Faithfully Rounded Floating-point Computations

MARKO LANGE, Waseda University
SIEGFRIED M. RUMP, Hamburg University of Technology

We present a pair arithmetic for the four basic operations and square root. It can be regarded as a simplified, more efficient double-double arithmetic. We prove rigorous error bounds for the computed result depending on the relative rounding error unit $\mathbf{u}$ according to base β, the size of the arithmetic expression, and possibly a condition measure. Under precisely specified assumptions, the result is proved to be faithfully rounded for up to $1/\sqrt{\beta\mathbf{u}} - 2$ operations. The assumptions are weak enough to apply to many algorithms. For example, our findings cover a number of previously published algorithms to compute faithfully rounded results, among them Horner's scheme, products, sums and dot products, or Euclidean norm. Beyond that, several other problems can be analyzed such as polynomial interpolation, orientation problems, Householder transformations, or the smallest singular value of Hilbert matrices of large size.

Additional Key Words and Phrases: double-double, inaccurate cancellation, rigorous error bounds

1 INTRODUCTION

Usually, an arithmetic expression evaluated using floating-point arithmetic produces a reasonably good approximation of the true result; however, the accuracy may be severely reduced due to cancellation. One way to diminish that effect is compensated evaluation [12, 16, 18]. Another approach is to increase the precision, either by the hard-/software implementation of a high precision format [13–15] or the simulation via multiple standard precision numbers. One example for the latter is the double-double format [1], in which results are represented as unevaluated sums of two floating-point numbers. On these pairs, arithmetic operations are defined.

In some cases the accuracy of the computed result can be estimated a priori. Examples are products of floating-point numbers [6], Horner's scheme [5], summation and dot product [17], or the Euclidean norm of a vector [7]. For a detailed analysis with error bounds cf. [11].

In this note we present a general evaluation scheme for arithmetic expressions consisting of addition, subtraction, multiplication, division, and square root. For applications like the ones listed above, where the accuracy can be estimated a priori, this evaluation scheme enables us to compute a faithfully rounded result, provided that precisely specified conditions are met. The arithmetic expression may be evaluated in any order.

To achieve that, we extend floating-point numbers by an error term. For such pairs with a significant and an error part, we define the classical arithmetic operations and square root. The significant part of the result is always equal to the result obtained by ordinary floating-point arithmetic, whereas the additional error term increases the precision. A main difference to the double-double approach is the omission of the final normalization steps. Therefore, the digits of the significant and the error part may overlap.

This research was partially supported by CREST, Japan Science and Technology Agency (JST).
Authors' addresses: M. Lange, Waseda University, Faculty of Science and Engineering, 3–4–1 Okubo, Shinjuku-ku, Tokyo 169–8555, Japan, m.lange@aoni.waseda.jp
S. M. Rump, Institute for Reliable Computing, Hamburg University of Technology, Am Schwarzenberg-Campus 3, Hamburg 21071, Germany, and Waseda University, Faculty of Science and Engineering, 3–4–1 Okubo, Shinjuku-ku, Tokyo 169–8555, Japan, rump@tuhh.de.
2017. XXXX-XXXX/2017/9-ART1

Our main results are as follows. Let an arithmetic expression comprising of additions, subtractions, multiplications, divisions, and square roots be given. Suppose that in the subtrees of square roots and of denominators of divisions no cancellation on computed data occurs, and that the number of operations in those subtrees is roughly bounded by $\mathbf{u}^{-1/2}$. Then our pair arithmetic computes a result with an error bounded by $\gamma k^2 \hbar \mathbf{u}^2$, where γ is an explicit known small constant close to 1, k denotes the total number of operations, and $\hbar$ is related to the condition number. Moreover, if no cancellation occurs on computed data, then $\hbar = 1$. In that case, for up to about $(\beta\mathbf{u})^{-1/2}$ operations, which is about $6.7 \cdot 10^7$ for binary64, the computed result is a faithful rounding of the correct result.

Our aim is to keep the additional effort for computing the error terms as small as possible, eventually accepting a mandatory restriction on the number of operations of about $\mathbf{u}^{-1/2}$. That might be a practical issue when using IEEE 754 binary32, and we could lift that bound to about $\mathbf{u}^{-1}$ by investing more computational effort. However, rather than using our pair arithmetic on binary32 as a base type, the computation could be performed directly and faster in binary64. Therefore, we opted on fast pair algorithms eventually accepting the limit of about 10^8 operations for base type binary64.

Having said that, in this note we assume only rather general properties to be satisfied by the underlying floating-point arithmetic. Our estimates hold true for any base β, with any rounding of ties, and without explicit restrictions of the mantissa or exponent length of the underlying format.

In the sequel, we start with some notations, followed by definitions, the main results and proofs. In the last section, we give some rationale on how to add other functions to our pair arithmetic and discuss a selection of applications. We assume familiarity by the reader with floating-point error estimates; a very nice source is [16].

2 NOTATION AND ASSUMPTIONS

Let $\mathbb{F}$ be a set of floating-point numbers with base β adhering to the IEEE 754 standard [9, 10], and denote by $\mathrm{fl}\colon \mathbb{R} \to \mathbb{F}$ a rounding to the nearest floating-point number with any rounding of ties.

For an arithmetic expression consisting of several operations, we use the short notation float(...) to indicate that for each operation the corresponding floating-point operation is to be used. For example, $g = \mathrm{float}(t + (af + be))$ means $g = \mathrm{fl}(t + \mathrm{fl}(\mathrm{fl}(a \cdot f) + \mathrm{fl}(b \cdot e)))$. The order of execution will always be made unique using parentheses.

The error constant $\mathbf{u}$ to an m-digit floating-point system with base β is $\mathbf{u} := \frac{1}{2}\beta^{1-m}$. Furthermore, we abbreviate

$$\mathbf{v} := \frac{\mathbf{u}}{1+\mathbf{u}}, \quad \omega := 1 + \mathbf{v}, \quad \text{and} \quad \bar{\omega} := 1 + \mathbf{u}. \tag{1}$$

The constants defined in (1) satisfy the relations

$$\bar{\omega}\mathbf{v} = \mathbf{u}, \quad \bar{\omega} = \frac{1}{1-\mathbf{v}}, \quad \text{and} \quad \omega\bar{\omega} = 1 + 2\mathbf{u},$$

which we often use throughout this note. In the following sections we suppose the floating-point operations to satisfy the first error model (cf. [16]). To be precise:

Assumption 1. *Let c be the result of a floating-point operation and let $\hat{c}$ denote the true result of the corresponding real operation. Then*

$$c = \hat{c}(1+\varepsilon) \qquad \textit{with} \quad |\varepsilon| \le \mathbf{v}. \tag{2}$$

Particularly, this assumption is satisfied if neither underflow errors nor overflow occur[1]. Another frequently used equation is an immediate consequence of (2):

$$\bar{\omega}^{-1} s\hat{c} = (1-\mathbf{v})s\hat{c} \le sc \le (1+\mathbf{v})s\hat{c} = \omega s\hat{c}, \tag{3}$$

[1]Note that floating-point addition and subtraction with result in the underflow range is error-free.

which is satisfied for any real number s with $s\hat{c} \geq 0$.

For reasons of clarity, we skip the difference operation from the following discussion and only consider the set of operations $\{+, \cdot, /, \sqrt{}\}$. This is because the algorithm for subtraction is directly derived from our algorithm for addition by negating the second argument, and because the negation of a floating-point number is error-free.

We aim to define floating-point operations with an error term so that for certain arithmetic expressions a faithfully rounded result is computed. The definition of the latter is as follows.

Definition 2.1. Let $\tilde{r} \in \mathbb{F}$ be given and define

$$\operatorname{pred}(\tilde{r}) := \max\{f \in \mathbb{F}\colon f < \tilde{r}\} \quad \text{and} \quad \operatorname{succ}(\tilde{r}) := \min\{f \in \mathbb{F}\colon \tilde{r} < f\}. \tag{4}$$

A floating-point number $\tilde{r}$ is defined to be a faithful rounding of $r \in \mathbb{R}$ if

$$\operatorname{pred}(\tilde{r}) < r < \operatorname{succ}(\tilde{r}). \tag{5}$$

Note that $\tilde{r} = r$ is the only possible faithful rounding if $r \in \mathbb{F}$, otherwise there are two candidates.

To prove a faithfully rounded result only depending on the number of operations but independent of the structure of the expression, we exploit the notation of the *No Inaccurate Cancellation* (NIC) principle. In [4] that was used to identify algorithms computing accurate results basically independent of the condition number. A famous example is to treat Hilbert matrices as Cauchy matrices allowing to faithfully compute the inverse up to about dimension 10^8 solely in binary64.

Definition 2.2. Let T be an evaluation tree with input data p and inner nodes comprising of operations from the set $\{+, \cdot, /, \sqrt{}\}$. Suppose that each individual operation satisfies Assumption 1. If sums, where at least one addend is not input data, are not performed on numbers with opposite signs, then (T, p) complies with the *No Inaccurate Cancellation* (NIC) principle.

Any deterministic instance of an algorithm is representable as an evaluation tree. Therefore the following results are as well applicable to the broader concept of algorithms. For reasons of simplicity, we use evaluation trees.

Definition 2.3. Consider an evaluation tree T with input data $p \in \mathbb{F}^n$. Let any pair of input numbers p_i and p_j that is added in T with negative result be replaced by $p_i' := -p_i$ and $p_j' := -p_j$, respectively. Moreover, let all other input numbers p_k be replaced by their absolute value $p_k' := |p_k|$. The so obtained data p' is called NIC remodeled input data to (T, p).

Evidently, for any tree T an instance (T, p') with NIC remodeled input data always complies with the NIC principle.

Assumption 2. *For an evaluation tree T with input data p we henceforth assume that all intermediate operations are well defined on the field of real numbers $\mathbb{R}$.*

3 PAIR ARITHMETIC AND PRELIMINARY RESULTS

Our basic pair operations to be analyzed are as follows:

Function $(c,g) \leftarrow$ CPairSum$((a,e),(b,f))$

$c \leftarrow \mathrm{fl}(a+b)$
$t \leftarrow a+b-c$ // TwoSum
$g \leftarrow \mathrm{float}(t+(e+f))$

Function $(c,g) \leftarrow$ CPairProd$((a,e),(b,f))$

$c \leftarrow \mathrm{fl}(ab)$
$t \leftarrow ab-c$ // FMA or TwoProduct
$g \leftarrow \mathrm{float}(t+(af+be))$

Function $(c,g) \leftarrow$ CPairDiv$((a,e),(b,f))$

$c \leftarrow \mathrm{fl}(a/b)$
$t \leftarrow a-bc$ // FMA
$g \leftarrow \mathrm{float}(((t+e)-cf)/(b+f))$

Function $(c,g) \leftarrow$ CPairSqrt$((a,e))$

$c \leftarrow \mathrm{fl}(\sqrt{a})$
$t \leftarrow a-c^2$ // FMA
$g \leftarrow \mathrm{float}((t+e)/(c+c))$

It is known [2, 16] that, independent of the choice of base β, the residuals $a+b-c$ for the sum, $ab-c$ for the product, $a-bc$ for the division, and $a-c^2$ for the square root are floating-point numbers, so that no float$(\cdot)$ is necessary in the lines computing t. The residuals can be computed correctly using the error-free transformation TwoSum [12][2] for the sum and fused multiply-and-add (FMA) for the other operations.

Once more we want to stress that the significant part of our floating-point pairs is identical to the result of the respective evaluation in base precision. The second term is a floating-point approximation of the actual error. This leads to the preservation of some beneficial properties.

Lemma 3.1. *To a given evaluation tree T consider an instance (T,p) that satisfies the NIC principle for the given input data p. Let c be the result evaluated using floating-point arithmetic, and $\hat{c}$ be the true result of the expression. Then there exists a positive real factor γ satisfying $c=\gamma\hat{c}$, i.e., the sign is preserved.*

Proof. We proceed by induction, where for all input data and operations solely on input data the assertion is obviously true. Henceforth, assume that the statement is true for the child element(s) of the root r of an evaluation tree with height $h \geq 3$. Following, we consider addition and multiplication; the respective arguments for division and square root can be easily adapted.

Let a and b be the results evaluated in the left and right subtree of r, with $\hat{a}$ and $\hat{b}$ denoting the results computed in real arithmetic, respectively. By the induction hypothesis there exist $\gamma_a, \gamma_b > 0$ satisfying $a=\gamma_a\hat{a}$ and $b=\gamma_b\hat{b}$. Choose $s \in \{-1,1\}$ such that $s\hat{c} \geq 0$.

For addition, due to the NIC principle, $s\hat{c} \geq 0$ implies $s\hat{a}, s\hat{b}, sa, sb \geq 0$. Using (3), we have

$$\bar{\omega}^{-1}\min\{\gamma_a,\gamma_b\}s\hat{c} \leq \bar{\omega}^{-1}(\gamma_a s\hat{a}+\gamma_b s\hat{b}) = \bar{\omega}^{-1}s(a+b) \leq sc$$

[2]It is known [3] that intermediate overflow may occur although the input and the results are within the floating-point range. That can be avoided using a branch and FastTwoSum.

and

$$sc \le \omega s(a+b) \le \omega(\gamma_a s\hat{a} + \gamma_b s\hat{b}) \le \omega \max\{\gamma_a, \gamma_b\} s\hat{c}.$$

Hence, there exists γ in the positive interval $[\bar{\omega}^{-1} \min\{\gamma_a, \gamma_b\},\ \omega \max\{\gamma_a, \gamma_b\}]$ satisfying $c = \gamma\hat{c}$.

For multiplication, again choosing s so that $s\hat{c} \ge 0$, we have

$$\bar{\omega}^{-1}\gamma_a\gamma_b s\hat{c} = \bar{\omega}^{-1} sab \le sc \le \omega sab = \omega\gamma_a\gamma_b s\hat{c},$$

by which the existence of some feasible $\gamma \in [\bar{\omega}^{-1}\gamma_a\gamma_b,\ \omega\gamma_a\gamma_b]$ is evident. □

Lemma 3.2. *For an evaluation tree T and input data p, let p' be the NIC remodeled input data. Denote by $\hat{c}$ and $\hat{C}$ the true result of evaluating T for input data p and p', respectively. If (T, p) complies with the NIC principle, then $|\hat{c}| = \hat{C}$.*

Proof. Using Definitions 2.2 and 2.3 together with Assumption 2, the statement follows by a simple induction argument. □

For given $r, \delta \in \mathbb{R}$, $\tilde{r} = \mathrm{fl}(r)$ and supposing binary arithmetic, it was shown in [18] that $|\delta| < \frac{\mathbf{u}}{2}|\tilde{r}|$ implies that $\mathrm{fl}(r)$ is a faithful rounding of $r + \delta$. In the following we need another criterion based on r rather than on $\tilde{r}$. Moreover, the criterion is extended to general base β.

Lemma 3.3. *Let $r, \delta \in \mathbb{R}$ be given and assume*

$$|\delta| \le \frac{1}{\beta}\mathbf{u}|r|. \tag{6}$$

Then $\mathrm{fl}(r)$ is a faithful rounding of $r + \delta$.

Proof. Without loss of generality, due to the symmetry of $\mathbb{F}$ and the trivial case $r = 0$, we may assume that r is positive. Let $\tilde{r} := \mathrm{fl}(r)$ and denote by q a second nearest floating-point number to r, in the sense that

$$q \ne \tilde{r} \qquad \text{and} \qquad \forall f \in \mathbb{F} \setminus \{\tilde{r}\}\colon\ |r - q| \le |r - f|.$$

Moreover, let $\varrho := |r - \frac{\tilde{r}+q}{2}|$ denote the difference between r and the midpoint of $\tilde{r}$ and q. Then

$$|r - q| \ge |r - \tilde{r}| \quad \Longrightarrow \quad |r - q| = \left|\frac{\tilde{r} + q}{2} - q\right| + \left|r - \frac{\tilde{r} + q}{2}\right| = \frac{|\tilde{r} - q|}{2} + \varrho.$$

Since the fraction $\frac{|\tilde{r}-q|}{|\tilde{r}+q|}$ is minimal if $\tilde{r}$ is a power of β and q its predecessor, we have

$$\frac{|r - q|}{r} \ge \frac{\frac{|\tilde{r}-q|}{2} + \varrho}{\frac{|\tilde{r}+q|}{2} + \varrho} \ge \frac{|\tilde{r} - q|}{|\tilde{r} + q|} \ge \frac{2\mathbf{u}}{\beta + \mathrm{pred}(\beta)} > \frac{\mathbf{u}}{\beta}.$$

Finally, by $|r - q| = \min\{|r - \mathrm{pred}(\tilde{r})|, |r - \mathrm{succ}(\tilde{r})|\}$, we validate

$$\mathrm{pred}(\tilde{r}) \le r - |r - q| < r - \frac{\mathbf{u}}{\beta}r \le r + \delta \le r + \frac{\mathbf{u}}{\beta}r < r + |r - q| \le \mathrm{succ}(\tilde{r}),$$

so that (2.1) completes the proof. □

4 MAIN RESULTS

In this section we specify criteria ensuring a faithful rounding of the computed floating-point approximation. For this purpose we first introduce an error estimate that bounds the approximation error of our pair arithmetic depending on $\mathbf{u}$, the size of the expression, and a quantity related to the condition.

Definition 4.1. For $0 \le k \in \mathbb{N}$, we define

$$\varphi_k := k(\omega\bar{\omega})^k\mathbf{u}, \quad \text{and} \quad \psi_k := (k^2+2k)(\omega\bar{\omega})^k\mathbf{u}^2. \tag{7}$$

Let $\hat{c}, \hat{C} \in \mathbb{R}$ and $s \in \{-1, 1\}$ satisfy $\hat{C} \ge |\hat{c}| = s\hat{c}$. We call a pair $(c, g) \in \mathbb{F} \times \mathbb{F}$ a k-approximation of $\hat{c}$ with respect to $\hat{C}$ if

$$|g| \le \varphi_k\hat{C}, \quad |\hat{c} - (c+g)| \le \psi_k\hat{C}, \tag{8}$$

and $\hat{C} > |\hat{c}|$ implies

$$|c - \hat{c}| \le (\bar{\omega}^k - 1)\hat{C}, \tag{9a}$$

whereas $\hat{C} = |\hat{c}|$ requires

$$\bar{\omega}^{-k}\hat{C} \le sc \le \bar{\omega}^k\hat{C}. \tag{9b}$$

If $\hat{C} = |\hat{c}|$, we also say that (c, g) is a proper k-approximation of $\hat{c}$.

Please note that (9b) is stronger than (9a) because

$$\bar{\omega}^{-k}\hat{C} \le sc \le \bar{\omega}^k\hat{C} \implies (\bar{\omega}^{-k} - 1)\hat{C} \le s(c - \hat{c}) \le (\bar{\omega}^k - 1)\hat{C} \implies |c - \hat{c}| \le (\bar{\omega}^k - 1)\hat{C}.$$

Conversely, (9a) implies the right inequality of (9b) by

$$sc \le |c - \hat{c}| + s\hat{c} \le (\bar{\omega}^k - 1)\hat{C} + \hat{C} = \bar{\omega}^k\hat{C}. \tag{10}$$

Hence, a k-approximation of $\hat{c}$ with respect to $\hat{C}$, whether $\hat{C} > |\hat{c}|$ or $\hat{C} = |\hat{c}|$, always satisfies (9a) and (10).

On the other hand, it is worth mentioning that the left inequality in (9b) is closely related to the NIC principle. In particular, this condition comprises the preservation of the sign addressed in Lemma 3.1. Indeed (9b) is a property of any floating-point result of an NIC conform expression with up to k operations. This statement will be clear from the proof of Theorem 4.2.

Using the above introduced quantities for the qualification of the approximation error, we may now state our main results. After providing some auxiliary facts in Subsection 4.1, the proofs are given in Subsection 4.2.

THEOREM 4.2. *Let an arithmetic expression be given by an evaluation tree T with n leaves, where to each inner node j an operation $\circ_j$ out of $\{+, \cdot, /, \sqrt{}\}$ is assigned. Moreover, to every node j, inner node or leaf, let an integer k_j be assigned according to*

$$k_j := \begin{cases} 0 & \text{if } j \text{ is leaf} \\ \max\{k_{\mathrm{left}(j)}, k_{\mathrm{right}(j)}\} + 1 & \text{if } \circ_j = + \\ k_{\mathrm{left}(j)} + k_{\mathrm{right}(j)} + 1 & \text{if } \circ_j = \cdot \\ k_{\mathrm{left}(j)} + k_{\mathrm{right}(j)} + 2 & \text{if } \circ_j = / \\ \max\{k_{\mathrm{child}(j)}, \min\{k_{\mathrm{child}(j)} + 1, 7\}\} & \text{if } \circ_j = \sqrt{}, \end{cases} \tag{11}$$

For given input data $p \in \mathbb{F}^n$, let $(p_i, 0)$ be the pairs at the leaves of T, and denote by (c, g) the result evaluated at root r using our pair arithmetic. Furthermore, let $\hat{c}$ be the true result of the expression for input data p, and let $\hat{C}$ be the true result for the NIC remodeled input data p'.

Suppose that all denominators and all expressions below a square root comply with the NIC principle. Furthermore, suppose that k_j is not larger than $\mathbf{u}^{-\frac{1}{2}}$ for any node j comprising of division or square root. Otherwise k_j is unbounded. Then (c, g) is a k_r-approximation of $\hat{c}$ with respect to $\hat{C}$.

Remark 1. *Note that if the whole expression (T, p) complies with the NIC principle, then $|\hat{c}| = \hat{C}$ by Lemma 3.2, so that the error is bounded by*

$$|\hat{c} - (c + g)| \leq \psi_{k_r} |\hat{c}|,$$

independent of a condition number.

Theorem 4.3. *Let (c, g) be a k-approximation of $\hat{c}$ with respect to $\hat{C}$ and define*

$$\boldsymbol{\hbar} := \frac{\hat{C}}{|\hat{c}|} \qquad \text{with the convention} \quad \frac{0}{0} := 1. \tag{12}$$

If k is restricted via

$$k \leq \frac{1}{\sqrt{\beta \boldsymbol{\hbar} \mathbf{u}}} - 2, \tag{13}$$

then $\mathrm{fl}(c + g)$ is a faithful rounding of $\hat{c}$. For any expression complying with the NIC principle, condition (13) reduces to $k \leq (\beta \mathbf{u})^{-\frac{1}{2}} - 2$. In particular, for binary64 that means $k \leq 67,108,862$.

Note that Theorem 4.3 makes no statement for $\hat{C} > |\hat{c}| = 0$ since a faithfully rounded result cannot be guaranteed for this case. On the other hand, if the NIC principle is satisfied, then $|\hat{c}| = 0 \Leftrightarrow \hat{C} = 0$ and always $\boldsymbol{\hbar} = 1$.

4.1 Auxiliaries

In the following subsection we prove the respective error estimates individually for each operation defined in Section 3. For this purpose, we first go through some auxiliary facts.

A central role plays the following inequality

$$\forall \lambda \geq \mu \geq 0, q \geq 0, r \geq 1 \colon \quad (1 + \mu)^q (1 + \lambda)^{r-1} (1 + \lambda - (q + r)\lambda) \leq 1,$$

which is certainly true if $1 + \lambda - (q + r)\lambda \leq 0$, and otherwise it follows by taking the logarithm on both sides, using $\mu \leq \lambda$ and three times $\ln(1 + x) \leq x$ for $x > -1$. In particular, the above inequality will be used on powers of ω and $\bar{\omega}$. Using the equivalence

$$(1 + \mu)^q (1 + \lambda)^{r-1} (1 + \lambda - (q + r)\lambda) \leq 1 \iff (1 + \mu)^q (1 + \lambda)^r - 1 \leq (q + r)\lambda (1 + \mu)^q (1 + \lambda)^{r-1}$$

and setting $\lambda = \mathbf{u}$, $\mu = \mathbf{v}$, we obtain

$$\forall q \geq 0, r \geq 1 \colon \quad \omega^q \bar{\omega}^r - 1 \leq (q + r)\mathbf{u} \omega^q \bar{\omega}^{r-1}. \tag{14}$$

Moreover, setting $\lambda = \mathbf{v}$, $q = 0$ and $\lambda = \mathbf{u}$, $q = 0$, respectively, we derive

$$\forall r \geq 1 \colon \quad \omega^r - 1 \leq r\mathbf{v}\omega^{r-1} \quad \text{and} \quad \bar{\omega}^r - 1 \leq r\mathbf{u}\bar{\omega}^{r-1}. \tag{15}$$

In the cases where $r \geq 1$ may not be satisfied, we further exploit

$$\forall q \geq 0 \colon \quad \bar{\omega}^q - 1 \leq q\mathbf{u}\bar{\omega}^q, \tag{16}$$

which follows from the equivalence with $(1 - q\mathbf{u})(1 + \mathbf{u})^q \leq 1$ and a similar argument as above.

In the proofs of Theorem 4.3 and Lemma 4.6, we exploit that for any nonnegative λ and ν satisfying $r \leq (1 - \frac{\lambda\nu}{2})\nu$,

$$(1 + \lambda)^r \leq 1 + \lambda\nu. \tag{17}$$

To see that we use that the Taylor series implies $x - \frac{x^2}{2} \leq \ln(1+x) \leq x$ for nonnegative x, such that

$$r\ln(1+\lambda) \leq r\lambda \leq \left(1 - \frac{\lambda\nu}{2}\right)\nu\lambda \leq \ln(1+\lambda\nu).$$

Finally, by

$$k \leq \mathbf{u}^{-\frac{1}{2}}, q \geq 0 \implies \bar{\omega}^q \leq \left(1+k^{-2}\right)^q = \exp(q\ln(1+k^{-2})) \leq \exp(q/k^2), \tag{18}$$

we have a relation that is beneficial in the proof of Lemma 4.7.

4.2 Proofs

For the purpose of introducing our line of argumentation and because it can be proved independently, we first give the argument for Theorem 4.3.

Proof of Theorem 4.3. We use (17) with $\lambda := \beta\mathbf{u}$ and $\nu := (\lambda\boldsymbol{\hbar})^{-\frac{1}{2}} \geq k+2$, by which

$$\frac{\psi_k}{\mathbf{u}^2} = (k+2)k(\omega\bar{\omega})^k \leq \nu(\nu-2)(1+\lambda)^{\nu-2} \leq \nu(\nu-2)(1+\lambda\nu) = (\nu-2)(\nu+\boldsymbol{\hbar}^{-1}) \leq \nu^2 - 2\boldsymbol{\hbar}^{-1},$$

such that $\psi_k\boldsymbol{\hbar} \leq ((\beta\mathbf{u}\boldsymbol{\hbar})^{-1} - 2\boldsymbol{\hbar}^{-1})\mathbf{u}^2\boldsymbol{\hbar} = (1-2\beta\mathbf{u})\frac{\mathbf{u}}{\beta} < \mathbf{u}$. Setting $r := c+g$ and $\delta := \hat{c} - r$ yields $|r| = |\hat{c}-\delta| \geq |\hat{c}| - \psi_k\hat{C} = (1-\psi_k\boldsymbol{\hbar})|\hat{c}|$ and therefore, using (8),

$$|\delta| = |\hat{c} - (c+g)| \leq \frac{\psi_k\hat{C}}{(1-\psi_k\boldsymbol{\hbar})|\hat{c}|}|r| = \frac{\psi_k\boldsymbol{\hbar}}{1-\psi_k\boldsymbol{\hbar}}|r| \leq \frac{(1-2\beta\mathbf{u})\mathbf{u}/\beta}{1-\mathbf{u}}|r| \leq \frac{\mathbf{u}}{\beta}|r|.$$

Together with Lemma 3.3, this proves the result. □

The proof of Theorem 4.2 requires appropriate statements for each pair operation. The respective lemmas and their proofs are listed below.

Lemma 4.4. *Let (a,e) be a k_a-approximation of $\hat{a}$ with respect to $\hat{A}$, let (b,f) be a k_b-approximation of $\hat{b}$ with respect to $\hat{B}$, and define*

$$k := 1 + \max\{k_a, k_b\}, \quad \hat{c} := \hat{a} + \hat{b}, \quad \text{as well as} \quad \hat{C} := \begin{cases} |\hat{a}+\hat{b}| & \text{if } k_a = k_b = 0, \\ \hat{A} + \hat{B} & \text{otherwise.} \end{cases}$$

Then the result (c,g) of CPairSum *is a k-approximation of $\hat{c}$ with respect to $\hat{C}$.*

Proof. After possible renaming, we assume without loss of generality that $k_a \geq k_b$. If $k_a = 0$, then CPairSum behaves like TwoSum [12], i.e., the sum is error free. Henceforth we assume $k_a \geq 1$. The first error model yields $t = a+b-c = a+b-(1-\varepsilon_1)(a+b) = \varepsilon_1(a+b)$ as well as

$$g = (1+\varepsilon_3)\big(\varepsilon_1(a+b) + (1+\varepsilon_2)(e+f)\big) \qquad \text{with} \quad |\varepsilon_i| \leq \mathbf{v}.$$

Using $k_a \geq k_b$, the absolute value of g is therefore bounded by

$$\begin{aligned} |g| &\leq (1+\mathbf{v})\mathbf{v}(|a|+|b|) + (1+\mathbf{v})^2(|e|+|f|) \\ &\leq \omega\mathbf{v}(\bar{\omega}^{k_a}\hat{A} + \bar{\omega}^{k_b}\hat{B}) + \omega^2(\varphi_{k_a}\hat{A} + \varphi_{k_b}\hat{B}) \\ &\leq \omega\mathbf{v}\bar{\omega}^{k_a}\hat{C} + \omega^2\varphi_{k_a}\hat{C} \leq \varphi_k\hat{C}. \end{aligned}$$

Moreover, $|a+b| \leq \bar{\omega}^{k_a}\hat{A} + \bar{\omega}^{k_b}\hat{B} \leq \bar{\omega}^{k_a}\hat{C}$ and $|e+f| \leq \varphi_{k_a}\hat{A} + \varphi_{k_b}\hat{B} \leq \varphi_{k_a}\hat{C}$ are used to derive

$$|t+e+f-g| = |\varepsilon_3\varepsilon_1(a+b) + (\varepsilon_2+\varepsilon_3+\varepsilon_2\varepsilon_3)(e+f)| \leq \mathbf{v}^2\bar{\omega}^{k_a}\hat{C} + 2\mathbf{v}\omega\varphi_{k_a}\hat{C} \leq (\mathbf{u}^2\bar{\omega}^{k_a} + 2\mathbf{u}\varphi_{k_a})\hat{C}.$$

Using the above inequality together with

$$|\hat{c} - (a+b+e+f)| \leq |\hat{a}-a-e| + |\hat{b}-b-f| \leq \psi_{k_a}\hat{A} + \psi_{k_b}\hat{B} \leq \psi_{k_a}\hat{C},$$

we validate

$$|\hat{c} - (c+g)| \le |\hat{c} - (a+b+e+f)| + |t+e+f-g| \le (k_a^2 + 2k_a + 1 + 2k_a)(\omega\bar{\omega})^{k_a}\mathbf{u}^2\hat{C}.$$

By $k = k_a + 1$, this gives $|\hat{c} - (c+g)| \le (k^2 + 2k - 2)(\omega\bar{\omega})^{k-1}\mathbf{u}^2\hat{C} \le \psi_k\hat{C}$ and proves (8).

Next, we consider (9). Inequality (9a), and thus also (10), follows by

$$\begin{aligned}
|c - \hat{c}| &= |(1-\varepsilon_1)(a - \hat{a} + b - \hat{b}) - \varepsilon_1\hat{a} - \varepsilon_1\hat{b}| \\
&\le \bar{\omega}(|a - \hat{a}| + |b - \hat{b}|) + \mathbf{u}|\hat{a}| + \mathbf{u}|\hat{b}| \\
&\le \bar{\omega}\big((\bar{\omega}^{k-1} - 1)\hat{A} + (\bar{\omega}^{k-1} - 1)\hat{B}\big) + \mathbf{u}\hat{A} + \mathbf{u}\hat{B} = (\bar{\omega}^k - 1)\hat{C}.
\end{aligned}$$

Finally, for $\hat{C} = |\hat{c}| = s\hat{c}$, we have $\hat{A} + \hat{B} = s\hat{a} + s\hat{b}$, which implies

$$\bar{\omega}^{-k}\hat{C} = \bar{\omega}^{-1}(\bar{\omega}^{-k+1}\hat{A} + \bar{\omega}^{-k+1}\hat{B}) \le \bar{\omega}^{-1}(\bar{\omega}^{-k_a}\hat{A} + \bar{\omega}^{-k_b}\hat{B}) \le \bar{\omega}^{-1}(sa + sb) \le sc,$$

validates the left inequality in (9b), and finishes the argument. □

Lemma 4.5. *Let (a, e) be a k_a-approximation of $\hat{a}$ with respect to $\hat{A}$, let (b, f) be a k_b-approximation of $\hat{b}$ with respect to $\hat{B}$, and define*

$$k := k_a + k_b + 1, \quad \hat{c} := \hat{a}\hat{b}, \quad \text{and} \quad \hat{C} := \hat{A}\hat{B}.$$

Then the result (c, g) of CPairProd *is a k-approximation of $\hat{c}$ with respect to $\hat{C}$.*

Proof. The following argument is mainly based on the first error model yielding $ab - c = ab - (1-\varepsilon_1)ab = \varepsilon_1 ab$ as well as

$$g = (1+\varepsilon_5)\big[\varepsilon_1 ab + (1+\varepsilon_4)\big((1+\varepsilon_2)af + (1+\varepsilon_3)be\big)\big] \qquad \text{with} \quad |\varepsilon_i| \le \mathbf{v}.$$

After possible renaming, we henceforth assume without loss of generality that $k_a \ge k_b$. Define

$$\tau := \begin{cases} 0 & \text{if } k_b = 0, \\ 1 & \text{otherwise.} \end{cases} \tag{19}$$

Taking into account that $k_b = 0 \implies f = 0 = \varepsilon_2 = \varepsilon_4$, the error term g is bounded via

$$\begin{aligned}
|g| &\le \omega\mathbf{v}|ab| + \omega^{\tau+2}(|af| + |be|) \\
&\le \omega\mathbf{v}\bar{\omega}^{k_a}\hat{A}\bar{\omega}^{k_b}\hat{B} + \omega^{k_b+2}(\bar{\omega}^{k_a}\varphi_{k_b} + \bar{\omega}^{k_b}\varphi_{k_a})\hat{A}\hat{B} \\
&\le \Big[(\omega\bar{\omega})^{k_a+k_b+1}\mathbf{u} + (\omega\bar{\omega})^{k_a+1}\varphi_{k_b} + (\omega\bar{\omega})^{k_b+1}\varphi_{k_a}\Big]\hat{C} = \varphi_k\hat{C}.
\end{aligned}$$

By (9a), (8), (16), and (10), we derive

$$\begin{aligned}
|\hat{a}\hat{b} - (ab + af + be)| &= |(\hat{a} - a)(\hat{b} - b) + a(\hat{b} - b - f) + b(\hat{a} - a - e)| \\
&\le (\bar{\omega}^{k_a} - 1)\hat{A}(\bar{\omega}^{k_b} - 1)\hat{B} + |a|\psi_{k_b}\hat{B} + |b|\psi_{k_a}\hat{A} \\
&\le k_a\mathbf{u}\bar{\omega}^{k_a}\hat{A}k_b\mathbf{u}\bar{\omega}^{k_b}\hat{B} + \bar{\omega}^{k_a}\hat{A}\psi_{k_b}\hat{B} + \bar{\omega}^{k_b}\hat{B}\psi_{k_a}\hat{A} \\
&\le \big(k_ak_b + k_b^2 + 2k_b + k_a^2 + 2k_a\big)(\omega\bar{\omega})^k\mathbf{u}^2\hat{C}.
\end{aligned}$$

Moreover, using (15), we obtain

$$\begin{aligned}
|g - (\varepsilon_1 ab + af + be)| &\le (\omega - 1)\mathbf{v}|ab| + (\omega^{\tau+2} - 1)(|af| + |be|) \\
&\le \mathbf{v}^2\bar{\omega}^{k_a}\hat{A}\bar{\omega}^{k_b}\hat{B} + (\tau+2)\mathbf{v}\omega^{\tau+1}\big(\bar{\omega}^{k_a}\varphi_{k_b} + \bar{\omega}^{k_b}\varphi_{k_a}\big)\hat{A}\hat{B} \\
&\le \big(1 + (\tau+2)(k_b + k_a)\big)(\omega\bar{\omega})^k\mathbf{u}^2\hat{C}.
\end{aligned}$$

Adding the above expressions and using

$$\tau(k_a + k_b) - k_a k_b - 1 \le \tau(k_a + k_b) - k_a k_b - \tau^2 = (k_a - \tau)(\tau - k_b) \le 0$$

yield

$$\begin{aligned}
|\hat{c} - (c + g)| &\le |\hat{a}\hat{b} - (ab + af + be)| + |ab - c + af + be - g| \\
&\le \big(k_a k_b + k_b^2 + 2k_b + k_a^2 + 2k_a + 1 + (\tau + 2)(k_b + k_a)\big)\,(\omega\bar{\omega})^k \mathbf{u}^2 \hat{C} \\
&= \big(k^2 + 2k + \tau(k_a + k_b) - k_a k_b - 2\big)\,(\omega\bar{\omega})^k \mathbf{u}^2 \hat{C} \le \psi_k \hat{C},
\end{aligned}$$

which finishes the argument for (8).

Now (9a) follows by (10) and

$$\begin{aligned}
|c - \hat{c}| &= \big|(1 - \varepsilon_1)\big((a - \hat{a})b + \hat{a}(b - \hat{b})\big) - \varepsilon_1 \hat{a}\hat{b}\big| \\
&\le \bar{\omega}\big((\bar{\omega}^{k_a} - 1)\hat{A}\bar{\omega}^{k_b}\hat{B} + \hat{A}(\bar{\omega}^{k_b} - 1)\hat{B}\big) + \mathbf{u}\hat{A}\hat{B} \\
&= \bar{\omega}\big((\bar{\omega}^{k_a}\bar{\omega}^{k_b} - 1)\hat{A}\hat{B}\big) + \mathbf{u}\hat{A}\hat{B} = (\bar{\omega}^k - 1)\hat{C}.
\end{aligned}$$

Finally, if $\hat{C} = |\hat{c}| = s\hat{c}$, then $\hat{A} = |\hat{a}| = s_a\hat{a}$ and $\hat{B} = |\hat{b}| = s_b\hat{b}$ with $s = s_a s_b$, so that (9b) and (3) yield

$$\bar{\omega}^{-k}\hat{C} = \bar{\omega}^{-1}\bar{\omega}^{-k_a}\hat{A}\bar{\omega}^{-k_b}\hat{B} \le \bar{\omega}^{-1}s_a a s_b b \le sc \le \bar{\omega}s_a a s_b b \le \bar{\omega}\bar{\omega}^{k_a}\hat{A}\bar{\omega}^{k_b}\hat{B} = \bar{\omega}^k\hat{C},$$

which completes the proof. □

Lemma 4.6. *Let (a, e) be a k_a-approximation of $\hat{a}$ with respect to $\hat{A}$, let (b, f) be a proper k_b-approximation of $\hat{b}$, and define*

$$k := k_a + k_b + 2, \quad \hat{c} := \frac{\hat{a}}{\hat{b}}, \quad \text{and} \quad \hat{C} := \frac{\hat{A}}{|\hat{b}|}.$$

If $k \le \mathbf{u}^{-\frac{1}{2}}$, then the result (c, g) of CPairDiv *is a k-approximation of $\hat{c}$ with respect to $\hat{C}$.*

Proof. The first standard model yields $c = (1 - \varepsilon_1)\frac{a}{b}$, and, using $t = a - bc = \varepsilon_1 a \in \mathbb{F}$,

$$g = (1 + \varepsilon_6)\frac{(1 + \varepsilon_5)\big((1 + \varepsilon_2)(\varepsilon_1 a + e) - (1 + \varepsilon_3)cf\big)}{(1 + \varepsilon_4)(b + f)} \qquad \text{with} \quad |\varepsilon_i| \le \mathbf{v}.$$

The restriction $k_b \le k - 2 \le \mathbf{u}^{-\frac{1}{2}} - 2$, the definition (7) of ψ_k, and $\omega\bar{\omega} = 1 + 2\mathbf{u}$ imply

$$\psi_{k_b} \le (k_b^2 + 2k_b)(1 + 2\mathbf{u})^{k_b}\mathbf{u}^2 \le (\mathbf{u}^{-1} - 2\mathbf{u}^{-\frac{1}{2}})(1 + 2\mathbf{u})^{1/\sqrt{\mathbf{u}} - 2}\mathbf{u}^2.$$

By (17) with $\lambda = 2\mathbf{u}$, $\nu = \mathbf{u}^{-\frac{1}{2}}$ and $r = \nu - 2$, we see

$$\psi_{k_b} \le (\mathbf{u}^{-1} - 2\mathbf{u}^{-\frac{1}{2}})(1 + 2\mathbf{u}^{\frac{1}{2}})\mathbf{u}^2 = \mathbf{u} - 4\mathbf{u}^2 \le \mathbf{v}.$$

Once more defining τ as in (19), we conclude

$$|b + f| \ge |\hat{b}| - |b + f - \hat{b}| \ge (1 - \psi_{k_b})|\hat{b}| \ge (1 - \tau\mathbf{v})|\hat{b}| = \bar{\omega}^{-\tau}|\hat{b}|. \tag{20}$$

Another consequence of (17) with $\lambda = \mathbf{u}$, $\nu = \mathbf{u}^{-1/2}$, and $r = \nu - 1$ is

$$\omega\bar{\omega}^{k_b} \le \bar{\omega}^{k_b + 1} \le (1 + \mathbf{u})^{1/\sqrt{\mathbf{u}} - 1} \le 1 + \mathbf{u}^{\frac{1}{2}} = \frac{1 - \sqrt{\mathbf{u}} - \mathbf{u} - \mathbf{u}\sqrt{\mathbf{u}} - 2\mathbf{u}^2}{\bar{\omega}(1 - 2\sqrt{\mathbf{u}})} \le \frac{\mathbf{u}^{-\frac{1}{2}} - 1}{\bar{\omega}(\mathbf{u}^{-\frac{1}{2}} - 2)} \le \frac{k_b + 1}{\bar{\omega}k_b}.$$

Together with the conditions (8) and (9b) for $\hat{b}$, this implies

$$|cf| \le \omega\frac{|a||f|}{|b|} \le \omega\frac{|a|\varphi_{k_b}|\hat{b}|}{\bar{\omega}^{-k_b}|\hat{b}|} = \omega\bar{\omega}^{k_b}k_b(\omega\bar{\omega})^{k_b}\mathbf{u}|a| \le \frac{k_b + 1}{\bar{\omega}}(\omega\bar{\omega})^{k_b}\mathbf{u}|a|. \tag{21}$$

We use (20) and (21) to derive

$$\begin{aligned}\frac{|\varepsilon_1 a| + |e| + |cf|}{|b+f|} &\le \frac{\mathbf{v}|a| + \varphi_{k_a}\hat{A} + \frac{k_b+1}{\bar{\omega}}(\omega\bar{\omega})^{k_b}\mathbf{u}|a|}{\bar{\omega}^{-\tau}|\hat{b}|} \\ &\le \left(\bar{\omega}^{\tau}\mathbf{v}\bar{\omega}^{k_a} + \bar{\omega}^{\tau}k_a(\omega\bar{\omega})^{k_a}\mathbf{u} + (k_b+1)(\omega\bar{\omega})^{k_b}\mathbf{u}\bar{\omega}^{k_a}\right)\frac{\hat{A}}{|\hat{b}|} \\ &\le (1+k_a+k_b+1)(\omega\bar{\omega})^{k_a+k_b}\mathbf{u}\hat{C} = k(\omega\bar{\omega})^{k-2}\mathbf{u}\hat{C},\end{aligned}$$

so that

$$|g| \le \omega^3\bar{\omega}\frac{|\varepsilon_1 a| + |e| + |cf|}{|b+f|} \le \omega^3\bar{\omega}k(\omega\bar{\omega})^{k-2}\mathbf{u}\hat{C} \le k(\omega\bar{\omega})^{k}\mathbf{u}\hat{C} = \varphi_k\hat{C}$$

and, using (14),

$$\left|g - \frac{a - bc - cf + e}{b+f}\right| \le (\omega^3\bar{\omega} - 1)\frac{|\varepsilon_1 a| + |e| + |cf|}{|b+f|} \le 4\omega^3\mathbf{u}k(\omega\bar{\omega})^{k-2}\mathbf{u}\hat{C} \le 4k(\omega\bar{\omega})^{k}\mathbf{u}^2\hat{C}.$$

Moreover, we exploit (20), to validate

$$\left|\frac{\hat{a}}{\hat{b}} - \frac{a+e}{b+f}\right| = \left|\frac{\hat{a}-a-e}{b+f} + \frac{(b+f-\hat{b})\hat{a}}{(b+f)\hat{b}}\right| \le \frac{\psi_{k_a}\hat{A}}{\bar{\omega}^{-1}|\hat{b}|} + \frac{\psi_{k_b}|\hat{b}||\hat{a}|}{\bar{\omega}^{-1}|\hat{b}|^2} \le \bar{\omega}(\psi_{k_a} + \psi_{k_b})\hat{C}.$$

Then, adding these two expressions yields

$$\begin{aligned}|\hat{c} - (c+g)| &\le \left|\frac{\hat{a}}{\hat{b}} - \frac{a+e}{b+f}\right| + \left|\frac{a-bc-cf+e}{b+f} - g\right| \\ &\le (k_a^2 + 2k_a + k_b^2 + 2k_b)(\omega\bar{\omega})^k\mathbf{u}^2\hat{C} + 4k(\omega\bar{\omega})^k\mathbf{u}^2\hat{C} \\ &= \left(k^2 + 2k - 2k_ak_b\right)(\omega\bar{\omega})^k\mathbf{u}^2\hat{C} \le \psi_k\hat{C},\end{aligned}$$

which completes the argument for (8).

Once more using the inequalities $\bar{\omega}^{-k_b}|\hat{b}| \le |b| \le \bar{\omega}^{k_b}|\hat{b}|$ due to (9b), we derive

$$\begin{aligned}|c - \hat{c}| &= \left|\frac{(1-\varepsilon_1)(a-\hat{a})}{b} - \frac{\varepsilon_1\hat{a}}{b} + \frac{(\hat{b}-b)\hat{a}}{b\,\hat{b}}\right| \\ &\le \frac{\bar{\omega}(\bar{\omega}^{k_a}-1)\hat{A}}{\bar{\omega}^{-k_b}|\hat{b}|} + \frac{\mathbf{u}|\hat{a}|}{\bar{\omega}^{-k_b}|\hat{b}|} + \left|\frac{\hat{b}-b}{b}\right|\hat{C} \\ &\le \frac{\bar{\omega}(\bar{\omega}^{k_a}-1)\hat{A}}{\bar{\omega}^{-k_b}|\hat{b}|} + \frac{\mathbf{u}\hat{A}}{\bar{\omega}^{-k_b}|\hat{b}|} + \frac{1-\bar{\omega}^{-k_b}}{\bar{\omega}^{-k_b}}\hat{C} = (\bar{\omega}^{k-1} - 1)\hat{C}\end{aligned}$$

and validate (9a). If $\hat{C} = |\hat{c}| = s\hat{c}$ and $\hat{B} = |\hat{b}| = s_b\hat{b}$, then $\hat{A} = |\hat{a}| = s_b s\hat{a}$, so that (9b) is satisfied for a and b. Using (3), we conclude

$$\hat{C} = |\hat{c}| \quad\Longrightarrow\quad \bar{\omega}^{-k+1}\hat{C} = \bar{\omega}^{-1}\frac{\bar{\omega}^{-k_a}\hat{A}}{\bar{\omega}^{k_b}\hat{B}} \le \bar{\omega}^{-1}\frac{s_b s a}{s_b b} \le sc \le \bar{\omega}\frac{s_b s a}{s_b b} \le \bar{\omega}\frac{\bar{\omega}^{k_a}\hat{A}}{\bar{\omega}^{-k_b}\hat{B}} = \bar{\omega}^{k-1}\hat{C}$$

and finish the proof. □

Lemma 4.7. *Let (a, e) be a proper k_a-approximation of $\hat{a}$, and define*

$$\hat{c} := \sqrt{\hat{a}} \quad \text{and} \quad k := \begin{cases} k_a + 1 & \text{if } k_a \le 6, \\ k_a & \text{otherwise.}\end{cases}$$

If $k \le \mathbf{u}^{-\frac{1}{2}}$, then the result (c, g) of CPairSqrt *is a proper k-approximation of $\hat{c}$.*

Proof. In the following we exploit the nonnegativity of $\hat{a}$ and $\hat{c}$ due to Assumption 2, by which $\hat{C} = \hat{c}$ and $\hat{A} = \hat{a}$. The first standard model yields $c = (1+\varepsilon_1)\sqrt{a}$ and

$$g = (1+\varepsilon_4)\frac{(1+\varepsilon_2)(a-c^2+e)}{(1+\varepsilon_3)\big((1+\varepsilon_1)\sqrt{a}+c\big)} = \frac{(1+\varepsilon_4)(1+\varepsilon_2)}{1+\varepsilon_3}\,\frac{2+\varepsilon_1}{2+2\varepsilon_1}\,\frac{a-c^2+e}{\sqrt{a}+c} \tag{22}$$

with $|\varepsilon_i| \le \mathbf{v}$, where $\varepsilon_3 = 0$ for base $\beta = 2$. The inequalities

$$\bar{\omega}^{-\frac{k_a}{2}}\,\hat{c} \le \sqrt{a} \le \bar{\omega}^{\frac{k_a}{2}}\,\hat{c} \qquad \text{and} \qquad \bar{\omega}^{-1}\sqrt{a} \le c \le \bar{\omega}\sqrt{a} \tag{23}$$

derived from (9b) and (3), respectively, as well as

$$1 + \bar{\omega}^{-1} = 2 - \mathbf{v} \ge 2\sqrt{1-\mathbf{v}} = 2\bar{\omega}^{-\frac{1}{2}} \tag{24}$$

are useful in the following argument. By $\frac{a-c^2}{\sqrt{a}+c} = -\varepsilon_1\sqrt{a}$, $|e| \le \varphi_{k_a}\hat{a}$, and (23), we have

$$\left|\frac{a-c^2+e}{\sqrt{a}+c}\right| \le \mathbf{v}\sqrt{a} + \frac{\varphi_{k_a}\hat{a}}{\sqrt{a}+c} \le \mathbf{v}\sqrt{a} + \frac{\varphi_{k_a}\hat{a}}{(1+\bar{\omega}^{-1})\sqrt{a}} \le \left(\mathbf{v}\bar{\omega}^{\frac{k_a}{2}} + \frac{\varphi_{k_a}}{2}\bar{\omega}^{\frac{k_a+1}{2}}\right)\hat{c}. \tag{25}$$

Moreover, defining

$$\tau := \begin{cases} 0 & \text{if } k_a = 0, \\ 1 & \text{otherwise,} \end{cases}$$

and using $k_a = 0 \implies \varepsilon_2 = 0$ as well as $\frac{2-\mathbf{v}}{2-2\mathbf{v}} = 1 + \frac{\mathbf{u}}{2} \le \bar{\omega}\omega^{-\frac{1}{2}}$ and $\frac{1+\mathbf{v}}{1-\mathbf{v}} = \omega\bar{\omega}$, we derive

$$\left|\frac{(1+\varepsilon_4)(1+\varepsilon_2)}{1+\varepsilon_3}\,\frac{2+\varepsilon_1}{2+2\varepsilon_1} - 1\right| \le \frac{(1+\mathbf{v})|1+\varepsilon_2|}{1-\mathbf{v}}\,\frac{2-\mathbf{v}}{2-2\mathbf{v}} - 1 \le \omega^{\tau+\frac{1}{2}}\bar{\omega}^2 - 1, \tag{26}$$

such that, using (22), (26), and (25),

$$|g| \le \omega^{\tau+\frac{1}{2}}\bar{\omega}^2\left(\mathbf{v}\bar{\omega}^{\frac{k_a}{2}} + \frac{\varphi_{k_a}}{2}\bar{\omega}^{\frac{k_a+1}{2}}\right)\hat{c} \le \left(\omega^{\tau+\frac{1}{2}}\bar{\omega}^{\frac{k_a+2}{2}}\mathbf{u} + \frac{\varphi_{k_a}}{2}\bar{\omega}^{\frac{k_a+4}{2}}(\omega\bar{\omega})\right)\hat{c}. \tag{27}$$

Hence, $|g| \le \varphi_k\hat{c}$ follows immediately for $k_a = 0 = \varphi_{k_a}$. On the other hand, for $1 \le k_a \le 6$ we have $k = k_a + 1$, so that (18) yields $\bar{\omega}^{\frac{k_a+4}{2}} \le \exp(\frac{k_a+4}{2(k_a+1)^2}) \le 2$, and (27) validates $|g| \le \varphi_{k_a+1}\hat{c}$ again. For $k_a \ge 7$, where $k = k_a$, we adapt (27) and once more exploit (18) to obtain

$$|g| \le \left(\omega^{\frac{3}{2}}\bar{\omega}^{\frac{k_a+2}{2}}\mathbf{u} + \frac{\varphi_{k_a}}{2}\bar{\omega}^{\frac{k_a+8}{2}}\right)\hat{c} \le \left(1 + \frac{k_a}{2}\exp\left(\frac{k_a+8}{2k_a^2}\right)\right)(\omega\bar{\omega})^{k_a}\mathbf{u}\hat{c}.$$

It is then easy to check that for $k_a = 7$ the right-hand side is smaller than $\varphi_{k_a}\hat{c}$. By the monotonicity of the exponential term, we conclude $|g| \le \varphi_{k_a}\hat{c}$ for all $k_a \ge 7$. This finishes the proof of the left inequality $|g| \le \varphi_k\hat{c}$ in (8).

The argument for $|\hat{c} - (c+g)| \le \psi_k\hat{c}$ is similar. The assumptions imply that the maximum $M(\sqrt{a}, c)$ of $\left|\hat{c} - \frac{\hat{a}+\sqrt{a}c}{\sqrt{a}+c}\right|$ is assumed at both the left or right bounds on $\sqrt{a}$ and c according to (23), respectively. By (23), $\hat{a} = \hat{c}^2$, (16), and (15), we derive that both values of $M(\sqrt{a}, c)$ are equal and

$$\begin{aligned}\left|\hat{c} - \frac{\hat{a}+\sqrt{a}c}{\sqrt{a}+c}\right| &\le \left|\hat{c} - \frac{\hat{a}+\bar{\omega}^{-\frac{k_a}{2}}\hat{c}\,\bar{\omega}^{-\frac{k_a+2}{2}}\hat{c}}{\bar{\omega}^{-\frac{k_a}{2}}\hat{c} + \bar{\omega}^{-\frac{k_a+2}{2}}\hat{c}}\right| = \left|\hat{c} - \frac{\hat{a}+\bar{\omega}^{\frac{k_a}{2}}\hat{c}\,\bar{\omega}^{\frac{k_a+2}{2}}\hat{c}}{\bar{\omega}^{\frac{k_a}{2}}\hat{c} + \bar{\omega}^{\frac{k_a+2}{2}}\hat{c}}\right| \\ &= \frac{\left(\bar{\omega}^{\frac{k_a}{2}}-1\right)\left(\bar{\omega}^{\frac{k_a+2}{2}}-1\right)}{\bar{\omega}^{\frac{k_a}{2}} + \bar{\omega}^{\frac{k_a+2}{2}}}\hat{c} \\ &\le \frac{\frac{k_a}{2}\bar{\omega}^{\frac{k_a}{2}}\mathbf{u}\,\frac{k_a+2}{2}\bar{\omega}^{\frac{k_a}{2}}\mathbf{u}}{\bar{\omega}^{\frac{k_a}{2}} + \bar{\omega}^{\frac{k_a+2}{2}}}\hat{c} = \frac{k_a^2+2k_a}{4(\bar{\omega}^{-1}+1)}\bar{\omega}^{\frac{k_a-2}{2}}\mathbf{u}^2\hat{c}.\end{aligned}$$

Using (23), (24), (26), (14), and (25) yields

$$\begin{aligned}
|\hat{c}-(c+g)| &= \left|\left(\frac{\hat{a}-a-e}{\sqrt{a}+c}\right)+\left(\hat{c}-\frac{\hat{a}+\sqrt{a}c}{\sqrt{a}+c}\right)+\left(\frac{a-c^2+e}{\sqrt{a}+c}-g\right)\right| \\
&\leq \frac{\psi_{k_a}\hat{a}}{(1+\bar{\omega}^{-1})\sqrt{a}}+\frac{k_a^2+2k_a}{4(1+\bar{\omega}^{-1})}\bar{\omega}^{\frac{k_a-2}{2}}\mathbf{u}^2\hat{c}+\left(\omega^{\tau+\frac{1}{2}}\bar{\omega}^2-1\right)\left|\frac{a-c^2+e}{\sqrt{a}+c}\right| \\
&\leq \left(\frac{\psi_{k_a}}{2}+\frac{k_a^2+2k_a}{8\bar{\omega}}\mathbf{u}^2\right)\bar{\omega}^{\frac{k_a+1}{2}}\hat{c}+\frac{2\tau+5}{2}\omega^{\tau+\frac{1}{2}}\bar{\omega}\mathbf{u}\left(\mathbf{v}\bar{\omega}^{\frac{k_a}{2}}+\frac{\varphi_{k_a}}{2}\bar{\omega}^{\frac{k_a+1}{2}}\right)\hat{c} \\
&\leq \left(\frac{5}{8}\bar{\omega}^{\frac{k_a-1}{2}}k_a^2+3\bar{\omega}^{\frac{k_a+2}{2}}k_a+\frac{2\tau+5}{2}(\omega\bar{\omega})^{-\tau}\right)(\omega\bar{\omega})^{k_a+1}\mathbf{u}^2\hat{c}.
\end{aligned}$$

For $k_a = 0 = \tau$ the inequality $|\hat{c}-(c+g)| \leq \psi_{k_a+1}\hat{c} = \psi_k\hat{c}$ is evident, and for $k_a \in \{1,2\}$, using $\mathbf{u} \leq k^{-2} \leq \frac{1}{4}$, the validity of the same is straightforward to check. Moreover, using (18) with $k_a \leq k$, we derive

$$|\hat{c}-(c+g)| \leq \left(\frac{5}{8}\exp\left(\frac{k_a-1}{2k_a^2}\right)k_a^2+3\exp\left(\frac{k_a+2}{2k_a^2}\right)k_a+\frac{7}{2}\right)(\omega\bar{\omega})^{k_a+1}\mathbf{u}^2\hat{c}. \tag{28}$$

Hence, $|\hat{c}-(c+g)| \leq \psi_{k_a+1}\hat{c}$ is satisfied for $k_a = 3$, and by monotonicity of the exponential terms also for all $k_a > 3$. This leaves us with the case $k_a \geq 7$ where $k = k_a$. If we replace $(\omega\bar{\omega})^{k_a+1}$ in (28) with $(\omega\bar{\omega})^{k_a}$ by exploiting $\omega\bar{\omega} \leq \exp(\frac{2}{k_a^2})$, we obtain

$$|\hat{c}-(c+g)| \leq \left(\frac{5}{8}\exp\left(\frac{k_a+3}{2k_a^2}\right)k_a^2+3\exp\left(\frac{k_a+6}{2k_a^2}\right)k_a+\frac{7}{2}\exp\left(\frac{2}{k_a^2}\right)\right)(\omega\bar{\omega})^{k_a}\mathbf{u}^2\hat{c}.$$

Now, in the same manner as above, $|\hat{c}-(c+g)| \leq \psi_{k_a}\hat{c} = \psi_k\hat{c}$ for all $k_a \geq 7$.

Finally, the inequalities in (9b) follow, using (23), by

$$\bar{\omega}^{-k}\hat{c} \leq \bar{\omega}^{-1}\bar{\omega}^{-\frac{k_a}{2}}\hat{c} \leq \bar{\omega}^{-1}\sqrt{a} \leq c \leq \bar{\omega}\sqrt{a} \leq \bar{\omega}\bar{\omega}^{\frac{k_a}{2}}\hat{c} \leq \bar{\omega}^k\hat{c},$$

which completes the proof. □

Proof of Theorem 4.2. Evidently, the input pairs $(p_i, 0)$ are proper 0-approximations of p_i, and the definition of k_j in (11) is consistent with the definitions of k in the Lemmas 4.4–4.7.

Thus, if the input pairs at some inner node j are proper $k_{\text{left}(j)}$- and $k_{\text{right}(j)}$-approximations of the true results $\hat{a}_j$ and $\hat{b}_j$, respectively, then the result (c_j, g_j) of our pair arithmetic for multiplication, division, and square root is a proper k_j-approximation of the true result $\hat{c}_j$, where k_j is defined according to the rules in the respective lemmas. Only addition demands special attention. If the real summands $\hat{a}_j$ and $\hat{b}_j$ satisfy $\hat{a}_j\hat{b}_j \geq 0$, then $\hat{A}_j = |\hat{a}_j|, \hat{B}_j = |\hat{b}_j|$ implies $\hat{C}_j = |\hat{c}_j|$. On the other hand, due to the NIC principle, different signs are only permitted if both addends are input data. In that case $k_{\text{left}(j)} = k_{\text{right}(j)} = 0$ and $\hat{C}_j = |\hat{a}+\hat{b}| = |\hat{c}|$ by definition. Hence, for each node j in the right subtree of a divisor or below a square root, (c_j, g_j) is a proper k_j-approximation of $\hat{c}_j$.

We may therefore replace the nodes comprising of division or square root by leaves containing the respective k_j-approximations. The remaining tree entails only summation and multiplication, so that there is no limit on the quantities k_j assigned to the remaining nodes. By Definition 2.3, every intermediate result of the expression described via (T, p') is nonnegative, such that each operation is consistent with the definition of $\hat{C}$ in Lemma 4.4 and Lemma 4.5, respectively. The statement in Theorem 4.2 follows. □

5 APPLICATIONS AND CONCLUSION

We first comment on possible extensions. Our pair arithmetic covers the four basic operations and the square root. We may ask whether other function could be added in a similar manner, for example, the square of a floating-point number. Of course, that is covered by multiplication; however, the function square may allow for better estimates. Whilst a simpler implementation of CPairSquare derives from CPairProd in the obvious way, the error estimates and in particular the value of k derived from k_a does not improve. Indeed, $(a + e)^2 = a^2 + 2 * e * a + e^2$ implies $\varphi_k \hat{a}^2 \approx 2\varphi_{k_a}\hat{a}^2$, so that k must be at least of the order $2k_a$. Together with the inevitable rounding errors, we obtain $k = 2k_a + 1$, the same as for CPairProd.

For other functions $f(a)$ the first problem is to obtain $c = \mathrm{fl}(f(a))$. Even ignoring the table maker's dilemma [16], the second problem is to compute the residual $c - f(a)$ together with an error bound. In principle, that is possible using some Taylor series expansion, however, with the current IEEE 754 standard [10] it is at least not as simple as for the four basic operations and the square root.

Before addressing possible applications, we want to recall that increasing the precision of each individual computation does not necessarily mean to increase the accuracy of the result. This is true for the double-double format as well as for our proposed floating-point arithmetic with error term. Exemplary, let $\mathbf{u} = 2^{-53}$ be the relative rounding error unit of IEEE 754 binary64, and consider the expression

$$t = ((((1 + \mathbf{u}) + \mathbf{u}^2) - \mathbf{u}) - \mathbf{u}^2) - 1.$$

Note that all quantities involved are representable in binary32. Obviously $t = 0$, the same as computed in binary32. However, binary64 produces $t = -\mathbf{u}$, whereas our pair arithmetic or double-double in binary64 yields $t = -\mathbf{u}^2$. Multiplying the result by some number does not change the correct binary32 value, but produces an arbitrarily large error for binary64, double-double, or our pair arithmetic.

As has been mentioned in the introduction, algorithms for computing faithfully rounded results have been introduced for some specific problems such as powers or products of floating-point numbers [6], Horner's scheme [5], summation and dot product [17], or the Euclidean norm of a vector [7]. Similar estimates are an immediate consequence of our main results. Throughout this last section, we assume that no underflow error or overflow occurs, ensuring that the error model (2) is satisfied.

Corollary 5.1. *For given $x \in \mathbb{F}^n$ denote by (c, g) the product of all x_i computed by our pair arithmetic in any order. If $n \le (\beta\mathbf{u})^{-\frac{1}{2}} - 1$, then $\mathrm{fl}(c + g)$ is a faithful rounding of $\prod_{i=1}^{n} x_i$.*

The proof reduces to counting the number $n - 1$ of operations and applying Theorem 4.2, where $k = n - 1$. In [6] a compensated algorithm is discussed, very similar to our pair arithmetic. Both approaches require the same number of floating-point operations. In [6] it is proved that a faithfully rounded result is computed in IEEE754 binary32 or binary64 provided that

$$n < \frac{\sqrt{1 - \mathbf{u}}}{\sqrt{4 + 2\mathbf{u}} + 2\sqrt{(1 - \mathbf{u})\mathbf{u}}}\mathbf{u}^{-1/2}.$$

A computation shows that, apart from a factor $\sqrt{2}$, both bounds for n are roughly the same. However, our result is applicable for any base β and any order of evaluation. The latter, for instance, allows to vectorize the operations for faster computations via SIMD instructions.

Corollary 5.2. *Let a polynomial* $p(x) := \sum_{i=0}^{n} p_i x^i$ *with* $p_0, \dots, p_n \in \mathbb{F}$ *and* $x \in \mathbb{F}$ *be given, and denote by* (c, g) *the evaluation of* $p(x)$ *by our pair arithmetic using Horner's scheme. If*

$$n \le \frac{1}{2\sqrt{\beta \boldsymbol{\hbar} \mathbf{u}}} - 1 \qquad \text{for} \quad \boldsymbol{\hbar} := \frac{\sum_{i=0}^{n} |p_i||x^i|}{|\sum_{i=0}^{n} p_i x^i|},$$

then $\mathrm{fl}(c + g)$ *is a faithful rounding of* $p(x)$.

Here $\boldsymbol{\hbar}$ reflects the condition number of the evaluation of p at x. Note that $\hat{C} = \sum_{i=0}^{n} |p_i||x^i|$. Evaluation by Horner's scheme requires $2n$ operations, and Theorems 4.2 and 4.3 show the result.

In [5] Graillat uses compensated floating-point operations, similar to our pair arithmetic. He shows that in binary arithmetic his computed result is faithfully rounded provided that

$$\boldsymbol{\hbar} < \frac{(1 - \mathbf{u})(1 - 2n\mathbf{u})^2}{4n^2\mathbf{u}(2 + \mathbf{u})}.$$

A calculation reveals that both conditions are almost identical.

When evaluating a polynomial by Horner's scheme, the error bound necessarily involves the condition number $\boldsymbol{\hbar}$ of evaluation. That is not necessary if the polynomial is given by its roots. If the polynomial has real coefficients, the roots are real or conjugate complex. In that case our pair arithmetic is applicable and the NIC principle is satisfied. That is clear for factors $x - r_i$ where $x, r_i \in \mathbb{F}$. On the other hand, for complex roots $s \pm t\sqrt{-1}$ the factor becomes $(x - s)^2 + t^2$, satisfying the NIC principle as well.

The calculation of k in Theorem 4.2 is as follows. Let p real and q complex roots be given, such that $n = p + 2q$. The trees for each $x - r_i$ are assigned by $k_1 = 1$, whereas the trees for $(x - s)^2 + t^2$ yield $k_2 = 4$, in total $k = pk_1 + qk_2 + p + q - 1 = 2p + 5q - 1 \le \frac{5}{2}n - 1$. Adding one to multiply by the leading coefficient, polynomial evaluation by its roots is faithful by our pair arithmetic for polynomial degrees up to $n \le \frac{2}{5}(\beta\mathbf{u})^{-\frac{1}{2}} - \frac{4}{5}$. In binary64 that means $n \le 26,843,544$.

That example can be extended to polynomial interpolation. Let $p \in \mathbb{R}[x]$ be the polynomial of degree n satisfying $p(x_i) = y_i$ for $i \in \{1, \dots, n\}$, where we assume that the corresponding Vandermonde matrix is regular. Then the polynomial value at some $x \in \mathbb{R}$ satisfies

$$p(x) = \sum_{i=0}^{n} \frac{\prod_{j \ne i} x - x_j}{\prod_{j \ne i} x_i - x_j} y_i =: \sum_{i=0}^{n} \Theta_i(x)\, y_i. \tag{29}$$

All denominators of that expression comply with the NIC principle, thus Theorem 4.2 is applicable. Each of the $n+1$ summands in (29) consists of $2n$ subtractions, $2n-1$ multiplications, and 1 division. Thus, at a node j representing one of these summands, we have $k_j = 4n + 1$. In the worst case, where the sum is evaluated recursively, the final value computes to $k = 5n + 1$, so that a faithfully rounded result is guaranteed if $n \le \frac{1}{5}(\beta\boldsymbol{\hbar}\mathbf{u})^{-\frac{1}{2}} - \frac{3}{5}$, where $\boldsymbol{\hbar} := \sum_{i=0}^{n} |\Theta_i(x) y_i| / |p(x)|$.

We note that in practice it may be more beneficial to use

$$R := \prod_{j=0}^{n} x - x_j, \quad p(x) = \sum_{i=0}^{n} \frac{R}{(x - x_i) \prod_{j \ne i} x_i - x_j}\, y_i,$$

where the factors $\prod_{j \ne i} x_i - x_j$ may be evaluated beforehand to ensure an efficient evaluation of multiple polynomial values. Then Theorem 4.2 is applicable as well with adapted constants.

Corollary 5.3. *For given* $x \in \mathbb{F}^n$ *denote by* (c, g) *the sum of the* x_i *computed by our pair arithmetic in any order. If*

$$n \le \frac{1}{\sqrt{\beta \boldsymbol{\hbar} \mathbf{u}}} - 1 \qquad \text{for} \quad \boldsymbol{\hbar} := \frac{\sum_{i=1}^{n} |x_i|}{|\sum_{i=1}^{n} x_i|},$$

then $\mathrm{fl}(c+g)$ *is a faithful rounding of* $\sum_{i=1}^{n} x_i$. *If binary summation is applied, the same is true if* $\lceil \log_2(n) \rceil \le (\beta \boldsymbol{k} \mathbf{u})^{-\frac{1}{2}} - 2$.

Here, for any order of summation, the k in Theorem 4.2 is bounded by $n-1$, whereas bounded by $\lceil \log_2(n) \rceil + 1$ in case of binary summation. This implies the result.

In [17] a summation based on error-free transformations is given. It is shown that a pair (c, g) is computed with

$$\left| \sum_{i=1}^{n} x_i - (c+g) \right| \le \gamma_{n-2}\gamma_{n-1} \sum_{i=1}^{n} |x_i|,$$

where $\gamma_k := \frac{k\mathbf{u}}{1-k\mathbf{u}}$ [16], as usual, implicitly assumes $k\mathbf{u} < 1$. Hence Lemma 3.3 implies that $\mathrm{fl}(c+g)$ is a faithful rounding of the true sum if

$$\frac{(n-2)(n-1)}{(1-(n-2)\mathbf{u})(1-(n-1)\mathbf{u})} \le \frac{1}{\beta \boldsymbol{k} \mathbf{u}},$$

which is very similar to our result for recursive summation. In [17], however, the estimate was proved only for recursive summation and base $\beta = 2$.

Corollary 5.4. *For given* $x, y \in \mathbb{F}^n$ *denote by* (c, g) *the dot product* $x^T y$ *computed by our pair arithmetic, where summation may be performed in any order. If*

$$n \le \frac{1}{\sqrt{\beta \boldsymbol{k} \mathbf{u}}} - 2 \qquad \text{for} \quad \boldsymbol{k} := \frac{|x|^T |y|}{|x^T y|}, \tag{30}$$

then $\mathrm{fl}(c+g)$ *is a faithful rounding of* $x^T y$. *If binary summation is applied, the same is true if* $\lceil \log_2(n) \rceil \le (\beta \boldsymbol{k} \mathbf{u})^{-\frac{1}{2}} - 3$.

The dot product is computed by a summation tree where each leaf becomes a product node. Thus, the maximal height of the dot product tree is n, and the result follows. The result in [17] mentioned above can be applied to the sum of $a_i + e_i := x_i y_i$ computed by the error-free transformation TwoProduct. However, without further investigation that increases the number of summands to $2n$, so that the bound on n becomes roughly half of that in (30).

To compute faithful results for problems involving a Hilbert matrix, it is treated as a Cauchy matrix with $H_{ij} = C_{ij} = (x_i + y_j)^{-1}$ for $x = (1, \ldots, n)$ and $y = (0, \ldots, n-1)$. Then evaluating Gaussian elimination steps by

$$C_{ij} = C_{ij} - \frac{C_{ik}C_{kj}}{C_{ii}} = C_{ij} \frac{(x_i - x_k)(y_j - y_k)}{(x_k + y_j)(x_i + y_k)}$$

satisfies the NIC-principle. Similar concepts apply to the orientation problem [19] and Householder transformations with the "usual" choice of sign [8].

Corollary 5.5. *For given* $x \in \mathbb{F}^n$ *denote by* (c, g) *the Euclidean norm* $\|x\|_2$ *computed by our pair arithmetic. If* $n \le (\beta \mathbf{u})^{-\frac{1}{2}} - 3$, *then* $\mathrm{fl}(c+g)$ *is a faithful rounding of* $\|x\|_2$. *In case of binary summation, the same is true if* $\lceil \log_2(n) \rceil \le (\beta \mathbf{u})^{-\frac{1}{2}} - 4$.

Note that Theorem 4.2 is applicable because the NIC-principle is satisfied. The height of the summation tree is at most n, so that including the last operation square root the k in Theorem 4.2 is bounded by $n+1$ for any order of summation, and by $\lceil \log_2(n) \rceil + 2$ in case of binary summation. This implies the result. For binary summation in IEEE754 binary64 the maximal vector length for our pair arithmetic to produce a faithfully rounded result is about $2^{2^{26}}$.

In [7] double-double arithmetic is used to compute $\|x\|_2$ with recursive summation, adapted to the fact that the summands are nonnegative. That requires $13n + 1$ floating-point operations

as compared to $10n + 1$ for our pair arithmetic when using TwoProduct rather than CPairProd. However, in [7] up to a considerably larger maximum vector length $n < \frac{1}{24\mathbf{u}+\mathbf{u}^2} - 3$ a faithfully rounded result is guaranteed.

That is a principal trade-off: Faithful rounding is ensured for larger n at the cost of more operations. By carefully going through our estimates it becomes clear that including the normalization step of double-double arithmetic into our pair arithmetic ensures faithful rounding up to n close to $\mathbf{u}^{-1}$, however, adding 3 operations to each of our pair operations.

One may argue that our limit $(\beta\mathbf{u})^{-\frac{1}{2}}$ for k is less than 3000 in binary32. However, in order to compute a more accurate result it seems more efficient to use binary64 rather than a pair arithmetic based on binary32. In fact, applying our pair arithmetic seems useful for the most precise and easily accessible floating-point format, which is often binary64. In that case the limit $(\beta\mathbf{u})^{-\frac{1}{2}} = 2^{26}$ for k seems large enough for many applications. Therefore, we opted for faster operations rather than for a larger limit of k.

6 ACKNOWLEDGMENT

The authors want to express their warmest thanks to Dr. Florian Bünger, who read the manuscript very carefully and gave several, most valuable comments.

REFERENCES

[1] D.H. Bailey. A Fortran-90 based multiprecision system. *ACM Trans. Math. Software*, 21(4):379–387, 1995.
[2] G. Bohlender, W. Walter, P. Kornerup, and D. W. Matula. Semantics for exact floating point operations. In *Proceedings 10th IEEE Symposium on Computer Arithmetic*, pages 22–26, 1991.
[3] S. Boldo, S. Graillat, and J. M. Muller. On the robustness of the 2Sum and Fast2Sum algorithms. *ACM Trans. Math. Software*, 44(1):4:1–14, 2017.
[4] J. B. Demmel, I. Dumitriu, O. Holtz, and P. Koev. Accurate and efficient expression evaluation and linear algebra. *Acta Numerica*, 2008:87–145, 2008.
[5] S. Graillat. Provably faithful evaluation of polynomials. In *Proceedings of the 2006 ACM Symposium on Applied Computing*, Dijon, France, 2006.
[6] S. Graillat. Accurate floating-point product and exponentiation. *IEEE Trans. on Comp.*, 58(7):994–1000, 2009.
[7] S. Graillat, C. Lauter, P. Tang, N. Yamanaka, and S. Oishi. Efficient calculations of faithfully rounded l_2-norms of n-vectors. *ACM Trans. Math. Software*, 41(4):24:1–20, 2015.
[8] N.J. Higham. *Accuracy and Stability of Numerical Algorithms.* SIAM Publications, Philadelphia, 1996.
[9] *ANSI/IEEE 754-1985: IEEE Standard for Binary Floating-Point Arithmetic.* New York, 1985.
[10] *ANSI/IEEE 754-2008: IEEE Standard for Floating-Point Arithmetic.* New York, 2008.
[11] Mioara Joldes, Jean-Michel Muller, and Valentina Popescu. Tight and rigourous error bounds for basic building blocks of double-word arithmetic. working paper or preprint, July 2016.
[12] D.E. Knuth. *The Art of Computer Programming: Seminumerical Algorithms*, volume 2. Addison Wesley, Reading, Massachusetts, third edition, 1998.
[13] S. Linnainmaa. Software for doubled-precision floating point computations. *ACM Trans. Math. Software*, 7:272–283, 1981.
[14] O. Møller. Quasi double precision in floating-point arithmetic. *BIT Numerical Mathematics*, 5:37–50, 1965.
[15] MPFR: A C library for multiple-precision floating-point computations with exact rounding. Code and documentation available at `http://www.mpfr.org`.
[16] 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.
[17] T. Ogita, S.M. Rump, and S. Oishi. Accurate sum and dot product. *SIAM Journal on Scientific Computing (SISC)*, 26(6):1955–1988, 2005.
[18] S.M. Rump, T. Ogita, and S. Oishi. Accurate floating-point summation part I: Faithful rounding. *SIAM J. Sci. Comput.*, 31(1):189–224, 2008.
[19] J.R. Shewchuk. Adaptive precision floating-point arithmetic and fast robust geometric predicates. *Discrete Comput. Geom.*, 18(3):305–363, 1997.