Draft Outline for EWRI Conference Paper, Version 2

12-15-2010

Title:

[Insights from Developing an] automated framework and test suite for verifying a one-dimensional transport model

Outline:

* Introduction/Motivation
  + Our problem domain and scaling
  + Requirements of code testing
    - Catch bad coverage of unusual “corner cases”
    - Required accuracy for solving ADR (and reasons?)
    - Catch the known areas of vulnerability in our algorithm
    - Automatic, silent, repeatable tests requiring little intervention
    - Facilitate reports … “acceptance”
  + Lack of a published set of tests
* Software and Numerical Testing Concepts and Tools
  + Software testing
    - Unit, regression, system tests

(Here I do not know what you mean by regression)

* + - Automated testing and assertions
  + Numerical verification
    - Grid convergence as an silent, testable assertion
      * Very good at catches coding errors
      * Versus absolute standard
        + Convergence easier to specify
        + Becomes unreliable when answer is machine precision – switch to absolute accuracy.
        + Accuracy should not be completely ignored
    - Toolkit for convergence and accuracy tests (analytical, Richardson, MMS, PS not very useful)
    - Incremental design and how it relates to the toolkit. As complexity (nonlinearity etc) is layered in, we go from analytical -> Richardson -> MMS
    - Difficulties of Richardson due to boundary/IC compatibility
    - Difficulties of MMS (adequate derivative, scaling)
* Test Suite Design
  + Unit tests
  + Incremental convergence tests
    - Layering of complexity
    - How tailored to algorithm and scaling
    - All the specific tests????
  + Regression tests mined from failures
    - Symmetry/directionality
    - Boundaries
    - Automata and reporting
    - All the specific tests???
  + STM Algorithm and example results
  + Tests that double as reports
* Lessons/Challenges
  + Expressing numerical tests as assertions
    - converting analytical result comparisons
    - strict O(2) standards trigger too many failures
      * regression (better than before, but less than O(2)?
  + Reporting and the fact that some tests really are for human consumption.
  + The human factors
    - Factors that lead to reluctance
    - Tendency to blame algorithm, precision or smoothness over bugs is ubiquitous
  + “Tests are buggier than the code” phenomenon
  + Tests get changed.
  + FORTRAN compatibility with standard testing tools like continuous integration tools
* Conclusions

Figures/Tables:

Here are some initial ideas. Please share any thoughts.

* Figure illustrating the approach (Jamie will draft)
* Table listing the tests, possibly a reworking of the “report card”, perhaps with the addition of a reference section