Numerical validation of compensated summation algorithms with stochastic arithmetic

Stef Graillat, Fabienne Jézéquel & Romain Picot

LIP6, Université Pierre et Marie Curie
Sorbonne Universités, Paris, France

1st Japanese-French workshop on numerical validation
LIP6, Paris, France
23-25 March 2015
Introduction

Development of computational resources
⇒ control of rounding error propagation
  • stochastic arithmetic
    • estimates rounding errors
    • uses a random rounding mode

If the accuracy is not satisfactory
  • arbitrary precision
  • compensated algorithms which rely on Error Free Transformations (EFT)

With the random rounding mode, EFT are no more exact.

Can we use stochastic arithmetic to validate compensated summations?
Principles of stochastic arithmetic

The CADNA library

Compensated summation with the random rounding mode

Numerical validation of compensated summation using CADNA

On-going work and conclusion
Round-off error analysis
Several approaches

- **Inverse analysis**
  based on the “Wilkinson principle”: the computed solution is assumed to be the exact solution of a nearby problem
  - provides error bounds for the computed results

- **Interval arithmetic**
  The result of an operation between two intervals contains all values that can be obtained by performing this operation on elements from each interval.
  - guaranteed bounds for each computed result
  - the error may be overestimated
  - specific algorithms

- **Probabilistic approach**
  - uses a random rounding mode
  - estimates the number of exact significant digits of any computed result
How to estimate the impact of round-off errors?

If the exact result $r$ of an arithmetic operation is not a floating-point number, it is approximated by a floating-point number $R^-$ or $R^+$. 

\[ r \]

\[ R^- \quad R^+ \]

The random rounding mode

Approximation of $r$ by $R^-$ or $R^+$ with the probability 1/2

The CESTAC method [Vignes & La Porte, 1974]

The same code is run several times with the random rounding mode. Then different results are obtained.

Briefly, the part that is common to all the different results is assumed to be reliable and the part that is different in the results is affected by round-off errors.
The implementation of the CESTAC method in a code providing a result $R$ consists in:

- performing $N$ times this code with the random rounding mode to obtain $N$ samples $R_i$ of $R$,
- choosing as the computed result the mean value $\bar{R}$ of $R_i$, $i = 1, \ldots, N$,
- estimating the number of exact significant decimal digits of $\bar{R}$ with

$$C_R = \log_{10} \left( \frac{\sqrt{N} \mid \bar{R} \mid}{\sigma \tau_\beta} \right)$$

where

$$\bar{R} = \frac{1}{N} \sum_{i=1}^{N} R_i \quad \text{and} \quad \sigma^2 = \frac{1}{N-1} \sum_{i=1}^{N} (R_i - \bar{R})^2.$$ 

$\tau_\beta$ is the value of Student’s distribution for $N - 1$ degrees of freedom and a probability level $1 - \beta$.

In practice, $N = 3$ and $1 - \beta = 95\%$. 
Self-validation of the CESTAC method

The CESTAC method is based on a 1st order model.

- A multiplication of two insignificant results
- or a division by an insignificant result

may invalidate the 1st order approximation.

Therefore the CESTAC method requires a dynamical control of multiplications and divisions, during the execution of the code.
**The concept of computed zero**

**Definition** [Vignes, 1986]

Using the CESTAC method, a result $R$ is a **computed zero**, denoted by @.0, if

$$\forall i, R_i = 0 \text{ or } C_R \leq 0.$$ 

$R$ is a computed result which, because of round-off errors, cannot be distinguished from 0.
The stochastic definitions

[Vignes, 1993]

Definition
Let $X$ and $Y$ be two results computed using the CESTAC method ($N$-sample), $X$ is stochastically equal to $Y$, noted $X \simeq Y$, if and only if

$$X - Y = @.0.$$ 

Definition
Let $X$ and $Y$ be two results computed using the CESTAC method ($N$-sample).

- $X$ is stochastically strictly greater than $Y$, noted $X \succ Y$, if and only if

$$\overline{X} > \overline{Y} \quad \text{and} \quad X \not\simeq Y$$

- $X$ is stochastically greater than or equal to $Y$, noted $X \succeq Y$, if and only if

$$\overline{X} \geq \overline{Y} \quad \text{or} \quad X \simeq Y$$

Discrete Stochastic Arithmetic (DSA) is defined as the joint use of the CESTAC method, the computed zero and the stochastic relation definitions.
The CADNA library implements Discrete Stochastic Arithmetic. CADNA allows to estimate round-off error propagation in any scientific program written in Fortran or in C++. More precisely, CADNA enables one to:

- estimate the numerical quality of any result
- control branching statements
- perform a dynamic numerical debugging
- take into account uncertainty on data.
The CADNA library implements Discrete Stochastic Arithmetic.

CADNA allows to estimate round-off error propagation in any scientific program written in Fortran or in C++.

More precisely, CADNA enables one to:
- estimate the numerical quality of any result
- control branching statements
- perform a dynamic numerical debugging
- take into account uncertainty on data.

CADNA provides new numerical types, the stochastic types, which consist of:
- 3 floating point variables
- an integer variable to store the accuracy.

All operators and mathematical functions are redefined for these types.
⇒ CADNA requires only a few modifications in user programs.
Compensated summation with stochastic arithmetic
Accurate sum of two numbers

Algorithms that compute with rounding to nearest the sum and the associated rounding error:

- **TwoSum** [Knuth, 1998] & [Møller, 1965]
  6 floating-point operations

- **FastTwoSum** [Dekker, 1971]
  3 floating-point operations and a comparison

The function:

\[
\begin{align*}
\text{s} & \leftarrow a + b \\
\text{z} & \leftarrow s - a \\
\text{t} & \leftarrow b - z
\end{align*}
\]

- \(s\) is the floating-point number that is closest to \(a + b\)
- \(t\) is the error on the floating-point addition of \(a\) and \(b\):
  \(s + t = a + b\) exactly.

With another rounding mode this error may not be exactly representable.
Accurate sum of two numbers

Algorithms that compute with rounding to nearest the sum and the associated rounding error:

- **TwoSum** [Knuth, 1998] & [Møller, 1965]
  - 6 floating-point operations

- **FastTwoSum** [Dekker, 1971]
  - 3 floating-point operations and a comparison

\[
\begin{align*}
\text{function } & [s, t] = \text{FastTwoSum}(a, b) \\
1: & \text{if } |b| \geq |a| \\
2: & \text{exchange } a \text{ and } b \\
3: & \text{end if} \\
4: & s \leftarrow a + b \\
5: & z \leftarrow s - a \\
6: & t \leftarrow b - z
\end{align*}
\]

`s` is the floating-point number that is closest to \(a + b\) exactly.

With another rounding mode this error may not be exactly representable.
Accurate sum of two numbers

FastTwoSum with rounding to nearest:

function \([s, t] = \text{FastTwoSum}(a, b)\)

1: if \(|b| \geq |a|\) then
2: exchange \(a\) and \(b\)
3: end if
4: \(s \leftarrow a + b\)
5: \(z \leftarrow s - a\)
6: \(t \leftarrow b - z\)

- \(s\) is the floating-point number that is closest to \(a + b\)
- \(t\) is the error on the floating-point addition of \(a\) and \(b\):
  \(s + t = a + b\) exactly.
  With another rounding mode this error may not be exactly representable.
function \([s, t] = \text{FastTwoSum}(a, b)\)

1: \textbf{if} \(|b| \geq |a|\) \textbf{then}
2: \hspace{1em} exchange \(a\) and \(b\)
3: \hspace{1em} \textbf{end if}
4: \hspace{1em} \(s \leftarrow a + b\)
5: \hspace{1em} \(z \leftarrow s - a\)
6: \hspace{1em} \(t \leftarrow b - z\)

Proposition

\(z\) is computed exactly: \(z = s - a\).

Let \(u\) be the relative rounding error.
For IEEE 754 double precision, \(u = 2^{-53}\) and for single precision \(u = 2^{-24}\).

Proposition

Let \(e\) be the error on \(s\): \(a + b = s + e\).

\(|e - t| \leq 2u|e|\).
Summation with faithful rounding

\begin{algorithm}
function \( \text{res} = \text{Sum}(p_1, \ldots, p_n) \)
1: \( s_1 \leftarrow p_1 \)
2: for \( i = 2 \) to \( n \) do
3: \( s_i \leftarrow s_{i-1} + p_i \)
4: end for
5: \( \text{res} \leftarrow s_n \)
\end{algorithm}

Let \( s = \sum_{i=1}^{n} p_i \) and \( S = \sum_{i=1}^{n} |p_i| \).

**Theorem** [Higham, 2002]

With directed rounding,

\[ |s - \text{res}| \leq \gamma_{n-1}(2u)|S| \quad \text{with} \quad \gamma_n(2u) = \frac{2nu}{1-2nu}. \]

When the condition number \( |s|/S \) is large (greater than \( 1/u \)), then the \texttt{Sum} function does not even return one correct digit.

\( \Rightarrow \) Compensated summation
Compensated summation

Compensated summation algorithms:

- [Kahan, 1965]
- [Priest, 1992]
- [Ogita, Rump & Oishi, 2005]
Compensated summation

Compensated summation algorithms:

- [Kahan, 1965]
- [Priest, 1992]
- [Ogita, Rump & Oishi, 2005]
Compensated summation

**function** \( \text{res} = \text{FastCompSum}(p_1, \ldots, p_n) \) [Ogita, Rump & Oishi, 2005]

1: \( \pi_1 \leftarrow p_1 \)
2: \( \sigma_1 \leftarrow 0 \)
3: **for** \( i = 2 \) **to** \( n \) **do**
4: \( [\pi_i, q_i] \leftarrow \text{FastTwoSum}(\pi_{i-1}, p_i) \)
5: \( \sigma_i \leftarrow \sigma_{i-1} + q_i \)
6: **end for**
7: \( \text{res} \leftarrow \pi_n + \sigma_n \)
Compensated summation using \textit{FastCompSum}

Let \( s = \sum p_i \) and \( S = \sum |p_i| \).

**Proposition [Ogita, Rump & Oishi, 2005]**

With rounding to nearest, if \( nu < 1 \),

\[
|\text{res} - s| \leq u|s| + \gamma_{n-1}(u)S \quad \text{with} \quad \gamma_n(u) = \frac{nu}{1-nu}.
\]

What are the effects of the random rounding mode on \textit{FastCompSum}?
Let \( s = \sum p_i \) and \( S = \sum |p_i| \).

**Proposition** [Ogita, Rump & Oishi, 2005]

With rounding to nearest, if \( nu < 1 \),

\[
|res - s| \leq u|s| + \gamma_{n-1}^2(u)S \quad \text{with} \quad \gamma_n(u) = \frac{nu}{1 - nu}.
\]

What are the effects of the random rounding mode on \textbf{FastCompSum}?

**Proposition**

With directed rounding, if \( nu < \frac{1}{2} \),

\[
|res - s| \leq 2u|s| + 2(1 + 2u)\gamma_{n}^2(2u)S \quad \text{with} \quad \gamma_n(2u) = \frac{2nu}{1 - 2nu}.
\]

\( \Rightarrow \) The accuracy obtained with the random rounding mode is similar to the one obtained with rounding to nearest.
Accuracy estimated by CADNA
Sum of 200 randomly generated floating-point numbers in double precision

With the current precision, the **FastCompSum** computes results that could have been obtained with twice the working precision.
Numerical instabilities

Various types of instabilities are detected during the execution:

- using the \textit{Sum} algorithm,
  - cancellation

- using the \textit{FastCompSum} algorithm,
  - cancellation
  - unstable branching
  - non significant argument in the absolute value function.

Because no multiplication and no division is performed, no instability can invalidate the estimation of accuracy.
Execution times
Sum of 100 000 floating-point numbers in double precision

Times measured on an Intel Core 2 quad Q9550 CPU at 2.83 GHz:

<table>
<thead>
<tr>
<th>execution</th>
<th>instability detection</th>
<th>Sum execution time (s)</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE</td>
<td>-</td>
<td>3.25E-04</td>
<td>1</td>
</tr>
<tr>
<td>CADNA</td>
<td>all instabilities</td>
<td>1.40E-02</td>
<td>43.2</td>
</tr>
<tr>
<td></td>
<td>no instability</td>
<td>3.40E-03</td>
<td>10.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>execution</th>
<th>instability detection</th>
<th>FastCompSum execution time (s)</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE</td>
<td>-</td>
<td>2.00E-03</td>
<td>1</td>
</tr>
<tr>
<td>CADNA</td>
<td>all instabilities</td>
<td>6.11E-02</td>
<td>30.6</td>
</tr>
<tr>
<td></td>
<td>no instability</td>
<td>2.98E-02</td>
<td>14.9</td>
</tr>
</tbody>
</table>

Cost of FastCompSum over Sum :
- about 6 without CADNA
- from 4 to 9 with CADNA.
On going work: compensated sum using TwoSum
Sum of 200 randomly generated floating-point numbers in double precision

Accuracy estimated by CADNA

Compensated summation algorithms with stochastic arithmetic  March 2015
On going work: compensated dot product

Dot product of arrays of 200 randomly generated elements in double precision

CompDot2FMA (relying on TwoProdFMA and CompSum)

Accuracy estimated by CADNA

Compensated summation algorithms with stochastic arithmetic

March 2015 21
CompHornerFMA (relies on TwoProdFMA and TwoSum)

Accuracy estimated by CADNA

On going work: compensated Horner scheme
Computation in double precision of \((x - 1)^n\) for \(x\) close to 1

Compensated summation algorithms with stochastic arithmetic

March 2015
Conclusion

- Stochastic arithmetic can be used to validate compensated summations using \texttt{FastTwoSum}

- Experimental results satisfactory using stochastic arithmetic for
  - compensated summation using \texttt{TwoSum}
  - compensated dot product
  - compensated Horner scheme

Perspectives:

- error bound for \texttt{TwoSum} with directed rounding

With $1 - \beta = 95\%$ and $N = 3$,

- the probability of overestimating the number of exact significant digits of at least 1 is 0.054%.
- the probability of underestimating the number of exact significant digits of at least 1 is 29%.

By choosing a confidence interval at 95%, we prefer to guarantee a minimal number of exact significant digits with high probability (99.946%), even if we are often pessimistic by 1 digit.