Numerical Accuracy Stuff: Tools. . . and Prerequisites
Philippe Langlois

To cite this version:
Philippe Langlois. Numerical Accuracy Stuff: Tools. . . and Prerequisites. CTAOptSim General Workshop, Dec 2018, Montpellier, France. lirmm-02059798
Numerical Accuracy Stuff: Tools... and Prerequisites

Philippe Langlois
DALI, University of Perpignan Via Domitia
LIRMM, UMR 5506 CNRS-UM, France
Blind use of tools = Hazard
Motivations

- Blind use of tools = Hazard
- FPA is an error-prone subject
- Many many recent tools ... but free space towards panacea

Prerequisites

- Floating point arithmetic for dummies
- Errors and measures
- Accuracy vs. Precision: the rule of thumb
- Motto: Don’t forget the problem and its data!

Tools

- What tool for which question?
- Tools: some well-known oldies
- Tools: some works in progress
Context and motivations

Sources of errors in numerical computing

- Mathematical model
- Truncation errors
- Data uncertainties
- Rounding errors

Rounding errors may totally corrupt a FP computation

- Floating-point arithmetic approximates real one
- Accumulation of billions of floating point operations
  - May compensate...
  - but very few are enough to ruin effort
- Intrinsic difficulty to accurately solve the problem
  - Data dependency, condition
Evaluation of univariate polynomials with exact floating point coefficients

\[ p(x) = (x - 2)^9 \] around \( x = 2 \) in IEEE binary64

- expanded form
Example: Schoolbook level

Evaluation of univariate polynomials with exact floating point coefficients

\[ p(x) = (x - 2)^9 \] around \( x = 2 \) in IEEE binary64

- expanded form
- developed polynomial + Horner algorithm

Interesting example!

- Problem? No problem: exact data!
- One problem + one algorithm + one precision but different accuracy for different data

Algorithms:
- the rich vs. the poor
- the good vs. the ugly: summation
Example: Industrial case

OpenTelemac2D simulation of Malpasset dam break (1959)

- A five year old dam break: 433 dead people and huge damage
- Triangular mesh: 26000 elements and 53000 nodes
- Water flow simulation → 35min. after break, 2sec. time step

Reproducible simulation? Accurate simulation?

<table>
<thead>
<tr>
<th></th>
<th>velocity U</th>
<th>velocity V</th>
<th>depth H</th>
</tr>
</thead>
<tbody>
<tr>
<td>The sequential run</td>
<td>0.4029747E-02</td>
<td>0.7570773E-02</td>
<td>0.3500122E-01</td>
</tr>
<tr>
<td>one 64 procs run</td>
<td>0.4935279E-02</td>
<td>0.3422730E-02</td>
<td>0.2748817E-01</td>
</tr>
<tr>
<td>one 128 procs run</td>
<td>0.4512116E-02</td>
<td>0.7545233E-02</td>
<td>0.1327634E-01</td>
</tr>
</tbody>
</table>
Bitwise reproducibility failure: gouttedo test case

time step = 2

![Image of time step 2, number of processors: 0]

![Image of time step 2, number of processors: 2]


time step = 8

![Image of time step 8, number of processors: 0]

![Image of time step 8, number of processors: 2]
Context and motivations

Prerequisite
- FPA for dummies
- Errors and Measures
- Accuracy vs. Precision: The Rule of Thumb

Tools
- Old Folks
  - Interval arithmetic
  - CADNA, verrou
- Recent Tools
  - Herbgrind
  - FP Bench

Conclusion

References
Discretisation (toy system) and precision

- Normal floating point: \( x = (-1)^s \cdot m \cdot 2^e = \pm 1.x_1x_2\ldots x_{p-1} \times 2^e \)

- Precision: \( 2u = 1^+ - 1 = 2^{-p} \)

Rounding, correct rounding and unit roundoff

- \( \circ(x) = x \) for \( x \in F \), else \( \circ(x) = x(1 + e) \) with \( |e| \leq u/2 \) (or \( u \))

- Correct rounding: best accuracy for \( +, -, \times, /, \sqrt{\cdot} \)

- IEEE-754
  - binary32: \( u \approx 5 \cdot 10^{-8}, p = 24, e \in \{-126 \ldots 127\} \)
  - binary64: \( u \approx 10^{-16}, p = 53, e \in \{-1022 \ldots 1023\} \)
Floating Point Arithmetic is Error Prone

Counter intuitive FPA

- Add is not associative
- Absorption: \((1 + u) + u \neq 1 + (u + u)\)
- Catastrophic cancellation: \((1 + u) - 1 = 0\)
- Order matters: \((1 - 1) + u = u\)
- Exact subtraction \(x - y\) for \(1/2 \leq x/y \leq 2\) (Sterbenz)
- Error Free Transformations (EFT) for +, ×:
  - add: \(x + y = s + e\),
  - sub: \(x \times y = p + e\),
- everybody being computable FP values
Automatic Rounding Error Stuff is difficult

Track large errors?

- Small local errors may have large global effect
  - catastrophic cancellation = 1 accurate add + 1 exact sub
- Large local errors may have no global effect
  - error cancellations: $r = (x + y) + z$ for $x, y, z$ resp. computed by $1/u + 1$, $-(1/u + 1)$, $u$ yields exact $r = u$
- Expression error depends on argument values
  - $(x + y) + z$ is accurate except for catastrophic cancellation values

Motto: don’t forget the problem and its data!

Practical limitations: scaling and modularity effects

- Tuning $n$ FP operations between 2 precisions $= 2^n$ cases
- $f(t) + z$ with accurate $f(t) = x + y$ is accurate except for catastrophic cancellation values
Errors and Measures: A Large Array

Errors

- **Forward error**: $x - \hat{x}$, in the result space
- **Backward error**: $d - \hat{d}$, in the data space, for identified $\hat{d}$ such that $f(\hat{d}) = \hat{f}(d)$
- **Absolute vs. Relative error**
- **Maximum vs. Average error**
- **Error measures**: ULPs [1], bits, significant digits [4], no dimension value, interval
- **Error bounds**: proven vs. estimated vs. measured
### RoT: Accuracy \( \lesssim \) Condition Number \( \times u \)

- Forward error \( \lesssim \) condition \( \times \) backward error
- Backward stable in precision \( u \): relative backward error \( \approx u \)

### Condition number

- \( \lim_{\delta \to 0} \sup_{|\Delta x| \leq \delta} \frac{|\Delta y|}{|y|} / \frac{|\Delta x|}{|x|} \),
  with \( y + \Delta y = f(x + \Delta x) \) and \( y = f(x) \).
- Differentiable \( f \): \( \frac{|x||f'(x)|}{|f(x)|}, \frac{|x||J(x)|}{|f(x)|} \)
- Motto: depends both on problem \( f \) and data \( x \)
- Example for summation:
  - \( \text{cond}(\sum_n x_i) = \sum_n |x_i| / |\sum_n x_i| \)
  - Arbitrarily larger than \( 1/u \) when catastrophic cancellation in \( \sum_n x_i \)
Accuracy $\lesssim$ Condition number \times u
How to verify or validate the accuracy of a FP computation?

- Verify vs. validate
- [M] Backward error analysis, probabilistic analysis, ad-hoc rounding error analysis
- [T] Interval arithmetic, stochastic arithmetic, sensitivity analysis, static analysis (+arithmetic models), dynamic analysis (+bounds, +references), formal proof assistants

How to identify the error sources?

- [M] Numerical analysis vs. Rounding error analysis
- [M/T] Algorithm/Program instructions vs. Input data range
- [T] Shadow computation: random, stochastic, higher precision, EFT, “exact”, AD
How to improve the accuracy of a FP computation?

- From accurate *enough* to correctly rounded for a given precision
- [T] More hardware precision, extended precision libraries
- [M/T] More accurate algorithms: expression order, other expression, EFT
  - Hand-made vs. Automatic rewriting tools

Tools: Cost, Efficiency and Tuning

- Cost: reasonable computing *time overheads for running* solutions
- Efficiency: sharp vs. overestimated bound, false positive ratio, non robust optimization
- Tuning: rewrite with a minimal precision for a given accuracy
How to recover the numerical reproducibility of parallel FP computation?

- Reproducible *enough* (i.e. modulo validation) vs. bitwise identical
  - At least to debug parallel vs. sequential,
  - also to validate for production step, to certify for legal process
- Reproducible algorithms, libraries vs. hand-made corrections
1 Context and motivations

2 Prerequisite

- FPA for dummies
- Errors and Measures
- Accuracy vs. Precision: The Rule of Thumb

3 Tools

- Old Folks
  - Interval arithmetic
  - CADNA, verrou

- Recent Tools
  - Herbgrind
  - FP Bench

4 Conclusion

5 References
Interval Arithmetic (1966)

IA at a glance

- Data range or FP arithmetic \(\rightarrow\) intervals + interval operation
- A sure (●) but too conservative (●) propagation of absolute errors (●)
- Dependency problem, wrapping effect, variable decorrelation, conservative inclusion of convex set; intervals containing zero
  - width([x] − [x]) = 2 width((x)),
  - tight function range: tight interval \([F([x])]\)
- Best computing flow driven convex set?
  - endpoint pair, center+radius, subdivisions, Taylor expansions, affine arithmetic, zonotope, ...

\[
[x_{k+1}] = R([x_k] \text{ for } R = R(0, \pi/4) \text{ and } x_0 = [-\varepsilon, \varepsilon])
\]
Interval arithmetic

Interval RoT [2]

- \( \text{width}(f(X)) \leq \lambda_F(X) \text{width}(X) \), where \( \lambda_F \): Lipschitz-constant of \( f \).

![Graph showing interval sum: \( \log_{10} \text{width}(s_n) \) vs. \( \log_{10} \text{cond}(s_n) \) for \( s_n = 1 \) [3]](image)

Tools for Interval Arithmetic

IntLab (Rump), MPFI (Revol) and many other
### Stochastic Arithmetic (1986, 1995)

- Rounding errors are independent identically distributed (uniform) random variables (●) + (CLT) Gaussian distribution around the exact result (●) of their global effect
- Estimation of the number of significant digits with very few values: N=3 samples are enough

### Tools: Cadna (UPMC)

- Random IEEE rounding modes, synchronicity + *computing zero → self validation*
- Practical tool at industrial scale: languages, parallelism, support
- New stochastic numeric types + Library + source to source translator
- ×15-45 overhead: costly hardware rounding mode change

### Tools: verrou (EDF)

- Parametrized random rounding modes, asynchronicity,
- ×10-20 overhead, “no” warning, post-processing tests
- Binary instrumentation (Valgrind), excluded parts (libm)
Many recent tools (2013 →)

**Proven bounds for snippets**
- Abstract model of FPA, forward error: proven (●) but conservative (●)
- Small size targets: 10-20 LOC

**Rewriting snippets**
- 10 LOC

**Detecting candidate error causes**
- Dynamic analysis (Valgrind), shadow computation: MPFR
- False positive, overhead
- Small size targets (●) ... until Herbgrind: 300K LOC (●●●)
Herbgrind (2018)

- Dynamic analysis, binaries (Valgring)
- Large programs, different languages, libraries
- Numerical tricks detection: compensation, EFT
- Open platform: front-end to “small sized oriented tools”, ...
- Input range limitations

Steps

- Detecting FP errors: exact shadow computation (MPFR) for every FP assignation
- Collecting root cause information
  - selected error dependency chains, symbolic expression, input characteristics

Validation cases

- Gram-Schmidt Orthonormalization, PID controller
- GROMACS: molecular dynamics simulation
  - SPEC FPU, 42K LOC in C + 22K LOC in Fortan
- TRIANGLE: accurate and robust mesh generator
A community infrastructure for cooperation and comparison

- FPCore: description format for FP benchmarks
- Benchmarks: suite drawn for published results
  - 111 benchs (v1.1, oct. 2018)
  - FPTaylor (CPU. Utah), Herbie (PLSE, U. Washington), Rosa (AVA, MPI-SWS, Saarbrücken), Salsa (LAMPS, UPVD)

Pros & Cons

- FPCore for fair comparison
- Small size cases, numerically safe case (worst 30% cases error = 5-6 bits)
- Others benchmarks: SPEC FPU, Hamming’s book,...
Numerical accuracy stuff: large and old subject, large literature, many tools but free space for human expertise up to the ideal tools

- Our Motto = hard issue to automatic tools
- Herbgrind: a gap in recent developments?
- Corsika: tuning to low precision FP formats → full benefit of SIMD speedup e.g. . $AVX512 = 16 \times \text{binary32}$
Recent resources

- 30+ tools listed by M. Lam (JMU):
  https://w3.cs.jmu.edu/lam2mo/fpanalysis.html
- FPBench: http://fpbench.org,
  https://github.com/FPBench/FPBench

J.-M. Muller.
On the definition of ulp(x).

A. Neumaier.
Interval Methods for Systems of Equations.

N. Revol.
Influence of the Condition Number on Interval Computations: Illustration on Some Examples.
J. Vignes.

Zéro mathématique et zéro informatique.


(In French).