Getting tight error bounds in floating-point arithmetic: illustration with complex functions, and the real $x^n$ function

Jean-Michel Muller

collaborators: S. Graillat, C.-P. Jeannerod, N. Louvet V. Lefèvre
NSV–2014
Floating-Point Arithmetic

- too often, viewed as a set of cooking recipes;
- too many “theorems” that hold... provided no variable is very near a power of the radix, there is no underflow/overflow; or that are dangerously generalized from radix 2 to radix 10, etc.
- simple models such as the standard model

$$\circ(a \oplus b) = (a \oplus b) \cdot (1 + \delta), \quad |\delta| \leq u,$$

$$(u = 2^{-p} \text{ in radix } 2, \text{ precision-}p, \text{ rounded to nearest, arithmetic}) \text{ do not allow to catch subtle behaviors such as those in}$$

$$s = a + b ; z = s - a ; r = b - z$$

(fast2sum) and many others.

- by the way, are these “subtle behaviors” robust?
Long term goals

- revisit "folklore knowledge" on FP arithmetic, and determine which properties are really true, and in what context they are true;
- build new knowledge on FP arithmetic;
- try to get optimal/asymptotically optimal/close-to-optimal error bounds;

... all this in close collaboration with the formal proof folks.
Binary Floating-Point System

Parameters:

\[
\begin{cases}
\text{radix (or base)}: 2 & \text{here;}
\text{precision} & p \geq 1
\text{extremal exponents} & e_{\text{min}}, e_{\text{max}},
\end{cases}
\]

A finite FP number \( x \) is represented by 2 integers:

- integral significand: \( M, |M| \leq 2^p - 1; \)
- exponent \( e, e_{\text{min}} \leq e \leq e_{\text{max}}. \)

such that

\[
x = M \times 2^{e+1-p}
\]

with \( |M| \) largest under these constraints \((\rightarrow |M| \geq 2^{p-1}, \text{unless } e = e_{\text{min}})\).

(Real) significand of \( x \): the number \( m = M \times 2^{1-p} \), so that \( x = m \times 2^e. \)
Correct rounding

- In general, the sum, product, quotient, etc., of two FP numbers is not an FP number: it must be rounded;
- **correct rounding**: *Rounding function* $\circ$, and when $(a \top b)$ is performed, the returned value is $\circ(a \top b)$;

- default rounding function RN (round to nearest even):

  1. For all FP numbers $y$, $|\text{RN}(t) - t| \leq |y - t|$  
  2. If there are two FP numbers that satisfy (i), RN$(t)$ is the one whose integral significand is even.
In the following...

- a few “basic building blocks” of numerical computing: $ab \pm cd$, complex arithmetic, $x^n$;
- “usual” error bounds:
  - prove them;
  - try to improve them;
  - discuss their possible optimality or near-optimality.
- we assume than an FMA instruction is available: computes $\text{RN}(ab + c)$.

(FMA: first appeared in IBMP RS/6000, then PowerPC and Itanium, now specified by IEEE 754-2008)
Relative error due to roundings, $u$, and ulp notations

Let $t \in \mathbb{R}$, $2^e \leq t < 2^{e+1}$, with $e \geq e_{\text{min}}$;

- we have $2^e \leq \text{RN}(t) \leq 2^{e+1}$, and

\[
|\text{RN}(t) - t| \leq 2^{e-p}. \tag{1}
\]

→ upper bound on the relative error due to rounding $t$:

\[
\left| \frac{\text{RN}(t) - t}{t} \right| \leq u = 2^{-p}. \tag{2}
\]

- $u = 2^{-p}$: rounding unit.
- $\text{ulp}(t) = 2^{e-p+1}$. 
In precision-$p$ binary FP arithmetic, in the normal range, the relative error due to rounding to nearest is bounded by $u = 2^{-p}$. 

Always tight depends on $e$. 

Figure 1: In precision-$p$ binary FP arithmetic, in the normal range, the relative error due to rounding to nearest is bounded by $u = 2^{-p}$. 

$t = RN(t) \leq 2^{e-p} \leq u \cdot t$. 

|$t - \hat{t}| \leq 2^{e-p}$. 

$t = RN(t)$ tight only slightly above $2^e$. 

Relative error due to roundings, $u$, and ulp notations.
A small improvement

The bound on the relative error due to rounding can be slightly improved (using a remark by Jeannerod and Rump):

if $2^e \leq t < 2^{e+1}$, then $|t - \text{RN}(t)| \leq 2^{e-p} = u \cdot 2^e$, and

- if $t \geq 2^e \cdot (1 + u)$, then $|t - \text{RN}(t)|/t \leq u/(1 + u)$;
- if $t = 2^e \cdot (1 + \tau \cdot u)$ with $\tau \in [0, 1)$, then $|t - \text{RN}(t)|/t = \tau \cdot u/(1 + \tau \cdot u) < u/(1 + u)$,

→ the maximum relative error due to rounding is bounded by $\frac{u}{1 + u}$.

attained → no further “general” improvement.
“Wobbling” relative error

For $t \neq 0$, define

$$
\bar{t} = \frac{t}{2^{|\log_2 |t||}}.
$$

We have,

**Lemma 1**

*Let $t \in \mathbb{R}$. If*

$$
2^e \leq w \cdot 2^e \leq |t| < 2^{e+1}, \quad e \in \mathbb{Z}
$$

(in other words, if $1 \leq w \leq |\bar{t}|$) *then*

$$
\left| \frac{\text{RN}(t) - t}{t} \right| \leq \frac{u}{w}.
$$
Figure 2: The bound on the relative error due to rounding to nearest can be reduced to \( u/(1 + u) \). Furthermore, if we know that \( w \leq \bar{t} = t/2^e \), then \( |\text{RN}(t) - t|/t \leq u/w \).
Figure 3: Relative error due to rounding, namely $|RN(t) - t|/t$, for $\frac{1}{5} \leq t \leq 8$, and $p = 4$. 
First example: \( ab + cd \) with an FMA

Assume an \texttt{fma} instruction is available. Kahan’s algorithm for \( x = ab + cd \):

\[
\begin{align*}
\hat{w} & \leftarrow \text{RN}(cd) \\
e & \leftarrow \text{RN}(\hat{w} - cd) \\
\hat{f} & \leftarrow \text{RN}(ab + \hat{w}) \\
\hat{x} & \leftarrow \text{RN}(\hat{f} - e) \\
\text{Return } \hat{x}
\end{align*}
\]

- using std model (Higham, 2002):
  \[
  |\hat{x} - x| \leq J|x|
  \]
  with \( J = 2u + u^2 + (u + u^2)u\frac{|cd|}{|x|} \rightarrow \text{high accuracy as long as } u|cd| \gg |x| \)
- using properties of RN (Jeannerod, Louvet, M., 2011)
  \[
  |\hat{x} - x| \leq 2u|x|
  \]
  asymptotically optimal error bound.
- Complex multiplication \\& division.
A somewhat simpler algorithm for \( ab + cd \)

Cornea, Harrison and Tang (2002) approximate

\[
    r = ab + cd
\]

by \( \hat{r} \) obtained as follows

```plaintext
algorithm CHT(a, b, c, d)
    \( \hat{w}_1 := \text{RN}(ab) \); \( \hat{w}_2 := \text{RN}(cd) \);
    \( e_1 := \text{RN}(ab - \hat{w}_1) \); \( e_2 := \text{RN}(cd - \hat{w}_2) \); // exact operations
    \( \hat{f} := \text{RN}(\hat{w}_1 + \hat{w}_2) \);
    \( \hat{e} := \text{RN}(e_1 + e_2) \);
    \( \hat{r} := \text{RN}(\hat{f} + \hat{e}) \);
return \( \hat{r} \);
```

They show that the error is \( \mathcal{O}(u) \). Since the \( 2u \) relative error bound of Kahan’s algorithm was not known at that time, the CHT algorithm was favored.
A somewhat simpler algorithm for $ab + cd$

We have shown the following result (ACM TOMS, to appear).

**Theorem 2**

*Provided no underflow/overflow occurs, and assuming radix-2, precision-$p$ floating-point arithmetic, the relative error of Cornea et al’s algorithm is bounded by $2u + 7u^2 + 6u^3$.***

- improvement compared to the previous $O(u)$.
- however, does not help to choose between Kahan and CHT.
An almost-worst-case example...

Consider

\[
\begin{align*}
  a &= 2^p - 1, \\
  b &= 2^{p-3} + \frac{1}{2}, \\
  c &= 2^p - 1, \\
  d &= 2^{p-3} + \frac{1}{4},
\end{align*}
\]

One easily checks that \( a, b, c, \) and \( d \) are precision-\( p \) FP numbers. One easily finds:

\[
\begin{align*}
  ab + cd &= 2^{2p-2} + 2^{p-1} - \frac{3}{4}, \\
  \pi_1 &= 2^{2p-3} + 2^{p-2}, \\
  e_1 &= 2^{p-3} - \frac{1}{2}, \\
  \pi_2 &= 2^{2p-3}, \\
  e_2 &= 2^{p-3} - \frac{1}{4}, \\
  \pi &= 2^{2p-2}, \\
  e &= 2^{p-2} - \frac{3}{4}, \\
  s &= 2^{2p-2}.
\end{align*}
\]
An almost-worst-case “generic” example...

The relative error \(|s - (ab + cd)|/|ab + cd|\) is equal to

\[
\frac{2^{p-1} - \frac{3}{4}}{2^{2p-2} + 2^{p-1} - \frac{3}{4}} = \frac{2u - 3u^2}{1 + 2u - 3u^2} = 2u - 7u^2 + 20u^3 + \cdots
\]

This shows that our relative error bound

\[
2u + 7u^2 + 6u^3
\]

is asymptotically optimal (as \(u \to 0\) or, equivalently, as \(p \to \infty\)).

So that Kahan’s algorithm is to be preferred, unless one wishes to get the same result when computing \(ab + cd\) and \(cd + ab\) (e.g., to get a commutative complex \(\times\)).
The really difficult part... 

Is not the theorem that gives the upper bound. It is to find the “generic” (i.e., valid $\forall p$) example.

- perform the algorithm for zillions of different input values, for a given $p$, find the largest obtained relative errors,
- try to hint patterns,
- try to show that the chosen patterns effectively lead to an error close to (or, better, asymptotically equal to, or, even better, equal to) the bound.

Painful, error-prone $\rightarrow$ we are trying to (partly) automatize that step, using a “symbolic floating point” arithmetic written in Maple.
Complex multiplication and division

Given $x = a + ib$ and $y = c + id$, their product $z = xy$ can be expressed as

$$z = ac - bd + i(ad + bc);$$

and their quotient $x/y$ can be expressed as

$$q = \frac{ac + bd}{c^2 + d^2} + i \frac{bc - ad}{c^2 + d^2}.$$

In floating-point arithmetic, several issues:

- tradeoff accuracy vs speed,
- spurious overflow/underflow (e.g., $c^2 + d^2$ overflows, whereas the real and imaginary parts of $q$ are representable);

Here: accuracy problems. Scaling techniques to avoid spurious overflow/underflow dealt with in separately.

Focus on very simple algorithms.
Componentwise and normwise relative errors

When \( \hat{z} \) approximates \( z \):

- **componentwise error:**
  
  \[
  \max \left\{ \left| \frac{\Re(z) - \Re(\hat{z})}{\Re(z)} \right|; \left| \frac{\Im(z) - \Im(\hat{z})}{\Im(z)} \right| \right\};
  \]

- **normwise error:**
  
  \[
  \left| \frac{z - \hat{z}}{z} \right|.\]

Choosing between both kinds of error depends on the application.

- **componentwise error** \( \leq \epsilon \Rightarrow \) **normwise error** \( \leq \epsilon \);
- the converse is not true.
Naive multiplication algorithm without an FMA

\[ A_0 : (a + ib, c + id) \mapsto \text{RN} \left( \text{RN}(ac) - \text{RN}(bd) \right) + i \cdot \text{RN} \left( \text{RN}(ad) + \text{RN}(bc) \right) \]

- componentwise error: can be huge (yet finite);
- Normwise accuracy: studied by Brent, Percival, and Zimmermann (2007). The computed value has the form

\[ \hat{z}_0 = z(1 + \epsilon), \quad |\epsilon| < \sqrt{5} u, \]

→ the normwise relative error \( |\hat{z}_0/z - 1| \) is always \( \leq \sqrt{5} \cdot u \).

For any \( p \geq 2 \) they provide FP numbers \( a, b, c, d \) for which

\[ |\hat{z}_0/z - 1| = \sqrt{5} u - O(u^2) \rightarrow \text{the relative error bound } \sqrt{5} u \text{ is asymptotically optimal} \text{ as } u \rightarrow 0 \text{ (or, equivalently, as } p \rightarrow +\infty) \).

Can we do better if an FMA instruction is available?
Naive multiplication algorithm with an FMA

With an FMA, the simple way of evaluating \( ac - bd + i(ad + bc) \) becomes:

\[
\mathcal{A}_1 : (a + ib, c + id) \mapsto \text{RN} \left( ac - \text{RN}(bd) \right) + i \cdot \text{RN} \left( ad + \text{RN}(bc) \right)
\]

Algorithm \( \mathcal{A}_1 \) is just one of 4 variants that differ only in the choice of the products to which the FMA operations apply.

- componentwise error: can be huge (even infinite);
- normwise error:
  - for any of these 4 variants the computed complex product \( \hat{z}_1 \) satisfies
    \[
    |\hat{z}_1 - z| \leq 2u|z|
    \] (4)
  - we build inputs \( a, b, c, d \) for which \( |\hat{z}_1 / z - 1| = 2u - O(u^{1.5}) \) as \( u \to 0 \) ⇒ the error bound (4) is asymptotically optimal (given later on).

→ the FMA improves the situation from a normwise point of view.
Application of Kahan’s algorithm to the complex product

- $\mathbb{F}_p$: precision-$p$, radix-2 FP numbers with unlimited exponents;
- Evaluate separately the real and imaginary parts of $z = ac - bd + i(ad + bc)$ using Kahan’s algorithm;
- uses 8 floating-point operations;

$$A_2 : (a + ib, c + id) \mapsto \text{Kahan}(a, c, -b, d) + i \cdot \text{Kahan}(a, d, b, c)$$

- componentwise error $\leq 2u$ (asymptotically optimal);
- consequence: normwise error $\leq 2u$.

The normwise bound is asymptotically optimal.
Theorem 3

Let $a, b \in \mathbb{F}_p$ be given by

$$a = \text{pred}\left(\sqrt{2^{p-2}}\right), \quad b = 2^{p-1} + \left\lfloor \sqrt{2^{p-2}} \right\rfloor + 1,$$

where, for $t \in \mathbb{R}_{>0}$, $\text{pred}(t) = \max\{f \in \mathbb{F}_p : f < t\}$ denotes the predecessor of $t$ in $\mathbb{F}_p$. Let also $\hat{z}_1$ and $\hat{z}_2$ be the approximations to $z = (a + ib)^2$ computed by algorithms $A_1$ and $A_2$, respectively. If $p \geq 5$ then, barring underflow and overflow,

$$|\hat{z}_h/z - 1| > 2u - 8u^{1.5} - 4u^2, \quad h \in \{1, 2\}.$$
Iterated products and powers

Floating-point multiplication $a \times b$:
- exact result $z = ab$;
- computed result $\hat{z} = \text{RN}(z)$;

\[
(1 - u) \cdot z \leq \hat{z} \leq (1 + u) \cdot z; \tag{5}
\]

→ when we approximate $\pi_n = a_1 \cdot a_2 \cdots \cdots a_n$ by

$$
\hat{\pi}_n = \text{RN}(\cdots \text{RN}(\text{RN}(a_1 \cdot a_2) \cdot a_3) \cdots) \cdot a_n),
$$

we have

Property 1

\[
(1 - u)^{n-1} \pi_n \leq \hat{\pi}_n \leq (1 + u)^{n-1} \pi_n. \tag{6}
\]
→ relative error on the product $a_1 \cdot a_2 \cdots \cdots \cdot a_n$ bounded by

$$\psi_{n-1} = (1 + u)^{n-1} - 1.$$ 

if we define (Higham)

$$\gamma_k = \frac{ku}{1 - ku},$$

then, as long as $ku < 1$ (holds in practical cases),

$$k \cdot u \leq \psi_k \leq \gamma_k.$$ 

→ classical relative error bound: $\gamma_{n-1}$.

For “reasonable” $n$, $\psi_{n-1}$ is very slightly better than $\gamma_{n-1}$, yet $\gamma_{n-1}$ is easier to manipulate;

note that in single and double precision we never observed a relative error $\geq (n - 1) \cdot u$. 
Special case: \( n \leq 4 \)

As we have seen before, the relative error bound \( u \) can be replaced by

\[
\frac{u}{1 + u}.
\]

→ we can replace

\[
(1 - u)^{n-1}\pi_n \leq \hat{\pi}_n \leq (1 + u)^{n-1}\pi_n
\]

by

\[
\left(1 - \frac{u}{1 + u}\right)^{n-1}\pi_n \leq \hat{\pi}_n \leq \left(1 + \frac{u}{1 + u}\right)^{n-1}\pi_n.
\]
Special case: $n \leq 4$

Property 2

*If* $1 \leq k \leq 3$ *then*

\[
\left(1 + \frac{u}{1+u}\right)^k < 1 + k \cdot u.
\]

- $k = 2$:

\[
\left(1 + \frac{u}{1+u}\right)^2 - (1 + 2u) = -\frac{u^2 \cdot (1 + 2u)}{(1 + u)^2} < 0;
\]

- $k = 3$:

\[
\left(1 + \frac{u}{1+u}\right)^3 - (1 + 3u) = -\frac{u^3 \cdot (2 + 3u)}{(1 + u)^3} < 0.
\]

$k = n - 1 \rightarrow$ for $n \leq 4$, the relative error of the iterative product of $n$ FP numbers is bounded by $(n - 1) \cdot u.$
The particular case of computing powers

- “General” case of an iterated product: no proof for \( n \geq 5 \) that \((n - 1) \cdot u\) is a valid bound;
- focus on \( x^n \), where \( x \in \mathbb{F}_p \) and \( n \in \mathbb{N} \);
- we assume the “naive” algorithm is used:

\[
y \leftarrow x \\
\text{for } k = 2 \text{ to } n \text{ do} \\
\quad y \leftarrow \text{RN}(x \cdot y) \\
\text{end for} \\
\text{return } y
\]

- notation: \( \hat{x}_j = \text{value of } y \text{ after the iteration corresponding to } k = j \) in the for loop.
Main result

We are going to show:

**Theorem 4**

Assume $p \geq 5$ (holds in all practical cases). If

$$n \leq \sqrt{2^{1/3} - 1} \cdot 2^{p/2},$$

then

$$|\hat{x}_n - x^n| \leq (n - 1) \cdot u \cdot x^n.$$

- we can assume $1 \leq x < 2$;
- two cases: $x$ close to 1, and $x$ far from 1.
Preliminary results

First,

\[(1 - u)^{n-1} \geq 1 - (n - 1) \cdot u\]

for all \(n \geq 2\) and \(u \in [0, 1]\).

→ the left-hand bound of

\[(1 - u)^{n-1} \pi_n \leq \hat{\pi}_n \leq (1 + u)^{n-1} \pi_n.\]

suffices to show that

\[1 - (n - 1) \cdot u \cdot x_n \leq \hat{x}_n\]

→ to establish the Theorem, we only need to focus on the right-hand bound.
For $t \neq 0$, define

$$\bar{t} = \frac{t}{2^{\lfloor \log_2 |t| \rfloor}}.$$

We have,

**Lemma 5**

Let $t \in \mathbb{R}$. If

$$2^e \leq w \cdot 2^e \leq |t| < 2^{e+1}, \ e \in \mathbb{Z}$$

(in other words, if $1 \leq w \leq |\bar{t}|$) then

$$\left| \frac{\text{RN}(t) - t}{t} \right| \leq \frac{u}{w}. \quad (8)$$
Local maximum error for $x^6$ as a function of $x$ ($p = 53$)

Figure 4: The input interval $[1, 2)$ is divided into 512 equal-sized subintervals. In each subinterval, we calculate $x^6$ for 5000 consecutive FP numbers $x$, compute the relative error, and plot the largest attained error.
Main idea behind the proof

At least once in the execution of the algorithm, $\overline{x \cdot y}$ is far enough from 1 to sufficiently reduce the error bound on the multiplication $y \leftarrow \text{RN}(x \cdot y)$, so that the overall error bound becomes $\leq (n - 1) \cdot u$.

$$y \leftarrow x$$
$$\text{for } k = 2 \text{ to } n \text{ do}$$
$$\quad y \leftarrow \text{RN}(x \cdot y)$$
$$\text{end for}$$
$$\text{return } y$$

$$\psi_{n-1} = (1 + u)^{n-1} - 1 = (n - 1) u + \left(1/2 n^2 - 3/2 n + 1\right) u^2 + \cdots$$

$\rightarrow$ we have to save $\approx \frac{n^2}{2} u^2$, which requires one of the values $\overline{x \cdot y}$ to be larger than $\approx 1 + \frac{n^2}{2} u$. 
What we are going to show

Unless $x$ is very near 1, at least once $x \cdot y \geq 1 + n^2 u$, so that in (6) the term $(1 + u)^{n-1}$ can be replaced by

$$(1 + u)^{n-2} \cdot \left(1 + \frac{u}{1 + n^2 u}\right).$$

$\rightarrow$ we need to bound this last quantity. We have,

Lemma 6

If $0 \leq u \leq 2/(3n^2)$ and $n \geq 3$ then

$$(1 + u)^{n-2} \cdot \left(1 + \frac{u}{1 + n^2 u}\right) \leq 1 + (n - 1) \cdot u. \quad (9)$$

Proof: tedious...
Two remarks

Remark 1

Assume $n \leq \sqrt{2/3} \cdot 2^{p/2}$. If $\exists k \leq n$ s.t. $\text{RN}(x \cdot \hat{x}_{k-1}) \leq x \cdot \hat{x}_{k-1}$ (i.e., if in the algorithm at least one rounding is done downwards), then

$$\hat{x}_n \leq (1 + (n-1) \cdot u)x^n.$$

Proof.

We have

$$\hat{x}_n \leq (1 + u)^{n-2}x^n.$$

Lemma 6 implies $(1 + u)^{n-2} < 1 + (n - 1) \cdot u$. Therefore,

$$\hat{x}_n \leq (1 + (n - 1) \cdot u)x^n.$$
Two remarks

Remark 2

Assume \( n \leq \sqrt{2/3} \cdot 2^{p/2} \). If \( \exists k \leq n - 1 \), s.t. \( x \cdot \hat{x}_k \geq 1 + n^2 \cdot u \), then

\[
\hat{x}_n \leq (1 + (n - 1) \cdot u)x^n.
\]

Proof.

By combining Lemmas 5 and 6, if there exists \( k \), \( 1 \leq k \leq n - 1 \), such that

\[
x \cdot \hat{x}_k \geq 1 + n^2 \cdot u
\]

then

\[
\hat{x}_n \leq (1 + u)^{n-2} \cdot \left(1 + \frac{u}{1 + n^2 \cdot u}\right) \cdot x^n \leq (1 + (n - 1) \cdot u) \cdot x^n.
\]
Proof of Theorem 4

We assume \( n \geq 5 \). Proof articulated as follows

- if \( x \) is close enough to 1, then when computing RN\((x^2)\), the rounding is done downwards;
- in the other cases, \( \exists k \leq n - 1 \) such that \( x \cdot \hat{x}_k \geq 1 + n^2 \cdot u \).

Lemma 7

If \( 1 < x < 1 + 2^{p/2} \cdot u \), then \( \hat{x}_2 = \text{RN}(x^2) < x^2 \).

Proof.

\[
x < 1 + 2^{p/2} \cdot u \Rightarrow x = 1 + k \cdot 2^{-p+1} = 1 + 2ku, \quad \text{with } k < 2^{p/2-1}.
\]

We have \( x^2 = 1 + 2k \cdot 2^{-p+1} + k^2 \cdot 2^{-2p+2} \), which gives

\[
\text{RN}(x^2) = 1 + 2k \cdot 2^{-p+1} < x^2.
\]

In the following, we assume that no rounding is done downwards, which implies \( x \geq 1 + 2^{p/2} \cdot u \).
Proof of Theorem 4: case \( x^2 \leq 1 + n^2 u \)

- \( x \geq 1 + 2^{p/2}u > 1 + nu \Rightarrow x^n > (1 + nu)^n > 1 + n^2 u; \)
- no downward rounding \( \Rightarrow \hat{x}_{n-1} \cdot x > (1 + n^2 u). \)

Therefore

- if \( \hat{x}_{n-1}x < 2, \) then \( \hat{x}_{n-1}x \geq (1 + n^2 u), \) so that, from Remark 2, \( x^n \leq (1 + (n - 1) \cdot u) \cdot x^n; \)
- if \( \hat{x}_{n-1}x \geq 2, \) let \( k \) be the smallest integer such that \( \hat{x}_{k-1}x \geq 2. \)

\( x^2 \leq 1 + n^2 u \Rightarrow k \geq 3. \) We have

\[
\hat{x}_{k-1} \geq \frac{2}{x} \geq \frac{2}{\sqrt{1 + n^2 u}},
\]

hence

\[
\hat{x}_{k-2} \cdot x \geq \frac{2}{\sqrt{1 + n^2 u} \cdot (1 + u)}. \quad (10)
\]
\[
\hat{x}_{k-2} \cdot x \geq \frac{2}{\sqrt{1 + n^2 u \cdot (1 + u)}}.
\]

Define
\[
\alpha_p = \sqrt{\left(\frac{2^p + 1}{2^p + 1}\right)^{2/3}} - 1.
\]

For all \( p \geq 5 \), \( \alpha_p \geq \alpha_5 = 0.745 \cdots \), and \( \alpha_p \leq \sqrt{2^{2/3} - 1} = 0.766 \cdots \). If
\[
n \leq \alpha_p \cdot 2^{p/2},
\]
then
\[
\frac{2}{\sqrt{1 + n^2 u \cdot (1 + u)}} \geq 1 + n^2 u.
\]

\( \rightarrow \hat{x}_{k-2} \cdot x \geq 1 + n^2 u \). Also, \( \hat{x}_{k-2} \cdot x < 2 \) since \( k \) is the smallest integer such that \( \hat{x}_{k-1} x \geq 2 \). Therefore
\[
\hat{x}_{k-2} \cdot x \geq 1 + n^2 u.
\]

Which implies \( x^n \leq (1 + (n - 1) \cdot u) \cdot x^n \).
Proof of Theorem 4: case $x^2 > 1 + n^2u$

- if $x^2 < 2$ then $x^2 > 1 + n^2u \Rightarrow x^n \leq (1 + (n - 1) \cdot u)$;
- $x^2 = 2$ impossible ($x$ is rational);

$\Rightarrow$ we assume $x^2 > 2$ we also assume $x^2 < 2 + 2n^2u$ (otherwise, $x^2 \geq 1 + n^2u$). This gives

$$x^{n-1} < (2 + 2n^2u)^{\frac{n-1}{2}},$$

therefore, using the classical bound (Property 1),

$$\hat{x}_{n-1} < (2 + 2n^2u)^{\frac{n-1}{2}} \cdot (1 + u)^{n-2},$$

which implies

$$x \cdot \hat{x}_{n-1} < (2 + 2n^2u)^{\frac{n}{2}} \cdot (1 + u)^{n-2}. \quad (12)$$
Reminder:

\[ x \cdot \hat{x}_{n-1} < (2 + 2n^2 u)^{n/2} \cdot (1 + u)^{n-2} \text{ and } n \geq 5 \]

Define

\[ \beta = \sqrt{2^{1/3} - 1}. \]

If \( n \leq \beta \cdot 2^{p/2} \) then \( 2 + 2n^2 u \leq 2^{4/3} \), so that

\[ (2 + 2n^2 u)^{n/2} \cdot (1 + u)^{n-2} \leq 2^{2n/3} \cdot (1 + u)^{n-2}. \tag{13} \]

The function

\[ g(t) = 2^{t-1} - 2^{2t/3} \left( 1 + \frac{1}{2p} \right)^{t-2} = 2^{2t/3} \left[ 2^{t/3-1} - \left( 1 + \frac{1}{2p} \right)^{t-2} \right]. \]

is continuous, goes to \(+\infty\) as \( t \to +\infty \), has one root only:

\[ \frac{\log(2) + 2 \log \left( 1 + \frac{1}{2p} \right)}{\frac{1}{3} \log(2) - \log \left( 1 + \frac{1}{2p} \right)}, \]

which is < 4 as soon as \( p \geq 5 \Rightarrow \text{if } p \geq 5 \text{ then } x \cdot \hat{x}_{n-1} < 2^{n-1}. \]
Reminder: if $p \geq 5$ then $x \cdot \hat{x}_{n-1} < 2^{n-1}$.

- define $k$ as the smallest integer for which $x \cdot \hat{x}_{k-1} < 2^{k-1}$,
- $3 \leq k \leq n$ (we have assumed $x^2 > 2$),
- $x \cdot \hat{x}_{k-2} \geq 2^{k-2} \Rightarrow \hat{x}_{k-1} = \text{RN}(x \cdot \hat{x}_{k-2}) \geq 2^{k-2}$.

Therefore, $\hat{x}_{k-1}$ and $x \cdot \hat{x}_{k-1}$ belong to the same binade, therefore,

$$x \cdot \hat{x}_{k-1} \geq x > \sqrt{2}. \quad (14)$$

The constraint $n \leq \beta \cdot 2^{p/2}$ implies

$$1 + n^2 u \leq 1 + \beta^2 = 2^{1/3} < \sqrt{2}. \quad (15)$$

By combining (14) and (15) we obtain

$$x \cdot \hat{x}_{k-1} \geq 1 + n^2 u.$$

Therefore, using Remark 2, we deduce that $\hat{x}_n \leq (1 + (n - 1) \cdot u) \cdot x^n$. 
Final steps

\( \forall p \geq 5, \alpha_p \geq \beta \rightarrow \) combining the conditions found in the cases 
\( x^2 \leq 1 + n^2 u \) and \( x^2 > 1 + n^2 u \), we deduce

If \( p \geq 5 \) and \( n \leq \beta \cdot 2^{p/2} \), then for all \( x \),

\[
(1 - (n - 1) \cdot u) \cdot x^n \leq \hat{x}_n \leq (1 + (n - 1) \cdot u) \cdot x^n.
\]

where \( \beta = \sqrt{2^{1/3} - 1} = 0.5098245285339 \cdots \)

Q.E.D.

Questions:

○ is the restriction \( n \leq \beta \cdot 2^{p/2} \) problematic?

○ is the bound sharp?
On the restriction $n \leq \beta \cdot 2^{p/2}$

<table>
<thead>
<tr>
<th>format</th>
<th>$p$</th>
<th>$n_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>binary32/single</td>
<td>24</td>
<td>2088</td>
</tr>
<tr>
<td>binary64/double</td>
<td>53</td>
<td>48385542</td>
</tr>
<tr>
<td>binary128/quad</td>
<td>113</td>
<td>51953580258461959</td>
</tr>
</tbody>
</table>

With the first $n$ larger than the bound, $x^n$ under- or overflows, unless

- in single precision, $0.95905406 \leq x \leq 1.0433863$,
- in double precision, $0.999985359 \leq x \leq 1.000014669422$,

and nobody will use the “naive” algorithm for a huge $n$. 
On the restriction $n \leq \beta \cdot 2^{p/2}$

Furthermore, that restriction is not just a “proof artefact”. For very big $n$, the bound does not hold:

If $p = 10$ and $x = 891$, when computing $x^{2474}$, relative error $2473.299u$.

Notice that:

- for $p = 10$, $n_{\text{max}} = \beta \cdot 2^{p/2} = 16.31$;
- $2474$ is the smallest exponent for which the bound does not hold when $p = 10$. 
Tightness of the bound \((n - 1) \cdot u\)

Small \(p\) and not-too-large \(n\): an exhaustive test is possible.

**Table 1:** Actual maximum relative error assuming \(p = 8\), compared with \(\gamma_{n-1}\) and our bound \((n - 1)u\).

<table>
<thead>
<tr>
<th>(n)</th>
<th>actual maximum</th>
<th>(\gamma_{n-1})</th>
<th>our bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.73903u</td>
<td>3.0355u</td>
<td>3u</td>
</tr>
<tr>
<td>5</td>
<td>2.21152u</td>
<td>4.06349u</td>
<td>4u</td>
</tr>
<tr>
<td>6</td>
<td>2.53023u</td>
<td>5.099601u</td>
<td>5u</td>
</tr>
<tr>
<td>7</td>
<td>2.69634u</td>
<td>6.1440u</td>
<td>6u</td>
</tr>
<tr>
<td>8 = n_{max}</td>
<td>3.42929u</td>
<td>7.1967u</td>
<td>7u</td>
</tr>
</tbody>
</table>

→ our bound seems to be quite poor... however...
Tightness of the bound \((n - 1) \cdot u\)

For larger values of \(p\):

- **single precision** \((p = 24)\), exhaustive search still possible, largest error 4.328005619\(u\) for \(n = 6\), and 7.059603149\(u\) for \(n = 10\);
- **double precision** \((p = 53)\), we have a case with error 4.7805779\(u\) for \(n = 6\) and 7.8618\(\cdots u\) for \(n = 10\);
- **quad precision** \((p = 113)\), case with error 4.8827888185\(u\) for \(n = 6\);

→ we seem to get close to \((n - 1) \cdot u\) for large \(p\).
Building “bad cases” for the iterated product

Still in precision-$p$ binary FP arithmetic, we approximate

$$a_1 \cdot a_2 \cdots \cdots \cdot a_n,$$

by

$$\text{RN}(\cdots \text{RN}(\text{RN}(a_1 \cdot a_2) \cdot a_3) \cdots) \cdot a_n)$$

- $\pi_k = a_1 \cdots a_k$,
- $\hat{\pi}_k = \text{computed value}$,
- relative error $|\pi_n - \hat{\pi}_n|/\pi_n$ upper-bounded by $\gamma_{n-1}$,
- conjecture: if $n$ is “not too large” it is bounded by $(n - 1)u$.

Let us now show how to build $a_1, a_2, \ldots, a_n$ so that the relative error becomes extremely close to $(n - 1) \cdot u$. 
Building “bad cases” for the iterated product

- define \( a_1 = 1 + k_1 \cdot 2^{-p+1} \), and \( a_2 = 1 + k_2 \cdot 2^{-p+1} \). We have
  \[
  \pi_2 = a_1 a_2 = 1 + (k_1 + k_2) \cdot 2^{-p+1} + k_1 k_2 \cdot 2^{-2p+2}.
  \]

- If \( k_1 \) and \( k_2 \) are not too large, \( 1 + (k_1 + k_2) \cdot 2^{-p+1} \) is a FP number → we wish \( k_1 + k_2 \) to be as small as possible, while \( k_1 k_2 \cdot 2^{-2p+2} \) is as close as possible (yet less than) to \( 2^{-p} \). Hence a natural choice is
  \[
  k_1 = k_2 = \left\lceil 2^{\frac{p}{2} - 1} \right\rceil,
  \]
  which gives \( \hat{\pi}_2 < \pi_2 \).
- Now, if at step \( i - 1 \) we have
  \[
  \hat{\pi}_i = 1 + g_i \cdot 2^{-p+1}, \text{ with } \hat{\pi}_i < \pi_i,
  \]
  we choose \( a_{i+1} \) of the form \( 1 + k_{i+1} 2^{-p+1} \), with
  - \( k_{i+1} = \left\lceil \frac{2^{p-2}}{g_i} - 1 \right\rceil \) if \( g_i \leq 2^{\frac{p}{2} - 1} \);  
  - \( k_{i+1} = -\left\lceil \frac{2^{p-2}}{g_i} + 1 \right\rceil \) otherwise.
Building “bad cases” for the iterated product

Table 2: Relative errors achieved with the values $a_i$ generated by our method.

<table>
<thead>
<tr>
<th>$p$</th>
<th>$n$</th>
<th>relative error</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>10</td>
<td>8.99336984 $\cdots u$</td>
</tr>
<tr>
<td>24</td>
<td>100</td>
<td>98.9371972591 $\cdots u$</td>
</tr>
<tr>
<td>53</td>
<td>10</td>
<td>8.9999972447 $\cdots u$</td>
</tr>
<tr>
<td>53</td>
<td>100</td>
<td>98.9999970091 $\cdots u$</td>
</tr>
<tr>
<td>113</td>
<td>10</td>
<td>8.9999999999999973119 $\cdots u$</td>
</tr>
<tr>
<td>113</td>
<td>100</td>
<td>98.99999999999999701662 $\cdots u$</td>
</tr>
</tbody>
</table>
Conclusion on $x^n$

- error bound $(n - 1) \cdot u$ for computation of $x^n$ by the naive algorithm;
- valid for $n \leq \sqrt{2^{1/3} - 1} \cdot 2^{p/2} \rightarrow$ all practical cases;
- small improvement: the main interest lies in the simplicity of the bound;
- seems to be “asymptotically sharp” (as $p \rightarrow \infty$) but not sure;
- the bound $\gamma_{n-1}$ on iterated products is very sharp.

Thank you for your attention.