Contrôle dynamique de méthodes d’approximation

Fabienne Jézéquel

Habilitation à Diriger des Recherches
Laboratoire d’Informatique de Paris 6
1. Stochastic approach of round-off errors
   - Principles of stochastic arithmetics
   - Implementation of Discrete Stochastic Arithmetic

2. Dynamical control of approximations methods
   - Using sequences with a linear convergence
   - Using combined sequences

3. Dynamical control of multiple integrals computation
   - Using the principle of “iterated integrals”
   - Using cubature methods

4. Applications
   - In crystallography
   - In the neutron star theory
Round-off error analysis - 1/2

Several approaches

- **Direct analysis**
  estimation or bound of the direct error
  running error analysis, J.H. Wilkinson 1971
  SCALP, Ph. François 1989

- **Inverse analysis**
  based on the “Wilkinson principle”: the computed solution is assumed to be the exact solution of a nearby problem
  LAPACK, E. Anderson et al. 1999
  PRECISE, F. Chaitin-Chatelin et al. 2000

- **Methods based on algorithmic differentiation**
  first order approximation of the global round-off error
  the CENA method, Ph. Langlois 2001
Several approaches

- **Interval arithmetic**
  - guaranted bounds for each computed result
  - XSC languages, U. Kulisch et al. 1990
  - INTLAB, S.M. Rump 1998
  - MPFI, N. Revol and F. Rouillier 2003

- **Probabilistic approach**
  - uses a random rounding mode
  - the CESTAC method, M. La Porte and J. Vignes 1974
  - Monte Carlo Arithmetic, D.S. Parker 1997
The CESTAC method

- each arithmetical operation is performed $N$ times using the random rounding mode
  $\Rightarrow$ for each arithmetical operation, $N$ results $R_i$ are computed.

- computed result: $\overline{R} = \frac{1}{N} \sum_{i=1}^{N} R_i$.

- the number $C_{\overline{R}}$ of exact significant digits is estimated by

$$C_{\overline{R}} = \log_{10} \left( \frac{\sqrt{N} |\overline{R}|}{s \tau_{\beta}} \right) \quad \text{with} \quad s^2 = \frac{1}{N-1} \sum_{i=1}^{N} (R_i - \overline{R})^2$$

$\tau_{\beta}$ being the value of the Student distribution for $N - 1$ degrees of freedom and a probability level $(1 - \beta)$.

In practice, $N = 2$ or $N = 3$ and $\beta = 0.05$. 
The concept of computational zero

J. Vignes, 1986

Definition

During the run of a code using the CESTAC method, a result $R$ is a computational zero, denoted by $\theta.0$, if

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

Using the CESTAC method, the results of each arithmetical operation can be considered as realizations of a Gaussian random variable.

**Definition**

A stochastic number $X$ is a Gaussian random variable denoted by $(m, \sigma^2)$, where $m$ is the mean value of $X$ and $\sigma$ its standard deviation.

The number of significant digits common to all the elements of the confidence interval of $m$ at $1 - \beta$ and to $m$ is lower bounded by

$$C_{\beta,X} = \log_{10} \left( \frac{|m|}{\lambda_\beta \sigma} \right)$$

with $\beta = 0.05$, $\lambda_\beta \approx 1.96$. 
**Definition**

We define the elementary operations on two stochastic numbers $X_1 = (m_1, \sigma_1^2)$ and $X_2 = (m_2, \sigma_2^2)$ by:

- $X_1 s + X_2 \overset{\text{def}}{=} \left( m_1 + m_2, \sigma_1^2 + \sigma_2^2 \right)$
- $X_1 s - X_2 \overset{\text{def}}{=} \left( m_1 - m_2, \sigma_1^2 + \sigma_2^2 \right)$
- $X_1 s \times X_2 \overset{\text{def}}{=} \left( m_1 \ast m_2, m_2^2 \sigma_1^2 + m_1^2 \sigma_2^2 \right)$
- $X_1 s / X_2 \overset{\text{def}}{=} \left( m_1 / m_2, \left( \frac{\sigma_1}{m_2} \right)^2 + \left( \frac{m_1 \sigma_2}{m_2^2} \right)^2 \right)$, with $m_2 \neq 0$.

They correspond to the first order terms in $\frac{\sigma}{m}$ of operations between two independent Gaussian random variables.
The concept of stochastic zero

**Definition**

A stochastic number \( X \) is a **stochastic zero**, denoted by \( 0 \), if

\[
X = (0, 0) \quad \text{or} \quad C_{\beta, X} \leq 0.
\]

In accordance with the concept of stochastic zero, a new equality concept and new order relations have been defined.

**Definition**

Let \( X_1 = (m_1, \sigma_1^2) \) and \( X_2 = (m_2, \sigma_2^2) \).

- **Stochastic equality**, denoted by \( s= \), is defined as:
  \[
  X_1 s= X_2 \quad \text{if and only if} \quad X_1 s_\neq X_2 = 0.
  \]

- **Stochastic inequalities**, denoted by \( s> \) and \( s\geq \), are defined as:
  \[
  X_1 s> X_2 \quad \text{if and only if} \quad m_1 > m_2 \text{ and } X_1 s_\neq X_2,
  \]
  \[
  X_1 s\geq X_2 \quad \text{if and only if} \quad m_1 \geq m_2 \text{ or } X_1 s= X_2.
  \]
With DSA, a real number becomes an $N$-dimensional set.

Any operation on these $N$-dimensional sets is performed element per element using the random rounding mode.

By identifying $C_{\beta,x}$ and $C_{R}$, an equality concept and order relations have been defined for DSA.
Theorem

Let $X_i$ be the approximation in stochastic arithmetic of a mathematical value $x_i$ such that its exact significant bits are those of $x_i$ up to $p_i$ ($i = 1, 2$).

Let $\circ$ be an arithmetical operator: $\circ \in \{+, -, \times, /\}$ and $s\circ$ the corresponding stochastic operator: $s\circ \in \{s+, s-, s\times, s/\}$.

Then the exact significant bits of $X_1 \circ X_2$ are those of the mathematical value $x_1 \circ x_2$, up to $\max(p_1, p_2)$.

- proved for stochastic operations
- used in practice for results obtained in DSA
DSA on scalar architectures

DSA is implemented in the CADNA library.

A stochastic variable is an N-dimensional set of real numbers. Practically, $N = 2$ (or 3).

In Fortran 90, each stochastic variable $A$ is represented by a structure consisting of 2 real variables: $A\%x$ and $A\%y$.

Each stochastic operation $A \Omega B$ is overloaded as:

$$(A\%x, A\%y) \Omega (B\%x, B\%y) = (A\%x \omega B\%x, A\%y \omega B\%y)$$

$\omega$: arithmetic operation rounded up or down with probability $\frac{1}{2}$. 
Two types of versions of CADNA on parallel architectures:

- Numerical validation of parallel codes using PVM or MPI
- Parallelization of DSA to improve the performances of CADNA for sequential codes

(HDR J.-L. Lamotte 2004)
The problems:

1. Vectorial processors do not always respect the IEEE standard.
   - NEC SX5 respects the IEEE standard,
   - but not CRAY SV1.

2. Classical operation overloading inhibits vectorization,

Without the help of a vector preprocessor, we can only implement new array operators.

Run time (in seconds) of a code performing LU decomposition of a matrix of dimension 1000, without pivoting:

<table>
<thead>
<tr>
<th></th>
<th>Classical code</th>
<th>With CADNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC (Pentium III-450)</td>
<td>73</td>
<td>199</td>
</tr>
<tr>
<td>CRAY SV1</td>
<td>7</td>
<td>395</td>
</tr>
</tbody>
</table>
static rounding modes:

- additions and subtractions are rounded to zero,
- multiplications and divisions are rounded to the nearest.

For a stochastic vector operation $C = A \Omega B$, the elements of $C$ must be randomly rounded up or down.

The last bit in the mantissa is changed using vector logical operations and an array of random 0 or 1.
Without pivoting, the dimension of the matrix is 1000.

Although the CADNA library is inlined, the code has also been written with manual inlining of the array operators.

<table>
<thead>
<tr>
<th></th>
<th>Run time</th>
<th>Perf.</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>classical vector processor</td>
<td>7.1 s</td>
<td>94 Mflops</td>
<td>–</td>
</tr>
<tr>
<td>CADNA, scalar version</td>
<td>395 s</td>
<td>1.7 Mstops</td>
<td>56</td>
</tr>
<tr>
<td>CADNA, vector version</td>
<td>49 s</td>
<td>14 Mstops</td>
<td>7</td>
</tr>
<tr>
<td>with manual inlining</td>
<td>27 s</td>
<td>24 Mstops</td>
<td>3.8</td>
</tr>
</tbody>
</table>

*Mstops means millions of stochastic operations per second.*
A real-life example: the ORCA code

- numerical simulation of all the ocean streams
- about 50,000 line codes in 112 files

7 subroutines have been rewritten with the array formulation so that the vector version of CADNA can be used.

For 10 iterations, they represent 65% of the global classical run and 57% of the stochastic run on CRAY SV1.

For 10 time iterations,

<table>
<thead>
<tr>
<th>in the 7 routines</th>
<th>Run time</th>
<th>Perf.</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>classical</td>
<td>32 s</td>
<td>210 Mflops</td>
<td>–</td>
</tr>
<tr>
<td>CADNA, scalar</td>
<td>2868 s</td>
<td>2.3 Mstops</td>
<td>90</td>
</tr>
<tr>
<td>CADNA, vector</td>
<td>568 s</td>
<td>11.8 Mstops</td>
<td>18</td>
</tr>
</tbody>
</table>
Array implementation on NEC SX5

NEC SX5 respects the **IEEE 754 standard**
⇒ the 4 rounding modes defined in this standard are available.

The Fortran compiler has been updated in order to **vectorize** some derived type statements.

**Memory optimization:**

From an 8-byte word to another, a step of length 1 is performed. An even step-length is very time consuming.

In order to perform steps of odd length, stochastic arrays consist of:

- 2 real arrays in single precision
- 3 real arrays in double precision.
Let \( C = A \times B \), where \( A, B \) and \( C \) are stochastic arrays of size \( n \).

To save a switch of the rounding mode, the following property is used.

\[
u \times^+ v = -((-u) \times^- v) \quad \text{and} \quad u \times^- v = -((-u) \times^+ v)\]

A random logical array \( L \) is used.

\[
\text{where}(L(1:n))
\begin{align*}
C\%x &= A\%x \times B\%x \\
C\%y &= -((-A\%y) \times B\%y)
\end{align*}
\]

\[
\text{elsewhere}
\begin{align*}
C\%y &= A\%y \times B\%y \\
C\%x &= -((-A\%x) \times B\%x)
\end{align*}
\]

\[
\text{endwhere}
\]

\[
C = \begin{pmatrix}
C(1)\%x & C(1)\%y \\
C(2)\%x & C(2)\%y \\
C(3)\%x & C(3)\%y \\
C(4)\%x & C(4)\%y \\
\vdots & \vdots \\
C(n)\%x & C(n)\%y
\end{pmatrix}
\]

A code performing 1 600 000 multiplications of single precision arrays of size 500 runs at 276 MFlops.
Implementation of stochastic operations
Example: the stochastic multiplication

Let $C = A \ast B$, where $A$, $B$ and $C$ are stochastic arrays of size $n$.

To save a switch of the rounding mode, the following property is used.

$$u \ast^+ v = -((-u) \ast^- v) \quad \text{and} \quad u \ast^- v = -((-u) \ast^+ v)$$

A random logical array $L$ is used.

$$\text{where}(L(1:n))$$

$$C\%x = A\%x \ast B\%x$$
$$C\%y = -((-A\%y) \ast B\%y)$$

elsewhere

$$C\%y = A\%y \ast B\%y$$
$$C\%x = -((-A\%x) \ast B\%x)$$

endwhere

$$C = \begin{pmatrix}
C(1)%x & C(1)%y \\
C(2)%x & C(2)%y \\
C(3)%x & C(3)%y \\
\vdots & \vdots \\
C(n)%x & C(n)%y
\end{pmatrix}$$

A code performing 1 600 000 multiplications of single precision arrays of size 500 runs at 276 MFlops.
Updates of the Fortran compiler on NEC SX5

Inlining of an operation involving
- 2 arrays: since 2001
- scalar and arrays: since 2003

With a code performing 1 600 000 multiplications of double precision arrays of size 500:

<table>
<thead>
<tr>
<th></th>
<th>Run time</th>
<th>Perf.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before update of 2001</td>
<td>13.0 s</td>
<td>62 Msteps</td>
</tr>
<tr>
<td>After update of 2001</td>
<td>3.5 s</td>
<td>229 Msteps</td>
</tr>
<tr>
<td>With global variable (instead of SIZE)</td>
<td>2.0 s</td>
<td>400 Msteps</td>
</tr>
<tr>
<td>With global variable and manual inlining</td>
<td>1.7 s</td>
<td>471 Msteps</td>
</tr>
</tbody>
</table>
LU decomposition on NEC SX5

Without pivoting, the dimension of the matrix is 1000:

<table>
<thead>
<tr>
<th></th>
<th>Run time</th>
<th>Perf.</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>classical vector processor</td>
<td>0.55 s</td>
<td>1.221 Gflops</td>
<td>–</td>
</tr>
<tr>
<td>CADNA, scalar version</td>
<td>1207 s</td>
<td>0.56 Mstops</td>
<td>2194</td>
</tr>
<tr>
<td>CADNA, vector version</td>
<td>13.5 s</td>
<td>50 Mstops</td>
<td>25</td>
</tr>
<tr>
<td>with manual inlining</td>
<td>1.3 s</td>
<td>501 Mstops</td>
<td>2.4</td>
</tr>
</tbody>
</table>

Still problems of inlining with instructions involving:

- > 2 variables
- 1D sub-arrays of 2D arrays

Numerical accuracy of approximation methods

When an approximation $L(h)$ such that $\lim_{h \to 0} L(h) = L$ is computed, it is affected by:

- a truncation error $e_m(h)$
- a round-off error $e_c(h)$.

If $h$ decreases, $L(h)$: 

```
<table>
<thead>
<tr>
<th>s</th>
<th>exponent</th>
<th>mantissa</th>
</tr>
</thead>
</table>
```

\[ e_m(h) \rightarrow e_c(h) \]

As long as $e_c(h) < e_m(h)$, decreasing $h$ brings reliable information to the mantissa.

The optimal step is reached when $e_c(h) \approx e_m(h)$.

1. How to determine dynamically the optimal step?
2. Which digits in the approximation obtained are in common with $L$?
Significant digits common to two real numbers

**Definition**

Let $a$ and $b$ be two real numbers, the number of significant digits that are common to $a$ and $b$ can be defined in $\mathbb{R}$ by

1. for $a \neq b$, $C_{a,b} = \log_{10} \left| \frac{a + b}{2(a - b)} \right|$, 
2. $\forall a \in \mathbb{R}, \ C_{a,a} = +\infty$.

Example:

if $a = 2.4599976$ and $b = 2.4600012$, then $C_{a,b} \approx 5.8$. 
Theorem

Let \((I_n)\) be a sequence converging linearly to \(I\), i.e. which satisfies
\[ I_n - I = K \alpha^n + o(\alpha^n) \text{ where } K \in \mathbb{R} \text{ and } 0 < |\alpha| < 1, \]
then
\[ C_{I_n, I_{n+1}} = C_{I_n, I} + \log_{10} \left( \frac{1}{1 - \alpha} \right) + o(1). \]

If the convergence zone is reached, the significant decimal digits common to \(I_n\) and \(I_{n+1}\), are those of \(I\), up to \(\log_{10} \left( \frac{1}{1 - \alpha} \right)\).

If \(-1 < \alpha \leq \frac{1}{2}\), then
\[ -1 < \log_2 \left( \frac{1}{1 - \alpha} \right) \leq 1. \]
In this case, the significant bits common to \(I_n\) and \(I_{n+1}\) are those of \(I\), up to one.
Let us assume that the convergence zone is reached.

If \( I_n - I_{n+1} = @.0 \), the difference between \( I_n \) and \( I_{n+1} \) is due to round-off errors. Further iterations are useless.

Consequently

- the optimal iterate \( I_{n+1} \) can be dynamically determined
- if \( \alpha \leq \frac{1}{2} \), the exact significant bits of \( I_{n+1} \) are those of \( I \), up to one.

Theorem

Let \( L(h) \) be an approximation of order \( p \) of \( L \), i.e.

\[
L(h) - L = Kh^p + \mathcal{O}(h^q) \quad \text{with} \quad 1 \leq p < q, \quad K \in \mathbb{R}.
\]

If \( L_n \) is the approximation computed with the step \( h_0/2^n \), then

\[
C_{L_n,L_{n+1}} = C_{L_n,L} + \log_{10} \left( \frac{2^p}{2^p - 1} \right) + \mathcal{O}\left(2^{n(p-q)}\right).
\]

If the convergence zone is reached and \( L_n - L_{n+1} = \mathbb{O}.0 \), the exact significant bits of \( L_{n+1} \) are those of \( L \), up to one.
Dynamical control of the trapezoidal rule and Simpson’s rule

Corollary

If \( I_n \) is the approximation of \( I = \int_a^b f(x) \, dx \) computed with step \( h = \frac{b-a}{2^n} \) using the trapezoidal rule or Simpson’s rule, then

\[
C_{I_n, I_{n+1}} = C_{I_n, I} + \log_{10}(\beta) + O\left(\frac{1}{4^n}\right)
\]

- trapezoidal rule: \( \beta = \frac{4}{3} \) \( (p = 2) \)
- Simpson’s rule: \( \beta = \frac{16}{15} \) \( (p = 4) \)

If the convergence zone is reached and \( I_n - I_{n+1} = @.0 \), the exact significant bits of \( I_{n+1} \) are those of \( I \), up to one.

The Gauss-Legendre method

The approximation of $\int_{-1}^{1} f(x) dx$ by the Gauss-Legendre method with $\nu$ points is

$$\sum_{i=1}^{\nu} C_i f(x_i)$$

where for $i = 1, \ldots, \nu$,

- $\{x_i\}$ are the roots of the $\nu$-degree Legendre polynomial $P_{\nu}$
- $C_i = \frac{2}{(1-x_i^2)(P_{\nu}'(x_i))^2}$.

For the computation of $I = \int_{a}^{b} g(t) dt$, a change of variable is required:

$$I = \frac{(b - a)}{2} \int_{-1}^{1} g \left( \frac{(b - a)x + (b + a)}{2} \right) dx.$$
Theorem

Let \( I = \int_a^b g(t) \, dt \).

If \([a, b]\) is partitioned into \(2^n\) subintervals of same length on which the Gauss-Legendre method with \(\nu\) points is applied and \(I_n\) is the sum of the \(2^n\) approximations obtained, then

\[
I_n - I = \frac{K_\nu}{4^n \nu} + O\left(\frac{1}{2^n(2\nu+1)}\right).
\]

Corollary

\[
C_{I_n, I_{n+1}} = C_{I_n, I} + \log_{10} \left(\frac{4^\nu}{4^\nu - 1}\right) + O\left(\frac{1}{2^n}\right).
\]

If the convergence zone is reached and \(I_n - I_{n+1} = 0\), the exact significant bits of \(I_{n+1}\) are those of \(I\), up to one.
Romberg’s method

The approximation of $I = \int_a^b f(x)dx$ with Romberg’s method, requires the following computations ($h = \frac{b-a}{M}$, $M \geq 1$):

\[
\begin{array}{cccccc}
T_1(h) & T_1\left(\frac{h}{2}\right) & \cdots & T_1\left(\frac{h}{2^{n-3}}\right) & T_1\left(\frac{h}{2^{n-2}}\right) & T_1\left(\frac{h}{2^{n-1}}\right) \\
T_2(h) & T_2\left(\frac{h}{2}\right) & \cdots & T_2\left(\frac{h}{2^{n-3}}\right) & T_2\left(\frac{h}{2^{n-2}}\right) \\
T_3(h) & T_3\left(\frac{h}{2}\right) & \cdots & T_3\left(\frac{h}{2^{n-3}}\right) \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
T_{n-1}(h) & T_{n-1}\left(\frac{h}{2}\right) \\
T_n(h)
\end{array}
\]

The first row is computed using the trapezoidal rule with step $\frac{h}{2^j}$. For $r = 2, \ldots, n$ and $j = 0, \ldots, n - r$,

\[
T_r\left(\frac{h}{2^j}\right) = \frac{1}{4^{r-1} - 1} \left(4^{r-1} T_{r-1}\left(\frac{h}{2^{j+1}}\right) - T_{r-1}\left(\frac{h}{2^j}\right)\right).
\]
Theorem

If $T_n(h)$ is the approximation of $I = \int_a^b f(x)dx$ computed with $n$ iterations of Romberg’s method using the initial step $h = \frac{b-a}{M}$, then

$$C_{T_n(h), T_{n+1}(h)} = C_{T_n(h), I} + O\left(\frac{1}{n^2}\right).$$

If the convergence zone is reached and $T_n(h) - T_{n+1}(h) = @.0$, the exact significant digits of $T_{n+1}(h)$ are those of $I$. 

Dynamical control of Romberg’s method
Let \((u_m)\) be a sequence converging linearly to \(u\).

For all \(m\), let \((u_m,n)\) be a sequence converging linearly to \(u_m\).

\((u_m,n)\) is computed until, in the convergence zone, the difference between two successive iterates is \(\@.0\).

Let \(U_m\) be the approximation of \(u_m\) obtained.

\(\Rightarrow\) the bits common to \(u_m\) and \(u_{m+1}\) are those of \(u\), up to \(p\).

\(\Rightarrow\) the exact significant bits of \(U_m\) are those of \(u_m\), up to \(q\).

\(\Rightarrow\) if \(U_m - U_{m+1} = \@.0\),

the exact significant bits of \(U_{m+1}\) are those of \(u\), up to \(p + q\).
Let $g = \int_{0}^{\infty} \phi(x) \, dx$ and $g_m = \sum_{j=0}^{m} f_j$ with $f_j = \int_{jL}^{(j+1)L} \phi(x) \, dx$.

We assume that $(g_m)$ converges linearly to $g$.

An approximation of each integral can be computed in DSA, such that its exact significant bits are those of $f_j$, up to 1.

Let $G_m$ be the approximation of $g_m$ computed in DSA.

- The exact significant bits of $G_m$ are those of $g_m$, up to 1.
- If the convergence zone is reached, the significant bits common to $g_m$ and $g_{m+1}$ are those of $g$, up to $p$.
- If $G_m - G_{m+1} = @.0$, the exact significant bits of $G_{m+1}$ are those of $g$, up to $p+1$. 
Dynamical control of multiple integrals computation

\[ I = \int_{\Omega} f(\mathbf{x}) d\mathbf{x} \text{ with } \Omega \subset \mathbb{R}^N \]

can be approximated by:

\[ Q[f] = \sum_{j=1}^{\nu} a_j f(\mathbf{x}_j) \text{ with } a_j \in \mathbb{R} \text{ and } \mathbf{x}_j \in \Omega. \]

The approximation \( Q \) is called \textbf{cubature formula} if \( N \geq 2 \).

- polynomial-based methods
- Monte Carlo methods

Cubpack, R. Cools et al. 1992
VANI, C.-Y. Chen 1998
CLAVIS, S. Wedner 2000
Approximation using the principle of “iterated integrals”

Computation of 2-dimensional integrals

\[
\begin{align*}
\mathbf{s} &= \int_a^b \int_{y_1(x)}^{y_2(x)} f(x, y) \, dx \, dy = \int_a^b g(x) \, dx \quad \text{with} \quad g(x) = \int_{y_1(x)}^{y_2(x)} f(x, y) \, dy.
\end{align*}
\]

\[\forall x \in [a, b], \] an approximation \( G(x) \) can be computed in DSA such that its exact significant bits are those of \( g(x) \), up to \( \delta \).

Let \( S_n = \phi(\{G(x_i)\}) \) be the approximation of \( s \) computed in DSA and \( s_n = \phi(\{g(x_i)\}) \).

\[\Rightarrow \] the exact significant bits of \( S_n \) are those of \( s_n \), up to \( \delta \)

\[\Rightarrow \] if the convergence zone is reached, the significant bits common to \( s_{n-1} \) and \( s_n \) are common with \( s \), up to \( \delta \)

\[\Rightarrow \] if \( S_{n-1} - S_n = @.0 \),
the exact significant bits of \( S_n \) are those of \( s \), up to \( 2\delta \).
The exact significant bits of the approximation obtained are those of the mathematical value of the integral, up to $N\delta$.

- With Romberg’s method, $\delta = 0$.
- With the trapezoidal rule, $N\delta$ represents:
  - one bit if $N \leq 2$
  - one decimal digit if $N \leq 8$.
- With Simpson’s rule, $N\delta$ represents one bit if $N \leq 35$.
- With the Gauss-Legendre method with 6 points, $N\delta$ represents one bit if $N \leq 2838$. 
Computation of an integral involved in crystallography

\[ g(a) = \int_0^{+\infty} f(x) \, dx, \]

with \( f(x) = [\exp(x) + \exp(-x)]^a - \exp(ax) - \exp(-ax) \) and \( 0 < a < 2 \).

\( g(5/3) \approx 4.45 \) (W. Harrison 1981)
\( g(5/3) \approx 4.6262911 \) (SIAM review 1996)

\( g(a) \) can be expressed as a series expansion:

\[ g(a) = \sum_{n=1}^{+\infty} \frac{\prod_{i=0}^{n-1} (a - i)}{(n!) (2n - a)} - \frac{1}{a}. \]

Several numerical problems may occur in the computation of $g(a)$:

- for high values of $x$, the computation of $f(x)$ may generate cancellations,
- the upper bound of the integral is infinite,
- the quadrature method used, e.g. Romberg’s method, generates both a truncation error and a round-off error.
In order to avoid cancellations, the same expression of the integrand is not used at both bounds of the interval.

\[ g(a) \approx \int_0^l f_1(x) \, dx + \sum_{j=1}^k \int_{j}^{(j+1)l} f_2(x) \, dx, \]

where

\[ f_1(x) = \exp(ax) \left[ (1 + \exp(-2x))^a - 1 - \exp(-2ax) \right] \]
\[ f_2(x) = \exp(ax) u(x) - \exp(-ax), \]

\[ u(x) = \lim_{n \to \infty} u_n(x) \text{ with } u_n(x) = \sum_{i=1}^{n-1} \frac{\exp(-2ix)}{i!} \prod_{j=0}^{i-1}(a - j). \]

Dynamical choice of several parameters:

- \( n \) such that \( u_n(x) \approx u(x) \)
- \( k \) such that \( \int_{j}^{(j+1)l} f_2(x) \, dx \approx \int_j^\infty f_2(x) \, dx \)
- the number of iterations with Romberg’s method
Proposition

One can compute an approximation $G(a)$ such that its exact significant digits are those of $g(a)$, up to $\delta = \log_{10}\left(\frac{2}{1 - \exp^{-/\min(a,2-a)}}\right)$.

<table>
<thead>
<tr>
<th>$a$</th>
<th>$\delta \approx$</th>
<th>$g(a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.34</td>
<td>exact: -1.694426169587958E+000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSA: -1.69442616958795E+000</td>
</tr>
<tr>
<td>5/3</td>
<td>0.39</td>
<td>exact: 4.6262911111983995E+000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSA: 4.62629111119839E+000</td>
</tr>
<tr>
<td>1.9999</td>
<td>3.6</td>
<td>exact: 1.999899986776092E+004</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSA: 1.99989997358E+004</td>
</tr>
</tbody>
</table>

The exact significant digits of $G(a)$ are in common with $g(a)$, up to $[\delta]$. 
Study of an integral involved in the neutron star theory

\[
\tau (\varepsilon, v) = \frac{1}{\omega(\varepsilon)} \int_{0}^{\frac{\pi}{2}} d\theta \sin (\theta) \int_{0}^{\infty} dn \left[ n^2 \int_{0}^{\infty} dp \ h(n, p, \theta, \varepsilon, v) \right]
\]

\((\varepsilon, v) \in [10^{-4}, 10^4] \times [10^{-4}, 10^3]\)

\(\omega\) is a normalization function

\[
h(n, p, \theta, \varepsilon, v) = \psi(z)\Gamma(n - \varepsilon - z) + \psi(-z)\Gamma(n - \varepsilon + z)
- \psi(z)\Gamma(n + \varepsilon - z) - \psi(z)\Gamma(n + \varepsilon + z)
\]

with \(z = \sqrt{p^2 + (v \sin(\theta))^2}\), \(\psi(x) = \frac{1}{\exp(x)+1}\), \(\Gamma(x) = \frac{x}{\exp(x)-1}\).

The numerical problems:

- **two infinite bounds**
  \[ \int_{0}^{\infty} \ldots \text{is replaced by} \sum_{j=0}^{k} \int_{jL}^{(j+1)L} \ldots \]
  \[ \Rightarrow \text{Dynamical choice of} \ k \]

- **\( \Gamma(x) = \frac{x}{\exp(x) - 1} \) generates cancellations if} x \approx 0.**
  A series expansion of \( \Gamma(x) \) is used:
  \[ \Gamma(x) \approx \frac{1}{1 + \frac{x}{2} + \ldots + \frac{x^{n-1}}{n!}} \]
  \[ \Rightarrow \text{Dynamical choice of} \ n \]

- **With the principle of “iterated integrals”, the Gauss-Legendre method is used and generates both a truncation error and a round-off error**
  \[ \Rightarrow \text{Dynamical control of the Gauss-Legendre method} \]
The numerical problems:

- two infinite bounds
  \[ \int_0^\infty \ldots \text{ is replaced by } \sum_{j=0}^k \int_{jL}^{(j+1)L} \ldots \]
  \[ \Rightarrow \text{Dynamical choice of } k \]

- \( \Gamma(x) = \frac{x}{\exp(x) - 1} \) generates cancellations if \( x \approx 0 \).
  a series expansion of \( \Gamma(x) \) is used:
  \[ \Gamma(x) \approx \frac{1}{1 + \frac{x}{2} + \ldots + \frac{x^{n-1}}{n!}} \]
  \[ \Rightarrow \text{Dynamical choice of } n \]

- With the principle of “iterated integrals”, the Gauss-Legendre method is used and generates both a truncation error and a round-off error
  \[ \Rightarrow \text{Dynamical control of the Gauss-Legendre method} \]
\( \tau (\varepsilon, \nu) \) has been computed using DSA in single precision for 5752 points \((\varepsilon, \nu)\) defined by:

\[
\begin{align*}
\varepsilon &= 10^a \quad \text{with} \quad a = -4.0, -3.9, -3.8, \ldots, 4.0 \\
\nu &= 10^b \quad \text{with} \quad b = -4.0, -3.9, -3.8, \ldots, 3.0.
\end{align*}
\]

The run time of the code varies from 45 s to 3347 s depending on the values of \(\varepsilon\) and \(\nu\), the average run time being 389 s.
Proposition

One can compute an approximation of $\tau(\varepsilon, \nu)$ such that its exact significant digits are those of $\tau(\varepsilon, \nu)$, up to 2.

<table>
<thead>
<tr>
<th>nb. of exact significant digits</th>
<th>occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>217</td>
</tr>
<tr>
<td>5</td>
<td>665</td>
</tr>
<tr>
<td>6</td>
<td>3347</td>
</tr>
<tr>
<td>7</td>
<td>1522</td>
</tr>
</tbody>
</table>

$\Rightarrow$ we can guarantee 1 to 5 significant digits in the results obtained.
Study of an integral involved in the neutron star theory
Numerical results
Dynamical control of converging sequences computation

Let \( u = \lim_{n \to \infty} u_n \). From two iterates in the convergence zone, one can determine the first digits of \( u \).

If \( u_n - u_{n+1} = 0.0 \), one can determine which exact significant digits of \( u_{n+1} \) are in common with \( u \).

Combination of theoretical results if several sequences are involved

For the approximation of an integral, one has to take into account:
- the dimension of the integral
- the number of improper bounds
- the possible approximation of the integrand by its series expansion
- the convergence speed of the sequences involved
Conclusion and perspectives - 2/2

- Adaptive strategies
- Other approximation methods
- Approximation of multiple integrals
  - other cubature methods
  - singular integrals
  - Monte Carlo methods
- Dynamical control of vector sequences computation
  PhD R. Adout
  - acceleration of the restarted GMRES method
  - dynamical control of the dimension of the Krylov subspace
- Automatic methods for round-off error analysis
  - DSA for MATLAB
  - compiler with DSA features
  - linear algebra library
  - grid computing: new methodologies