Let’s talk about precision

Hadrien Grasland
CNRS – LAL

2018-12-17

Know your friend / enemy

- **IEEE floating-point in a nutshell:**

$$\pm 1.\underbrace{0100100111011}_{\text{mantissa}} \cdot 2^{\overbrace{\pm 010011101}^{\text{exponent}}}$$

- **Consequences:**
  - Base 2 → Even good old 0.1 isn't exact!
  - Precision relative to exponent / order of magnitude
  - Unbounded loss of accuracy on subtract / add
  - Very small / large numbers need care

Comparing FP numbers

- **How to tell if val ≈ ref ?**
  - FP precision is relative → Relative comparison often best
  - Typical algorithm looks like $|val - ref| < tol \cdot |ref|$
  - Nice side-effect: tolerance is (mostly) data-agnostic
- **Two limits of relative comparisons**
  - Orders of magnitude may matter (e.g. spatial tolerances)
  - Breaks down when reference is close to zero

Some choice can be good

- **When in doubt, start with relative comparisons**
- **If they prove inadequate, consider other algs…**
  - Absolute comparisons : |val – ref| < tol
  - « Small enough » : |val| < tol
  - « Close or small » : relative unless val & ref are both small
  - L2 norm of difference of matrices vs ref matrix, etc.

Too much choice will kill you

- FP test assertions currently used in ACTS :
  - BOOST_CHECK_CLOSE(val, ref, tol)
  - BOOST_CHECK_CLOSE_FRACTION(val, ref, tol)
  - BOOST_CHECK_SMALL(val, tol)
  - BOOST_TEST(val == ref[, tol]
  - BOOST_CHECK(val.isApprox(ref[, tol]))
  - checkCloseXyz(val, ref)
  - STL container element-wise comparison (*test-specific*)

Consistency matters

- **The previous assertions disagree on many things:**
  - Are relative tolerances given as fractions? Percentages?
  - Can I compare floats with integers? Doubles?
  - Does it work with scalars? Eigen types? STL containers?
  - Is there a default tolerance? A hidden global one?
  - What happens when a value/reference is near zero?
  - Are matrices compared element-wise or by L2 norm?
  - How good is the error reporting?

Trying to improve upon this

- **Key goal: Assertions should be easy to understand**
  - Follow typical & shared conventions
  - Inputs are explicit (nothing global, nothing hardcoded)
  - Simple, general-purpose and predictable logic
- **Some flexibility on comparison algs, input types**
- **Report errors as clearly as possible**
- **My attempt at resolving this: acts-core!490**

One remaining problem

```
/root/acts-core/Tests/Integration/PropagationTestHelper.hpp(527):
error: in "covariance_transport_disc_disc_/_45":
check Acts::Test::checkCloseOrSmall((calculated_cov),
(obtained_cov), (reltol), (1e-4)) has failed. [...]
```

The failure occured during a matrix comparison, where the value was

35447.7	**31.4111**	-1.80979	59.5127	0.291849
31.4111	25761.4	53.0901	1.53086	-8.93186
-1.80979	53.0901	0.112616	**3.72723e-06**	-0.0356915
59.5127	1.53086	**3.72723e-06**	0.1	**-1.98435e-11**
0.291849	-8.93186	-0.0356915	**-1.98435e-11**	0.1

and the reference was

35448	**20.9458**	-1.8162	59.5128	0.291879
20.9458	25864.9	53.1939	1.52074	-8.93245
-1.8162	53.1939	0.112616	**1.91157e-06**	-0.0356914
59.5128	1.52074	**1.91157e-06**	0.1	**0**
0.291879	-8.93245	-0.0356914	**0**	0.1

Help wanted!

- **Seeing this now because we used isApprox() before**
  - isApprox() based on L2 norm: $||val\text{-}ref|| < tol \cdot ||ref||$
  - Comparison dominated by large diagonal terms
  - But… does L2 norm make sense for covariance?
- **Question: how should I handle this issue?**
  - Is this difference physically significant?
  - Should I consider it to be a propagator bug?

Beyond that: single precision experiment

- **Step 1: Evaluate SP tolerances of ACTS code → OK**
  - acts-core!491: Making tests pass under Verrou emulation
  - Affected by previous issue, otherwise looking good…
- **Step 2: Find out what isn't acceptable → Ongoing**
  - Need review from someone who knows the physics!
- **Step 3: Fix the unacceptable part → TBD**
  - Look out for easy « precision bottlenecks »
  - Move what we can to SP, keep rest in double precision

Questions?