

# On the componentwise accuracy of complex floating-point division with an FMA

Claude-Pierre Jeannerod, Nicolas Louvet, Jean-Michel Muller

▶ **To cite this version:**

Claude-Pierre Jeannerod, Nicolas Louvet, Jean-Michel Muller. On the componentwise accuracy of complex floating-point division with an FMA. 21st IEEE Symposium on Computer Arithmetic, Apr 2013, Austin, TX, United States. pp.83–90. ensl-00734339v2

**HAL Id: ensl-00734339**

**https://ens-lyon.hal.science/ensl-00734339v2**

Submitted on 7 Feb 2013

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# On the componentwise accuracy of complex floating-point division with an FMA

Claude-Pierre Jeannerod
INRIA
claude-pierre.jeannerod@ens-lyon.fr

Nicolas Louvet
UCB Lyon 1
nicolas.louvet@ens-lyon.fr

Jean-Michel Muller
CNRS
jean-michel.muller@ens-lyon.fr

AriC INRIA project, Laboratoire LIP (CNRS, ENS de Lyon, INRIA, UCB Lyon 1), Université de Lyon, France

***Abstract*—This paper deals with the accuracy of complex division in radix-two floating-point arithmetic. Assuming that a fused multiply-add (FMA) instruction is available and that no underflow/overflow occurs, we study how to ensure high relative accuracy in the componentwise sense. Since this essentially reduces to evaluating accurately three expressions of the form $ac + bd$, an obvious approach would be to perform three calls to Kahan's compensated algorithm for 2 by 2 determinants. However, in the context of complex division, two of those expressions are such that $ac$ and $bd$ have the same sign, suggesting that cheaper schemes should be used here (since cancellation cannot occur). We first give a detailed accuracy analysis of such schemes for the sum of two nonnegative products, providing not only sharp bounds on both their absolute and relative errors, but also sufficient conditions for the output of one of them to coincide with the output of Kahan's algorithm. By combining Kahan's algorithm with this particular scheme, we then deduce two new division algorithms. Our first algorithm is a straight-line program whose componentwise relative error is always at most $5u + 13u^2$ with $u$ the unit roundoff; we also provide examples of inputs for which the error of this algorithm approaches $5u$, thus showing that our upper bound is essentially the best possible. When tests are allowed we show with a second algorithm that the bound above can be further reduced to $4.5u + 9u^2$, and that this improved bound is reasonably sharp.**

*Keywords.* Floating-point arithmetic; complex division; fused multiply-add (FMA); rounding error analysis.

## I. Introduction

For two complex numbers $x = a + ib$ and $y = c + id$ given by their real and imaginary parts, the result $z$ of the division of $x$ by $y$ is classically expressed as

$$z = \frac{ac + bd}{c^2 + d^2} + i\,\frac{bc - ad}{c^2 + d^2}. \tag{1}$$

When evaluating (1) in floating-point arithmetic and if neither underflow nor overflow occurs, then it is well known that high relative accuracy is ensured in the *normwise* sense [1, §3.6], [2, §4.5]. More precisely, assume standard binary floating-point arithmetic, in precision $p$ and with rounding 'to nearest even' (RN), and let $\widehat{z}$ denote the approximation to $z$ given by

$$\mathrm{Re}\,\widehat{z} = \mathrm{RN}\left(\frac{\mathrm{RN}\big(\mathrm{RN}(ac) + \mathrm{RN}(bd)\big)}{\mathrm{RN}\big(\mathrm{RN}(c^2) + \mathrm{RN}(d^2)\big)}\right) \tag{2}$$

for its real part and by a similar expression for its imaginary part $\mathrm{Im}\,\widehat{z}$. Then the normwise relative error

$$E_n(\widehat{z}) = |\widehat{z} - z|/|z|$$

is bounded by $\gamma u + O(u^2)$, where $u = 2^{-p}$ is the unit roundoff and $\gamma$ is a modest constant. For example, one can take $\gamma$ equal to $3 + \sqrt{5} = 5.236...$, as noted in [3, §3.6]. (A detailed proof of this fact is given in appendix.)

However, in the *componentwise* sense, the approximation $\widehat{z}$ can be highly inaccurate: there exist values for $a$, $b$, $c$, $d$ such that the componentwise relative error

$$E_c(\widehat{z}) = \max\big(|\mathrm{Re}\,\widehat{z} - \mathrm{Re}\,z|/|\mathrm{Re}\,z|, |\mathrm{Im}\,\widehat{z} - \mathrm{Im}\,z|/|\mathrm{Im}\,z|\big)$$

is (much) larger than one. For example, the input

$$(a, b, c, d) = (N, N + 1, -N, N - 1) \tag{3}$$

with $N = 2^{p-1} + 2^{p-2} - 1$ leads to $E_c(\widehat{z}) \approx 9.0 \times 10^{15} \gg 1$ when $p = 53$ and $\widehat{z}$ is computed according to (2). Here, the input consists of normal floating-point numbers and there is no underflow/overflow, but since $ac$ and $bd$ have opposite signs and similar magnitudes a cancellation occurs during the evaluation of $ac + bd$ as

$$\widehat{f_o} = \mathrm{RN}\big(\mathrm{RN}(ac) + \mathrm{RN}(bd)\big). \tag{4}$$

In this paper, we are interested in performing complex divisions very accurately in the componentwise sense, assuming that a fused multiply-add (FMA) instruction is available and that no underflow/overflow occurs. The FMA evaluates expressions of the form $ab + c$ with one rounding error only and since it is required by the 2008 revision of the IEEE 754 standard [4], one can expect that it will soon belong to the instruction set of most general-purpose processors.

From (1) it is clear that in order to achieve high relative accuracy in the componentwise sense, it suffices to evaluate with high relative accuracy two-dimensional dot products, that is, expressions of the form

$$g = ac + bd.$$

With an FMA, a naive computation would produce, say,

$$\widehat{f} = \mathrm{RN}\big(ac + \mathrm{RN}(bd)\big), \tag{5}$$

so that the expression in (2) can for example be replaced by

$$\mathrm{Re}\,\widehat{z} = \mathrm{RN}\left(\frac{\mathrm{RN}\big(ac + \mathrm{RN}(bd)\big)}{\mathrm{RN}\big(c^2 + \mathrm{RN}(d^2)\big)}\right).$$

Implementing complex division by means of such schemes is, however, not enough for our purposes. Indeed, considering

again (3) and $p = 53$ now gives $E_c(\widehat{z}) \approx 4.5 \times 10^{15}$, but this componentwise relative error is still (much) larger than one.

To ensure high accuracy, the FMA is typically used further to recover the rounding errors committed when evaluating the products $ac$ and $bd$, and then to compensate for them by addition; see for example [5, p. 273]. It turns out that if the FMA is already used to get $\widehat{f}$ as in (5), then only the rounding error for $bd$ needs to be recovered. This is the basis of the algorithm below, attributed to Kahan in [1, p. 65]:

**algorithm** Kahan$(a, b, c, d)$
  $\widehat{w} := \mathrm{RN}(bd)$;
  $e := \mathrm{RN}(bd - \widehat{w})$; // this operation is exact: $e = bd - \widehat{w}$.
  $\widehat{f} := \mathrm{RN}(ac + \widehat{w})$;
  $\widehat{g} := \mathrm{RN}\big(\widehat{f} + e\big)$;
  **return** $\widehat{g}$;

The error $e = bd - \mathrm{RN}(bd)$ is computed exactly thanks to the FMA instruction, and then added to $\widehat{f}$ in order to yield the approximation $\widehat{g}$ to $g = ac + bd$. In [6] it was shown that in the absence of underflow and overflow

$$|\widehat{g} - g| \leqslant 2u|g|, \tag{6}$$

so that Kahan's algorithm always approximates $g$ to high relative accuracy; it was also shown that the bound $2u$ is essentially best possible (even if $ac$ and $bd$ can be swapped) via the explicit construction of floating-point numbers $a, b, c, d$ for which $|\widehat{g} - g|/|g| = 2u + O(u^2)$.

Consequently, by using Kahan's algorithm to evaluate $ac + bd$, $bc - ad$, $c^2 + d^2$ in (1) and then performing two floating-point divisions, we obviously get high componentwise relative accuracy: considering for example the real part,

$$\mathrm{Re}\,\widehat{z} = \frac{g(1+\epsilon)}{(c^2+d^2)(1+\epsilon')}(1+\epsilon''),$$

where $|\epsilon|, |\epsilon'| \leqslant 2u$ by (6) and where the relative error $|\epsilon''|$ of floating-point division is bounded by $u$. Hence the real part of $\widehat{z}$ has the form $(1+\theta)\cdot \mathrm{Re}\, z$ with $|\theta| \leqslant \frac{1+2u}{1-2u}(1+u) - 1$ and since the same holds for the imaginary part, it follows that the componentwise relative error satisfies

$$E_c(\widehat{z}) \leqslant 5u + O(u^2). \tag{7}$$

Yet this approach raises two issues, which we address in this paper. First, among the three dot products $ac + bd$, $bc - ad$, $c^2 + d^2$ appearing in (1) only one of them can cancel: if $abcd \geqslant 0$ cancellation can occur only when evaluating $bc - ad$, while if $abcd < 0$ cancellation can occur only when evaluating $ac + bd$. Hence, there is always exactly one potentially 'difficult' case (one of the numerators) and two 'easy' cases (the denominator $c^2+d^2$ and the other numerator). Although it seems obvious not to use Kahan's compensated algorithm for such easy cases where cancellation cannot occur, the behavior of simpler algorithms like those producing $\widehat{f_o}$ in (4) or $\widehat{f}$ in (5) deserves further investigation: Is the relative error bound $2u$ given in (6) for $\widehat{g}$ also a sharp bound for $\widehat{f_o}$ and $\widehat{f}$? How are $\widehat{f}$ and $\widehat{g}$ related, and when do they coincide?

We address this first issue in Section II by presenting a thorough study of the case '$ac + bd$ with $ac \geqslant 0$ and $bd \geqslant 0$'. In particular,

- we show that without any further assumption on $a, b, c, d$ the bound $2u$ is indeed already sharp for $\widehat{f_o}$ and $\widehat{f}$;
- we also show that either $\widehat{f}$ and $\widehat{g}$ coincide, or one of them is a faithfully rounded value of the exact result $g$;
- third, we show that if $0 \leqslant bd \leqslant ac$ then $\widehat{f} = \widehat{g}$ and a sharp bound is $\frac{3}{2}u$.

Along with these relative error bounds, we also systematically provide bounds on absolute errors, which we express in terms of ulps (units in the last place) of $g$. In particular, this provides a good illustration of the impact the FMA can have on error bounds: for $|\widehat{f} - g|$ and $|\widehat{f_o} - g|$, that is, for the naive evaluation of $g$ with or without FMA, sharp bounds are proven to be $\mathrm{ulp}(g)$ and $\frac{5}{4}\mathrm{ulp}(g)$, respectively. Finally, in the more constrained case $g = c^2 + d^2$ of a sum of two squares, all our error bounds are shown to remain either sharp or reasonably tight for the basic IEEE formats. By combining the results in Section II with the bound in (6) we immediately deduce two other FMA-based, highly accurate algorithms for complex floating-point division:

- a straight-line program that evaluates the denominator $c^2+d^2$ using (5) and the two numerators using Kahan's algorithm, and whose relative error is again clearly bounded as in (7);
- an algorithm using tests in order to always reduce the error bound on $c^2 + d^2$ to $\frac{3}{2}u$, thus improving (7) to

$$E_c(\widehat{z}) \leqslant \tfrac{9}{2}u + O(u^2). \tag{8}$$

The second issue concerns the sharpness of the bounds in (7) and (8). Indeed, although it is straightforward to derive the two algorithms just mentioned and to perform their rounding error analysis, it is a priori not clear that sharp bounds for the numerators and for the denominator will lead to sharp bounds for the whole complex division algorithms.

Section III is devoted to this second issue, and we show in particular through numerical examples that for all the basic binary IEEE formats

- the error bound in $5u+O(u^2)$ is sharp for the straight-line program;
- the error bound in $\frac{9}{2}u + O(u^2)$ is reasonably sharp for the algorithm using tests.

Furthermore, for the straight-line program and when the precision $p$ is even, we show that the bound $5u + O(u^2)$ is asymptotically optimal by explicitly constructing an input $(a, b, c, d)$ for which $E_c(\widehat{z}) = 5u - O(u^{3/2})$.

**Assumptions and notation.** Throughout this paper we make the customary assumption that $p \geqslant 2$. Also, we write ulp to denote the *unit in the last place* function: $\mathrm{ulp}(0) = 0$ and for any nonzero real number $t$, $\mathrm{ulp}(t)$ is the unique integer power of two such that $2^{p-1} \leqslant |t|/\mathrm{ulp}(t) < 2^p$. Furthermore, we assume that underflows and overflows never occur, so that

$$|\mathrm{RN}(t) - t| \leqslant \tfrac{1}{2}\mathrm{ulp}(t) \leqslant u|t| \quad \text{for any real number } t. \tag{9}$$

In particular, the rightmost inequality leads to the so-called *standard model* of floating-point arithmetic [1, p. 40]. For some of our proofs we will also use the following classical modified version of (9): for any real number $t$,

$$|\mathrm{RN}(t) - t| \leqslant \tfrac{1}{2}\mathrm{ulp}(\mathrm{RN}(t)) \leqslant u|\mathrm{RN}(t)|.$$

## II. On the sum of two nonnegative products

In this section we focus on the approximation of $g = ac+bd$ by $\widehat{f_o}$, $\widehat{f}$, or $\widehat{g}$ in the case where

$$ac \geqslant 0 \qquad \text{and} \qquad bd \geqslant 0. \tag{10}$$

From [6] and [7] we already know that $\mathrm{ulp}(g)$ and $2u$ are sharp absolute/relative error bounds for $\widehat{g}$, and that $2u$ is a relative error bound for $\widehat{f_o}$. As a first extension of these results, we show in Section II-A that this bound $2u$ on $|\widehat{f_o}-g|/g$ is sharp, and also that a sharp bound on $|\widehat{f_o} - g|$ is $\frac{5}{4}\mathrm{ulp}(g)$. Then, we show in Section II-B that similarly to $\widehat{g}$, sharp absolute and relative error bounds for $\widehat{f}$ are $\mathrm{ulp}(g)$ and $2u$. Since $\widehat{f}$ and $\widehat{g}$ share the same error bounds, we study further how they relate to each other. Specifically, we prove in Section II-C that either $\widehat{f}$ and $\widehat{g}$ are equal, or one of them is a faithfully rounded value of $g$. We conclude in Section II-D by assuming further that

$$\mathrm{ulp}(bd) \leqslant \mathrm{ulp}(ac). \tag{11}$$

In this case, which may be of interest when tests are allowed and which is implied by $0 \leqslant bd \leqslant ac$, we prove that $\widehat{f}$ and $\widehat{g}$ always coincide, thus showing that the compensation step in Kahan's algorithm simply has no effect on the quality of the result. Furthermore, the error bounds $\mathrm{ulp}(g)$ and $2u$ given in the general case can now be improved to $\frac{3}{4}\mathrm{ulp}(g)$ and $\frac{3}{2}u$, which are shown to be sharp when (11) holds.

The table below summarizes the bounds presented in this section:

| $\widehat{r}$ | bound on $\lvert\widehat{r} - g\rvert$ | bound on $\lvert\widehat{r} - g\rvert/g$ |
|---|---|---|
| $\widehat{f_o}$ | $\frac{5}{4}\mathrm{ulp}(g)$ | $2u$ [7] |
| $\widehat{f}$ | $\mathrm{ulp}(g)$ | $2u$ |
| $\widehat{g}$ | $\mathrm{ulp}(g)$ [6] | $2u$ [6] |
| $\widehat{f} = \widehat{g}$ for (11) | $\frac{3}{4}\mathrm{ulp}(g)$ | $\frac{3}{2}u$ |

The sharpness of these error bounds is established using examples of the form $ac+bd$ parametrized by the precision $p$. We also investigated the sharpness of these error bounds for the evaluation of sums of squares $c^2 + d^2$: in most cases we show that the bounds remain asymptotically optimal as $p \to \infty$, but for some of the algorithms considered we have not found examples parametrized by $p$ to show this. In those cases, however, we illustrate the quality of the error bounds with examples in precisions $p = 53, 113$, which correspond to the binary64 and binary128 formats of the IEEE 754-2008 standard [4].

### A. Error bounds for the approximation $\widehat{f_o}$ without FMA

From [7] it is known that when approximating $g$ as in (10) using $\widehat{f_o}$ as in (4), the relative error is always at most of $2u$:

$$|\widehat{f_o} - g| \leqslant 2ug.$$

Since here $g = |g|$, this bound thus is the same as the bound obtained in (6) using Kahan's algorithm. Furthermore, just like for Kahan's algorithm, it is asymptotically optimal, as the example below shows.

**Example 1** ($2ug$ is an asymptotically optimal bound on $|\widehat{f_o} - g|$)**.** *For any $p \geqslant 2$, consider*

$$a = b = 2^p - 1, \quad c = 2^{p-1} + 2, \quad d = 2^{p-1} + 1.$$

*One easily shows that $g = 2^{2p} + 2^{p+1} - 3$ and $\widehat{f_o} = 2^{2p}$, so that we have*

$$\frac{|\widehat{f_o} - g|}{g} = \frac{2^{2p} - 3 \cdot 2^{p-1}}{2^{2p} + 2^{p+1} - 3} \cdot 2u.$$

*Thus, $|\widehat{f_o} - g|/(2ug)$ is in $1 - O(2^{-p})$ as $p \to \infty$.*

Let us now give an error bound in ulps for $\widehat{f_o}$ and an example showing the asymptotic optimality of this bound.

**Property 1.** $|\widehat{f_o} - g| \leqslant \frac{5}{4}\mathrm{ulp}(g)$.

*Proof:* Let $\widehat{v} = \mathrm{RN}(ac)$, $\widehat{w} = \mathrm{RN}(bd)$, and $f_o = \widehat{v} + \widehat{w}$. Then $|\widehat{f_o} - g| \leqslant |\widehat{f_o} - f_o| + |f_o - g|$ and we shall prove first that $|f_o - g|$ is at most $\frac{3}{4}\mathrm{ulp}(g)$. With $m = \min\{ac, bd\}$ and $m' = \max\{ac, bd\}$, we have

$$\begin{aligned} |f_o - g| &\leqslant |\widehat{v} - ac| + |\widehat{w} - bd| \\ &\leqslant \tfrac{1}{2}\mathrm{ulp}(ac) + \tfrac{1}{2}\mathrm{ulp}(bd) = \tfrac{1}{2}\mathrm{ulp}(m) + \tfrac{1}{2}\mathrm{ulp}(m'). \end{aligned}$$

From $0 \leqslant m \leqslant \frac{g}{2}$ and $0 \leqslant m' \leqslant g$ we get $\mathrm{ulp}(m) \leqslant \frac{1}{2}\mathrm{ulp}(g)$ and $\mathrm{ulp}(m') \leqslant \mathrm{ulp}(g)$, and then

$$|f_o - g| \leqslant \tfrac{3}{4}\mathrm{ulp}(g). \tag{12}$$

We conclude the proof by using the same case analysis as in [7, p. 1470]:

- If $\mathrm{ulp}(f_o) \leqslant \mathrm{ulp}(g)$ then $|\widehat{f_o} - f_o| \leqslant \frac{1}{2}\mathrm{ulp}(f_o) \leqslant \frac{1}{2}\mathrm{ulp}(g)$, which can be combined with (12) to give the result;
- If $\mathrm{ulp}(g) < \mathrm{ulp}(f_o)$ then $0 \leqslant g < 2^k \leqslant f_o$ for some integer $k$. Hence, using (12) and $\mathrm{ulp}(g) \leqslant \frac{1}{2}\mathrm{ulp}(f_o)$,

  $$0 \leqslant f_o - 2^k < f_o - g \leqslant \tfrac{3}{4}\mathrm{ulp}(g) \leqslant \tfrac{3}{8}\mathrm{ulp}(f_o).$$

  Consequently, the floating-point number $2^k$ must be equal to $\widehat{f_o} = \mathrm{RN}(f_o)$, and then $0 < \widehat{f_o} - g \leqslant f_o - g \leqslant \frac{3}{4}\mathrm{ulp}(g)$. Thus, in this case $|\widehat{f_o} - g|$ is also at most $\frac{5}{4}\mathrm{ulp}(g)$.

■

**Example 2** ($\frac{5}{4}\mathrm{ulp}(g)$ is an asymptotically optimal bound on $|\widehat{f_o} - g|$)**.** *Assuming $p \geqslant 7$, let*

$$a = c = 2^{p-1} + 2^{p-3} + 1, \quad b = d = 2^{p-1} + 2^{p-2} + 1.$$

*One easily shows that $g = 61 \cdot 2^{2p-6} + 2^{p+1} + 3 \cdot 2^{p-2} + 2$, $\widehat{f_o} = 61 \cdot 2^{2p-6} + 2^{p+2}$, and $\mathrm{ulp}(g) = 2^p$, so that*

$$|\widehat{f_o} - g| = \left(\tfrac{5}{4} - 2^{1-p}\right)\mathrm{ulp}(g).$$

**Special case of the sum of two squares.** When computing an expression of the form $c^2+d^2$ using (4), Example 2 shows that the absolute error bound $\frac{5}{4}\text{ulp}(g)$ is asymptotically optimal. Still in the particular case of an expression of the form $c^2+d^2$, Example 3 below shows that the relative bound $2u$ for $\widehat{f_o}$ remains asymptotically optimal when $p$ is even. When $p$ is odd, we did not manage to find an example parametrized by $p$ to prove that the error bound $2u$ is asymptotically optimal; nevertheless, when $p$ is odd, the quality of the bound is illustrated in Table I for $p = 53, 113$.

**Example 3.** *Assuming $p \geqslant 2$ is even, let us consider*

$$c = 2^{p/2-1} \quad \textit{and} \quad d = 2^{p-1} + 2^{p/2-1}.$$

*We have $g = 2^{2p-2} + 2^{\frac{3p}{2}-1} + 2^{p-1}$ and $\widehat{f_o} = 2^{2p-2} + 2^{\frac{3p}{2}-1}$, so that*

$$\frac{|\widehat{f_o} - g|}{g} = \frac{2^{2p-2}}{2^{2p-2} + 2^{\frac{3p}{2}-1} + 2^{p-1}} \cdot 2u.$$

*Hence $|\widehat{f_o} - g|/(2ug)$ is in $1 - O(2^{-\frac{p}{2}})$ as $p \to \infty$.*

• Relative error in $\widehat{f_o}$ close to $2u$:

| $p$ | example |
|---|---|
| 53 | $c = 4503608217436160$<br>$d = 4503599711256576$<br>$\lvert\widehat{f_o} - g\rvert/(ug) = 1.9374\ldots$ |
| 113 | $c = 5192296858544272361496373058600960$<br>$d = 5192296858534827718602488876630016$<br>$\lvert\widehat{f_o} - g\rvert/(ug) = 1.9374\ldots$ |

• Relative error in $\widehat{f} = \widehat{g}$ close to $2u$:

| $p$ | example |
|---|---|
| 53 | $c = 8426657115275263$<br>$d = 4503608217436160 \cdot 2^{26}$<br>$\lvert\widehat{f} - g\rvert/(ug) = 1.9980\ldots$ |
| 113 | $c = 9715274200149150133070733366001663$<br>$d = 5192296858544272361496373058600960 \cdot 2^{56}$<br>$\lvert\widehat{f} - g\rvert/(ug) = 1.9980\ldots$ |

• For $|d| \leqslant |c|$, errors in $\widehat{f} = \widehat{g}$ close to $\frac{3}{4}\text{ulp}(g)$ and $\frac{3}{2}u$:

| $p$ | example |
|---|---|
| 53 | $c = 4503600164241409$<br>$d = 4503599900000256$<br>$\lvert\widehat{f} - g\rvert/\text{ulp}(g) = 0.7480\ldots$<br>$\lvert\widehat{f} - g\rvert/(ug) = 1.4960\ldots$ |
| 113 | $c = 5192296858534828204991248632643585$<br>$d = 5192296858534827921264472108302336$<br>$\lvert\widehat{f} - g\rvert/\text{ulp}(g) = 0.7480\ldots$<br>$\lvert\widehat{f} - g\rvert/(ug) = 1.4960\ldots$ |

TABLE I
EXAMPLES OF THE FORM $c^2 + d^2$, FOR $p = 53, 113$.

### B. Error bounds for the naive approximation $\widehat{f}$ with an FMA

We now turn to the computation of $\widehat{f}$ as in (5). The property and example below show that the relative error bound $2u$ obtained for $\widehat{f_o}$ in Section II-A is still sharp, while the absolute error is now tightly bounded by one ulp of the exact result.

**Property 2.** $|\widehat{f} - g| \leqslant \text{ulp}(g) \leqslant 2ug$.

*Proof:* Since $g = f + e$, we have $|\widehat{f} - g| \leqslant |\widehat{f} - f| + |e|$. By definition, $|e| \leqslant \frac{1}{2}\text{ulp}(bd)$ and using $0 \leqslant bd \leqslant g$ leads to

$$|e| \leqslant \tfrac{1}{2}\text{ulp}(g). \tag{13}$$

On the other hand, we can check that $|\widehat{f} - f| \leqslant \frac{1}{2}\text{ulp}(g)$ as follows:

- If $\text{ulp}(f) \leqslant \text{ulp}(g)$ then $|\widehat{f} - f| \leqslant \frac{1}{2}\text{ulp}(f) \leqslant \frac{1}{2}\text{ulp}(g)$;
- If $\text{ulp}(g) < \text{ulp}(f)$ then $0 \leqslant g < 2^k \leqslant f$ for some integer $k$. Since $\widehat{f} = \text{RN}(f)$ and $2^k$ is a floating-point number, this implies $|\widehat{f} - f| \leqslant f - 2^k < f - g = -e \leqslant \frac{1}{2}\text{ulp}(g)$.

Thus, in both cases we have $|\widehat{f} - g| \leqslant \text{ulp}(g)$ as wanted, and (9) implies further that $\text{ulp}(g) \leqslant 2ug$. ∎

**Example 4** ($\text{ulp}(g)$ and $2ug$ are asymptotically optimal bounds on $|\widehat{f} - g|$; Example 6.2 in [6])**.** *For any $p \geqslant 2$, consider*

$$a = 2^p - 2, \quad b = (2^p - 1)2^p, \quad c = d = 2^{p-1} + 1.$$

*One easily shows that $g = 2^{3p-1} + 2^{2p} - 2^p - 2$, $\text{ulp}(g) = 2^{2p}$, and $\widehat{f} = 2^{3p-1}$. Hence $|\widehat{f} - g| = (1 - 2^{-p} - 2^{1-2p})\text{ulp}(g)$ and*

$$\frac{|\widehat{f} - g|}{g} = \frac{2^{3p-1} - 2^{2p-1} - 2^p}{2^{3p-1} + 2^{2p} - 2^p - 2} \cdot 2u.$$

*Consequently, both $|\widehat{f} - g|/\text{ulp}(g)$ and $|\widehat{f} - g|/(2ug)$ are in $1 - O(2^{-p})$ as $p \to \infty$. In this example, since $e = 2^{2p-1} - 2^p$ and $\text{ulp}(\widehat{f}) = 2^{2p}$, it may also be noticed that the result $\widehat{g}$ returned by Kahan's algorithm satisfies $\widehat{g} = \widehat{f}$.*

**Special case of the sum of two squares.** Examples 6.3 and 6.4 in [6] are of the form $c^2 + d^2$, with $\widehat{f} = \widehat{g}$. Hence we already know from that paper that:

- the absolute error bound $|\widehat{f} - g| \leqslant \text{ulp}(g)$ is asymptotically optimal for $p$ even, and optimal for $p$ odd;
- the relative error bound $|\widehat{f} - g| \leqslant 2ug$ is asymptotically optimal when $p$ is even.

When $p$ is odd, numerical examples of the form $c^2 + d^2$ in precisions 53 and 113 are given in Table I that illustrate the quality of the relative error bound.

### C. Relationship between $\widehat{f}$ and $\widehat{g}$

The sharp bounds we have so far for $\widehat{f}$ and $\widehat{g}$ are the same: one ulp of $g$ for absolute errors, and $2u$ for relative errors. In this section, we go one step further by investigating more precisely how $\widehat{f}$ and $\widehat{g}$ are related. We show in Theorem 1 below that if $\widehat{f}$ and $\widehat{g}$ differ then they must be two consecutive floating-point numbers surrounding the exact value $g$. For this, we start with a lemma giving a necessary condition for $\widehat{f}$ to be different from $\widehat{g}$.

**Lemma 1.** *If $\widehat{f} \neq \widehat{g}$ then*

$$2^k \leqslant \widehat{w} < \widehat{f} < 2^{k+1} \textit{ for some } k \in \mathbb{Z}, \tag{14a}$$

*and*

$$|e| = \tfrac{1}{2}\mathrm{ulp}(\widehat{w}) = \tfrac{1}{2}\mathrm{ulp}(\widehat{f}). \tag{14b}$$

*Proof:* Using $0 \leqslant \widehat{w} \leqslant ac + \widehat{w}$ together with the monotonicity of rounding, we have

$$0 \leqslant \widehat{w} \leqslant \widehat{f}.$$

If $\widehat{f}$ is zero, this implies $\widehat{w} = 0$ and thus $bd = e = 0$, so that $\widehat{g} = 0 = \widehat{f}$. Hence $\widehat{f} \neq \widehat{g}$ implies $\widehat{f}$ is positive and, therefore, $2^k \leqslant \widehat{f} < 2^{k+1}$ for some integer $k$.

(i) If $\widehat{w} < 2^k$ then $\mathrm{ulp}(\widehat{w}) \leqslant \frac{1}{2}\mathrm{ulp}(\widehat{f})$ and then $|e| \leqslant \frac{1}{2}\mathrm{ulp}(\widehat{w}) \leqslant \frac{1}{4}\mathrm{ulp}(\widehat{f})$, so that $\widehat{g} = \mathrm{RN}(\widehat{f} + e)$ equals $\widehat{f}$.

(ii) If $\widehat{w} = \widehat{f}$ then $\widehat{g} = \mathrm{RN}(\widehat{w} + e) = \mathrm{RN}(bd) = \widehat{w}$ and thus $\widehat{f} = \widehat{g}$.

From (i) and (ii) we deduce that $\widehat{f} \neq \widehat{g}$ implies the condition in (14a). This condition gives in particular $\mathrm{ulp}(\widehat{w}) = \mathrm{ulp}(\widehat{f})$ and, since $\widehat{f}$ is a floating-point number,

$$2^k + \mathrm{ulp}(\widehat{f}) \leqslant \widehat{f} \leqslant 2^{k+1} - \mathrm{ulp}(\widehat{f}). \tag{15}$$

Furthermore, by definition $|e| \leqslant \frac{1}{2}\mathrm{ulp}(\widehat{w})$ and, on the other hand, $|e| < \frac{1}{2}\mathrm{ulp}(\widehat{w})$ implies $\mathrm{RN}(\widehat{f} + e) = \widehat{f}$. Thus, $\widehat{f} \neq \widehat{g}$ also implies the condition in (14b). ∎

**Theorem 1.** *Let* $m = \min(\widehat{f}, \widehat{g})$ *and* $m' = \max(\widehat{f}, \widehat{g})$. *If* $\widehat{f} \neq \widehat{g}$ *then* $g \in [m, m']$ *and* $m' = m + \mathrm{ulp}(m)$.

*Proof:* First, by Lemma 1 we have $|e| = \frac{1}{2}\mathrm{ulp}(\widehat{f})$. Hence, if $e \geqslant 0$ then $\widehat{g} = \mathrm{RN}(\widehat{f} + \frac{1}{2}\mathrm{ulp}(\widehat{f}))$ must be either $\widehat{f}$ or $\widehat{f} + \mathrm{ulp}(\widehat{f})$, while if $e < 0$ it must be either $\widehat{f} - \mathrm{ulp}(\widehat{f})$ or $\widehat{f}$. Since $\widehat{f} \neq \widehat{g}$, we deduce that in both cases

$$\widehat{g} = \widehat{f} + 2e. \tag{16}$$

Note also that $|\widehat{f} - \widehat{g}| = \mathrm{ulp}(m)$. Indeed, if $m = \widehat{f}$ then this follows immediately from (16); else, we have $\widehat{g} < \widehat{f}$ and, since $\widehat{g}$ is a floating-point number, using (15) gives $\mathrm{ulp}(\widehat{g}) = \mathrm{ulp}(\widehat{f})$. Together with (16), this implies $|\widehat{f} - \widehat{g}| = \mathrm{ulp}(\widehat{g}) = \mathrm{ulp}(m)$.

Writing $\epsilon$ for the error committed when rounding $ac + \widehat{w}$ to nearest, we have $\widehat{f} = ac + \widehat{w} + \epsilon$ and $|\epsilon| \leqslant \frac{1}{2}\mathrm{ulp}(\widehat{f}) = |e|$. Hence, using the identity $e = bd - \widehat{w}$,

$$g = \widehat{f} + e - \epsilon.$$

If $e > 0$ then $-\epsilon \in [-e, e]$, so that $g$ belongs to $[\widehat{f}, \widehat{f} + 2e] = [\widehat{f}, \widehat{g}]$; if $e \leqslant 0$ then $-\epsilon \in [e, -e]$ and in this case $g$ belongs to $[\widehat{f} + 2e, \widehat{f}] = [\widehat{g}, \widehat{f}]$. We conclude that $g$ ranges in $[m, m']$ with $m' - m = |\widehat{f} - \widehat{g}| = \mathrm{ulp}(m)$. ∎

Recall from [8, Definition 2.3] that a *faithful rounding* of a real number $t$ is any floating-point number $s$ whose floating-point predecessor and successor satisfy the strict inequalities $\mathrm{pred}(s) < t < \mathrm{succ}(s)$. Consequently, Theorem 1 shows in particular that if $\widehat{f}$ and $\widehat{g}$ differ then at least one of them must be a faithful rounding of the exact result $g$.

Since $|\widehat{f} - \widehat{g}| = \mathrm{ulp}(m)$ and $0 \leqslant m \leqslant g$, another consequence of the theorem above is that

$$|\widehat{f} - \widehat{g}| \leqslant \mathrm{ulp}(g).$$

This bound improves upon the bound $|\widehat{f} - \widehat{g}| \leqslant 2\mathrm{ulp}(g)$ immediately implied by $|\widehat{f} - g| \leqslant \mathrm{ulp}(g)$ and $|\widehat{g} - g| \leqslant \mathrm{ulp}(g)$; furthermore, it is shown to be optimal in the example below.

**Example 5** ($\mathrm{ulp}(g)$ is an optimal bound on $|\widehat{f} - \widehat{g}|$)**.** *Assuming* $p \geqslant 4$*, let us consider*

$$a = c = 2^{p-1} + 1, \quad b = 2^p + 2^{\lceil \frac{p}{2} \rceil}, \quad d = 2^p + 2^{\lfloor \frac{p}{2} \rfloor}.$$

*Then one can check that* $|\widehat{f} - \widehat{g}| = \mathrm{ulp}(g)$. *It may also be checked that in this example the result computed by the naive algorithm with FMA is always more accurate than the one computed with Kahan's algorithm, since we have*

$$|\widehat{f} - g| = 2^{-p-1}\mathrm{ulp}(g) \quad \textit{and} \quad |\widehat{g} - g| = (1 - 2^{-p-1})\mathrm{ulp}(g).$$

*D. Special case where* $\mathrm{ulp}(bd) \leqslant \mathrm{ulp}(ac)$

The goal of this section is to show that if our input $a, b, c, d$ satisfy (10) and (11), then $\widehat{f}$ is always equal to $\widehat{g}$ and the error bounds $\mathrm{ulp}(g)$ and $2u$ can be decreased further.

**Lemma 2.** *If* $\mathrm{ulp}(bd) \leqslant \mathrm{ulp}(ac)$ *then* $|e| \leqslant \frac{1}{4}\mathrm{ulp}(g)$.

*Proof:* Since $|e| \leqslant \frac{1}{2}\mathrm{ulp}(bd)$, it suffices to show that $\mathrm{ulp}(bd) \leqslant \frac{1}{2}\mathrm{ulp}(g)$. We do this by considering two cases.

If $bd \leqslant ac$ then $bd \leqslant g/2$ and thus $\mathrm{ulp}(bd) \leqslant \frac{1}{2}\mathrm{ulp}(g)$.

Assume now that $ac < bd$. This implies $\mathrm{ulp}(ac) \leqslant \mathrm{ulp}(bd)$ as well as $g > 0$ and $ac < g/2$. Consequently, $\mathrm{ulp}(bd) = \mathrm{ulp}(ac) \leqslant 2^{1-p}ac < 2^{-p}g < \mathrm{ulp}(g)$, and the conclusion follows from $\mathrm{ulp}(bd) < \mathrm{ulp}(g)$ being equivalent to $\mathrm{ulp}(bd) \leqslant \frac{1}{2}\mathrm{ulp}(g)$. ∎

**Theorem 2.** *If* $\mathrm{ulp}(bd) \leqslant \mathrm{ulp}(ac)$ *then* $\widehat{f} = \widehat{g}$.

*Proof:* If $\widehat{f} = 0$ then it has already been seen in the proof of Lemma 1 that $\widehat{g} = 0$ as well. Let us now assume that $\widehat{f} > 0$. Since $\widehat{f} = \mathrm{RN}(f)$ is a floating-point number, we have $\widehat{f} \leqslant (2^p - 1)\mathrm{ulp}(\widehat{f})$ and it follows that for RN 'to nearest even'

$$f < \left(2^p - \tfrac{1}{2}\right)\mathrm{ulp}(\widehat{f}).$$

On the other hand, $0 \leqslant \widehat{w} \leqslant f = ac + \widehat{w}$, which implies $\mathrm{ulp}(\widehat{w}) \leqslant \mathrm{ulp}(f) \leqslant \mathrm{ulp}(\widehat{f})$ and then

$$|e| \leqslant \tfrac{1}{2}\mathrm{ulp}(\widehat{f}).$$

Hence $g = f + e \leqslant f + |e| < 2^p\mathrm{ulp}(\widehat{f})$ and, using $g \geqslant 0$,

$$\mathrm{ulp}(g) \leqslant \mathrm{ulp}(\widehat{f}).$$

By Lemma 2 we thus obtain $|e| \leqslant \frac{1}{4}\mathrm{ulp}(\widehat{f})$, from which we conclude that $\widehat{g} = \mathrm{RN}(\widehat{f} + e) = \widehat{f}$. ∎

The condition $\mathrm{ulp}(bd) \leqslant \mathrm{ulp}(ac)$ is sufficient but *not* necessary to have $\widehat{f}$ equal to $\widehat{g}$; see Example 4. In addition, this condition is equivalent to

$$\text{exponent of } bd \leqslant \text{exponent of } ac,$$

and, for $ac$ and $bd$ nonnegative, it is implied by

$$bd \leqslant ac.$$

Furthermore, in this special case where both (10) and (11) hold, Property 3 below shows that we can also improve the bounds we had given for the general case where only (10) holds. To get this, all we have to do is to replace, in the proof of Property 2, the bound in (13) by the bound in Lemma 2.

**Property 3.** *If* $\text{ulp}(bd) \leqslant \text{ulp}(ac)$ *then* $|\widehat{f} - g| \leqslant \frac{3}{4}\text{ulp}(g) \leqslant \frac{3}{2}ug$.

The next example shows that these new bounds are sharp.

**Example 6** (In the special case where $\text{ulp}(bd) \leqslant \text{ulp}(ac)$ the bound $\frac{3}{4}\text{ulp}(g)$ is optimal and the bound $\frac{3}{2}ug$ is asymptotically optimal)**.** *Assuming* $p \geqslant 6$*, consider*

$$a = 2^{p-1}+2^{\lfloor \frac{p}{2} \rfloor},\ b = c = 2^{p-1}+2^{\lceil \frac{p}{2} \rceil -1},\ d = 2^{p-1}+2^{\lfloor \frac{p}{2} \rfloor -1}.$$

*In this example, we have* $\text{ulp}(ac) = \text{ulp}(bd)$*,* $|\widehat{f} - g|$ *equals* $\frac{3}{4}\text{ulp}(g)$*, and* $|\widehat{f} - g|/(\frac{3}{2}ug)$ *is in* $1 - O(2^{-\frac{p}{2}})$ *as* $p \to \infty$.

**Special case of the sum of two squares.** Example 7 below shows that the error bounds of Property 3 remain asymptotically optimal when evaluating an expression of the form $c^2+d^2$ with $p$ even. For $p$ odd, we give examples for $p = 53, 113$ in Table I showing that our bounds are reasonably sharp.

**Example 7.** *Assuming* $p \geqslant 6$ *even, consider*

$$c = 2^{p-1} + 2^{p/2+1} \quad \textit{and} \quad d = 2^{p-1} + 2^{p/2-1} + 1.$$

*With these inputs,* $0 \leqslant d \leqslant c$ *and* $\text{ulp}(d^2) = \text{ulp}(c^2)$*. Also, both* $|\widehat{f} - g|/(\frac{3}{4}\text{ulp}(g))$ *and* $|\widehat{f} - g|/(\frac{3}{2}ug)$ *are in* $1 - O(2^{-\frac{p}{2}})$ *as* $p \to \infty$.

## III. Two complex division algorithms and their componentwise relative accuracy

When the denominator $c^2 + d^2$ in (1) is evaluated as $\widehat{\delta} = \text{RN}(c^2 + \text{RN}(d^2))$ with no comparison between $|c|$ and $|d|$, Property 2 shows that $\widehat{\delta} = (c^2 + d^2)(1 + \epsilon_1)$ with $|\epsilon_1| \leqslant 2u$. On the other hand, Property 3 suggests that one should choose to compare $|c|$ and $|d|$, and to compute $\widehat{\delta} = \text{RN}(c^2 + \text{RN}(d^2))$ if $|d| \leqslant |c|$ and $\widehat{\delta} = \text{RN}(d^2 + \text{RN}(c^2))$ otherwise: this ensures the smaller bound $|\epsilon_1| \leqslant \frac{3}{2}u$.

It is well known that conditional branches can limit the performance of programs on pipelined architectures, essentially because each wrong branch prediction causes the pipeline to stall (see Chapters 3 and 4, and Appendix A in [9]). Hence, using a conditional branch to choose how to compute $\widehat{\delta}$ depending on a comparison between $|d|$ and $|c|$ may have a bad impact on the performance of the program.

However, some architectures support predicated instructions [9, p. 340], [5, Chap. 2]. Each predicated instruction in a program refers to a condition: if the condition is true then the instruction is executed normally, otherwise it is executed as a 'no-operation' causing no particular hazard in the pipeline. The IEEE 754-2008 standard [4] also describes new operations with two floating-point numbers as operands:

- minNumMag, which returns the one of smallest magnitude (or, in the case of equal magnitudes, the minimum);
- maxNumMag, which returns the one of largest magnitude (or, in the case of equal magnitudes, the maximum).

The operations minNumMag and maxNumMag can be used to sort two floating-point numbers by order of magnitude: they are already available for example on the Itanium processor through the instructions famin and famax [5, p. 291]. Either predicated instructions, or instructions implementing the minNumMag and maxNumMag operations, could be used in our case to benefit from Property 3 while avoiding conditional branches.

In this section, we consider two complex division algorithms. We first analyze a straight-line algorithm, with a componentwise relative error bound of $5u + O(u^2)$; this bound is proved to be asymptotically optimal when $p$ is even, and its quality is illustrated with numerical examples when $p = 53, 113$. We next describe a second division algorithm in which a comparison between $|c|$ and $|d|$ is used to ensure a componentwise relative error bound of $\frac{9}{2}u+O(u^2)$, and whose quality is again illustrated with numerical examples for all the basic binary IEEE formats.

### A. Straight-line version: no tests allowed

We assume here that we can neither compare the magnitudes of $c$ and $d$, nor determine which one of the two numerators $ac + bd$ and $bc - ad$ is a sum of two products of the same sign. Hence, we shall compute the numerators using Kahan's algorithm and the denominator using (5).

**algorithm** CompDivS($a + ib$, $c + id$)
  $\widehat{\delta} := \text{RN}(c^2 + \text{RN}(d^2))$;
  $\widehat{g}_{\text{re}} := \text{Kahan}(a, b, c, d)$; // evaluates $ac + bd$
  $\widehat{g}_{\text{im}} := \text{Kahan}(b, -a, c, d)$; // evaluates $bc - ad$
  $\widehat{z}_{\text{re}} := \text{RN}\big(\widehat{g}_{\text{re}}/\widehat{\delta}\big)$; $\widehat{z}_{\text{im}} := \text{RN}\big(\widehat{g}_{\text{im}}/\widehat{\delta}\big)$;
  **return** $\widehat{z}_{\text{re}} + i\,\widehat{z}_{\text{im}}$;

For reasons of symmetry, for bounding the componentwise relative error of that algorithm it suffices to consider the relative error in the real part of the quotient. Using Property 2 together with (6) and (9) leads to

$$\widehat{z}_{\text{re}} = \frac{ac + bd}{c^2 + d^2} \cdot \frac{(1 + \epsilon_2)}{(1 + \epsilon_1)}(1 + \epsilon_3),$$

with $|\epsilon_1|, |\epsilon_2| \leqslant 2u$ and $|\epsilon_3| \leqslant u$. Hence $\widehat{z}_{\text{re}} = (1 + \epsilon) \cdot \text{Re}\,(z)$ with $|\epsilon|$ bounded by $\frac{(1+2u)}{(1-2u)}(1 + u) - 1$, which is less than $5u + 13u^2$ if and only if $p \geqslant 5$.

**Property 4.** *For* $p \geqslant 5$ *and in the absence of underflow and overflow, the quotient* $\widehat{z}$ *computed by CompDivS satisfies*

$$E_c(\widehat{z}) \leqslant 5u + 13u^2.$$

The following example shows that, when $p$ is even, the bound given by Property 4 is asymptotically optimal: here, the relative error in the real part of the computed quotient is $5u - O(u^{3/2})$. Since this example is more involved than the previous ones, we give a detailed analysis.

**Example 8** (For $p$ even, the bound in Property (4) is asymptotically optimal)**.** *Let*

$$\begin{aligned} a &= 2^p - 5\cdot 2^{\frac{p}{2}-1},\\ b &= -2^{-\frac{p}{2}}\cdot\left(2^p - 5\cdot 2^{\frac{p}{2}-1} + 3\right),\\ c &= 2^p - 2,\\ d &= 2^{\frac{p}{2}+1}\cdot\left(2^{p-1} + 2^{\frac{p}{2}-1}\right), \end{aligned}$$

*and assume $p$ is even. Then, the quotient $\widehat{z}$ computed by CompDivS satisfies*

$$\frac{|\mathit{Re}\,\widehat{z} - \mathit{Re}\,z|}{|\mathit{Re}\,z|} = 5u - O(u^{3/2}). \tag{17}$$

**Proof of (17).** Let us define $R = 2^{p/2}$. We have

$$\begin{aligned} \mathrm{Re}\,z &= -\frac{2R^3 + 5R^2 - 4R}{2R^6 + 4R^5 + 4R^4 - 8R^2 + 8}\\ &= -\frac{1}{R^3} - \frac{1}{2R^4} + \frac{5}{R^5} + O\left(\frac{1}{R^6}\right). \end{aligned}$$

Elementary manipulations show that the value of $\widehat{\delta}$ in Algorithm CompDivS is $R^6 + 2R^5$ and that the value of $\widehat{g}_{\mathrm{re}}$ is $-R^3 - \frac{5}{2}R^2$. Hence

$$\frac{\widehat{g}_{\mathrm{re}}}{\widehat{\delta}} = \frac{-R^3 - \frac{5}{2}R^2}{R^6 + 2R^5} = -\frac{1}{R^3} - \frac{1}{2R^4} + \frac{1}{R^5} + O\left(\frac{1}{R^6}\right),$$

and we will show that $\widehat{z}_{\mathrm{re}} = \tau$, where $\tau$ is the floating-point number defined by

$$\tau := -\frac{1}{R^3} - \frac{1}{2R^4} = -2^{-3p/2} - 2^{2p-1}.$$

Since $\widehat{z}_{\mathrm{re}} = \mathrm{RN}\big(\widehat{g}_{\mathrm{re}}/\widehat{\delta}\big)$, it suffices to show that

$$\tau \leqslant \frac{\widehat{g}_{\mathrm{re}}}{\widehat{\delta}} < \tau + \frac{1}{2}\mathrm{ulp}(\tau),$$

and since $\mathrm{ulp}(\tau) = 2^{-5p/2+1} = 2/R^5$, this is equivalent to

$$\underbrace{-\frac{1}{R^3} - \frac{1}{2R^4}}_{\mathrm{Left}(R)} \leqslant \underbrace{\frac{-R^3 - \frac{5}{2}R^2}{R^6 + 2R^5}}_{\mathrm{Med}(R)} < \underbrace{-\frac{1}{R^3} - \frac{1}{2R^4} + \frac{1}{R^5}}_{\mathrm{Right}(R)}. \tag{18}$$

An elementary analysis of the functions $\mathrm{Med}(R) - \mathrm{Left}(R)$ and $\mathrm{Right}(R) - \mathrm{Med}(R)$ allows us to check that (18) is satisfied when $R > 0$, which is true by definition of $R$. Hence we deduce that CompDivS returns $\widehat{z}_{\mathrm{re}} = \tau$. It follows that

$$\begin{aligned} \left|\frac{\widehat{z}_{\mathrm{re}}}{\mathrm{Re}\,z} - 1\right| &= \frac{10R^5 + 2R^4 - 8R^3 - 4R^2 + 8R + 4}{2R^7 + 5R^6 - 4R^5}\\ &= \frac{5}{R^2} - \frac{23}{2R^3} + \frac{139}{4R^4} + O\left(\frac{1}{R^5}\right), \end{aligned}$$

which leads to (17). ∎

For example, when $p = 24$, which corresponds to the binary32 IEEE format, the value in (17) is about $4.997$. When $p$ is odd, we have not yet constructed an example parametrized by $p$. Nevertheless, the examples given in Table II already illustrate well the quality of the bound in Property 4 for other formats of practical interest like binary64 ($p = 53$) and binary128 ($p = 113$).

| $p$ | example |
|---|---|
| 53 | $a = 2^{52} + 1$<br>$b = -142398041 = -\left\lceil \frac{2^{105}+2^{104}}{d} \right\rceil - 38644$<br>$c = 2^{52}$<br>$d = 94906267 \cdot 2^{52} = \left(1 + \lceil 2^{53/2} \rceil\right) \cdot 2^{52}$<br>$\lvert\widehat{z}_{\mathrm{re}} - \mathrm{Re}\,z\rvert/(u\lvert\mathrm{Re}\,z\rvert) = 4.9987\ldots$ |
| 113 | $a = 2^{112} + 1$<br>$b = -152857240142482713 = -\left\lceil \frac{2^{225}+2^{224}}{d} \right\rceil - 1864174$<br>$c = 2^{112}$<br>$d = 101904826760412363 \cdot 2^{112} = \left(1 + \lceil 2^{113/2} \rceil\right) \cdot 2^{112}$<br>$\lvert\widehat{z}_{\mathrm{re}} - \mathrm{Re}\,z\rvert/(u\lvert\mathrm{Re}\,z\rvert) = 4.9999\ldots$ |

TABLE II
RELATIVE ERROR IN $\widehat{z}_{\mathrm{RE}}$ COMPUTED USING COMPDIVS CLOSE TO THE UPPER BOUND $5u + 13u^2$.

### *B. If tests are allowed*

If tests are allowed, then one can compare $|c|$ and $|d|$ to select between $\mathrm{RN}(c^2 + \mathrm{RN}(d^2))$ and $\mathrm{RN}(d^2 + \mathrm{RN}(c^2))$ the computation that ensures the smallest relative error bound according to Property 3, and still use Kahan's algorithm for the evaluation of both numerators $ac + bd$ and $bc - ad$. This gives the following complex division algorithm:

**algorithm** CompDivT($a + ib$, $c + id$)
  **if** $|d| \leqslant |c|$ **then** $\widehat{\delta} := \mathrm{RN}(c^2 + \mathrm{RN}(d^2))$;
  **else** $\widehat{\delta} := \mathrm{RN}(d^2 + \mathrm{RN}(c^2))$;
  $\widehat{g}_{\mathrm{re}} := \mathrm{Kahan}(a, b, c, d)$; // evaluates $ac + bd$
  $\widehat{g}_{\mathrm{im}} := \mathrm{Kahan}(b, -a, c, d)$; // evaluates $bc - ad$
  $\widehat{z}_{\mathrm{re}} := \mathrm{RN}\big(\widehat{g}_{\mathrm{re}}/\widehat{\delta}\big)$; $\widehat{z}_{\mathrm{im}} := \mathrm{RN}\big(\widehat{g}_{\mathrm{im}}/\widehat{\delta}\big)$;
  **return** $\widehat{z}_{\mathrm{re}} + i\,\widehat{z}_{\mathrm{im}}$;

When CompDivT is implemented on an architecture supporting the operations minNumMag and maxNumMag defined in the IEEE 754-2008 standard [4], its first two lines may be replaced by the following straight-line code:

$\underline{c} := \mathrm{maxNumMag}(c, d)$;
$\underline{d} := \mathrm{minNumMag}(c, d)$;
// Here, $\underline{c}^2 + \underline{d}^2 = c^2 + d^2$ and $|\underline{d}| \leqslant |\underline{c}|$.
$\widehat{\delta} := \mathrm{RN}(\underline{c}^2 + \mathrm{RN}(\underline{d}^2))$;

Similar techniques avoiding branches can also be used more generally on architectures supporting predicated instructions [9, Chap. 4, p. 340], [5, Chap. 2].

According to Property 3 we now have $\widehat{\delta} = (c^2 + d^2)(1 + \epsilon_1)$ with $|\epsilon_1| \leqslant \frac{3}{2}u$, which leads to the following error bound.

**Property 5.** *For $p \geqslant 6$ and in the absence of underflow and overflow, the quotient $\widehat{z}$ computed by CompDivT satisfies*

$$E_c(\widehat{z}) \leqslant \tfrac{9}{2}u + 9u^2.$$

The examples given in Table III show that the bound $\frac{9}{2}u + 9u^2$ if reasonably sharp for the basic binary IEEE formats ($p = 24, 53, 113$).

| $p$ | example |
|---|---|
| 24 | $a = 8391768$ |
| | $b = -8392368$ |
| | $c = 8391504$ |
| | $d = 8390648$ |
| | $\lvert\widehat{z}_{\text{re}} - \text{Re}\, z\rvert/(u\lvert\text{Re}\, z\rvert) = 4.4932\ldots$ |
| 53 | $a = 4503599627378010$ |
| | $b = -4503599627377047$ |
| | $c = 6369051672541039$ |
| | $d = 6369051672534109$ |
| | $\lvert\widehat{z}_{\text{re}} - \text{Re}\, z\rvert/(u\lvert\text{Re}\, z\rvert) = 4.4421\ldots$ |
| 113 | $a = 7343016637207168931428032607349357$ |
| | $b = -7343016637207168931428032607349412$ |
| | $c = 7343016637207168931428032607356045$ |
| | $d = 7343016637207168931428032607355264$ |
| | $\lvert\widehat{z}_{\text{re}} - \text{Re}\, z\rvert/(u\lvert\text{Re}\, z\rvert) = 4.4620\ldots$ |

TABLE III
RELATIVE ERROR IN $\widehat{z}_{\text{re}}$ COMPUTED USING COMPDIVT (GIVEN WITH $|d| \leqslant |c|$) CLOSE TO THE UPPER BOUND $\frac{9}{2}u + 9u^2$.

Concerning the computation of the numerators $ac + bd$ and $bc - ad$, Property 3 could also be used: instead of using Kahan's algorithm for computing both numerators, one could determine which one is the sum of two products with the same sign, and then use Property 3 as in CompDivT to evaluate the corresponding numerator with a relative error bounded by $\frac{3}{2}u$. However, this would only improve the error bound for one of the components of the computed quotient, and the componentwise relative error bound would remain $\frac{9}{2}u + 9u^2$ as with CompDivT: for this reason, we have not considered this algorithm here.

## IV. CONCLUDING REMARKS AND FUTURE WORK

By combining Kahan's algorithm with a cheaper scheme, we have introduced two complex division algorithms that take advantage of the availability of an FMA instruction. We have also given sharp componentwise relative error bounds for these algorithms. This work should be pursued further in the following three directions. First, we should implement our algorithms in order to compare their running times and measure the overhead induced by the use of Kahan's algorithm to improve the accuracy. Second, extensions to non binary radices and other rounding modes should be proposed. Third, it would be interesting to study how our division algorithms can be combined with the scaling techniques from [10], [11], [12], [13] in order to avoid spurious under/overflows while preserving high componentwise accuracy.

## ACKNOWLEDGMENTS

This research was partly supported by the French *Agence Nationale de la Recherche (ANR)*, under grants ANR 11 BS02 013 (HPAC project) and ANR 2010 BLAN 0203 01 (TaMaDi project).

## REFERENCES

[1] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, 2nd ed. Philadelphia, PA, USA: SIAM, 2002.
[2] J.-M. Muller, N. Brisebarre, F. de Dinechin, C.-P. Jeannerod, V. Lefèvre, G. Melquiond, N. Revol, D. Stehlé, and S. Torres, *Handbook of Floating-Point Arithmetic*. Birkhäuser Boston, 2010.
[3] M. Baudin, "Error bounds of complex arithmetic," June 2011, available at http://forge.scilab.org/upload/compdiv/files/complexerrorbounds_v0.2.pdf.
[4] IEEE Computer Society, *IEEE Standard for Floating-Point Arithmetic*. IEEE Standard 754-2008, Aug. 2008, available at http://ieeexplore.ieee.org/servlet/opac?punumber=4610933.
[5] M. Cornea, J. Harrison, and P. T. P. Tang, *Scientific Computing on Itanium®-based Systems*. Intel Press, Hillsboro, OR, 2002.
[6] C.-P. Jeannerod, N. Louvet, and J.-M. Muller, "Further analysis of Kahan's algorithm for the accurate computation of $2 \times 2$ determinants," *Mathematics of Computation*, 2012, to appear. Preliminary version available at http://hal-ens-lyon.archives-ouvertes.fr/ensl-00649347/en/.
[7] R. P. Brent, C. Percival, and P. Zimmermann, "Error bounds on complex floating-point multiplication," *Mathematics of Computation*, vol. 76, pp. 1469–1481, 2007.
[8] S. M. Rump, T. Ogita, and S. Oishi, "Accurate floating-point summation part I: Faithful rounding," *SIAM Journal on Scientific Computing*, vol. 31, no. 1, pp. 189–224, 2008.
[9] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*, 3rd ed. Morgan Kaufmann, 2003.
[10] R. L. Smith, "Algorithm 116: Complex division," *Comm. ACM*, vol. 5, no. 8, p. 435, 1962.
[11] G. W. Stewart, "A note on complex division," *ACM Trans. Math. Software*, vol. 11, no. 3, pp. 238–241, 1985.
[12] D. M. Priest, "Efficient scaling for complex division," *ACM Trans. Math. Software*, vol. 30, no. 4, Dec. 2004.
[13] M. Baudin and R. L. Smith, "A robust complex division in Scilab," October 2012, available at http://arxiv.org/abs/1210.4539.

## APPENDIX

**Property 6** (§3.5 and §3.6 in [3]). *Given $x = a + ib$ and $y = c + id$ two complex numbers with floating-point coefficients, let $\widehat{z}$ be the evaluation of $z = x/y$ computed according to (2). In the absence of underflow and overflow,*

$$E_n(\widehat{z}) \leqslant \frac{(3+\sqrt{5})u + \sqrt{5}u^2}{1-2u} < (3+\sqrt{5})u + 13u^2,$$

*where the latter inequality holds assuming $p \geqslant 7$ .*

*Proof:* Writing $\nu = (ac+bd)+i(bc-ad)$ and $\delta = c^2+d^2$, we have $z = \nu/\delta$. Let also $\varphi = \widehat{\nu}/\widehat{\delta}$ with

$$\widehat{\nu} = \text{RN}\big(\text{RN}(ac) + \text{RN}(bd)\big) + i\,\text{RN}\big(\text{RN}(bc) - \text{RN}(ad)\big)$$

and

$$\widehat{\delta} = \text{RN}\big(\text{RN}(c^2) + \text{RN}(d^2)\big).$$

Then $\widehat{z}$ satisfies $\widehat{z} = \text{RN}(\text{Re}\,\varphi) + i\,\text{RN}(\text{Im}\,\varphi)$ and we have

$$\begin{aligned}|\widehat{z} - \varphi|^2 &= \big(\text{RN}(\text{Re}\,\varphi) - \text{Re}\,\varphi\big)^2 + \big(\text{RN}(\text{Im}\,\varphi) - \text{Im}\,\varphi\big)^2 \\ &\leqslant u^2(\text{Re}\,\varphi)^2 + u^2(\text{Im}\,\varphi)^2.\end{aligned}$$

Hence $|\widehat{z} - \varphi| \leqslant u|\varphi|$, which implies $\widehat{z} = \varphi(1+\epsilon_1)$ for some $\epsilon_1 \in \mathbb{C}$ such that $|\epsilon_1| \leqslant u$. Now, by [7] we know that $\widehat{\nu} = \nu(1+\epsilon_2)$ for some $\epsilon_2 \in \mathbb{C}$ such that $|\epsilon_2| \leqslant \sqrt{5}u$, and also that $\widehat{\delta} = \delta(1+\epsilon_3)$ for some $\epsilon_3 \in \mathbb{R}$ such that $|\epsilon_3| \leqslant 2u$. Therefore, $\widehat{z} = z(1+\epsilon)$ with $\epsilon$ given by

$$\epsilon = \frac{(1+\epsilon_1)(1+\epsilon_2)}{1+\epsilon_3} - 1.$$

The bounds on $|\epsilon|$ then follow immediately from the bounds on the $|\epsilon_i|$'s and from the fact that $u = 2^{-p}$. ∎