Extension of floating-point filters to absolute and relative errors for numerical computation

Yuki Ohta\textsuperscript{1} and Katsuhisa Ozaki\textsuperscript{2}
\textsuperscript{1} Functional Control Systems, Graduate School Engineering and Science, Shibaura Institute of Technology
\textsuperscript{2} Department of Mathematical Sciences, College of Systems Engineering and Science, Shibaura Institute of Technology
E-mail: nb15102@shibaura-it.ac.jp, ozaki@sic.shibaura-it.ac.jp

Abstract. Although numerical computation is very fast, however, the results may not be accurate due to the accumulation of rounding errors. Consequently, much research has focussed on ways to verifying the accuracy of approximate solutions. Floating-point filters are one such technique. These can, for example, be used to guarantee the signs of computed results, such as those of the matrix determinants that are so important in the computational geometry field. In this paper, we extend floating-point filters to guarantee absolute and relative errors.

1. Introduction
Floating-point number and floating-point arithmetic defined by IEEE 754 [1] are widely used by modern computers. However although numerical computation is fast, the computed results may not be exact due to accumulation of rounding errors. For example, the result of computing $1 + 2^{-53}$ using the IEEE 754 binary64 format (otherwise known as double precision floating-point numbers) is 1, according to the rule of rounding to the nearest and breaking ties by rounding to even (roundTiesToEven). Hence, the result of computing
\[
(((1 + 2^{-53}) + 2^{-53}) + \cdots + 2^{-53}) + 2^{-53}
\]
is 1, independent of the number of terms. If we define the $n$-vector $p$ as
\[
p_1 = 1, \quad p_i = 2^{-53}, \quad p_{n-1} = -1, \quad p_n = -2^{-53},
\]
the result of computing $\sum_{i=1}^{n} p_i$ in the recursive order is $-2^{-53}$, despite the fact that the exact result is positive for $n \geq 5$. These examples show some of the potential accuracy issues with numerical computation and why verifying accuracy is important. Floating-point filters are one approach this problem [2, 3, 4, 5].

In this paper, we propose computable methods that provide guaranteed absolute or relative error bounds based on a previously known rounding error analysis technique. Here, computable means that all error bound checks are carried out using only floating-point arithmetic in roundTiesToEven mode and we do not require any switches of rounding mode.
2. Notation and error bounds
First, we introduce several floating-point arithmetic lemmas, which will be used to develop the floating-point filters. The floating-point numbers and its arithmetic used in this paper are ruled by the IEEE 754 standard [1].

2.1. Floating-point arithmetic and rounding errors
We define the function \( \text{fl} \) as follows. For \( x, y \in \mathbb{F} \), \( \text{fl}(x \circ y) \) returns \( z \), where \( z \) is the nearest floating-point number to \( x \circ y \). If there are two candidates for \( z \), namely, \( x \circ y \) lies at the mid-point between neighboring floating-point numbers, then \( \text{fl}(x \circ y) \) returns the result of applying the roundTiesToEven rule. In addition, \( \text{float}(\cdot) \) evaluates the expression in any order of floating-point arithmetic. This notation is the same as that used in [6].

From now on, we omit the \( \text{fl}(\cdot) \), for brevity. For example, \( \text{fl}((a + b) + (c + d)) \) means \( \text{fl}(\text{fl}(a + b) + \text{fl}(c + d)) \) for \( a, b, c, d \in \mathbb{F} \). Let the constants \( u, u_N \) and \( u_S \) be defined as follows.
- \( u \): the relative rounding error unit
- \( u_N \): the smallest positive normalized floating-point number
- \( u_S \): the smallest positive floating-point number

In binary64, these constants are \((u, u_N, u_S) = (2^{-53}, 2^{-1022}, 2^{-1074})\).

Next, we introduce the following theorem for addition, subtraction and multiplication using floating-point arithmetic.

Theorem 1 ([7, 8, 9]) For \( x, y \in \mathbb{F} \), the following holds:
\[ x \circ y = (1 + \delta_1)\text{fl}(x \circ y) + \eta_1, \quad |\delta_1| \leq u, \quad |\eta_1| \leq \frac{u_S}{2}, \quad \delta_1 \cdot \eta_1 = 0, \]
where \( \circ = \{+, -, \cdot\} \). Especially, \( \eta_1 = 0 \) for \( \circ \in \{+, -\} \). Moreover, if \( \text{fl}(|x \cdot y|) \geq u_N \), then \( \eta_1 = 0 \); otherwise, \( \delta_1 = 0 \).

We also need to introduce several additional functions. For \( a, b, c \in \mathbb{F} \) and \( x \in \mathbb{R} \), we define
\[ \text{ufp}(x) := \begin{cases} 2^{\lfloor \log_2|x| \rfloor} & x \neq 0 \\ 0 & x = 0 \end{cases}, \quad (1) \]
\[ \text{RN}(x) := \min\{f \in \mathbb{F} \mid |f - x|\}, \]
\[ \text{succ}(x) := \min\{f \in \mathbb{F} \mid f > x\}, \quad (2) \]
\[ \text{pred}(x) := \max\{f \in \mathbb{F} \mid f < x\}, \]
\[ \text{FMA}(a, b, c) := \min\{f \in \mathbb{F} \mid |f - (ab + c)|\}. \quad (3) \]
Here, \( \text{FMA}(\cdot) \) means the fused multiply-add function. The rounding mode of the functions \( \text{RN}(x) \) and \( \text{FMA}(\cdot) \) is roundTiesToEven in IEEE 754. From (1), (2) and the definition of floating-point numbers, we have
\[ |a| < 2\text{ufp}(a) \quad a \neq 0, \]
\[ |a| \leq 2\text{ufp}(a) - \max\{2u\text{ufp}(a), u_S\} \quad a \neq 0, \quad (4) \]
\[ \text{succ}(a) = a + \max\{2u\text{ufp}(a), u_S\} \quad a \geq 0, \]
\[ \text{succ}(a) \geq a + 2u\text{ufp}(a) \quad a \geq 0, \quad (5) \]
where \( a \in \mathbb{F} \).

Now, we show 4 lemmas. Using the roundTiesToEven rule, the following lemma holds trivially.
Lemma 1 For $a \in \mathbb{R}$, $b \in \mathbb{F}$ and $\circ = \{+,-,\cdot\}$,
\[
\begin{align*}
\text{RN}(a) > b & \Rightarrow a > b \\
\text{RN}(a) < b & \Rightarrow a < b
\end{align*}
\]
are satisfied.

Lemma 2 For $0 \leq a, b \in \mathbb{F}$,
\[
ab \leq \text{fl}(\text{succ}(ab)) + \frac{u_S}{2}
\]
holds true.

Proof 1 We consider the following case distinction:

- the case of $\text{fl}(ab) \geq u_N$

Let $X := \max\{2u \text{ufp}(a), u_S\}$. From (5), Theorem 1 and (4), we have
\[
\begin{align*}
\text{fl}(\text{succ}(ab)) &= \text{fl}((a + X)b) \\
&\geq (a + X)b - u(a + X)b \\
&= ab + ((1 - u)X - au)b \\
&\geq ab + ((1 - u)X - (2u \text{ufp}(a) - X)u)b \\
&= ab + (X - 2u \text{ufp}(a))b \\
&= ab + (\max\{2u \text{ufp}(a), u_S\} - 2u \text{ufp}(a))b \geq ab.
\end{align*}
\]
Thus, $ab \leq \text{fl}(\text{succ}(ab))$ is satisfied.

- the case of $\text{fl}(ab) < u_N$

From the assumption and Theorem 1,
\[
ab \leq \text{fl}(ab) + \frac{u_S}{2} \leq \text{fl}(\text{succ}(ab)) + \frac{u_S}{2}
\]
is satisfied.
Thus, we proved that (7) is satisfied.

Lemma 3 For $0 \leq c, d < u^{-1}$ and $c, d \in \mathbb{F} \cap \mathbb{N}_0$,
\[
cu + du^2 \leq \text{fl}((cu + (d + \text{ufp}(c))u^2)
\]
holds true.

Proof 2 For $c = 0$ or $d = 0$, (8) trivially holds. Hence, we assume $c \neq 0$ and $d \neq 0$ for the remainder of the proof. If $\text{fl}(cu + du^2) = cu + du^2$, then (8) holds, because $cu = \text{fl}(cu)$, $du^2 = \text{fl}(du^2)$ and $d < \text{fl}(d + \text{ufp}(c))$. Thus, we focus on the case where $\text{fl}(cu + du^2) \neq cu + du^2$.

Since $cu + du^2 \notin \mathbb{F}$, it lies between neighboring floating-point numbers. Thus, there exists a $k \in \mathbb{N}_0$ such that
\[
\alpha_1 := cu + 2ku \text{ufp}(c)u^2 < cu + du^2 < cu + 2(k + 1)ufp(c)u^2 =: \alpha_2,
\]
where $\alpha_2 = \text{succ}(\alpha_1)$ from $ufp(cu + du^2) = ufp(cu)$ and (3). Because
\[
|cu + (d + \text{ufp}(c))u^2 - \alpha_1| > |cu + (d + \text{ufp}(c))u^2 - \alpha_2|
\]
\[
\text{fl}(cu + (d + \text{ufp}(c))u^2) = \alpha_2. \text{ This completes the proof.}
\]
Lemma 4  For $0 \leq c, d < u^{-1}$, $c \neq 0$, $c, d \in \mathbb{F}$, $c, d \in \mathbb{Z}$ and $0 \leq f \in \mathbb{F}$,
\[
(cu + du^2) f \leq \text{fl}((cu + (d + 3\text{ufp}(c))u^2) f) + \frac{us}{2}
\]
is satisfied.

Proof 3  From Lemma 3, we have
\[
(cu + du^2) f \leq \text{fl}(cu + (d + \text{ufp}(c))u^2) f.
\]

Here again, we consider two cases.

- $\text{ufp}(\text{fl}(cu + (d + \text{ufp}(c))u^2) f) = \text{ufp}(\text{fl}(cu + du^2) f)$

  This assumption follows from (3) that
\[
\text{succ}(\text{fl}(cu + (d + \text{ufp}(c))u^2) f) = \text{fl}(cu + (d + \text{ufp}(c))u^2) f + 2\text{ufp}(c)u^2 = \text{fl}(cu + (d + 3\text{ufp}(c))u^2).
\]

Substituting (2) into (9) then completes the proof.

- $\text{ufp}(\text{fl}(cu + (d + \text{ufp}(c))u^2) f) > \text{ufp}(\text{fl}(cu + du^2) f)$

  There must exist a $g \in \mathbb{N}$, where $g$ is a power of two, such that
\[
\text{ufp}(\text{fl}(cu + (d + \text{ufp}(c))u^2) f) \geq g > \text{ufp}(\text{fl}(cu + du^2) f) \geq \text{ufp}(cu + du^2).
\]

Therefore, we have
\[
(cu + du^2) f \leq gf \leq \text{fl}(gf) + \frac{us}{2} \leq \text{fl}(\text{ufp}(\text{fl}(cu + (d + \text{ufp}(c))u^2) f)) + \frac{us}{2}
\]
\[
\leq \text{fl}((cu + (d + \text{ufp}(c))u^2) f) + \frac{us}{2}.
\]

This complete the proof.

We are now in a position to introduce several error bounds for use in numerical computation. Let $p \in \mathbb{F}^n$ and $x, y \in \mathbb{F}^n$. [6, 10, 11] give the following summation and inner product error bounds:
\[
\sum_{i=1}^{n} p_i - \text{float}(\sum_{i=1}^{n} p_i) \leq (n - 1)u \sum_{i=1}^{n} |p_i|,
\]
\[
\sum_{i=1}^{n} p_i - \text{float}(\sum_{i=1}^{n} p_i) \leq (n - 1)u \text{ufp}(\text{float}(\sum_{i=1}^{n} |p_i|)), \tag{10}
\]
\[
|x^T y - \text{float}(x^T y)| \leq (n + 2)u \text{ufp}(\text{float}(|x^T y|)) + \frac{n}{2}us, \tag{11}
\]
where $(n + 2)u \leq 1$ in (11). Here, $\text{float}(\cdot)$ is evaluated in the same order on both the left and right sides of (10) and (11). Let the polynomial of degree $n$ be
\[
p(x) = \sum_{i=0}^{n} a_i x^i = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n
\]
where \( a_0, \ldots, a_n, x \in \mathbb{F} \). Now, let \( H_n(x) \) be the Horner scheme for \( p(x) \) and define \( \hat{H}_n(x) \) by taking the absolute values of all elements in \( H_n(x) \), as follows:

\[
p(x) = H_n(x) = (\ldots((a_n x + a_{n-1}) x + a_{n-2}) \cdots + a_2) x + a_1) x + a_0, \\
\hat{H}_n(x) = (\ldots(|a_n| |x| + |a_{n-1}| |x| + |a_{n-2}|) \cdots + |a_2| |x| + |a_1| |x| + |a_0|).
\]

In [12], the error in Horner’s method can now be bounded as follows:

\[
|p(x) - \text{fl}(H_n(x))| \leq (2n u + 8n^2 u^2) \text{fl}(\hat{H}_n(x)),
\]

(12)

where we have assumed that no underflow occurs when evaluating \( \text{fl}(H_n(x)) \).

In [13], the error in \((a + b)(c + d)\) for \( a, b, c, d \in \mathbb{F} \) can be bounded by

\[
|\text{fl}((a + b)(c + d)) - (a + b)(c + d)| \leq (3u - (\phi - 14)u^2) \text{fl}(|(a + b)(c + d)|) + S,
\]

(13)

where

\[
\phi = 2 \left[ -1 + \sqrt{\frac{4u^{-1} + 45}{4}} \right], \quad 0 \leq S < \frac{1}{2} u_S + \frac{3}{2} uu_S < u_S.
\]

### 3. Floating-point filter

Here, we focus on evaluating an expression using floating-point arithmetic, and let \( t \in \mathbb{R} \) and \( x \in \mathbb{F} \) be the exact and computed results, respectively. We also assume that

\[
|t - x| \leq (cu + du^2) f + eu_S,
\]

(14)

where \( 0 \leq c, d \in \mathbb{F} \cap \mathbb{Z}, e \in \mathbb{F} \cap \mathbb{Z}/2, 0 \leq f \in \mathbb{F}, c, d < u^{-1}, |x| \leq f \) and \( e < u^{-1} \). Various error bounds can be represented in the form (14), such as (10), (11), (12) and (13).

The simplest way to develop a floating-point filter is to switch rounding modes. IEEE 754 also defines the directed floating-point rounding modes roundTowardPositive and roundTowardNegative, which we represent as \( \text{fl}_{\Delta}(\cdot) \) and \( \text{fl}_{\triangledown}(\cdot) \), respectively. From (14), if

\[
k > \text{fl}_{\Delta}((cu + du^2)f + eu_S)
\]

for \( 0 \leq k \in \mathbb{F} \), then the absolute error bound is strictly less than \( k \). Likewise, if

\[
X < \text{fl}_{\triangledown}(k(|x| - X)), \quad X = \text{fl}_{\Delta}((cu + du^2)f + eu_S)
\]

then the relative error bound is strictly less than \( k \).

Here, we propose floating-point filters that do not require a change of rounding mode, and instead only use the rounding to the nearest (roundTiesToEven) mode. Since this is the default rounding mode in many computational environments, the proposed floating-point filters are widely applicable.

### 3.1. Special Cases (no underflows)

First, we consider several special cases. If no underflow occurs in the floating-point arithmetic, we can set \( e = 0 \) in (14). We can then show the following theorem holds trivially using Lemma 1 and Lemma 3.
Theorem 2  Given (14) with \(e = 0\), if
\[
|x| > \text{fl}((cu + (d + \text{ufp}(c))u^2)f)
\]
then \(t\) and \(x\) have the same sign. In addition, if
\[
k > \text{fl}((cu + (d + \text{ufp}(c))u^2)f)
\]
then the absolute error of \(x\) is strictly less than \(k\).

Next, we propose a floating-point filter for the relative error.

Theorem 3  Assume (14) with \(e = 0\). Let \(X = \text{fl}((cu + (d + 3\text{ufp}(c))u^2)f)\), \(Y = \text{fl}(|x| - X)\) and \(r = \max\{2u\text{ufp}(Y), u_S\}\). If
\[
X < \text{fl}(k(Y - r)) \tag{15}
\]
for \(0 \leq k \in \mathbb{F}\), then the relative error bound is strictly less than \(k\).

Proof 4  By Lemma 4, we have \(|x - t|\leq X\). If \(t = 0\), then \(|x| \leq X\) and \(Y \leq 0\), meaning that \(\text{fl}(k(Y - r)) < 0\) and hence that (15) cannot hold. Therefore, we assume \(t \neq 0\) in the remainder of the proof. We have
\[
\frac{|x - t|}{|t|} < k \iff |x - t| < k|t|.
\]
Now, we evaluate the lower bound of \(|t|\) as follows by Theorem 1.
\[
|t| \geq |x| - |x - t| \geq |x| - X \geq Y - r = \text{fl}(Y - r), \quad Y - r \in \mathbb{F}.
\]
Therefore, if \(X < k\text{fl}(Y - r)\), then we have shown that \(|x - t| < k|t|\). By Lemma 1, if (15) holds, then \(X < k\text{fl}(Y - r)\). This complete the proof. \(\square\)

3.2. General Cases

Now, we give computable error bounds in the general form (14).

Theorem 4  Given (14), if
\[
|x| > \text{fl}((cu + (d + 3\text{ufp}(c))u^2)f + (e + 1)u_S)
\]
then \(t\) and \(x\) have same sign.

Proof 5  From (6), we have
\[
|x| > \text{fl}((cu + (d + 3\text{ufp}(c))u^2)f + (e + 1)u_S) > \text{fl}((cu + (d + 3\text{ufp}(c))u^2)f + u_S/2 + eu_S
\]
\[
> (cu + du^2)f + eu_S
\]
Here, Lemma 4 is used for the last inequality. \(\square\)

Theorem 5  Given (14), if
\[
k > \text{fl}((cu + (d + 3\text{ufp}(c))u^2)f + (e + 1)u_S)
\]
for \(0 \leq k \in \mathbb{F}\), then the absolute error bound is strictly less than \(k\).
This theorem can be proved in a similar way to Theorem 4.

Next, we propose a floating-point filter for the relative error.

**Theorem 6** Assume (14), and let $X = \text{fl}((cu + (d + 3ufp(c))u^2)f)$, $Y = \text{fl}(X + (e + 1)uS)$, $Z = \text{fl}(|x| - Y)$ and $r = \max\{2uufp(Z), uS\}$. If

$$Y < \text{fl}(k(Z - r)) \quad (16)$$

for $0 \leq k \in \mathbb{F}$, then the relative error bound is strictly less than $k$.

**Proof 6** If $t = 0$, we can derive $Z < 0$ by a similar argument to that in Theorem 3 and (16) cannot hold. Hence, we will assume $t \neq 0$ for the remainder of the proof. Then, we have

$$|x - t| < k \iff |x - t| < k|t| .$$

By Lemma 4, we have $|x - t| \leq Y$, and we can apply Theorem 1 to find the following lower bound on:

$$|t| \geq |x| - |x - t| \geq |x| - Y \geq Z - r = \text{fl}(Z - r)$$

By Lemma 1, if (16) holds, then $X < k\text{fl}(Z - S)$, and hence $|x - t| < k|t|$. □

### 3.3. Bound for Fused Multiply-Add

Here, we propose a computable bound for the Fused Multiply-Add operation. The following theorem follows immediately from Lemma 1 and Lemma 3.

**Theorem 7** Given (14), if

$$|x| > \text{FMA}\left((cu + (d + ufp(c))u^2), f, \text{fl}(e + 1)uS)\right)$$

then $t$ and $x$ have same sign. In addition, if

$$k > \text{FMA}\left((cu + (d + ufp(c))u^2), f, \text{fl}(e + 1)uS)\right)$$

then the absolute error bound on $x$ is strictly less than $k$.

Next, we propose a floating-point filter for the relative error.

**Theorem 8** Assume (14), and let $X = \text{FMA}\left((cu + (d + ufp(c))u^2), f, \text{fl}(e + 1)uS)\right)$, $Y = \text{fl}(|x| - X)$ and $r = \max\{2uufp(Y), uS\}$. If

$$|x| > \text{fl}(k(Y - r)) \quad (17)$$

for $0 \leq k \in \mathbb{F}$, then the relative error bound is strictly less than $k$.

**Proof 7** By a similar argument to that used for Theorems 3 and Theorems 6, we can assume $t \neq 0$ and to prove $|x - t| < k|t|$. By Lemma 4, $|x - t| \leq X$ and we can apply Theorem 1 to find the following lower bound on $|t|:

$$|t| \geq |x| - |x - t| \geq |x| - X \geq Y - r = \text{fl}(Y - r)$$

By Lemma 1, if (17) holds, then $X < k\text{fl}(Y - r)$, and hence $|x - t| / |t| < k$. □
Conclusion
In this paper, we have proposed several floating-point filters that guarantee the absolute or relative error is less than a positive constant $k$. These only use only floating-point arithmetic in rounding to the nearest (roundTiesToEven) mode, and hence, could be applied in a wide range of computational environments.

References
[1] ANSI, IEEE Standard for Floating-Point Arithmetic, Std 754-2008, 2008.
[2] Ozaki K., Ogita T and Oishi S, 2012, A robust algorithm for geometric predicate by error-free determinant transformation, *Information and Computation*, 216, 3–13.
[3] Brönnimann H, Burnikel C and Pion S, 2001, Interval arithmetic yields efficient dynamic filters for computational geometry, *Discrete Applied Mathematics: 14th European Workshop on Computational Geometry*, 109, 25–47.
[4] Burnikel C, Funke S and Seel M, 2001, Exact geometric computation using cascading, *International Journal of Computational Geometry & Applications*, 11, 245–266.
[5] Melquiond G and Pion S, 2007, Formally certified floating-point filters for homogeneous geometric predicates, *RAIRO - Theoretical Informatics and Applications*, 41, 57-69.
[6] Jeannerod C. -P., Rump S. M., 2013, Improved error bounds for inner products in floating-point arithmetic, *SIAM J. Matrix Anal. & Appl.*, 34, 338–344.
[7] Mehlhorn K and Näher S, 1995, Experiences with the Implementation of Geometric Algorithms, *Algorithms and Data Structures: 4th International Workshop (WADS95)*, 955, ed Selim G. Akl, Frank Dehne, Jörg-Rüdiger Sack, Nicola Santoro, (Berlin: Springer-Verlag) 223–231.
[8] Jeannerod C P and Rump S M, 2018, On relative errors of floating-point operations: optimal bounds and applications, *Mathematics of Computation*, (Providence: American Mathematical Society), 87, 803–819.
[9] Higham N J, 2002, Accuracy and Stability of Numerical Algorithms, *Society for Industrial and Applied Mathematics*, (Philadelphia: SIAM).
[10] Rump S M, 2012, Error estimation of floating-point summation and dot product, *BIT Numerical Mathematics*, 52, 201–220.
[11] Rump S M, 2015, Computable backward error bounds for basic algorithms in linear algebra, *Nonlinear Theory and Its Applications, IEICE*, 6, 360-363.
[12] Ohta Y and Ozaki K, 2018, Verification of comparison of two computed results, *Transactions of the Japan Society for Industrial and Applied Mathematics*, 28, 1–17 (in Japanese).
[13] Ozaki K, Bünger F, Ogita T, Oishi S and Rump S M, 2016, Simple floating-point filters for the two-dimensional orientation problem, *BIT Numerical Mathematics*, 56, 729–749.