To compute the vectors $\mathbf{y}$ and $\mathbf{z}$ we solve two linear systems with the matrix K. Indeed rewrite the products $F\mathbf{v}$ and $F\mathbf{b}$ by substituting the expressions $F = (\mathbf{0}_n, I_n)K^{-1}\begin{pmatrix} \mathbf{0}_n^T \\ I_n \end{pmatrix}$, $\begin{pmatrix} \mathbf{0}_n^T \\ I_n \end{pmatrix}\mathbf{b} = \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$, and $\begin{pmatrix} \mathbf{0}_n^T \\ I_n \end{pmatrix}\mathbf{v} = \begin{pmatrix} 0 \\ \mathbf{v} \end{pmatrix}$ and obtain that $\mathbf{y} = (\mathbf{0}_n, I_n)K^{-1}\begin{pmatrix} 0 \\ \mathbf{v} \end{pmatrix}$ and $\mathbf{z} = (\mathbf{0}_n, I_n)K^{-1}\begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$.

Now recall that in our tests we dealt with real persymmetric Hankel matrices M such that $JMJ = M$, so that $MJ = JM$ are symmetric Toeplitz matrices. We augmented them to obtain real persymmetric Hankel matrices K. This requirement set almost all the entries of the matrix K equal to the respective entries of the block submatrix M, except that we still had freedom to choose any entry v_0 of the vector $\mathbf{v}$. We have chosen that entry at random in the range $[-||M||, ||M||)$. The resulting matrices K were nonsingular in all our tests and the entry $x_{11} = \mathbf{e}_1^T K^{-1}\mathbf{e}_1$ never vanished, as we expected because the $n \times n$ input matrix M was nonsingular and the coordinate v_0 of the vector $\mathbf{v}$ was random. (If a matrix K were singular or if the entry x_{11} vanished, we would have regenerated the value v_0 and would have redefined the matrix K.) Furthermore in our tests we consistently arrived at well conditioned matrices K.

We solved Hankel linear systems with a matrix K by first reducing the task to Toeplitz linear systems with the matrix JK and then applied the Gohberg–Semencul's formula and expresses the

inverse via the solution $K^{-1}J(\mathbf{e}_1, \mathbf{e}_{n+1})$ of two Toeplitz linear systems with the matrix JK. The matrices JK were real symmetric because so were the matrices T that defined the persymmetric Hankel matrices $H = JT$ and $M = H + 10^{-9}J$. It followed that $K^{-1}J\mathbf{e}_{n+1} = JK^{-1}J\mathbf{e}_1$, and so we only needed to solve a single Toeplitz linear system $JK\mathbf{x} = \mathbf{e}_1$. We solved it by applying the algorithm in [VB99], based on the superfast algorithms in [KVB99], [VBHK01], and [VBK98] and incorporated the extended iterative refinement from [PQZb] to compute the solutions with high accuracy.

For comparison we also obtained the solutions of the linear systems $M\mathbf{x} = \mathbf{b}$ based on computing the SVD and QR factorization of the matrix M. In all cases we yielded the output within the error bound of 10^{-15} and displayed the respective data on the CPU running time in the columns QR and SVD, respectively.

7.3 Computation of nmbs

Input Matrices:

We first defined the auxiliary matrices $\widehat{M} = \begin{pmatrix} M_k & A \\ B & C \end{pmatrix}$ where the blocks M_k, A, B, and C were generated as in Section 7.1 but for n replaced by $n = 2^s - r$, $r = 4$. Then we defined the input matrices $M = \begin{pmatrix} \widehat{M} & E \\ G & K \end{pmatrix}$ where the block G was a random $r \times (2^s - r)$ Toeplitz matrix, $\begin{pmatrix} E \\ K \end{pmatrix} = \begin{pmatrix} \widehat{M} \\ G \end{pmatrix} T$, and T was a random $(n - r) \times r$ Toeplitz matrix.

We generated **multiplicative preconditioners** in the same way as in Section 7.1. Tables 7.4 and 7.5 display the results of the computation of nmbs for the matrices M based on GJENP with our randomized structured preconditioning (cf. (4.2)). The same format as in Tables 7.1 and 7.2 is used, except that now the tables display the data on the relative residual norms $\frac{||MB||}{||M||\ ||B||}$.

References

[AHU74] A. V. Aho, J. E. Hopcroft, J. D. Ullman, *The Design and Analysis of Algorithms.* Addison-Wesley, Reading, MA, 1974.

[B85] J. R. Bunch, Stability of Methods for Solving Toeplitz Systems of Equations, *SIAM J. Sci. Stat. Comput.*, **6(2)**, 349–364, 1985.

[BA80] R. R. Bitmead, B. D. O. Anderson, Asymptotically Fast Solution of Toeplitz and Related Systems of Linear Equations, *Linear Algebra and Its Applications*, **34**, 103–116, 1980.

[BEPP97/99] H. Brönnimann, I. Z. Emiris, V. Y. Pan, S. Pion, Sign Determination in Residue Number Systems, *Theoretical Computer Science*, **210**, **1**, 173–197, 1999. Proceedings version in *Proceedings of 13th Annual ACM Symposium on Computational Geometry*, 174–182, ACM Press, New York, 1997.

[CGLX] S. Chandrasekaran, M. Gu, X. S. Li, J. Xia, Superfast Multifrontal Method for Large Structured Linear Systems of Equations, *SIAM. J. on Matrix Analysis and Applications*, **31**, 1382–1411, 2009.

[CGSXZ] S. Chandrasekaran, M. Gu, X. Sun, J. Xia, J. Zhu, A Superfast Algorithm for Toeplitz Systems of linear Equations, *SIAM. J. on Matrix Analysis and Applications* **29**, **4**, 1247–1266, 2007.

Table 7.1: residual norms of the solutions of Toeplitz-like linear systems

multiplier	**size**	**iterations**	**min**	**max**	**mean**	**std**
circulant	32	0	7.8×10^{-15}	1.6×10^{-10}	3.6×10^{-12}	1.8×10^{-11}
circulant	32	1	8.3×10^{-16}	5.7×10^{-12}	7.4×10^{-14}	5.7×10^{-13}
circulant	64	0	5.9×10^{-14}	1.6×10^{-9}	2.4×10^{-11}	1.6×10^{-10}
circulant	64	1	1.7×10^{-15}	7.3×10^{-13}	4.9×10^{-14}	1.2×10^{-13}
circulant	128	0	3.1×10^{-13}	1.9×10^{-8}	3.5×10^{-10}	2.1×10^{-9}
circulant	128	1	5.2×10^{-15}	1.3×10^{-10}	1.6×10^{-12}	1.3×10^{-11}
circulant	256	0	2.7×10^{-12}	3.6×10^{-9}	1.7×10^{-10}	4.6×10^{-10}
circulant	256	1	8.8×10^{-15}	2.8×10^{-12}	1.6×10^{-13}	3.5×10^{-13}
circulant	1024	0	4.0×10^{-10}	3.8×10^{-9}	1.5×10^{-9}	1.5×10^{-9}
circulant	1024	1	1.2×10^{-13}	5.1×10^{-13}	2.3×10^{-13}	1.9×10^{-13}
Householder	32	0	4.9×10^{-11}	1.8×10^{-7}	5.8×10^{-9}	2.0×10^{-8}
Householder	32	1	4.6×10^{-16}	1.1×10^{-13}	6.1×10^{-15}	1.7×10^{-14}
Householder	64	0	3.2×10^{-10}	9.2×10^{-7}	4.1×10^{-8}	1.3×10^{-7}
Householder	64	1	8.3×10^{-16}	1.4×10^{-12}	2.8×10^{-14}	1.5×10^{-13}
Householder	128	0	2.6×10^{-9}	9.9×10^{-6}	2.6×10^{-7}	1.1×10^{-6}
Householder	128	1	1.4×10^{-15}	1.5×10^{-10}	1.5×10^{-12}	1.5×10^{-11}
Householder	128	3	1.5×10^{-15}	1.5×10^{-12}	3.1×10^{-14}	1.5×10^{-13}
Householder	256	0	1.1×10^{-8}	2.7×10^{-5}	8.8×10^{-7}	3.2×10^{-6}
Householder	256	1	2.8×10^{-15}	4.8×10^{-11}	8.2×10^{-13}	5.0×10^{-12}
Householder	256	3	2.8×10^{-15}	1.3×10^{-12}	5.9×10^{-14}	1.8×10^{-13}
Householder	1024	0	5.0×10^{-6}	3.4×10^{-5}	1.6×10^{-5}	1.3×10^{-5}
Householder	1024	1	1.6×10^{-12}	1.2×10^{-11}	6.7×10^{-12}	4.7×10^{-12}
Householder	1024	3	1.9×10^{-14}	1.7×10^{-13}	7.1×10^{-14}	6.5×10^{-14}

[CPV04] G. Codevico, V. Y. Pan, M. Van Barel, Newton-like Iteration Based on Cubic Polynomials for Structured Matrices, *Numerical Algorithms*, **36**, 365-380, 2004.

[CPW74] R.E. Cline, R.J. Plemmons, and G. Worm, Generalized inverses of certain Toeplitz matrices. *Linear Algebra and Its Applications,* **8**, 25–33, 1974.

[CW90] Coppersmith, S. Winograd, Matrix Multiplicaton via Arithmetic Progressions. *J. Symbolic Comput.*, **9(3)**, 251–280, 1990.

[D88] J. Demmel, The Probability That a Numerical Analysis Problem Is Difficult, *Math. of Computation*, **50**, 449–480, 1988.

[DL78] R. A. Demillo, R. J. Lipton, A Probabilistic Remark on Algebraic Program Testing, *Information Processing Letters*, **7**, **4**, 193–195, 1978.

[DS01] 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.

[E88] A. Edelman, Eigenvalues and Condition Numbers of Random Matrices, *SIAM J. on Matrix Analysis and Applications*, **9**, **4**, 543–560, 1988.

[EP03/05] I. Z. Emiris, V. Y. Pan, Improved Algorithms for Computing Determinants and Resultants, *J. of Complexity*, **21**, **1**, 43–71, 2005. Proceedings version in *Proceedings of 6th International Workshop on Computer Algebra in Scientific Computing (CASC '03)*, E. W. Mayr, V. G. Ganzha, E. V. Vorozhtzov (editors), 81–94, Technische Univ. München, Germany, 2003.

Table 7.2: residual norms of the solutions of linear systems in the case of general matrices M_k

multiplier	**size**	**iterations**	**min**	**max**	**mean**	**std**
circulant	32	0	9.1×10^{-15}	5.4×10^{-12}	5.7×10^{-13}	1.1×10^{-12}
circulant	32	1	1.1×10^{-15}	7.8×10^{-14}	9.7×10^{-15}	1.4×10^{-14}
circulant	64	0	4.7×10^{-14}	8.0×10^{-11}	4.0×10^{-12}	1.1×10^{-11}
circulant	64	1	1.9×10^{-15}	5.3×10^{-13}	2.3×10^{-14}	5.4×10^{-14}
circulant	128	0	2.8×10^{-13}	2.1×10^{-10}	1.6×10^{-11}	3.1×10^{-11}
circulant	128	1	4.3×10^{-15}	1.6×10^{-12}	6.6×10^{-14}	1.8×10^{-13}
circulant	256	0	1.7×10^{-12}	1.4×10^{-7}	2.0×10^{-9}	1.5×10^{-8}
circulant	256	1	8.3×10^{-15}	4.3×10^{-10}	4.5×10^{-12}	4.3×10^{-11}
circulant	1024	0	1.7×10^{-10}	4.4×10^{-9}	1.4×10^{-9}	2.1×10^{-9}
circulant	1024	1	3.4×10^{-14}	9.9×10^{-14}	6.8×10^{-14}	2.7×10^{-14}
Householder	32	0	5.5×10^{-15}	2.3×10^{-11}	1.0×10^{-12}	3.1×10^{-12}
Householder	32	1	4.1×10^{-16}	1.8×10^{-13}	4.5×10^{-15}	1.8×10^{-14}
Householder	64	0	2.9×10^{-14}	1.8×10^{-10}	3.6×10^{-12}	1.9×10^{-11}
Householder	64	1	5.8×10^{-16}	3.6×10^{-13}	9.0×10^{-15}	3.8×10^{-14}
Householder	128	0	1.2×10^{-13}	9.1×10^{-10}	2.4×10^{-11}	1.0×10^{-10}
Householder	128	1	1.2×10^{-15}	4.8×10^{-13}	1.9×10^{-14}	6.0×10^{-14}
Householder	256	0	1.1×10^{-12}	3.2×10^{-8}	4.5×10^{-10}	3.2×10^{-9}
Householder	256	1	2.0×10^{-15}	6.4×10^{-13}	2.7×10^{-14}	8.2×10^{-14}
Householder	1024	0	3.2×10^{-11}	2.7×10^{-9}	8.6×10^{-10}	1.3×10^{-9}
Householder	1024	1	1.6×10^{-14}	9.5×10^{-14}	4.3×10^{-14}	3.7×10^{-14}

Table 7.3: CPU time (in CPU cycles) for solving ill conditioned Hankel linear systems of equations

size	**AUGM**	**SVD**	**QR**	**SVD/AUGM**	**QR/AUGM**
512	29.3	5160.43	149.4	175.9	5.1
1024	101.4	76853.6	2014.7	757.9	19.9
2048	250.1	–	18021.2	–	72.0
4096	453.2	–	–	–	–
8192	779.1	–	–	–	–

[GI88] W. Gautschi, G. Inglese, Lower Bounds for the Condition Number of Vandermonde Matrices, *Numerische Math.*, **52**, 241–250, 1988.

[GKO95] I. Gohberg, T. Kailath, V. Olshevsky, Fast Gaussian Elimination with Partial Pivoting for Matrices with Displacement Structure, *Math. of Computation*, **64**, 1557-1576, 1995.

[GL96] G. H. Golub, C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, 1996 (third addition).

[H02] N. J. Higham, *Accuracy and Stability in Numerical Analysis*, SIAM, Philadelphia, 2002 (second edition).

[KVB99] P. Kravanja, M. Van Barel, Algorithms for Solving Rational Interpolation Problems Related to Fast and Superfast Solvers for Toeplitz Systems, *SPIE*, 359–370, 1999.

[LPS92] J. Laderman, V. Y. Pan, H. X. Sha, On Practical Algorithms for Accelerated Matrix Multiplication, *Linear Algebra and Its Applications*, **162–164**, 557–588, 1992.

Table 7.4: residual norms of approximate nmbs in the case of Toeplitz-like matrices M_k

multiplier	**size**	**iterations**	**min**	**max**	**mean**	**std**
circulant	32	0	4.4×10^{-13}	6.0×10^{-9}	2.0×10^{-10}	7.9×10^{-10}
circulant	32	1	4.4×10^{-17}	2.3×10^{-15}	3.2×10^{-16}	3.9×10^{-16}
circulant	64	0	6.1×10^{-13}	6.2×10^{-9}	2.3×10^{-10}	8.9×10^{-10}
circulant	64	1	3.9×10^{-17}	3.1×10^{-15}	3.0×10^{-16}	4.2×10^{-16}
circulant	128	0	1.9×10^{-13}	3.8×10^{-9}	1.3×10^{-10}	4.7×10^{-10}
circulant	128	1	3.6×10^{-17}	2.5×10^{-15}	3.7×10^{-16}	4.2×10^{-16}
circulant	256	0	1.7×10^{-12}	1.4×10^{-8}	3.8×10^{-10}	1.8×10^{-9}
circulant	256	1	2.1×10^{-17}	2.1×10^{-14}	5.8×10^{-16}	2.2×10^{-15}
circulant	1024	0	9.3×10^{-12}	5.4×10^{-11}	3.2×10^{-11}	2.2×10^{-11}
circulant	1024	1	1.7×10^{-16}	2.2×10^{-16}	2.0×10^{-16}	2.2×10^{-17}
circulant	1024	3	6.2×10^{-17}	4.7×10^{-16}	2.0×10^{-16}	1.8×10^{-16}
Householder	32	0	3.5×10^{-15}	2.6×10^{-5}	2.7×10^{-7}	2.6×10^{-6}
Householder	32	1	1.1×10^{-20}	1.9×10^{-9}	1.9×10^{-11}	1.9×10^{-10}
Householder	32	3	1.9×10^{-20}	5.5×10^{-15}	2.4×10^{-16}	6.6×10^{-16}
Householder	64	0	3.3×10^{-14}	4.3×10^{-7}	1.4×10^{-8}	5.1×10^{-8}
Householder	64	1	1.7×10^{-20}	8.1×10^{-15}	5.7×10^{-16}	1.4×10^{-15}
Householder	128	0	7.5×10^{-14}	4.1×10^{-7}	1.5×10^{-8}	4.7×10^{-8}
Householder	128	1	1.7×10^{-20}	4.2×10^{-14}	1.1×10^{-15}	4.8×10^{-15}
Householder	256	0	9.7×10^{-14}	1.4×10^{-6}	3.7×10^{-8}	1.6×10^{-7}
Householder	256	1	3.5×10^{-20}	1.6×10^{-13}	3.4×10^{-15}	1.7×10^{-14}
Householder	1024	0	1.8×10^{-11}	1.2×10^{-6}	3.3×10^{-7}	5.6×10^{-7}
Householder	1024	1	6.0×10^{-18}	1.3×10^{-12}	3.2×10^{-13}	6.4×10^{-13}
Householder	1024	3	1.5×10^{-19}	1.0×10^{-14}	3.2×10^{-15}	4.6×10^{-15}

[M80] M. Morf, Doubling Algorithms for Toeplitz and Related Equations, *Proceedings of IEEE International Conference on ASSP*, 954–959, IEEE Press, Piscataway, New Jersey, 1980.

[P90] V. Y. Pan, On Computations with Dense Structured Matrices, *Math. of Computation*, **55(191)**, 179–190, 1990. Proceedings version in *Proc. of Intern. Symposium on Symbolic and Algebraic Computation (ISSAC89)*, 34-42, ACM Press, New York, 1989.

[P92] V. Y. Pan, Parallel Solution of Toeplitz-like Linear Systems, *J. of Complexity*, **8**, 1–21, 1992.

[P93a] V. Y. Pan, Concurrent Iterative Algorithm for Toepliz-like Linear Systems, *IEEE Transactions on Parallel and Distributed Systems*, **4, 5**, 592–600, 1993.

[P93b] V. Y. Pan, Decreasing the Displacement Rank of a Matrix, *SIAM Journal on Matrix Analysis and Applications*, **14, 1**, 118–121, 1993.

[P01] V. Y. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms*, Birkhäuser/Springer, Boston/New York, 2001.

[P10a] F. Poloni, A Note on the O(n)-Storage Implementation of the GKO Algorithm, *Numerical Algorithms*, 2010.

[P10] V. Y. Pan, Newton's Iteration for Matrix Inversion, Advances and Extensions, in *Matrix methods: Theory, Algorithms and Applications* (dedicated to the Memory of Gene H. Golub, edited by Vadim Olshevsky and Eugene Tyrtyshnikov), pages 364–381, World Scientific Publishing, 2010.

Table 7.5: residual norms of approximate nmbs in the case of general matrices M_k

multiplier	**size**	**iterations**	**min**	**max**	**mean**	**std**
circulant	32	0	6.4×10^{-13}	8.0×10^{-9}	2.1×10^{-10}	9.9×10^{-10}
circulant	32	1	4.5×10^{-17}	1.8×10^{-15}	2.3×10^{-16}	2.6×10^{-16}
circulant	64	0	9.4×10^{-13}	5.3×10^{-9}	2.0×10^{-10}	7.5×10^{-10}
circulant	64	1	2.4×10^{-17}	4.1×10^{-15}	3.9×10^{-16}	5.9×10^{-16}
circulant	128	0	1.1×10^{-12}	2.3×10^{-9}	1.2×10^{-10}	3.0×10^{-10}
circulant	128	1	3.6×10^{-17}	3.8×10^{-15}	3.7×10^{-16}	6.1×10^{-16}
circulant	256	0	2.2×10^{-12}	5.3×10^{-8}	9.8×10^{-10}	6.1×10^{-9}
circulant	256	1	4.4×10^{-17}	6.2×10^{-15}	4.8×10^{-16}	8.6×10^{-16}
circulant	1024	0	1.0×10^{-11}	1.1×10^{-9}	3.3×10^{-10}	5.4×10^{-10}
circulant	1024	1	1.9×10^{-16}	6.3×10^{-16}	3.5×10^{-16}	2.0×10^{-16}
Householder	32	0	1.8×10^{-16}	3.3×10^{-11}	6.4×10^{-13}	3.4×10^{-12}
Householder	32	1	5.7×10^{-21}	1.8×10^{-17}	6.1×10^{-19}	2.3×10^{-18}
Householder	64	0	1.7×10^{-16}	2.9×10^{-11}	6.3×10^{-13}	3.2×10^{-12}
Householder	64	1	5.6×10^{-21}	5.7×10^{-17}	1.5×10^{-18}	6.7×10^{-18}
Householder	128	0	2.4×10^{-16}	9.9×10^{-11}	1.5×10^{-12}	1.0×10^{-11}
Householder	128	1	3.5×10^{-21}	6.9×10^{-17}	2.3×10^{-18}	7.6×10^{-18}
Householder	256	0	5.2×10^{-16}	5.8×10^{-11}	1.1×10^{-12}	5.9×10^{-12}
Householder	256	1	3.0×10^{-21}	9.4×10^{-17}	4.0×10^{-18}	1.1×10^{-17}
Householder	1024	0	6.3×10^{-14}	1.2×10^{-12}	5.6×10^{-13}	4.5×10^{-13}
Householder	1024	1	1.9×10^{-18}	4.1×10^{-17}	1.9×10^{-17}	1.8×10^{-17}

[PBRZ99] V. Y. Pan, S. Branham, R. Rosholt, A. Zheng, Newton's Iteration for Structured Matrices and Linear Systems of Equations, *SIAM volume on Fast Reliable Algorithms for Matrices with Structure* (edited by T. Kailath and A. H. Sayed), 189–210, SIAM Publications, Philadelphia, 1999.

[PGMQ] 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, and J. Verschelde editors), **409**, **2**, 255–268, 2008.

[PQ10] V. Y. Pan, G. Qian, Randomized Preprocessing of Homogeneous Linear Systems of Equations, *Linear Algebra and Its Applications*, **432**, 3272–3318, 2010.

[PQa] V. Y. Pan, G. Qian, Solving Linear Systems with Randomized Augmentation, Technical Reports TRs 2009009 and 2010009, *Ph.D. Program in Computer Science*, *Graduate Center, the City University of New York*, 2009 and 2010.

Available at http://www.cs.gc.cuny.edu/tr/techreport.php?id=352

[PQZb] V. Y. Pan, G. Qian, A. Zheng, Randomized Preconditioning of Linear Systems of Equations, Technical Report TR 2010012, *Ph.D. Program in Computer Science*, *Graduate Center, the City University of New York*, 2010.

Available at http://www.cs.gc.cuny.edu/tr/techreport.php?id=352

[PQZC] V. Y. Pan, G. Qian, A. Zheng, Z. Chen, Matrix Computations and Polynomial Root-finding with Preprocessing, *Linear Algebra and Its Applications*, in print.

[PRW02] V. Y. Pan, Y. Rami, X. Wang, Structured Matrices and Newtons Iteration: Unified Approach, *Linear Algebra and Its Applications*, **343/344**, 233–265, 2002.

[PVWC04] V. Y. Pan, M. Van Barel, X. Wang, G. Codevico, Iterative Inversion of Structured Matrices, *Theoretical Computer Science*, **315**, **2–3** (Special Issue on Algebraic and Numerical Computing), 581–592, 2004.

[PW08] V. Y. Pan, X. Wang, Degeneration of Integer Matrices Modulo an Integer, *Linear Algebra and Its Applications*, **429**, 2113–2130, 2008.

[PY99/01] V. Y. Pan, Y. Yu, Certified Computation of the Sign of a Matrix Determinant, *Algorithmica*, **30**, 708–724, 2001; Proc. version in *Proc. 10th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'99)*, 715–724, ACM Press, New York, and SIAM Publications, Philadelphia, 1999.

[PZa] V. Y. Pan, A.-L. Zheng, Real and Complex Polynomial Root-Finding with Eigen-Solving and Preprocessing, *Proc. International Symp. on Symbolic and Algebraic Computation (ISSAC 2010)*, ACM Press, New York (July 2010).

[R06] G. Rodriguez. Fast solution of Toeplitz- and Cauchy-like least squares problems, *SIAM J. Matrix Analysis and Applications*, **28, 3**, 724–748, 2006.

[S80] J. T. Schwartz, Fast Probabilistic Algorithms for Verification of Polynomial Identities, *Journal of ACM*, **27**, **4**, 701–717, 1980.

[S98] G. W. Stewart, *Matrix Algorithms, Vol I: Basic Decompositions*, SIAM, Philadelphia, 1998.

[SST06] 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.

[ST02] D. Spielman, S.-H. Teng, Smoothed Analysis of Algorithms, *Proc. of the International Congress of Mathematicians* (Beijing 2002), Vol. I, 597–606, Higher ED. Press, Beijing, 2002.

[VB99] M. Van Barel, A Supefast Toeplitz Solver, 1999.
Available at http://www.cs.kuleuven.be/ marc/software/index.html

[VBHK01] 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.

[VBK98] M. Van Barel, P. Kravanja, A Stabilized Superfast Solver for Indefinite Hankel Systems, *Linear Algebra and its Applications*, **284**, **1–3**, 335–355, 1998.

[W04] M. Wschebor, Smoothed Analysis of $\kappa(a)$, *J. of Complexity*, **20**, 97–107, 2004.

[W07] X. Wang, Affect of Small Rank Modification on the Condition Number of a Matrix, *Computer and Math. (with Applications)*, **54**, 819–825, 2007.

[Z79] R. E. Zippel, Probabilistic Algorithms for Sparse Polynomials, *Proceedings of EUROSAM'79, Lecture Notes in Computer Science*, **72**, 216–226, Springer, Berlin, 1979.