SAM: a multiprecision stochastic arithmetic library

Stef Graillat, Fabienne Jézéquel, Shiyue Wang and Yuxiang Zhu

LIP6/PEQUAN - Université Pierre et Marie Curie (Paris 6) - CNRS

SCAN 2010, 14th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics

ENS Lyon, France, September 27-30, 2010

The need for arbitrary precision

Floating-point arithmetic precision

- IEEE single precision: 32 bits (24-bit mantissa)
- IEEE double precision: 64 bits (53-bit mantissa)
- extended precision: 80 to 128 bits

Because of round-off errors, some problems must be solved with a longer floating-point format.

`http://crd.lbl.gov/~dhbailey/dhbpapers/hpmpd.pdf`

$\mapsto$ Arbitrary precision libraries

- ARPREC

  `http://crd.lbl.gov/~dhbailey/mpdist`
- MPFR

  `http://www.mpfr.org`

Numerical validation & arbitrary precision

In arbitrary precision, round-off errors still occur...
and require to be controlled!

MPFI: interval arithmetic in arbitrary precision, based on MPFR
`http://mpfi.gforge.inria.fr`
☹ interval arithmetic not well suited for the validation of huge applications

CADNA: stochastic arithmetic
`http://www.lip6.fr/cadna`
☺ used for the validation of real-life applications
☹ in single or double precision

⇨ SAM: Stochastic Arithmetic in Multiprecision

The CESTAC method

Stochastic arithmetic is based on the CESTAC method
M. La Porte and J. Vignes, 1974

It consists in performing the same code several times with different round-off error propagations. 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.

Implementation of the CESTAC method

- each arithmetical operation is performed N times using a 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

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

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

  In practice, $N = 3$ and $\beta = 0.05$.

Self-validation of the CESTAC method

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

- A multiplication of two non-significant results
- or a division by a non-significant 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 computational zero

J. Vignes, 1986

Definition

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

$$\forall i, \quad R_i = 0 \ \text{ or } \ C_{\overline{R}} \leq 0.$$

It means that R is a computed result which, because of round-off errors, cannot be distinguished from 0.

The stochastic relations

Definition

Let X and Y be two results computed using the CESTAC method.

- Stochastically equality, denoted by $s=$, is defined as:
  $X \; s= \; Y$ if and only if $X - Y = @.0$.
- Stochastically inequalities, denoted by $s>$ and $s\geq$, are defined as:
  $X \; s> \; Y$ if and only if $\overline{X} > \overline{Y}$ and $X \; s\neq \; Y$.
  $X \; s\geq \; Y$ if and only if $\overline{X} \geq \overline{Y}$ and $X \; s= \; Y$.

DSA (Discrete Stochastic Arithmetic) is the joint use of the CESTAC method, the computational zero and the stochastic relations.

The problem of stopping criteria

Let a general iterative algorithm be: $U_{n+1} = F(U_n)$, U_0 being a data.

```
WHILE (ABS(X-Y) > ε) DO
   X = Y
   Y = F(X)
ENDDO
```

ε too low $\Longrightarrow$ a risk of infinite loop
ε too high $\Longrightarrow$ a too early termination.

The optimal choice from the computer point of view:
$X - Y$ is a **computational zero** $(X \; s= \; Y)$

$\mapsto$ New methodologies for numerical algorithms have been developed.

The SAM library - I

The SAM library implements in arbitrary precision the features of DSA:

- the stochastic types
- the concept of computational zero
- the stochastic operators.

Arithmetic and relational operators in SAM take into account round-off error propagation.

The particularity of SAM (compared to CADNA) is the arbitrary precision of stochastic variables.

SAM with 24-bit or 53-bit mantissa length is similar to CADNA.

The SAM library - II

- The SAM library is written in C++ and is based on MPFR.
- All operators are overloaded
  $\mapsto$ for a program in C++ to be used with SAM, only a few modifications are needed.
- Classical variables $\rightarrow$ stochastic variables (of `mp_st` type) consisting of
  - three variables of MPFR type
  - one integer variable to store the accuracy.

How to implement SAM

The use of the SAM library involves several steps:

- declaration of the SAM library for the compiler
  ```
  #include "sam.h"
  ```
- initialization of the SAM library
  ```
  sam_init(nb_instabilities, nb_bits);
  ```
- substitution of `float` or `double` by the stochastic type `mp_st` in variable declarations
- change of output statements to print stochastic results with their accuracy, *only the significant digits not affected by round-off errors are displayed*
- termination of the SAM library
  ```
  sam_end();
  ```

Example of SAM code

$$f(x,y) = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + \frac{x}{2y}$$

is computed with $x = 77617$, $y = 33096$.

S. Rump, 1988

```
#include "sam.h"
#include <stdio.h>
int main() {
   sam_init(-1,122);
   mp_st x = 77617; mp_st y = 33096; mp_st res;
   res=333.75*y*y*y*y*y*y+x*x*(11*x*x*y*y-y*y*y*y*y*y
      -121*y*y*y*y-2.0)+5.5*y*y*y*y*y*y*y*y+x/(2*y);
   printf("res=%s\n",strp(res));
   sam_end();
}
```

Output of the SAM code

Using SAM with 122-bit mantissa length, one obtains:

```
SAM software -- University P. et M. Curie -- LIP6
Self-validation detection:  ON
Mathematical instabilities detection:  ON
Branching instabilities detection:  ON
Intrinsic instabilities detection:  ON
Cancellation instabilities detection:  ON
----------------------------
res=-0.82739605994682136814116509547981629 2
----------------------------
SAM software -- University P. et M. Curie -- LIP6
No instability detected
```

Computation of $f(77617, 33096)$

single precision	1.172603
double precision	1.1726039400531
extended precision	1.17260394005317 8
Variable precision interval arithmetic	$[-0.827396059946821368141165095479816292005,$ $-0.827396059946821368141165095479816291986]$
SAM, 121 bits	@.0
SAM, 122 bits	$-0.82739605994682136814116509547981629 2$

Computing a second order recurrent sequence

$U_{n+1} = 111 - \frac{1130}{U_n} + \frac{3000}{U_n U_{n-1}}$ with $U_0 = 5.5$, $U_1 = \frac{61}{11}$ J.-M. Muller, 1989

The exact limit is 6.

Using IEEE double precision arithmetic with rounding to the nearest:

```
U(11) = +5.861018785996283e+00
U(12) = +5.882524608269310e+00
U(13) = +5.918655323805488e+00
U(14) = +6.243961815306110e+00
U(15) = +1.120308737284091e+01
U(16) = +5.302171264499677e+01
U(17) = +9.473842279276452e+01
U(18) = +9.966965087355071e+01
U(19) = +9.998025776093678e+01
U(20) = +9.999882245337588e+01
...
U(29) = +9.999999999999999e+01
U(30) = +1.000000000000000e+02
```

Using SAM in double precision (53 bits):

```
U(3) = 0.559016393442 6E+1
...
U(11) = 0.586E+1
U(12) = 0.59E+1
U(13) = 0.6E+1
U(14) = @.0
U(15) = @.0
U(16) = @.0
U(17) = @.0
U(18) = 0.9E+2
U(19) = 0.99E+2
U(20) = 0.999E+2
...
U(30) = 0.100000000000000E+3
--------------------------------
SAM software -- University P. et M. Curie -- LIP6
CRITICAL WARNING: the self-validation detects major problem(s).
The results are NOT guaranteed.
There are 12 numerical instabilities
9 UNSTABLE DIVISION(S)
3 UNSTABLE MULTIPLICATION(S)
```

using SAM with 100 bits, one obtains:

```
U(3) = 0.55901639344262295081967213l1E+1
...
U(23) = 0.598E+1
U(24) = 0.59E+1
U(25) = @.0
U(26) = @.0
U(27) = @.0
U(28) = @.0
U(29) = 0.9E+2
U(30) = 0.99E+2
U(31) = 0.999E+2
...
U(52) = 0.10000000000000000000000000000E+3
-------------------------------
SAM software -- University P. et M. Curie -- LIP6
CRITICAL WARNING: the self-validation detects major problem(s).
The results are NOT guaranteed.
There are 12 numerical instabilities
9 UNSTABLE DIVISION(S)
3 UNSTABLE MULTIPLICATION(S)
```

Application of SAM to chaotic dynamical systems

Logistic iteration:

$$x_{n+1} = a x_n (1 - x_n) \text{ with } a > 0 \text{ and } 0 < x_0 < 1$$

- $a < 3$: $\forall x_0$, this sequence converges to a unique fixed point.
- $3 \leq a \leq 3.57$: $\forall x_0$, this sequence is periodic, the periodicity depending only on a. Furthermore the periodicity is multiplied by 2 for some values of a called “bifurcations”.
- $3.57 < a < 4$: this sequence is usually chaotic, but there are certain isolated values of a that appear to show periodic behavior.
- $a \geq 4$: the values eventually leave the interval [0,1] and diverge for almost all initial values.

Logistic map

The logistic map has been computed with $x_0 = 0.6$ using SAM and MPFI

- In stochastic arithmetic, iterations have been performed until the current iterate is a computational zero, *i.e.* all its digits are affected by round-off errors.
- In interval arithmetic, iterations have been performed until the two bounds of the interval have no common significant digit.

Comparison of SAM and MPFI - I

Number N of iterations performed with SAM and MPFI, for $x_{n+1} = ax_n(1 - x_n)$ with $x_0 = 0.6$.

a		# bits	N
3.575	SAM	24	142
	SAM	53	372
	SAM	100	802
	SAM	200	1554
	SAM	2000	15912
	MPFI	24	12
	MPFI	53	27
	MPFI	100	53
	MPFI	200	108
	MPFI	2000	1087
3.6	SAM	24	62
	SAM	53	152
	SAM	100	338
	SAM	200	724
	MPFI	24	12
	MPFI	53	27
	MPFI	100	53
	MPFI	200	107

Comparison of SAM and MPFI - II

Number N of iterations performed with SAM and MPFI,
$x_{n+1} = -a(x_n - \frac{1}{2})^2 + \frac{a}{4}$

a		# bits	N
3.575	SAM	24	156
	SAM	53	362
	SAM	100	738
	SAM	200	1558
	SAM	2000	15958
	MPFI	24	93
	MPFI	53	303
	MPFI	100	707
	MPFI	200	1517
	MPFI	2000	15865
3.6	SAM	24	56
	SAM	53	156
	SAM	100	344
	SAM	200	730
	MPFI	24	49
	MPFI	53	143
	MPFI	100	329
	MPFI	200	713

Performance test

Run time (in seconds) of SAM and MPFI for the matrix multiplication $M * M$, with $M_{i,j} = i + j + 1$.

Matrix Size: $N = 100$

ID: Instability Detection

# bits	24	53	100	500	1000	5000
MPFI	0.288	0.316	0.420	0.500	0.652	1.996
SAM without ID	1.004	1.100	1.212	1.308	1.640	3.348
SAM with ID	7.596	8.617	10.813	28.290	70.432	908.709
SAM without ID/MPFI	3.49	3.48	2.89	2.62	2.52	1.68

The ratio SAM/MPFI is independent of N.

Conclusion and future work

Conclusion:

- ☺ validation of scientific codes in any working precision
- ☹ cost of the detection of numerical instabilities

Future work:

- Lorenz attractor
- multiple roots of polynomial
- computation of integrals

Thank you for your attention