



FLUCTUAT: VERIFICATION OF ACCURACY PROPERTIES OF NUMERICAL COMPONENTS AND SYNCHRONOUS EMBEDDED SOFTWARE

Journée précision numérique | Védrine Franck
Joint work with Eric Goubault & Sylvie Putot & Karim Tekkal + Maxime Jacquemin



VERIFICATION OF POLYNOMIAL APPROXIMATIONS

```

1 #include "daed_builtins.h"
2 int main(void)
3 {
4   float t,x,y,z,T1,X1,Y1,Z1;
5
6   t = FBETWEEN(0,1);
7   x = (t-1)*(t-1)*(t-1)*(t-1);
8   y = t*t*t*t - 4*t*t*t + 6*t*t - 4*t + 1;
9   z = (((t-4)*t + 6)*t - 4)*t + 1;
10
11  T1 = FBETWEEN_WITH_ERROR(0,1,0.01,0.02);
12  X1 = (T1-1)*(T1-1)*(T1-1)*(T1-1);
13  Y1 = T1*T1*T1*T1 - 4*T1*T1*T1 + 6*T1*T1 - 4*T1 + 1;
14  Z1 = (((T1-4)*T1 + 6)*T1 - 4)*T1 + 1;
15 }
16

```

Variables / Files

- T1 (float)
- X1 (float)
- Y1 (float)
- Z1 (float)
- main (integer)
- t (float)
- x (float)
- polynome.c

Variable Interval

Float :	-9.37500954e-1	1.00000000
Real :	-9.37500000e-1	1.00000000
Global error :	-1.07288361e-6	1.07288361e-6
Relative error :	-∞	+∞
Higher Order error :	0	0
At current point (9) :	-5.96046e-07	5.96046e-07

Interactive static analysis

```

> the analysis has stopped on AssignControlPoint at /home/fvedrine/DEMO_FLUCTUAT/Fluctuat_Polynome/polynome.c:9,36 z = t - (float) 4 * t + (float) 6 * t - (float) 4 * t + (float) 1
> the analysis has stopped on AssignLibraryControlPoint at /home/fvedrine/DEMO_FLUCTUAT/Fluctuat_Polynome/polynome.c:11,38 T1 = __BUILTIN_DAED_FLOAT_WITH_ERROR((float) 0, (float) 1, (float) 0.01, (float) 0.02)
> p x
float in [0.000000000000000e+00, 1.000000000000000e+00], real in [0.000000000000000e+00, 1.000000000000000e+00], error in [-4.1723254540215860e-07, 4.1723254540215860e-07]
> p y
float in [-2.1875019073486328e+00, 2.7500019073486328e+00], real in [-2.187500000000000e+00, 2.750000000000000e+00], error in [-2.2053718566894531e-06, 2.2053718566894531e-06]
> p z
float in [-9.3750095367431641e-01, 1.000000000000000e+00], real in [-9.375000000000000e-01, 1.000000000000000e+00], error in [-1.0728836059570312e-06, 1.0728836059570312e-06]
>

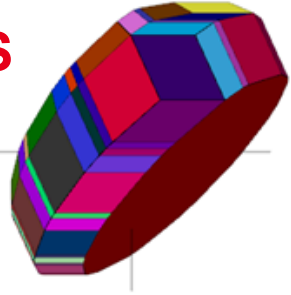
```

Last analysis : 0.00 sec / 16384 Kilo Bytes

VERIFICATION OF INTERPOLATION TABLE

```
double y[10] = { 0.0, 2.0, 3.0, 2.0, 0.0, -2.0, -3.0, -1.9, 0.1, 2.1 };
double in = interval_with_error(0, 10, -1e-6, 1e-6), out;
SPLIT
int index = (int) in;
MERGE
if (index < 0) out = y[0];
else if (index >= 9) out = y[9];
else out = y[index] + (in - index) * (y[index+1] - y[index]);
MERGE
```

- 11 regular paths
- 110 possible unstable branches
- 20 effective unstable branches \Rightarrow 40 unstable paths
- 51 abstract interpretation of the code between **SPLIT/MERGE**
- out: $float \in [-3.0, 3.0]$, $ideal \in [-3.0 - 1.11 \times 10^{-16}, 3.0]$,
 $error \in [-4.0 \times 10^{-6} - 4.85 \times 10^{-15}, 4.0 \times 10^{-6} + 4.44 \times 10^{-15}]$



Why it works

- all computations are linear
- almost no approximation in Fluctuat
- domain and error = same type of transfer function
- $1 + x + x^2 + \dots + x^n$ bounded by $1/(1-x)$

Transfer function

$$(M) = \begin{pmatrix} 0 & 1 \\ 1.4 & -0.7 \end{pmatrix}$$

$$S_n = \sum_{k=0}^n (M)^k \cdot (V) + (V_0) - (V)$$

The screenshot shows the Fluctuat - Filter application window. The main editor displays C code for a second-order filter analysis. The code includes assertions for error bounds and uses the Fluctuat analysis engine. The output window shows the results of the worstcase generation, including a table of results and a list of variables with their intervals.

```

1 #include "daed_builtins.h"
2
3 /*
4  * Second order filter
5  * =====
6  * - Sensors are modeled by "DBETWEEN" assertions
7  * - Output is stored in variable S
8  *
9  * Fluctuat analysis tips
10 * =====
11 * - Increase analyzer internal accuracy to 500 bits and unfold the entire loop.
12 */
13
14 int main(int N) {
15     double S,S0,S1,E,E0,E1;
16     int i,j;
17
18     S = 0.0;
19     S0 = 0.0;
20     S1 = 0.0;
21     E = DBETWEEN(-1.0,1.0);
22     E0 = DBETWEEN(-1.0,1.0);
23     N = 200;
24
25     for (i=1;i<N;i++) {
26         E1 = E0;
27         E0 = E;
28         E = DBETWEEN(-1.0,1.0);
29         S1 = S0;
30         S0 = S;
31         S = 0.7*E - 1.3*E0 + 1.1*E1 + 1.4*S0 - 0.7*S1;
32         DPRINT(S);
33         DPRINT(S0);
34         DPRINT(S1);
35     }
36     DSENSITIVITY(S);
37 }
    
```

Worstcase generation Output:

Maximum value: Minimum value: Maximum error: Minimum error:

Variable E: 1
 Variable E0: 1
 Variable E (1st iteration): 1
 Variable E (2th iteration): 1
 Variable E (11th iteration): 1
 Variable E (10th iteration): -1

Results:

	Min	Max
S	3.84810436	3.84810437

Variables / Files:

- E (double)
- E0 (double)
- E1 (double)
- N (integer)
- S (double)
- S0 (double)
- S1 (double)
- i (integer)

Variable Interval:

Float: -3.84810437 3.84810437

Real: -3.84810437 3.84810437

Global error: -1.64569317e-14 1.64569317e-14

Relative error: -∞ ∞

Higher Order error: 0 0

At current point:

CONVERGENCE PROOF FOR CONVERGENT LINEAR FILTERS

```

Fluctuat - Filter
File Analysis Results Error-Mode Display ?
4 * Second order filter
5 * =====
6 * - Sensors are modeled by "DBETWEEN" assertions
7 * - Output is stored in variable S
8 *
9 * Fluctuat analysis tips
10 * =====
11 * - Increase analyzer internal accuracy to 500 bits and unfold the entire loop.
12 */
13
14 int main(int N) {
15     double S,S0,S1,E,E0,E1;
16     int i,j;
17
18     S = 0.0;
19     S0 = 0.0;
20     S1 = 0.0;
21     E = DBETWEEN(-1.0,1.0);
22     E0 = DBETWEEN(-1.0,1.0);
23     N = TOP(int);
24
25     for (i=1;i<N;i++) {
26         E1 = E0;
27         E0 = E;
28         E = DBETWEEN(-1.0,1.0);
29         S1 = S0;
30         S0 = S;
31         S = 0.7*E - 1.3*E0 + 1.1*E1 + 1.4*S0 - 0.7*S1;
32         DPRINT(S);
33         DPRINT(S0);
34         DPRINT(S1);
35     }
36
37     DSENSITIVITY(S);
38 }
39

```

Fluctuat Resources

Project Name : Filter
Directory : C:/Users/fv174595/AppData/Local/Fluctuat/DEMO/ALGO/Filter//Fluctuat_Filter/

General | **Loops / Widening** | Subdivisions | Misc | Fixed-Point

Unfolding parameters
Initial Unfolding : 200
Depth of inter-loop relational history : -1
Cyclic unfolding : 200
Maximum iterations : 100

Widening
Widening Threshold : 18
Narrowing Threshold : 5

Progressive reduction of precision
Number of iterations before reduction : 20
Loss of precision (bits) : 2 every 1 iteration(s)
Minimal precision : 20

Convergence
 Exact error calculation for constants
 Convergence of higher order errors

Reset parameters Start Analysis Cancel

Standard parameters...

Variables / Files

- E (double)
- E0 (double)
- E1 (double)
- N (integer)
- S (double)
- S0 (double)
- S1 (double)

filter.c

Variable Interval

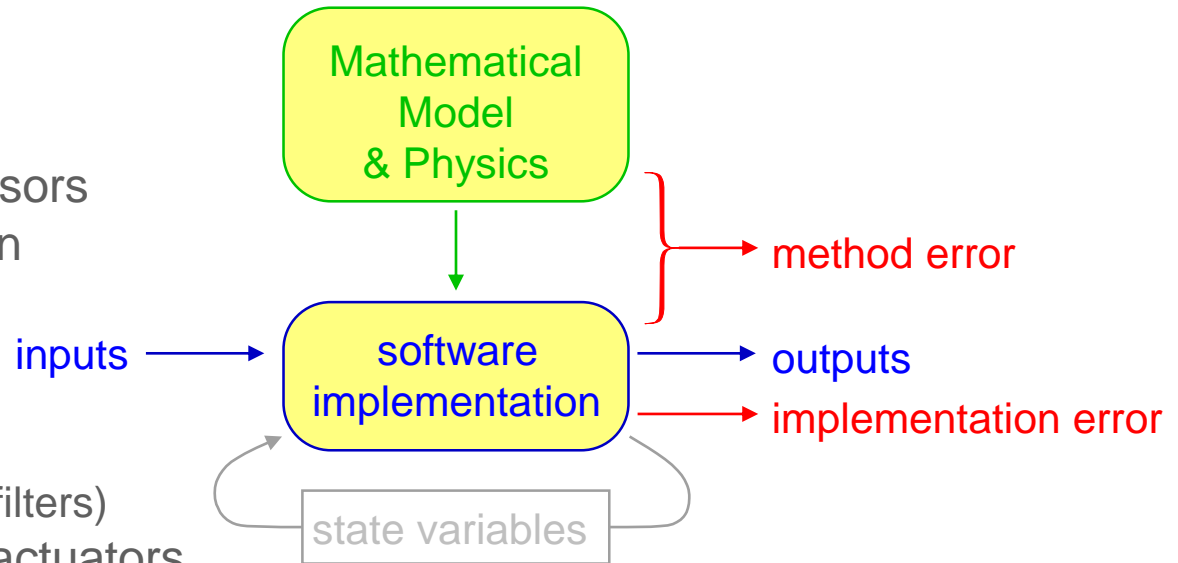
Float : -1.18997538e1 1.18997538e1
Real : -1.18997538e1 1.18997538e1
Global error : -4.65480457e-14 4.65480457e-14
Relative error : -∞ ∞
Higher Order error : 0 0
At current point :

L 96%

EMBEDDED SYNCHRONOUS SYSTEM

- **Infinite loop**

- Take inputs from sensors
- Do some computation
 - in constrained time
 - without any allocation
 - read/write internal state variables (ex filters)
- Produce outputs for actuators



- **Questions**

- Is the accuracy of outputs bounded over time? within acceptable bounds?
- Is the system robust ?
 - small perturbation of inputs produces a small perturbation of outputs

- **Objective**

- Ensure that the implementation is conform to the model
- Avoid to take irreversible decisions on wrong computations

FLUCTUAT CAN BOUND THE ACCURACY

- **if state variables only concern**
 - convergent linear filters of the inputs
 - and/or convergent linear filters of the outputs
- **if no discontinuous unstable branches can occur**
- **if all atomic operations respect the operation hypotheses**
 - no possible division by zero,
 - no negative square root
- **Then Fluctuat can bound the accuracy of the output**
 - for input values in full range
 - in theory for every simulation cycle,
 - unroll the loops on first cycles and see the evolution of error on the filters' state variables
 - depends on time, find adequate parameters (cyclic unfolding, widening and narrowing threshold) to achieve the proof for all cycles

- **Fluctuat has many reasons to fail at bounding domains and accuracy for a full range of inputs**
 - acceptable bounds may not exist
 - accumulation of over-approximations
- **Apply the methodology from some test-cases**
 - compare with observed accuracy with MPFR
 - compare with stochastic intended accuracy with Verrou , Verificarlo, Cadna
 - guaranteed intended accuracy around the test-case
 - investigation and annotations for the unstable branches
 - verification of the results on the first iteration cycles
- **More and more complex system – ex geolocalization with solvers inside**

INTEREST OF USING SEVERAL TOOLS FOR VERIFICATION

- **To find bugs when different tools return unexpectedly different results**
 - ex: if you forget to `std::` when calling the `abs` function, a discontinuous unstable branch will reveal an implicit `int` conversion
- **In a certification context (DO178C – avionics, FDA – medical, ...)**
- **To progressively add annotations and go to proofs**

- **Fluctuat has shown that Abstract Interpretation can effectively deal with accuracy properties to infer and prove accuracy bounds**
- **Fluctuat has features that needs to be developed to target HPC codes**
 - ability to analyze parts of code on large input ranges
 - create a summary that computes an approximation of the precision loss of a function
 - challenge: more symbolic computations for the error
 - ability to keep a trace of the origins of the numerical errors (data flow)
 - to couple with debugging capabilities to explain abnormal behavior
 - challenge: avoid explosion of the number of noise symbols
 - ability to generate and simulate a worst-case
 - to couple with debugging capabilities
 - challenge: define a semantics different but close to IEEE-754 to run this worst-case
 - ability to generates dependencies from a specific parameter/input
 - challenge: adequate user-interface to quickly investigate these dependencies

Thanks for
your attention!

Commissariat à l'énergie atomique et aux énergies alternatives
Institut List | CEA SACLAY NANO-INNOV | BAT. 861 – PC142
91191 Gif-sur-Yvette Cedex - FRANCE
www-list.cea.fr

Établissement public à caractère industriel et commercial | RCS Paris B 775 685 019