Algorithmica (2001) 30: 708–724
DOI: 10.1007/s00453-001-0032-8

Algorithmica

Certification of Numerical Computation of the Sign of the Determinant of a Matrix[1]

V. Y. Pan[2] and Y. Yu[3]

Abstract. Certified computation of the sign of the determinant of a matrix is a central problem in computational geometry. Certification by known methods is practically difficult because the magnitude of the determinant of an integer input matrix A may vary dramatically, from 1 to $\|A\|^n$, and the round-off error bound of the determinant computation varies proportionally. Because of such a variation, high precision computation of $\det A$ is required to ensure that the error bound is smaller than the magnitude of the determinant. We observe, however, that our certification task of determining only a single bit of $\det A$, that is, the bit carrying the sign, does not require us to estimate the latter round-off error. Instead, we solve a much simpler task of computing numerically the factorization of a matrix by Gaussian elimination with pivoting (which is a subroutine of LAPACK and LINPACK) and of estimating the minimum distance $1/\|A^{-1}\|$ from A to a singular matrix. Such an estimate gives us a desired range for the round-off error of the factorization such that the invariance of the sign of $\det A$ is ensured as long as the error varies in this range. Based on these simple but novel observations, we devise new effective arithmetic filters for the certification of the sign, compare them with the known filters, and confirm the efficiency of our techniques by some numerical tests.

Key Words. Sign of matrix determinant, Arithmetic filters, Certified geometric computations, Convex hull, Distance to a singular matrix.

1. Introduction

1.1. *The Problem and the Background.* The classical problem of computing $\det A$, the determinant of an $n \times n$ matrix A, has a long history (see, e.g., [M1], [M2], [D1], [F], [R], [E1], [B], [P], and [BP]). Recently, it turned out that some of the most fundamental problems of computational geometry (such as the computation of convex hulls and Voronoi diagrams, of the orientation of a polyhedron or an algebraic variety, and of the arrangement of lines and line segments) are reduced to the computation of $\det A$ or, more precisely, its sign, that is, testing whether $\det A = 0$, $\det A > 0$, or $\det A < 0$ [BEPP1], [BEPP2], [BKM$^+$], [EC], [E2], [FvW1], [Y], [YD].

In many areas of computational geometry, lower-dimensional problems must be solved, and then n ranges between 2 and 10, usually staying below 5. In this class of applications, the matrix A is filled with "long" numbers, representing the real data

[1] The results of this paper have been presented at the Annual ACM–SIAM Symposium on Discrete Algorithms (SODA 99), Baltimore, MD, January 1999. This research was supported by NSF Grants CCR 9625344 and CCR 9732206 and PSC CUNY Awards 668365 and 669363.
[2] Department of Mathematics and Computer Science, Lehman College, CUNY, Bronx, NY 10468, USA. vpan@lehman.cuny.edu.
[3] Ph.D. Program in Computer Science, CUNY, New York, NY 10036, USA.

Received March 29, 1998; revised December 14, 1998. Communicated by G. F. Italiano.
Online publication April 30, 2001.

with a high precision (and thus allowing us to treat the important case of a nearly singular input). In another major class of applications [DL1], [DL2], [ES], [AF], [FR], [MA], n is large (say, in the range from 100 to 500), whereas the matrix A is filled with relatively short integers (say, represented with 5–10 bits). Such applications include the computation of the orientation of a polyhedron or an algebraic variety in a high-dimensional space (for instance, such computations are required in the area of convex optimization in statistical physics and chemistry).

In both cases we may apply the well-known methods to compute $\det A$ based on the triangular ($PLUP_1$) or orthogonal (QR) factorization of the matrix A. The high speed of these computations is ensured as they are performed numerically, with a fixed (single or double) precision, which currently has much faster computer implementation than rational, integer, and multiple precision arithmetic. The major problem, however, is to certify that the output is correct in the presence of round-off errors.

To overcome this problem, one can compute $\det A$ by using symbolic techniques for computations with no errors, even though floating point numerical computations with round-off to single precision are performed faster in the present day computer environment. In symbolic setting, $\det A$ can be computed by Krylov space methods [W], [KP1], [KP2], [BP], [K2], which are particularly efficient where A is a sparse matrix and which have good parallel implementation. The first subroutine specialized for certified computation of the sign of $\det A$ (written as a part of a convex hull computation routine) was due to Clarkson [C]. His paper [C] was a substantial advance in this area, though the correctness certification of the output of the proposed algorithm (based on the modified Gram–Schmidt method) complicated and slowed down the computation. The algorithm in [ABD$^+$] competes with the one in [C] for $n \leq 4$ but does not work well for larger n.

Recent progress reported in [BEPP1] and [BEPP2] relies on using a symbolic algorithm that computes $\det A$ modulo several primes $p_1, \ldots, p_k$ such that their product exceeds $|\det A|$. The papers propose effective algorithms for the recovery of the sign of $\det A$ from these data, based on some novel application and extensions of the Chinese remainder algorithm, which in [BEPP1] and [BEPP2] reduce to single or double precision computation. The bottleneck of the algorithms is the relatively expensive computation of $(\det A) \bmod m_i$, for $i = 1, \ldots, k$. The number k of the pairwise relatively prime moduli m_i involved and, consequently, the computational cost decrease if $|\det A|$ is shown to be smaller. The subsequent recovery of the sign of $\det A$ uses numerical computations with round-off to single precision, but this stage is not the bottleneck.

With the latter minor exception, all the cited algorithms assume exact integer computations with no round-off errors. Such computations start with the artificial trick of conversion of the input data into integers and are well known to be slower by orders of magnitude than the IEEE 754 standard floating point computations [IEEE]. The problem with the floating point computation of the sign of $\det A$ is in the potential corruption of the result due to the round-off errors and, therefore, in the need for the tight output error estimate. Namely, if the computed approximation, d^*, to $\det A$ and an available upper estimate, e_d^+, for the error satisfy $|d^*| > e_d^+$, then it is certified that

$$\operatorname{sign} d^* = \operatorname{sign}(\det A), \tag{1.1}$$

that is, $\operatorname{sign}(\det A)$ is computed correctly. Otherwise, the result of the computations is not certified, and one must repeat numerical computations with a higher precision or

shift to exact integer or rational computations. The typical policy is to start with lower precision numerical computations, which should produce certified solutions for most of the input instances, and then one repeats the computations with increased precision only in relatively rare cases where the certification fails. The process stops as soon as the certified solution is obtained. Such an approach, called *arithmetic filtering*, has been successfully applied to the computation of several algebraic and geometric predicates, with the sign of $\det A$ being the most celebrated example [ABD$^+$], [BKM$^+$], [FvW2], [PYS], [YD]. The major technical problem in this area is in obtaining sharper round-off error bounds for the output of floating point computations.

In the case of the computation of $\det A$, however, the available techniques typically produce overestimated error bounds. The algorithms very frequently report failure even where (1.1) actually holds true, that is, where $\text{sign}(\det A)$ is computed correctly. Such a phenomenon occurs because the range for the values of $|\det A|$ is huge (from 0 to Hadamard's determinant bound D^*, which can be as large as $\|A\|^n$), and the known techniques of error analysis only guarantee round-off error bounds of order $D^* n^2 \epsilon$, ϵ being the *machine epsilon* (also called *unit round-off*). Such bounds can be large even where $|\det A|$ is actually small (see Section 8.3). Therefore, the round-off error bounds for computing $\det A$ may exceed the value of $|\det A|$ substantially.

1.2. *Our Results.* In this paper we propose some simple and effective but (surprisingly) novel techniques for the certification of the sign of $\det A$ computed numerically. Our main algorithm, Algorithm 4.1, based on these techniques, either certifies that the sign of $\det A$ has been computed correctly or shows which increase of the precision of computing should yield the certification. If the required increase of the precision is too large to be handled by the available computer support, then usually the computed upper bound $|d^*| + e_d$ on $|\det A|$ is relatively small, and the transition to the symbolic approach is appropriate. Motivated by this observation, we comment (in Sections 4 and 7) on how to estimate from above $|\det A|$, which is an input value for the latter approach.

All computations required in the resulting algorithms for $\text{sign}(\det A)$ (including ones applied at the certification stage) essentially amount to the invocation of some simple and widely available subroutines of LAPACK, LINPACK, and MATLAB for matrix computations (see [GL]), which we select and combine in an appropriate order (see Sections 4, 6, and 7). A major step of our algorithm is the estimation of the distance to a closest singular matrix, and we describe several known techniques for this task, depending, in particular, on allowing or not allowing randomization. We also briefly examine modifications of our approach in Section 8.

The major technical novelty behind our improvement versus that in [PYS] is the certification of the sign of $\det A$ without estimating the round-off error of computing $\det A$. Instead, we estimate the minimum distance from the matrix $\tilde{L}\tilde{U}$ to a singular matrix, $\tilde{L}$ and $\tilde{U}$ being the computed approximations to the factors L and U of the triangular ($PLUP_1$) factorization of A. By a well-known theorem [EY], [K1], this distance is equal to $1/N$, for $N = \|\tilde{U}^{-1}\tilde{L}^{-1}\|$.

Due to the celebrated techniques of backward error analysis, the round-off errors of computing $\tilde{L}$ and $\tilde{U}$ can be equivalently represented by the errors caused by perturbation of A by some matrix E whose norm is easily estimated from above. On the other hand, we deduce easily that $\det(A + E)$ does not change its sign when E ranges in the ball

centered in the origin and having a radius smaller than the distance from A to a closest singular matrix, that is, smaller than $1/N = 1/\|\tilde{U}^{-1}\tilde{L}^{-1}\|$. Now, since the matrices $\tilde{L}$ and $\tilde{U}$ are two available triangular matrices, it is not hard to obtain a certified and quite tight upper bound on N.

As mentioned, n is large in a large class of applications to geometric and algebraic computations, and the papers on the sign of determinant usually include asymptotic estimates in terms of n for the arithmetic complexity of the proposed algorithms, even though such estimates are only relevant for large n. For our algorithms, the worst case estimates are $O(n^3)$, the same as the experimental ones of [C] (whereas the worst case bound in [C] has a larger order of $n^3 \log n$). To our advantage versus [C] and other cited works, however, we do not need to stay with the (rational or integer) infinite precision computations and may exploit faster numerical computations with finite precision. Another advantage of our approach (particularly over [C] and [ABD$^+$]) is that the subroutines for matrix computations employed in our algorithms have very efficient implementation on parallel computers. The results of our experiments (reported in Appendix C) show that our arithmetic filter is clearly superior to the one in [PYS], specialized for the computation of det A. (Since the advantages of floating point finite precision computations are well known, our tests had a limited goal of the comparison of our new filter with the ones of [PYS].)

1.3. *The Order of Our Presentation.* We present our results in the following order. In the next section we state some definitions. In Section 3 we recall some known estimates for the errors of Gaussian elimination due to round-off. In Section 4 we relate the certification of the sign of det A in the presence of round-off errors to the minimum distance from A to a singular matrix. Then we propose our main algorithm for computing and certifying the sign of det A based on this relation. In Section 5 we show how to use the computed information in the case where the algorithm does not produce a certified correct output. In Section 6 we elaborate the stage of estimating the minimum distance to a singular matrix, which is a major block of our algorithm of Section 4. In Section 7 we complement the algorithm by presenting some techniques for computing or estimating from above $|\det A|$. In Section 8 we comment on some variations of our approach and some alternatives. In particular, we indicate some reasons for the deficiency of an approach in [PYS] and of one based on the Barrlund and Sun theorem; we also point out the modifications of our approach that use Gauss–Jordan, $PLDM^{\mathrm{T}}P_1$ and QR (rather than $PLUP_1$) factorizations of A. In the appendixes we recall some techniques for computing $\tilde{U}^{-1}\tilde{L}^{-1}$, complement the approach in [PYS] by using the LDL^{T} factorization of $A^{\mathrm{T}}A$, and present the results of our numerical experiments.

Sections 1–8 and Appendixes A and B, are due to the first author, Appendix C on numerical tests is due to both authors.

2. Some Definitions. Hereafter, $\mathbf{A}_k$ denotes the kth column vector of a matrix A. $w_{i,j}$ denotes the (i, j)th entry of a matrix $W = (w_{i,j})$. $\|\cdot\| = \|\cdot\|_h$ denotes a fixed operator matrix norm, in particular we use the 2-norm $\|\cdot\|_2$, the row norm $\|W\|_\infty = \max_i \sum_j |w_{i,j}|$, and the column norm $\|W\|_1 = \max_j \sum_i |w_{i,j}|$ for a matrix $W = (w_{i,j})$ (see [GL] or [H]). The transpose of W is denoted $W^{\mathrm{T}} = (w_{j,i})$, whereas $|W|$ denotes

the matrix $(|w_{i,j}|)$. We write $|W| \le |V|$ iff $V = (v_{i,j})$ and $|w_{i,j}| \le |v_{i,j}|$ for all i and j. I denotes the $n \times n$ identity matrix. $\operatorname{diag}(w_{i,i})$ denotes the diagonal matrix with the diagonal entries $w_{i,i}$, $i = 1, \ldots, n$. Det W and $\operatorname{sign}(\det W)$ denote the determinant of a square matrix W and its sign, respectively. A triangular matrix will be called *unit* (respectively, *proper*) triangular if its diagonal is filled with ones (respectively, zeros).

3. Round-Off Errors of Matrix Factorization by Gaussian Elimination. Det A and its sign for a given matrix A can be immediately obtained from the triangular ($PLUP_1$) factorization of A, but the problem is to analyze the effect of the round-off errors when the factorization is computed numerically. In this analysis we apply some known estimates, which we recall in this section.

THEOREM 3.1 (see Theorem 9.3 on p. 175 of [H]). *Let*

$$A = PA'P_1, \qquad \tilde{A} = A' + E = \tilde{L}\tilde{U}, \tag{3.1}$$

where A, A', $\tilde{A}$, E, P, P_1, $\tilde{L}$, and $\tilde{U}$ are $n \times n$ matrices, P and P_1 are permutation matrices, $\tilde{L} = (\tilde{l}_{i,j})$ is a unit lower triangular matrix (so that $\det \tilde{L} = 1$), and $\tilde{U} = (\tilde{u}_{i,j})$ is an upper triangular matrix, $\tilde{L}$ and $\tilde{U}$ are computed numerically, with unit round-off ϵ (machine epsilon), by means of Gaussian elimination (with complete pivoting) applied to the matrix A. Then

$$|E| \le \gamma_n |\tilde{L}| \cdot |\tilde{U}|, \qquad \gamma_n = n\epsilon/(1 - n\epsilon).$$

Two special cases of this result cover Gaussian elimination with partial pivoting (for $P_1 = I$) and with no pivoting (for $P_1 = P = I$).

By (3.1) we have

$$\det A = (\det P)(\det A') \det P_1, \tag{3.2}$$

$$\det \tilde{A} = \det \tilde{U} = \tilde{u}_{1,1}\tilde{u}_{2,2} \cdots \tilde{u}_{n,n}. \tag{3.3}$$

Since $\det P$ and $\det P_1$ are readily available (they equal 1 or -1), (3.2) and (3.3) define $\operatorname{sign}(\det A)$ provided that the diagonal entries of $\tilde{U}$ are available and that ϵ is sufficiently small to guarantee that

$$\operatorname{sign}(\det A') = \operatorname{sign}(\det \tilde{A}). \tag{3.4}$$

In the next section we show how to verify (3.4) by using the following corollary of Theorem 3.1.

COROLLARY 3.1. *Under the assumptions of Theorem* 3.1, *we have*

$$\|E\| \le e = \|(|\tilde{L}| \cdot |\tilde{U}|)\|\gamma_n \tag{3.5}$$

for any fixed operator matrix norm.

The computation of the row and column norms of $|\tilde{L}| \cdot |\tilde{U}|$ involves only $O(n^2)$ arithmetic operations because $\|(|\tilde{L}| \cdot |\tilde{U}|)\|_\infty = \|(|\tilde{L}|(u_i))\|_\infty$, $\|(|\tilde{L}| \cdot |\tilde{U}|)\|_1 = \|((l_j)^T |\tilde{U}|)\|_1$, where (l_j) and (u_i) are two vectors with the components $l_j = \sum_i |\tilde{l}_{i,j}|$ and $u_i = \sum_j |\tilde{u}_{i,j}|$.

By Corollary 3.1, the rounding error of the computation of the $PLUP_1$ factorization of A is bounded in terms of γ_n and the norm of the matrix $|\tilde{L}| \cdot |\tilde{U}|$. It is known that in the case of using complete pivoting, the (k, j)th entries $\tilde{a}_{k,j}$, $\tilde{l}_{k,j}$, and $\tilde{u}_{k,j}$ of the matrices $\tilde{A}$, $\tilde{L}$, and $\tilde{U}$ of (3.1), respectively, satisfy the bounds

$$|\tilde{l}_{k,j}| \le 1, \qquad |\tilde{u}_{k,j}| \le \rho_k \max_{g,h} |\tilde{a}_{g,h}|,$$

for all k and j, where $\rho_k \le k^{1/2}(2 \cdot 3^{1/2} \cdots k^{1/(k-1)})^{1/2} \le 1.8k^{(\ln k)/4}$ (see p. 119 of [GL]). The same bounds are known in the case of partial pivoting, but theoretically only for $\rho_k \le 2^{k-1}$. In practice, however, $\rho_k = O(k)$ even in the case of using partial pivoting (see p. 116 of [GL]).

4. Certification of the Sign in Terms of the Smallest Distance to a Singular Matrix. Here is a sufficient condition for (3.4),

$$|\det \tilde{U}| = |\det \tilde{A}| > |(\det A) - \det(P\tilde{A}P_1)| = e_d.$$

For a large class of matrices A and sufficiently small machine epsilon, the latter condition can be verified based on the straightforward crude estimate:

$$e_d \le n^2 e_+ D_+,$$

where e_+ denotes the maximum absolute value of the entries of E, and $D_+ = \prod_{k=1}^n (\|\mathbf{A}_k\|_2 + ne_+)$. This estimate is based on Hadamard's bound,

$$|\det A| \le \prod_{k=1}^n \|\mathbf{A}_k\|_2 \tag{4.1}$$

(see p. 287 of [H]). An upper bound $e^* = \gamma_n \max_{i,j}(|\tilde{L}| \cdot |\tilde{U}|)_{i,j}$ on e_+ is implied by Theorem 3.1. Here $(|\tilde{L}| \cdot |\tilde{U}|)_{i,j}$ denotes the (i, j)th entry of the matrix $|\tilde{L}| \cdot |\tilde{U}|$. Then we obtain the following estimate:

$$e_d \le e_d^+ = D^+ n^2 e^*, \qquad D^+ = \prod_{k=1}^n (\|\mathbf{A}_k\|_2 + ne^*). \tag{4.2}$$

Our more refined techniques for the verification of (3.4) rely on the next two results.

THEOREM 4.1. *For two given matrices W and $W + \Delta$, for a fixed matrix norm $\|\cdot\|$, and for all singular matrices S, let*

$$\max\{\|W - S\|, \|W + \Delta - S\|\} > \|\Delta\|. \tag{4.3}$$

Then

$$\operatorname{sign}(\det W) = \operatorname{sign}(\det(W + \Delta)). \tag{4.4}$$

PROOF. Unless (4.4) holds, $S = W + t\Delta$ is a singular matrix for some real t, $0 \le t \le 1$. Clearly, for such a matrix S, we have

$$\|W - S\| = t\|\Delta\| \le \|\Delta\|,$$

$$\|W + \Delta - S\| = (1 - t)\|\Delta\| \le \|\Delta\|,$$

and (4.3) is violated. □

The next theorem was first proved by Eckart and Young, in 1939 [EY], in the special case of the matrix norm $\|\cdot\| = \|.\|_2$ and much later by Gastinel (see [K1]), in the general case of any operator matrix norm $\|\cdot\|$.

THEOREM 4.2. *For any fixed nonsingular matrix W and any fixed operator matrix norm $\|\cdot\|$, we have $1/\min\|W - S\| = \|W^{-1}\|$, where the minimum is over all singular matrices S.*

Combining Theorems 4.1 and 4.2 implies the next result.

COROLLARY 4.1. *Under the assumptions of Theorem* 4.1, *if* $1/\min\{\|W^{-1}\|, \|(W + \Delta)^{-1}\|\} > \|\Delta\|$, *then* (4.4) *holds.*

Apply Corollary 4.1 to $W = A'$ and $\Delta = \tilde{A} - A'$ to obtain the next result.

COROLLARY 4.2. *Equation* (3.4) *holds if* $\|E\| < 1/\min\{\|(A')^{-1}\|, \|\tilde{A}^{-1}\|\}$.

Now we are ready to propose an algorithm for computing sign(det A). At stage 2, we rely on the easily computable bound (4.2), which suffices for a very large class of inputs. Thus, most of the input instances will be filtered out at stage 2. For the remaining group of inputs, for which we do not get certification at stage 2, we perform the next stage 3, where we rely on Corollary 4.2. In Section 5 we discuss the policy of handling the rarer cases where even stage 3 does not produce certification.

Algorithm 4.1.

Input: an $n \times n$ matrix A, a fixed matrix norm ($\|\cdot\|_\infty$ or $\|\cdot\|_2$), and a unit round-off ϵ.
Output: either the certified value of sign(det A) or FAILURE (see Section 5).
Computations:

1. Apply Gaussian elimination (with complete, partial, or no pivoting) using the unit round-off ϵ, to compute the matrices $\tilde{L}$ and $\tilde{U}$ of (3.1).
2. Compute an upper bound on e_d (in particular, we may use the simple crude bound e_d^+ of (4.2)) and check if $|\det \tilde{U}|$ exceeds this bound. If so, compute and output sign(det A) based on (3.2)–(3.4).

3. Otherwise, estimate the norm $N = \|\tilde{A}^{-1}\| = \|\tilde{U}^{-1}\tilde{L}^{-1}\|$ from above and/or below (see Section 6) to decide whether

$$eN < 1. \tag{4.5}$$

If the latter inequality is verified, compute and output sign(det A) based on (3.2)–(3.4). Otherwise output FAILURE.

The complexity of the computations by the algorithm is dominated by the estimated complexity of stages 1 and 3. We refer the reader to [GL] and [BP] on the complexity of stage 1 and to Section 6 on the complexity of stage 3. In both cases we need $O(n^3)$ arithmetic operations. At stage 1, we may also need $O(n^3)$ or $O(n^2)$ comparisons for complete or partial pivoting, respectively.

5. Some Recipes in the Case of FAILURE. Suppose that Algorithm 4.1 has output FAILURE and that an upper bound N^+ on N and the value $f = eN^+$ are available. Then we have several options:

1. Repeat the computation but with the unit round-off $\epsilon_{\text{new}} = c\epsilon_{\text{old}}/f$ for some heuristic choice of $c < 1$. The value c can be adapted dynamically, depending on the resulting change of the value eN^+. If the latter value changes proportionally to ϵ_{new} (as can be expected unless A is a very ill-conditioned matrix), then (4.5) holds for $\epsilon = \epsilon_{\text{new}} = c\epsilon_{\text{old}}/f$ and for any $c < 1$. Recomputation of $\tilde{L}$ and $\tilde{U}$ for ϵ_{new} can be simplified because several leading digits in the representation of the computed values stay invariant, and only the remaining trailing digits must be recomputed (compare [EPY]).
2. Unless it is known already that $N^- \geq 1/e$, try to improve the upper estimate N^+ for $N = \|\tilde{U}^{-1}\tilde{L}^{-1}\|$ at stage 3 (see the next section).
3. If the value f is too large so that numerical computations with a unit round-off $\epsilon_{\text{new}} < \epsilon_{\text{old}}/f$ become too expensive, shift to the symbolic algorithms of [BEPP1] and [BEPP2]. In this case, one needs an a priori upper bound on $|\det A|$. Hadamard's bound (4.1) or one of the bounds $|\det A| \leq e_d^+ + |\det \tilde{u}|$ and $|\det A| \leq \prod_{i=1}^n (|u_{i,i}| + \|E\|)$ (see (3.5)) can be used, but it may pay to refine these bounds by performing some additional computations (see Section 7).

6. Estimating the Minimum Distance to a Singular Matrix

6.1. *Estimating the Distance from Above.* To estimate from above the minimum distance from the matrix $\tilde{A}$ to a singular matrix or, equivalently, to estimate $N - \|\tilde{U}^{-1}\tilde{L}^{-1}\|$ from below, we may apply the simple iterative algorithm in [CMSW], which, according to Golub and van Loan [GL], "produces a good order-of-magnitude" lower bound N^- on N at the cost of performing $O(jn^2)$ arithmetic operations in j iterations (practically, j is much smaller than n). If (4.5) does not hold even for N replaced by N^-, then we may apply the recipes of Section 5, for some heuristic choice of N^+.

6.2. *Two-Sided Estimates with Randomization.* Recall that $N = \|\tilde{A}^{-1}\|_2 = \|\tilde{U}^{-1}\tilde{L}^{-1}\|_2 = \sigma_1(\tilde{U}^{-1}\tilde{L}^{-1}) = 1/\sigma_n(\tilde{L}\tilde{U}) = 1/\sigma_n(\tilde{A})$, $\sigma_k(W)$ denoting the kth largest singular value of a matrix W. This reduces our problem to estimating the smallest singular value $\sigma_n(\tilde{A}) \geq 0$, which is a well-known problem of matrix computations whose solution does not require full computation of the singular value decomposition (SVD) of $\tilde{A}$. The problem can be solved by using the inverse power iteration or, better, the Lanczos algorithm [GL, Section 8.2.2 and Chapter 9]. Both of these algorithms require an initial vector, which can be chosen at random. Both use $O(n^2)$ flops per iteration, but the Lanczos algorithm converges faster (see p. 477 of [GL]). (The cost of each iteration in our case is dominated by the cost of the solution of four linear systems of equations with the already available triangular coefficient matrices $\tilde{L}^{\mathrm{T}}$, $\tilde{U}^{\mathrm{T}}$, $\tilde{U}$, and $\tilde{L}$.) The probability of obtaining accurate approximation to $\sigma_n(\tilde{A})$ depends on the number of iterative steps. Dixon [D2] shows that, with a probability at least $P_{I\text{-POWER}}(k, \Theta) = 1 - 0.8\Theta^{-k/2}n^{1/2}$, the lower bound $N_k^- = (\mathbf{x}^{\mathrm{T}}(\tilde{A}\tilde{A}^{\mathrm{T}})^{-1}\mathbf{x})^{1/2k} \leq \|\tilde{A}^{-1}\|_2 = 1/\sigma_n(\tilde{A})$ on N is also an upper bound within the factor $\Theta > 1$, that is, $\Theta N_k^- \geq \|\tilde{A}^{-1}\|_2$, for $k \geq 1$ and a random choice of a vector $\mathbf{x}$ on the unit sphere $S_n = \{\mathbf{x}\colon\ \mathbf{x}^{\mathrm{T}}\mathbf{x} = 1\}$, under the uniform probability distribution on S_n. Dixon deduced this estimate for the inverse power method applied to approximation of the smallest eigenvalue of $\tilde{A}\tilde{A}^{\mathrm{T}}$, which is the smallest singular value of $\tilde{A}$.

An alternative approach produces two-sided estimates for N by means of the Lanczos algorithm. The Lanczos algorithm computes σ_n^* satisfying $\sigma_n^* \geq \sigma_n(\tilde{L}\tilde{U})$. The estimates for the probability $P_{\text{LANCZOS}}(l, \Theta)$ that $\sigma_n^*/\sigma_n(\tilde{L}\tilde{U}) \leq \Theta$, for a fixed $\Theta > 1$ and nonsingular $\tilde{L}$ and $\tilde{U}$, can be expressed as functions in the number l of Lanczos iterations for a random choice of the initial vector (see [KW1] and [KW2]).

6.3. *Deterministic Lower Estimates for the Distance.* Section 8.3 of [H], pp. 159–161, shows some techniques for rapid computation of some crude upper bounds on $\|T^{-1}\|$ for triangular matrices T, and this can be immediately translated into rapid computation of some crude upper bounds on $N = \|\tilde{U}^{-1}\tilde{L}^{-1}\| \leq \|\tilde{U}^{-1}\| \cdot \|\tilde{L}^{-1}\|$. To yield more refined upper bounds on N (or, equivalently, some crude lower bounds on $1/N$), one may actually compute the inverses $\tilde{U}^{-1}$ and $\tilde{L}^{-1}$, then their product X and finally (an upper bound on) the norm $N = \|\tilde{U}^{-1}\tilde{L}^{-1}\|$. Detailed presentation of such computations can be found in Sections 13.2–13.3, pp. 265–275, of [H]. For the reader's convenience, we sketch some of these algorithms in Appendix A.

Assuming the computations with unit round-off ϵ, Higham [H] presents the estimates for the residual norms $\|\tilde{A}X - I\| = r(X)$, $\|X\tilde{A} - I\| = r^*(X)$, and the error norms $\|\tilde{A}^{-1} - X\| = e(X)$ for the computed approximations X to $\tilde{U}^{-1}\tilde{L}^{-1} = \tilde{A}^{-1}$. The estimates are given in the form

$$
\begin{aligned}
r(X) &\leq c_n\epsilon\|\tilde{L}\| \cdot \|\tilde{U}\| \cdot \|X\|,\\
r^*(X) &\leq c_n^*\epsilon\|\tilde{L}\| \cdot \|\tilde{U}\| \cdot \|X\|,\\
e(X) &\leq c_n'\epsilon\|\tilde{L}\| \cdot \|\tilde{U}\| \cdot \|X\| \cdot \|\tilde{A}^{-1}\|.
\end{aligned}
$$

Here c_n, c_n^*, and c_n' are constants independent of ϵ and $\tilde{A}$, which can be elaborated by using the error analysis techniques of [H].

Instead of applying these estimates, we may directly compute $\tilde{A}X - I$ or $X\tilde{A} - I$ (in $O(n^3)$ operations) and arrive at the residual norms $r(X)$ or $r^*(X)$ within the round-off

error bound $\gamma_n \|\tilde{A}\| \cdot \|X\|$, $\gamma_n = \epsilon n/(1-\epsilon n)$ (see p. 78 of [H]). For computing the row and column norms of the matrix, we may apply recursive pairwise summation, whose relative round-off error is at most $\gamma_{\lceil \log_2 n \rceil}$ (see p. 92 of [H]). If $r = \min\{r(X), r^*(X)\} < 1$, we immediately estimate that $e(X) \le r\|X\|/(1-r)$.

REMARK 6.1. To improve the approximation to $\tilde{A}^{-1}$ by $X = X_0$, one may apply Newton's iteration, $X_{i+1} = X_i(2I - AX_i)$, whose ith iterative step for every i squares the residual matrices $\tilde{A}X_i - I$ and $X_i\tilde{A} - I$ and, consequently, squares the upper bounds on their norms (see [PS]).

7. Estimating the Magnitude of the Determinant. With some additional work, one may try to refine the upper bounds of Section 5 on $|\det A|$ by relying on the following well-known fact:

FACT 7.1.

$$|\det A| = \prod_{i=1}^{k} \sigma_i,$$

where $\sigma_1, \dots, \sigma_r$ are the singular values of A, $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_r > 0$, $\sigma_{r+i} = 0$ for $i = 1, \dots, n-r$, $r = \operatorname{rank} A$.

Consequently, $d^+ = \prod_{i=1}^{n} \sigma_i^+ \ge |\det A|$ if $\sigma_i^+ \ge \sigma_i$, $i = 1, \dots, n$, and our problem is reduced to approximating $\sigma_1, \dots, \sigma_n$ from above. The known SVD algorithms [GL, Section 8.6 and p. 254] for the latter task use $7n^3/3 + O(n^2)$ flops. A faster though cruder solution may rely on approximating from above a few smallest singular values $\sigma_n, \sigma_{n-1}, \dots$ by means of the Lanczos algorithm and applying the readily available upper bound $\sigma = \min\{\|\tilde{A}\|_1, \|\tilde{A}\|_\infty\}\|$ on all other σ_i.

8. Some Modifications and Alternative Approaches

8.1. *LDM^T Factorization.* Instead of LU factorization of A', one may rely on its LDM^T factorization, where L and M are unit lower triangular matrices and D is a diagonal matrix. One may immediately obtain LDM^T factorization by extending the LU factorization as follows: compute the matrices $D = \operatorname{diag}(u_{i,i})$ and $M^T = D^{-1}U$ and then substitute $U = DM^T$ into the LU factorization. Alternatively, one may compute the LDM^T factorization directly, by means of the Gauss–Jordan transformation. In both cases our analysis can be easily extended (see pp. 275–281 of [H]).

8.2. *Solution by Estimating the Magnitude of the Diagonal Perturbation.* Consider the relative round-off errors of the diagonal entries of the factor U of $A' = LU$. To prove that (3.4) holds, it suffices to verify that all these errors are less than 1. The theorem of Barrlund and Sun (see p. 194 of [H]) gives us the following sufficient condition: $\|\tilde{G}\|_2 < 1$ and simultaneously $\operatorname{diag}(|\tilde{G}|(I - |\tilde{G}|)^{-1}|\tilde{U}|) < \operatorname{diag}(|\tilde{U}|)$ for $\tilde{G} = \tilde{L}^{-1}(\tilde{A} - A')\tilde{U}^{-1}$.

This approach has some similarity to one of Sections 4–5 (in particular $\|\tilde{G}\|$ can be estimated similarly to N^+ of Section 6 and Appendix A) but requires stronger assumptions and involves a more complicated expression, $|\tilde{G}|(I - |\tilde{G}|)|U|$, whose computation has larger round-off errors.

8.3. *The Straightforward Approach in* [PYS]. Unlike our approach, the paper [PYS] relies on the following equations:

$$\delta = \det(A' + E) - \det A' = \sum_{i,j} e_{i,j} \tilde{D}_{i,j}, \tag{8.1}$$

where $E = (e_{i,j}) = \tilde{A} - A'$, $\tilde{D}_{i,j} = \det \tilde{A}_{i,j}$, $\tilde{A}_{i,j}$ is the $(n-1) \times (n-1)$ matrix obtained by replacing the first $j-1$ columns of A' by ones of $\tilde{A}$ and by deleting the ith row and the jth column of the resulting matrix, for $i, j = 1, \ldots, n$. Then the relations $|\delta| \le n^2 e_+ \max_{i,j} |\tilde{D}_{i,j}| \le n^2 e_+ (\|A\| + n e_+)^{n-1}$ for a fixed matrix norm are deduced. This upper estimate for $|\delta|$, however, is overly pessimistic because it relies on the rough bound $|\tilde{D}_{i,j}| \le (\|A\| + n e_+)^{n-1}$ (which could be only slightly improved by using Hadamard's bound (4.1)) and on ignoring possible cancellation in the summation in (8.1). Generally, it is hard to obtain a sharp estimate for δ, and our approach benefits from proposing a solution that avoids estimating δ.

8.4. *QR Factorization Versus $PLUP_1$ Factorization.* The proposed algorithms and their analyses can be immediately extended based on the QR rather than the $PLUP_1$ factorization of A. Some estimates for the cost of computing the factorization and for the perturbation of the input, which would accommodate the round-off errors, can be taken from [PYS] but refined by carefully estimating from above the norm $\|R^{-1}Q^{\mathrm{T}}\|$ (see Chapter 18 of [H]). This analysis gives the upper hand to Householder transformations as a method of choice for the computation of the QR factorization. (Actually, besides the sign of $\det Q$, which is usually readily available, we only need the diagonal entries of R for our purpose of the sign computation.) Relying on the QR factorization has advantage over using triangular factorizations when the sign of $\det A$ must be computed for a dynamically updated matrix A (see [GGMS] and [PYS]). To decrease the round-off errors, one may apply a scaled version of the QR factorization, making it free of square root computation (see [C]).

Acknowledgment. Victor Pan thanks Dario Bini for the preliminary discussions on some alternative approaches to certified numerical computation of the sign of matrix determinant.

Appendix A. Computation of the Inverse. To compute the inverse $\tilde{A}^{-1} = \tilde{U}^{-1}\tilde{L}^{-1}$, we may rely on the following simple observations (see pp. 170–174 of [H]).

THEOREM A.1. *Let $T = (t_1, \ldots, t_n)$ be an $n \times n$ proper lower (respectively, upper) triangular matrix (with zeros on its diagonal). Let T_i denote the matrix obtained by*

zeroing all the columns of T except for its ith (*respectively,* $(n+1-i)$th) *column,* $i = 1, \dots, n$. *Then*

$$I + T = (I + T_1)(I + T_2) \cdots (I + T_{n-1}),$$
$$(I + T_i)^{-1} = I - T_i, \qquad i = 1, \dots, n-1.$$

COROLLARY A.1. *Under the assumptions of Theorem* A.1, *we have*

$$(I + T)^{-1} = (I - T_{n-1})(I - T_{n-2}) \cdots (I - T_1). \tag{A.1}$$

Based on Corollary A.1, we may compute $\tilde{L}^{-1}$ and $\hat{U}^{-1}$ where $\hat{U}$ is a unit triangular matrix obtained from $\tilde{U}$ by scaling the rows and/or columns of $\tilde{U}$. The overall round-off error estimate for computing $\tilde{L}^{-1}$ depends on the order in which we multiply the matrices $I - T_i$ in (A.1) for $T = \hat{L} - I$ (e.g., from left to right, from right to left, or according to some mixed policy) and similarly for $T = \hat{U} - I$. (At the latter stage the variation also depends on the choice of the scaling of $\tilde{U}$, which defines the transition to the matrix $\hat{U}$.)

Appendix B. Sign Determination via LDL^T Factorization of A^TA. Computation of the LDL^T factorization of the symmetric matrix A^TA is simple and has very good numerical stability (see pp. 138–139 of [GL] and pp. 207–209 of [H]). Namely, the two matrices of the round-off errors of computing A^TA and the LDL^T factors of A^TA have norms bounded from above by $\gamma_n \|A\| \cdot \|A^T\|$ and $(\gamma_{n+1}/(1-\gamma_{n+1})) \sum_i a_{i,i}^2$, respectively [H, p. 78]. Thus, such a factorization may serve as a basis for good numerical approximation of $(\det A)^2 = \det(A^TA) = \prod_i D_{i,i}$ and then of $|\det A|$.

We next extend this observation to a certified algorithm for $\operatorname{sign}(\det A)$. We assume that we have computed $\tilde{d} \geq 0$ and a relatively small $\Delta > 0$ such that

$$\tilde{d} - \Delta < |\det A| < \tilde{d} + \Delta.$$

This approach inherits the disadvantage of the work in [PYS] because it involves numerical computation of $|\det A|$ with finite precision and with a large round-off error. We include it for the sake of completeness as a natural variation of the approach in [PYS]. If $\tilde{d} < \Delta$, then $|\det A| < 2\Delta$, and the symbolic algorithms of [BEPP1] and [BEPP2] effectively compute $\operatorname{sign}(\det A)$. If $\tilde{d} \geq \Delta$, we choose some prime $p \geq 4\Delta$ lying close to 4Δ and compute $d_p = (\det A) \bmod p$ defined as a unique value in the semi-open interval $[-p/2, p/2)$ such that p divides $d_p - \det A$. Similarly we define $\tilde{d}_p = \tilde{d} \bmod p$. (To obtain d_p, we first compute (modulo p) a PLU factorization of A and $\operatorname{sign}(\det P)$ and then compute $d_p = ((\prod_i (u_{i,i} \bmod p)) \operatorname{sign}(\det P)) \bmod p$. Note that the computation modulo p only requires $2\lceil \log_2 p \rceil$-bit precision.)

Finally, we explain the transition from Δ, p, d_p, and $\tilde{d}_p$ to $\operatorname{sign}(\det A)$. Clearly, we have either

$$-\Delta < \det A - \tilde{d} = (d_p - \tilde{d}_p) \bmod p < \Delta$$

or

$$-\Delta < \det A + \tilde{d} = (d_p + \tilde{d}_p) \bmod p < \Delta.$$

These two cases cannot occur simultaneously since $(2\tilde{d}) \bmod p = (2\tilde{d}_p) \bmod p \geq 2\Delta$, and we may easily check which of them actually occurs. In the former case, $\det A > \tilde{d} - \Delta \geq 0$, and we output $\text{sign}(\det A) = 1$. Otherwise, $\det A < \Delta - \tilde{d} \leq 0$, and we output $\text{sign}(\det A) = -1$.

Appendix C. Numerical Experiments. In this section we report the results of numerical tests that we performed to compare our arithmetic filter based on Algorithm 4.1 with those in [PYS], specialized for the computation of $\det A$. The comparison made under the same format used in [PYS] clearly favors our filter.

In the experiments, we computed numerically the determinants of $n \times n$ matrices A for $2 \leq n \leq 12$, based on computing the LU, PLU, and $PLUP_1$ factorizations of A. Instead of using the packages of subroutines, we just implemented the respective simple algorithms displayed in [GL]. We defined the input matrices A in the form PLU, with P being a random permutation matrix and with triangular matrices L and U filled with random integers. When these integers were chosen long, both groups of algorithms (ours and ones in [PYS]) worked efficiently and output certified $\text{sign}(\det A)$ for almost all input instances. The comparison became much more meaningful, when we restricted it to the case of integer matrices A having small $|\det A|$. We arrived at this case by choosing small integer diagonal entries of L and U.

Namely, the input matrices A have been composed by using the following steps to produce random nonsingular matrices either with determinants 1 or with relatively small known determinants:

1. For an auxiliary pair of lower and upper triangular matrices $L^{(0)}$ and $U^{(0)}$, respectively, let their nonzero off-diagonal entries be random integers in the interval $(-10, 10)$.
2. Either let the diagonal entries $l_{i,i}^{(0)}$ and $u_{i,i}^{(0)}$ also be chosen at random in the same way (see Table 1) or set $l_{i,i}^{(0)} = u_{i,i}^{(0)} = 1$ for all i (see Table 2). In the former case, if $l_{i,i}^{(0)} = 0$ or $u_{i,i}^{(0)} = 0$, for some i, $1 \leq i \geq n$, then set the entry to 1 to avoid arriving at a singular matrix.
3. Compute $A = L^{(0)}U^{(0)}$.
4. Swap a random pair of rows in the matrix A.
5. Repeat step 4 m times, where m is a random integer in the range $[0, n)$.

The algorithms have been implemented with C++ and built as a console application with Microsoft Visual C++ 5.0 compiler and linker. All numerical operations have been performed with double precision floating point arithmetic. The double precision representation of a number uses 64 bits: 1 for the sign, 11 for the exponent, and 52 for the mantissa. Its range is $\pm 1.7 \times 10^{308}$ with at least 15 decimal digits of precision. The test results have been collected on a Pentium 100 MHz PC, running under Windows 95's DOS session. The system pseudorandom number generator functions **srand()** and **rand ()** have been used to generate input matrices.

Tables 1 and 2 present the results of our experiments. One thousand random matrices of each size (from 2 to 12 in the case of small determinants and from 2 to 10 in the case of determinants 1) have been tested. The relative errors $e_0 = |\det \tilde{A} - \det A'|/|\det A'|$, $e_{\text{pys}} = n^2 a^+ \epsilon / |\det A'|$, and $e_{\text{new}} = eN$ have been evaluated in each case, and their average values

Table 1. Average estimated errors of 1000 random matrices.

Algorithm	e_0	e_{pys}	e_{new}	$\lvert \det A \rvert$	$1/\lVert A^{-1} \rVert$	NV_{pys}	NV_{new}
			Size 2				
LU	1.958e-016	1.707e-013	5.906e-014	8.781e+002	9.036e+000	102	102
PLU	2.625e-016	4.454e-014	6.266e-014	8.845e+002	9.262e+000	3	3
$PLUP_1$	1.935e-016	3.189e-014	5.084e-014	8.834e+002	9.253e+000	0	0
			Size 3				
LU	2.636e-015	2.322e-010	4.828e-012	2.400e+004	3.360e+000	76	72
PLU	2.404e-015	4.011e-011	1.705e-012	2.408e+004	3.438e+000	3	3
$PLUP_1$	2.184e-015	1.077e-011	1.160e-012	2.402e+004	3.441e+000	0	0
			Size 4				
LU	2.265e-014	4.030e-004	3.705e-010	6.943e+005	1.483e+000	62	64
PLU	1.363e-014	1.756e-009	4.177e-011	6.991e+005	1.481e+000	1	1
$PLUP_1$	1.041e-014	1.678e-009	1.525e-011	6.984e+005	1.480e+000	0	0
			Size 5				
LU	1.797e-013	3.935e-003	3.761e-009	1.563e+007	5.724e-001	56	40
PLU	8.941e-014	5.184e-006	2.116e-010	1.530e+007	5.652e-001	0	0
$PLUP_1$	7.952e-014	6.080e-007	1.437e-010	1.530e+007	5.652e-001	0	0
			Size 6				
LU	9.590e-013	1.980e-002	1.255e-008	4.585e+008	2.615e-001	134	37
PLU	1.078e-012	5.420e-005	2.438e-009	4.097e+008	2.395e-001	0	0
$PLUP_1$	9.341e-013	1.005e-005	1.132e-009	4.097e+008	2.395e-001	0	0
			Size 7				
LU	7.345e-013	5.393e-002	4.752e-008	3.433e+010	1.802e-001	317	37
PLU	9.060e-012	5.940e-003	3.327e-008	2.473e+010	1.423e-001	4	0
$PLUP_1$	1.046e-011	7.431e-004	1.502e-008	2.463e+010	1.417e-001	0	0
			Size 8				
LU	4.674e-012	1.175e-001	1.751e-005	6.615e+011	8.039e-002	619	29
PLU	3.221e-011	2.448e-002	1.061e-006	5.662e+011	6.086e-002	37	0
$PLUP_1$	2.304e-011	1.404e-002	5.653e-008	5.541e+011	5.957e-002	16	0
			Size 9				
LU	9.134e-013	2.053e-001	1.586e-008	2.701e+013	6.165e-002	839	19
PLU	1.055e-011	6.904e-002	5.593e-008	1.201e+013	2.812e-002	160	0
$PLUP_1$	1.356e-011	3.917e-002	6.251e-008	1.108e+013	2.596e-002	89	0
			Size 10				
LU	9.726e-013	2.240e-001	7.454e-009	1.963e+015	6.443e-002	964	21
PLU	5.562e-012	1.304e-001	2.899e-008	8.986e+014	1.949e-002	425	0
$PLUP_1$	9.713e-012	1.013e-001	5.244e-008	7.089e+014	1.551e-002	271	0
			Size 11				
LU	4.879e-013	4.320e-001	9.386e-009	5.658e+016	7.637e-002	995	32
PLU	4.061e-012	2.047e-001	1.454e-008	1.308e+016	1.431e-002	740	0
$PLUP_1$	1.578e-011	1.700e-001	5.002e-008	8.611e+015	9.995e-003	595	0
			Size 12				
LU	NA	NA	NA	NA	NA	1000	24
PLU	6.190e-012	2.802e-001	5.254e-008	3.347e+018	3.118e-002	912	0
$PLUP_1$	4.520e-012	2.429e-001	2.767e-008	1.494e+018	1.581e-002	802	0

Table 2. Average estimated errors of 1000 random matrices with ± 1 determinants.

Algorithm	e_0	e_{pys}	e_{new}	$\lvert \det A \rvert$	$1/\lVert A^{-1} \rVert$	NV_{pys}	NV_{new}
			Size 2				
LU	7.904e-016	1.809e-012	1.656e-013	1.000e+000	1.000e+000	92	92
PLU	9.345e-016	1.036e-013	5.165e-013	1.000e+000	1.000e+000	2	2
$PLUP_1$	9.268e-016	1.034e-013	5.155e-013	1.000e+000	1.000e+000	0	0
			Size 3				
LU	2.326e-014	2.800e-008	3.315e-010	1.000e+000	1.000e+000	85	76
PLU	3.583e-014	1.064e-009	4.950e-011	1.000e+000	1.000e+000	4	3
$PLUP_1$	2.828e-014	6.154e-010	3.819e-011	1.000e+000	1.000e+000	1	0
			Size 4				
LU	1.004e-012	1.924e-003	5.165e-008	1.000e+000	1.000e+000	54	53
PLU	7.900e-013	1.285e-005	5.243e-009	1.000e+000	1.000e+000	0	0
$PLUP_1$	6.655e-013	4.342e-007	2.049e-009	1.000e+000	1.000e+000	0	0
			Size 5				
LU	2.608e-011	3.990e-002	1.504e-005	1.000e+000	1.000e+000	152	52
PLU	2.507e-011	9.849e-004	3.504e-007	1.000e+000	1.000e+000	1	0
$PLUP_1$	2.143e-011	2.059e-004	1.021e-007	1.000e+000	1.000e+000	0	0
			Size 6				
LU	2.809e-010	2.083e-001	3.227e-005	1.000e+000	1.000e+000	626	41
PLU	4.969e-010	1.319e-001	1.051e-005	1.000e+000	1.000e+000	104	0
$PLUP_1$	4.670e-010	8.723e-002	3.762e-006	1.000e+000	1.000e+000	21	0
			Size 7				
LU	2.245e-009	3.712e-001	4.674e-004	1.000e+000	1.000e+000	992	41
PLU	4.464e-009	4.782e-001	1.818e-003	1.000e+000	1.000e+000	937	0
$PLUP_1$	2.534e-009	4.393e-001	3.227e-005	1.000e+000	1.000e+000	885	0
			Size 8				
LU	NA*	NA	NA	NA	NA	1000	114
PLU	NA	NA	NA	NA	NA	1000	8
$PLUP_1$	NA	NA	NA	NA	NA	1000	0
			Size 9				
LU	NA	NA	NA	NA	NA	1000	349
PLU	NA	NA	NA	NA	NA	1000	54
$PLUP_1$	NA	NA	NA	NA	NA	1000	34
			Size 10				
LU	NA	NA	NA	NA	NA	1000	713
PLU	NA	NA	NA	NA	NA	1000	281
$PLUP_1$	NA	NA	NA	NA	NA	1000	242

*NA stands for "not available" in the case where computation of det A by the algorithm in [PYS] failed in all cases.

have been presented in the tables. The values of $|\det A|$ and $1/\|\tilde{A}^{-1}\| = 1/\|\tilde{U}^{-1}\tilde{L}'^{-1}\|$ (average over the results of both approaches [PYS] and Algorithm 4.1) have also been computed and presented in Tables 1 and 2. Two integer counters V_{pys} and V_{new} have been used to keep track of the number of cases where the [PYS] algorithm and our new algorithm failed to verify the computation results, respectively. More specifically, whenever division by zero occurred in the [PYS] algorithm or whenever we observed that $e_{\text{pys}} \geq 1$, the counter V_{pys} was incremented by one, and this case was excluded from the average count of the data represented in the first five columns of the tables. Likewise, whenever division by zero occurred in Algorithm 4.1 or whenever we observed that $e_{\text{new}} \geq 1$, the counter V_{new} was incremented by one and the case was similarly excluded from the average count of the data.

The tables show that the number of cases rejected by the new algorithm are consistently smaller than that of the [PYS] algorithm. For the accepted cases, the relative error estimates obtained by the new algorithm are much closer to but no smaller than the "true" relative errors.

References

[ABD+] F. Avnaim, J.-D. Boissonnat, O. Devillers, F. P. Preparata, M. Yvinec, Evaluating Signs of Determinants Using Single-Precision Arithmetic, *Algorithmica*, **17**, 111–132, 1997.

[AF] D. Avis, K. Fukuda, A Pivoting Algorithm for Convex Hulls and Vertex Enumeration of Arrangements and Polyhedra, *Discrete Comput. Geometry*, **8**, 295–313, 1992.

[B] E. H. Bareiss, Sylvester's Identity and Multistep Integer-Preserving Gaussian Elimination, *Math. Comp.*, **22**, 565–578, 1968.

[BEPP1] H. Brönnimann, I. Z. Emiris, V. Y. Pan, S. Pion, Computing Exact Geometric Predicates Using Modular Arithmetic with Single Precision, *Proc. 13th Ann. ACM Symp. on Computational Geometry*, pp. 174–182, ACM Press, New York, 1997.

[BEPP2] H. Brönnimann, I. Z. Emiris, V. Y. Pan, S. Pion, Sign Determination in Residue Number Systems, *Theoret. Comput. Sci.*, **210**(1), 173–197, 1999.

[BKM+] C. Burnikel, J. Könnemann, K. Mehlhorn, S. Näher, S. Schirra, C. Uhrig, Exact Geometric Computation in *LEDA*, *Proc. 11th Ann. ACM Symp. on Computational Geometry*, pp. C18–C19, 1995. Package available at http://www.mpi-sb.mpg.de/LEDA/leda.html.

[BP] D. Bini, V. Y. Pan, *Polynomial and Matrix Computations*, vol. 1, Birkhäuser, Boston, 1994.

[C] K. L. Clarkson, Safe and Effective Determinant Evaluation, *Proc. 33rd Ann. IEEE Symp. on Foundations of Computer Science*, pp. 387–395, IEEE Computer Society Press, Los Alamitos, CA, 1992.

[CMSW] A. K. Cline, C. B. Moler, G. W. Stewart, J. H. Wilkinson, An Estimate for the Condition Number of a Matrix, *SIAM J. Numer. Anal.*, **16**, 368–375, 1979.

[D1] E. Durand, *Solutions Numériques des Équations Algébriques*, vol. II, Masson, Paris, 1961.

[D2] J. J. D. Dixon, Estimating Extremal Eigenvalues and Conditional Numbers of Matrices, *SIAM J. Numer. Anal.*, **20**(4), 812–814, 1983.

[DL1] M. Deza, M. Laurent, Applications of Cut Polyhedra, *J. Comput. Appl. Math.*, **55**(1), 191–216, and **55**(2), 217–247, 1994.

[DL2] M. M. Deza, M. Laurent, *Geometry of Cuts and Metrics*, Springer-Verlag, Berlin, 1997.

[E1] J. Edmonds, Systems of Distinct Representatives and Linear Algebra, *J. Res. Nat. Bur. Standards*, Sect. B, **71**(4), 241–245, 1967.

[E2] I. Z. Emiris, A Complete Implementation for Computing General Dimensional Convex Hulls, *Internat. J. Comput. Geom. Appl.*, **8**(2), 223–253, 1998.

[EC] I. Z. Emiris, J. F. Canny, A General Approach to Removing Degeneracies, *SIAM J. Comput.*, **24**(3), 650–664, 1995.

[EPY] I. Z. Emiris, V. Y. Pan, Y. Yu, Modular Arithmetic for Linear Algebra Computations in the Real Field, *J. Symbolic Comput.*, **21**, 1–17, 1998.

[ES] R. M. Erdahl, V. H. Smith (editors), *Density Matrices and Density Functionals* (Proc. of the A. John Coleman Symp.), Reidel, Dordrecht, 1987.

[EY] C. Eckart, G. Young, A Principal Axis Transformation for Non-Hermitian Matrices, *Bull. Amer. Math. Soc.* (New Series), **45**, 118–121, 1939.

[F] L. Fox, *An Introduction to Numerical Linear Algebra*, Clarendon Press, Oxford, 1964.

[FR] K. Fukuda, V. Rosta, Combinatorial Face Enumeration in Convex Polytopes, *Comput. Geom., Theory Appl.*, **4**, 191–198, 1994.

[FvW1] S. Fortune, C. J. van Wyk, Efficient Exact Arithmetic for Computational Geometry, *Proc. 9th Ann. ACM Symp. on Computational Geometry*, pp. 163–172, 1993.

[FvW2] S. Fortune, C. J. van Wyk, Static Analysis Yields Efficient Exact Integer Arithmetic for Computational Geometry, *ACM Trans. Graph*, **15**(3), 223–248, July 1996.

[GGMS] P. E. Gill, G. H. Golub, W. Murray, M. A. Saunders, Methods for Modifying Matrix Factorizations, *Math. Comput.*, **28**, 505–535, 1974.

[GL] G. H. Golub, C. F. van Loan, *Matrix Computations* (third edition), Johns Hopkins University Press, Baltimore, MD, 1996.

[H] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, Philadelphia, PA, 1996.

[IEEE] IEEE Standard 754-1985 for Binary Floating Point Arithmetic, reprinted in *SIGPLAN Notices*, **22**(2), 9–25, 1987.

[K1] W. Kahan, Numerical Linear Algebra, *Canad. Math. Bull.*, **9**, 757–801, 1966.

[K2] E. Kaltofen, Analysis of Coppersmith's Block Wiedemann Algorithm for the Parallel Solution of Sparse Linear Systems, *Math. Comput.*, **64**(210), 777–806, 1995.

[KP1] E. Kaltofen, V. Y. Pan, Processor Efficient Parallel Solution of Linear Systems over an Abstract Field, *Proc. 3rd Ann. ACM Symp. on Parallel Algorithms and Architectures*, pp. 180–191, ACM Press, New York, 1991.

[KP2] E. Kaltofen, V. Y. Pan, Processor Efficient Parallel Solution of Linear Systems, II. The Positive Characteristic and Singular Cases, *Proc. 33rd Ann. IEEE Symp. on Foundation of Computer Science*, pp. 714–723, IEEE Computer Society Press, Los Alamitos, CA, 1992.

[KW1] J. Kuczyński, H. Woźniakowski, Estimating the Largest Eigenvalue by the Power and Lanczos Algorithms with a Random Start, *SIAM J. Matrix Anal. Appl.*, **13**(4), 1094–1122, 1992.

[KW2] J. Kuczyński, H. Woźniakowski, Probabilistic Bounds on the Extremal Eigenvalues and Condition Number by the Lanczos Algorithm, *SIAM J. Matrix Anal. Appl.*, **15**(2), 672–691, 1994.

[M1] T. Muir, *The Theory of Determinants in Historical Order of Development*, four volumes bound as two (I, 1693–1841; II, 1841–1860; III, 1861–1880; IV, 1881–1900), Dover, New York, 1960; *Contributions to the History of Determinants*, 1900–1920, Blackie and Son, London, 1930 and 1950.

[M2] R. H. Macmillan, A New Method for the Numerical Evaluation of Determinants, *J. Roy. Aeronaut. Soc.*, **59**, 772ff, 1955.

[MA] R. E. M. Moore, I. O. Angell, Voronoi Polygons and Polyhedra, *J. Comput. Phys.*, **105**, 301–305, 1993.

[P] V. Y. Pan, Computing the Determinant and the Characteristic Polynomial of a Matrix via Solving Linear Systems of Equations, *Inform. Process. Lett.*, **28**(2), 71–75, 1988.

[PS] V. Y. Pan, R. Schreiber, An Improved Newton Iteration for the Generalized Inverse of a Matrix, with Applications, *SIAM J. Sci. Statist. Comput.*, **12**(5), 1109–1131, 1991.

[PYS] V. Y. Pan, Y. Yu, C. Stewart, Algebraic and Numerical Techniques for the Computation of Matrix Determinants, *Comput. Math. Appl.*, **34**(1), 43–70, 1997.

[R] J. B. Rosser, A Method of Computing Exact Inverses of Matrices with Integer Coefficients, *J. Res. Nat. Bur. Standards*, Sect. B, **49**, 349–358, 1952.

[W] D. H. Wiedemann, Solving Sparse Linear Equations over Finite Fields, *IEEE Trans. Inform. Theory*, **32**(1), 54–62, 1986.

[Y] C. Yap, Towards Exact Geometric Computation, *Comput. Geom. Theory Appl.*, **7**, 3–23, 1997.

[YD] C. K. Yap, T. Dubhe, The Exact Computation Paradigm, in D. Du and F. Hwang, editors, *Computing in Euclidean Geometry*. World Scientific Press, Singapore, 1995.