Stability of a method for multiplying complex matrices with three real matrix multiplications

Higham, Nicholas J.

1992

MIMS EPrint: **2006.169**

ISSN 1749-9097

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 681–687, July 1992

STABILITY OF A METHOD FOR MULTIPLYING COMPLEX MATRICES WITH THREE REAL MATRIX MULTIPLICATIONS*

NICHOLAS J. HIGHAM†

Abstract. By use of a simple identity, the product of two complex matrices can be formed with three real matrix multiplications and five real matrix additions, instead of the four real matrix multiplications and two real matrix additions required by the conventional approach. This alternative method reduces the number of arithmetic operations, even for small dimensions, achieving a saving of up to 25 percent. The numerical stability of the method is investigated. The method is found to be less stable than conventional multiplication but stable enough to warrant practical use. Issues involved in the choice of method for complex matrix multiplication are discussed, including the relative efficiency of real and complex arithmetic and the backward stability of block algorithms.

Key words. matrix multiplication, complex matrix, Strassen's method, Winograd's identity, numerical stability, error analysis, level-3 BLAS

AMS(MOS) subject classifications. 65F05, 65G05

1. Introduction.

How many real multiplications are required to multiply two complex numbers? In view of the familiar identity

$$z = (a+ib)(c+id) = ac - bd + i(ad+bc),$$

the answer might appear to be four. However, it is possible to make do with three multiplications, because

$$z = ac - bd + i\big[(a+b)(c+d) - ac - bd\big]. \tag{1.1}$$

This formula was suggested by Peter Ungar in 1963, according to Knuth [14, p. 647]. That three multiplications (or divisions) are *necessary* for evaluating z was proved by Winograd [17].

Ungar's formula does not rely on commutativity, so it can be generalized to matrix multiplication, as noted by Fam [10]. Let $A = A_1 + iA_2$ and $B = B_1 + iB_2$, where $A_j, B_j \in \mathbb{R}^{n\times n}$, and define $C = C_1 + iC_2 = AB$. (We concentrate on square matrices, although everything we say extends easily to rectangular matrices.) Then C can be formed using three real matrix multiplications as

$$\begin{aligned} T_1 &= A_1B_1, \qquad T_2 = A_2B_2,\\ C_1 &= T_1 - T_2,\\ C_2 &= (A_1+A_2)(B_1+B_2) - T_1 - T_2, \end{aligned} \tag{1.2}$$

which we will refer to as the "3M method." This computation involves $3n^3$ scalar multiplications and $3n^3 + 2n^2$ scalar additions. Straightforward evaluation of the conventional formula $C = A_1B_1 - A_2B_2 + i(A_1B_2 + A_2B_1)$ requires $4n^3$ multiplications and $4n^3 - 2n^2$ additions. Thus, the 3M method requires strictly less arithmetic operations than the conventional means of multiplying complex matrices for $n \geq 3$, and it achieves a saving of about 25 percent for $n \geq 30$ (say). Similar savings occur in the important special case where A or B is triangular.

* Received by the editors February 12, 1990; accepted for publication November 2, 1990.

† Department of Mathematics, University of Manchester, Manchester, M13 9PL, England (na.nhigham@na-net.ornl.gov).

It is rare in matrix computations to be able to produce such a clear-cut computational saving over a standard technique, and the 3M method therefore deserves careful consideration for practical use.

Since an increase in speed is often accompanied by a loss of numerical stability, it is important to investigate the behaviour of the 3M method in the presence of rounding errors. Doubts about the stability are raised by the comment of Knuth [14, p. 647] on a variation of (1.1) (see (2.7)): "Beware numerical instability." We investigate the stability in §2 and show that the 3M method is stable in a certain sense, although it does not match the stability properties of conventional multiplication.

In §3 we offer some guidance on the choice of method for multiplying complex matrices.

This work was motivated by the knowledge that the 3M method is being used in Fortran routines CGEMMS and ZGEMMS in IBM's ESSL library. The 3M method is also used by routines of the same name in Cray's UNICOS library [5]. All these routines form complex matrix products by using Strassen's fast matrix multiplication method [15] to evaluate the real matrix products in (1.2). Although the ESSL documentation warns about potential instability of Strassen's method [13, p. 344], it contains no comment on the stability of the 3M method itself.

2. Numerical stability.

A simple example reveals a fundamental weakness of the 3M method. Consider the computation of the scalar

$$z = x + iy = (\theta + i/\theta)^2 = \theta^2 - 1/\theta^2 + 2i.$$

Suppose we use floating point arithmetic with unit roundoff u. If y is computed the usual way, as $y = \theta(1/\theta) + (1/\theta)\theta$, then no cancellation occurs and the computed $\widehat{y}$ has high relative accuracy: $|\widehat{y} - y|/|y| = O(u)$. The 3M method computes

$$y = \left(\theta + \frac{1}{\theta}\right)\left(\theta + \frac{1}{\theta}\right) - \theta^2 - \frac{1}{\theta^2}.$$

If $|\theta|$ is large this formula expresses a number of order 1 as the difference of large numbers. The computed $\widehat{y}$ will almost certainly be contaminated by rounding errors of order $u\theta^2$, in which case the relative error is large: $|\widehat{y}-y|/|y| = O(u\theta^2)$. However, if we measure the error in $\widehat{y}$ relative to z, then it is acceptably small: $|\widehat{y}-y|/|z| = O(u)$.

This example suggests that the 3M method may be stable in a weaker sense than conventional multiplication. In the rest of this section we establish the stability properties in a precise form, for general n.

For the error analysis we assume that the floating point arithmetic obeys the model

$$\begin{aligned} fl(x \operatorname{op} y) &= (x \operatorname{op} y)(1+\delta), \quad |\delta| \le u, \quad \operatorname{op} = *, /, \\ fl(x \pm y) &= x(1+\alpha) \pm y(1+\beta), \qquad |\alpha|, |\beta| \le u, \end{aligned}$$

where the latter equation allows for possible lack of a guard digit in addition and subtraction. Standard analysis (analogous to that in [11, p. 66], for example) shows that if $A, B \in \mathbb{R}^{n\times n}$ and we compute $\widehat{C} = fl(AB)$ by conventional multiplication, then

$$|\widehat{C} - AB| \le nu|A||B| + O(u^2). \tag{2.1}$$

Here, $|\cdot|$ denotes the operation of replacing each matrix element by its absolute value, and the matrix inequality is interpreted componentwise.

Now we consider the product $C_1 + iC_2 = (A_1 + iA_2)(B_1 + iB_2)$ for $n \times n$ complex matrices, as in §1. Using (2.1) we find that the computed product from conventional multiplication,

$$\widehat{C} = fl\big(A_1B_1 - A_2B_2 + i(A_1B_2 + A_2B_1)\big),$$

satisfies

$$|\widehat{C}_1 - C_1| \le (n+1)u\big(|A_1||B_1| + |A_2||B_2|\big) + O(u^2), \tag{2.2}$$

$$|\widehat{C}_2 - C_2| \le (n+1)u\big(|A_1||B_2| + |A_2||B_1|\big) + O(u^2). \tag{2.3}$$

It is easy to verify that, apart from the factors $n+1$, these bounds reflect the sensitivity of the product AB to perturbations in A and B of the form $A_j \to A_j + \Delta A_j$, where $|\Delta A_j| \le u|A_j|$.

For the 3M method C_1 is computed in the conventional way, and so (2.2) holds. It is straightforward but tedious to show that $\widehat{C}_2$ satisfies

$$\begin{aligned} |\widehat{C}_2 - C_2| \le (n+4)u\big[&(|A_1| + |A_2|)(|B_1| + |B_2|) \\ &+ |A_1||B_1| + |A_2||B_2|\big] + O(u^2). \end{aligned} \tag{2.4}$$

Two notable features of the bound (2.4) are as follows. First, it is of a different and weaker form than (2.3); in fact, it exceeds the sum of the bounds (2.2) and (2.3). Second and more pleasing, it retains the property of (2.2) and (2.3) of being invariant under diagonal scalings

$$C = AB \to D_1AD_2 \cdot D_2^{-1}BD_3 = D_1CD_3, \quad D_j \text{ diagonal},$$

in the sense that the upper bound ΔC_2 in (2.4) scales also according to $D_1\Delta C_2 D_3$. (The "hidden" second-order terms in (2.2)–(2.4) are invariant under these diagonal scalings.)

The disparity between (2.3) and (2.4) is, in part, a consequence of the differing numerical cancellation properties of the two methods. It is easy to show that there are always subtractions of like-signed numbers in the 3M method, whereas if A_1, A_2, B_1, and B_2 have nonnegative elements (for example), then no numerical cancellation takes place in conventional multiplication.

We can define a measure of stability with respect to which the 3M method matches conventional multiplication by taking norms in (2.3) and (2.4). We obtain the weaker bounds

$$\|\widehat{C}_2 - C_2\|_\infty \le 2(n+1)u\|A\|_\infty\|B\|_\infty + O(u^2), \tag{2.5}$$

$$\|\widehat{C}_2 - C_2\|_\infty \le 4(n+4)u\|A\|_\infty\|B\|_\infty + O(u^2). \tag{2.6}$$

Combining these with an analogous weakening of (2.2), we find that for both conventional multiplication and the 3M method, the computed complex matrix $\widehat{C}$ satisfies

$$\|\widehat{C} - C\|_\infty \le c_n u\|A\|_\infty\|B\|_\infty + O(u^2),$$

where $c_n = O(n)$.

Our findings can be summarised as follows. The 3M method produces a computed product $\widehat{C}$ whose imaginary part may be contaminated by relative errors much larger than those for conventional multiplication (or equivalently, much larger than can be accounted for by small componentwise perturbations in the data A and B). However, if the errors are measured relative to $\|A\|_\infty \|B\|_\infty$, which is a natural quantity to use for comparison when employing matrix norms, then they are just as small as for conventional multiplication.

We conclude this section with several further comments.

(1) It does not seem possible to improve the stability of the 3M method by "tinkering" with the basic formula. The symmetric formula

$$z = a(c+d) - (a+b)d + i\big[a(c+d) + (b-a)c\big], \tag{2.7}$$

mentioned by Knuth [14, p. 647] as an alternative to (1.1), is "worse" in the sense that either of the real and imaginary parts can be relatively inaccurate. Of course, by adapting formula (1.1) we can arrange that only the real part be of questionable accuracy.

(2) The 3M formula resembles Winograd's identity for computing the inner product of vectors $x, y \in \mathbb{R}^n$. The identity is [16], for even n,

$$x^T y = \sum_{i=1}^{n/2} (x_{2i-1} + y_{2i})(x_{2i} + y_{2i-1}) - \sum_{i=1}^{n/2} x_{2i-1}x_{2i} - \sum_{i=1}^{n/2} y_{2i-1}y_{2i}. \tag{2.8}$$

Setting $n = 2$, we have

$$x^T y = (x_1 + y_2)(x_2 + y_1) - x_1 x_2 - y_1 y_2. \tag{2.9}$$

For comparison, the 3M formula (1.1) uses the identity

$$x^T y = (x_1 + x_2)(y_1 + y_2) - x_1 y_2 - x_2 y_1 \tag{2.10}$$

to compute the imaginary part of z. Although they look similar, formulas (2.9) and (2.10) have quite different stability properties, because (2.9) exploits commutativity ($y_2 x_2 = x_2 y_2$ and $y_2 y_1 = y_1 y_2$), while (2.10) does not. Thus only (2.10) permits the generalisation where the x_j and y_j are matrices. On the other hand, Winograd's identity can be used to trade half the multiplications for additions in a matrix product AB (this being the main application of Winograd's identity), but the analogue of (2.10) for n-vectors cannot be employed in this fashion.

A further difference is that (2.9), and more generally (2.8), is numerically unstable in the sense that the best available normwise error bound is of the form

$$|fl(x^T y) - x^T y| \le c_n u \big(\|x\|_\infty + \|y\|_\infty\big)^2 + O(u^2), \tag{2.11}$$

which can be arbitrarily weaker than the bound

$$|fl(x^T y) - x^T y| \le c_n u \|x\|_\infty \|y\|_\infty + O(u^2),$$

which holds for conventional multiplication (and for (2.10)).

The instability of Winograd's identity was first pointed out by Brent [4], who proves a bound of the form (2.11). He shows that the instability can be overcome by scaling x and y so that $\|x\|_\infty \approx \|y\|_\infty$ before applying the identity.

(3) To put the stability of the 3M method into perspective it is worth noting that it is at least as stable as Strassen's method. The best available bound for Strassen's method for forming $C = AB$, where $A, B \in \mathbb{R}^{n \times n}$, is [12]

$$\|\widehat{C} - C\|_\infty \le f(n, n_0) u \|A\|_\infty \|B\|_\infty + O(u^2),$$

where $f(n, n_0) \approx n n_0^2 (n/n_0)^{3.6}$ and where $n_0 \le n$ is the threshhold such that conventional multiplication is used for matrices of dimension n_0 or less. Thus Strassen's method satisfies a normwise bound only, and has a potentially much larger constant in the bound than the 3M method.

(4) It is straightforward to show that if the 3M method is implemented using Strassen's method to form the real matrix products, then the computed complex $\widehat{C}$ satisfies

$$\|\widehat{C} - C\|_\infty \le 6(f(n, n_0) + 4) u \|A\|_\infty \|B\|_\infty + O(u^2). \tag{2.12}$$

In other words, the 3M method combined with Strassen's method has the same stability properties as Strassen's method alone.

(5) We have done numerical experiments in MATLAB to confirm the theoretical analysis. Our experience is that for "random" matrices the 3M method is quite likely to produce a computed answer of similar quality to that from conventional multiplication. (The same is true for Strassen's method; see [12].) However, it is easy to generate examples where instability occurs—for example, by generalizing the example at the beginning of this section.

3. Practical considerations. What method should we use to multiply complex matrices? If the best possible accuracy is required, or if execution time is not a primary concern, then the multiplication should be done in the conventional manner. When implementing conventional matrix multiplication in Fortran, we have the choice of splitting the computation into its real and imaginary parts at the beginning, as is necessary to apply the 3M method, or of using "complex arithmetic," which effectively means resorting to real arithmetic only at the scalar level. These two approaches carry out the same (real) arithmetic operations in different orders, and so satisfy the same error bounds (2.2) and (2.3). The choice of which approach to use can therefore be guided by considerations other than accuracy, such as the relative efficiency of real and complex arithmetic implementations, which depends on various factors, including memory reference time, the overhead of invoking complex arithmetic routines, and the intrinsic costs of real and complex arithmetic. The relative efficiency can vary greatly between machines and compilers. The LINPACK manual [8, p. 1.25, Appendix B] reports the execution times of `CGEFA` (complex LU factorization) and `SGEFA` (real LU factorization) for the LINPACK test sites (21 computing environments). For $n = 100$, the ratio "`CGEFA/SGEFA`" varies between 1.64 and 8.98, with an average of 4.31.

If a faster multiplication is desired, the most promising possibilities involve the 3M method and Strassen's method. Recent experience with Strassen's method on real matrices has shown that on certain machines it can produce useful speedups for n in the hundreds [1], [2]. If the computing environment is such that complex arithmetic is implemented very efficiently, it may be best to use Strassen's method alone in complex arithmetic. For example, in experiments in Algol-W on an IBM 360/67, Brent [3] found that a complex matrix multiplication took less than three times as long as a real matrix multiplication, for both the conventional method and Strassen's method. Thus it is probably not worth using the 3M method in this environment.

The evidence quoted above from the LINPACK manual suggests that in many Fortran environments complex arithmetic will exceed real arithmetic in cost by a factor of more than three. In such situations it is appropriate to use the 3M method in conjunction with Strassen's method, as is done in the ESSL library [13] and the UNICOS library [5], and as is discussed at the end of §2.

A prominent source of Fortran 77 matrix multiplication routines is the level-3 basic linear algebra subprograms (BLAS3) [9]. The BLAS3 specifications define what each routine must do but not how it must do it. Thus there is freedom of implementation, subject to the requirement of retaining numerical stability. One of the main uses of the BLAS3 is as modules in block algorithms for solving linear equation and eigenvalue problems, for example, in LAPACK [6]. Two important questions in this context are whether the block algorithms remain backward stable when they are built upon BLAS3 operations satisfying bounds of the form (2.12), and, if they do, whether the backward error results are sufficiently strong for a given application. In joint work with Demmel [7], we have shown that a wide class of block algorithms satisfy a backward error bound of the form (2.12) if the BLAS3 themselves satisfy (2.12). In combination with the work here, this provides motivation for preparing complex BLAS3 routines based on the 3M method combined with Strassen's method.

Acknowledgement. I thank Des Higham for his helpful comments on the manuscript and Phuong Vu for pointing out reference [10].

REFERENCES

[1] D. H. BAILEY, *Extra high speed matrix multiplication on the Cray-*2, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 603–607.

[2] D. H. BAILEY, K. LEE, AND H. D. SIMON, *Using Strassen's algorithm to accelerate the solution of linear systems*, J. Supercomputing, 4 (1991), pp. 357–371.

[3] R. P. BRENT, *Algorithms for matrix multiplication*, Tech. Report CS 157, Computer Science Department, Stanford University, Stanford, CA, 1970.

[4] ———, *Error analysis of algorithms for matrix multiplication and triangular decomposition using Winograd's identity*, Numer. Math., 16 (1970), pp. 145–156.

[5] CRAY RESEARCH INC., UNICOS *Math and Scientific Library Reference Manual*, Number SR-2081, Version 5.0, March 1989, Eagan, MN.

[6] J. W. DEMMEL, J. J. DONGARRA, J. J. DU CROZ, A. GREENBAUM, S. J. HAMMARLING, AND D. C. SORENSEN, *Prospectus for the development of a linear algebra library for high-performance computers*, Tech. Memorandum No. 97, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, 1987.

[7] J. W. DEMMEL AND N. J. HIGHAM, *Stability of block algorithms with fast level* 3 BLAS, LAPACK Working Note #22 and Numerical Analysis Report No. 188, University of Manchester, Manchester, England, 1990; to appear in ACM Trans. Math. Software.

[8] J. J. DONGARRA, J. R. BUNCH, C. B. MOLER, AND G. W. STEWART, LINPACK *Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[9] J. J. DONGARRA, J. J. DU CROZ, I. S. DUFF, AND S. J. HAMMARLING, *A set of level* 3 *basic linear algebra subprograms*, ACM Trans. Math. Software, 16 (1990), pp. 1–17.

[10] A. T. FAM, *Efficient complex matrix multiplication*, IEEE Trans. Comput., C-37 (1988), pp. 877–879.

[11] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[12] N. J. HIGHAM, *Exploiting fast matrix multiplication within the level* 3 BLAS, ACM Trans. Math. Software, 16 (1990), pp. 352–368.

[13] IBM, *Engineering and Scientific Subroutine Library, Guide and Reference, Release* 3, Fourth Edition (Program Number 5668-863), 1988.

[14] D. E. KNUTH, *The Art of Computer Programming, Volume* 2, *Seminumerical Algorithms*, Second Edition, Addison–Wesley, Reading, MA, 1981.

[15] V. STRASSEN, *Gaussian elimination is not optimal*, Numer. Math., 13 (1969), pp. 354–356.
[16] S. WINOGRAD, *A new algorithm for inner product*, IEEE Trans. Comput., C-18 (1968), pp. 693–694.
[17] ———, *On multiplication of* 2 × 2 *matrices*, Linear Algebra Appl., 4 (1971), pp. 381–388.