Evolving from "run to completion" to checking the output

A. Dotti for the Physics Validation Task Force

Parallel Session 2B – Hadronic Testing

U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

Overview

- **Automatic** testing is done with ctest/cdash
- Almost all tests pass if "run to completion"
- In few cases **output is verified against expected results**:
  - Ctest group: PhysicsChecks
  - Test67 : EM testing of gamma incident on Ge detector (only test with a reference to an article)
  - Test73 : MSC test based on LHCb geometry test
  - SimplifiedCaloriemter : full physics simulation of hadronic showers
    - Uses external package StatTest (developed by us) to statistically check results against a reference
- In addition: phonon advanced example compares to reference, however statistics is not adequate for PhysicsChecks (to be reviewed)

Introduction

- Last year we presented the first experiences with ctest/cdash based physics validation https://indico.cern.ch/contributionDisplay.py?contribId=3&sessionId=21&confId=199138
- Not so much has been done since then for hadronics
  - SimplifiedCalorimeter is currently the only test being run every night
  - g4tools and g4analysis **are now mature enough** and can simplify a lot the creation of physics testing

Extend tests

- If a test produces physics results, it should add some physics validation
  - Run time should approximately 2 hours: restricted statistics
    - But we can change this: run PhysicsChecks only on weekends for several hours
    - Tests anyway run in parallel
- Two ways of adding checks:
  - Ad-hoc checking
  - Comparison to reference

Ad-hoc checks

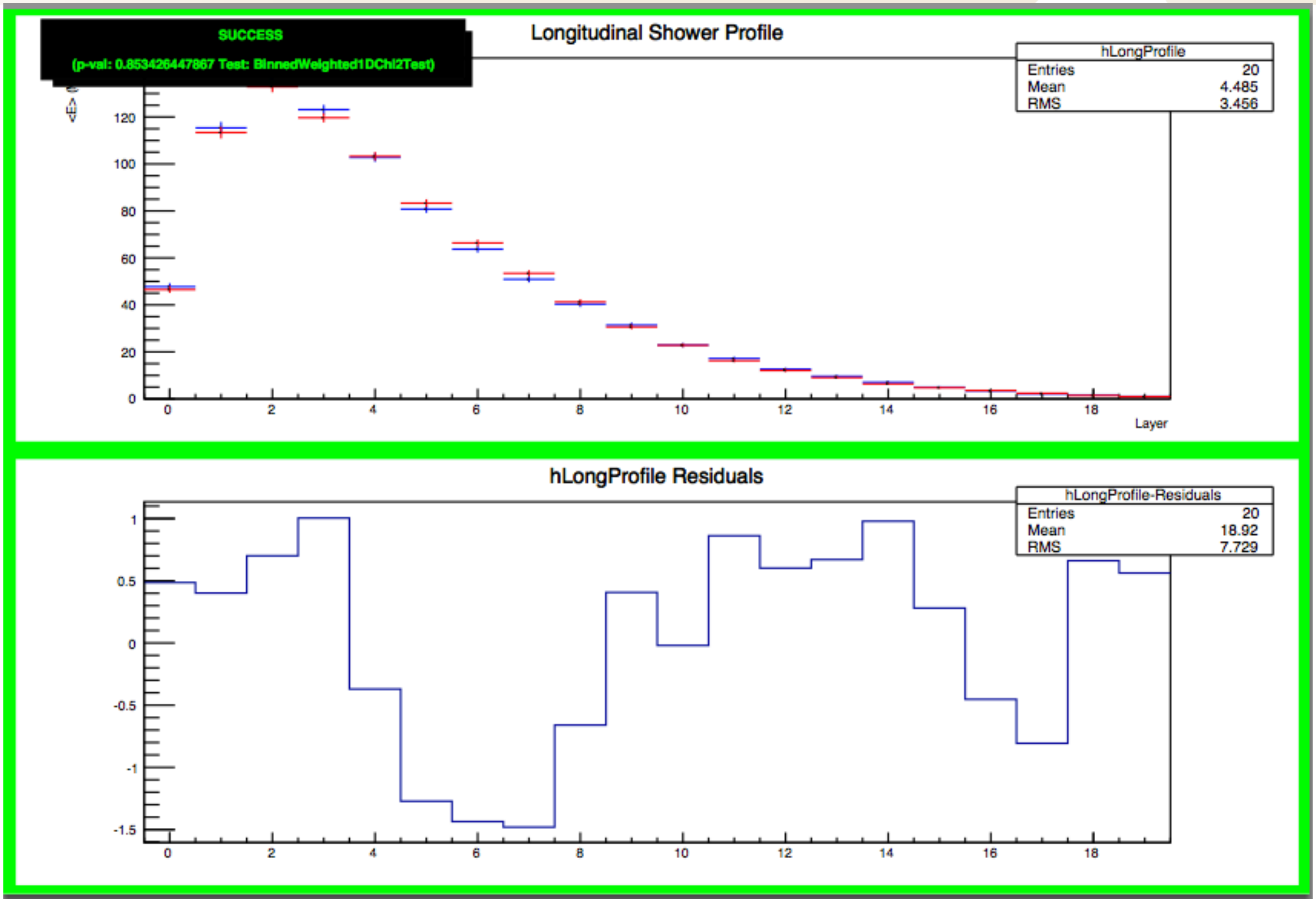
- Simplest solution, just add code that checks the output and at the end of run, write some logs.
  - **If it goes to G4cerr this will trigger a "failure" in cdash**

- Test73 and test67 implement this method
- **Pro**: simplest, developer can make very specific checks
- **Cons**: code duplication in some cases, cannot be shared with other similar tests

Comparison to reference

- **StatTest package** uses ROOT to compare a test output file (in ROOT format) of a test and its output to a reference file
- It needs a simple text file for configuration:
  - Which histograms / tree variables to check
  - Which statistical test run (several provided), typically use p-value, limits for "OK", "BAD", "NotSure"
- Produce text summary and **graphical output**

Example



SUCCESS

(p-val: 0.853426447867 Test: BinnedWeighted1DChi2Test)

Longitudinal Shower Profile

```
hLongProfile
Entries        20
Mean        4.485
RMS         3.456
```

hLongProfile Residuals

```
hLongProfile-Residuals
Entries        20
Mean        18.92
RMS         7.729
```

StatTest in your code

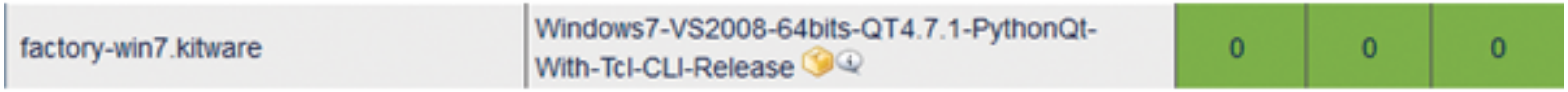
- Use of StatAccep test is more general:
  - An additional test is added AFTER the MC simulation as a separate step
  - To add a new test, it is a matter of producing the test configuration file (very easy in simple cases, believe me!)
- Can be used in CTest (see Parallel 1B) with macro STATTEST_ADD_TEST
- Can be used "stand-alone" in bash shell

Extensions

- I believe these tools are not used widely for two reasons:
  1. Access of graphical output files
  2. Reference handling

Access to results

- Results (graphics) are stored locally on the machine where job is runs
  - **Difficult to access reports, deleted next night**
  - CDash reports error numbers and log summary, not enough for physics checks
- Solution, since version 2.0 CDash supports **file_upload** command
  - Add file_upload to CMake file
  - Files are available from web-page



factory-win7.kitware	Windows7-VS2008-64bits-QT4.7.1-PythonQt-With-Tcl-CLI-Release	0	0	0

Reference handling

- Currently reference files (ROOT binaries) are **stored together with source tree**
- Periodically we need to update the reference with new results

- Proposal new repository and some additional utility in StatTest (with CMake interfaces):
  - upload_result
  - make_reference

Repository for references

- Similar to our other svn repositories:

```
g4tests/trunk/references/testName
       /tags/references/testName/_symbols/2013-09-16
       /branches/references/testName/_symbols/CurrentReference
```

- **Workflow**:
  1. **Checkout references** from CurrentReferences
  2. Run test **comparing with downloaded reference**
  3. After test is concluded**, commit files to SVN**
  4. **Create a tag** with date
  5. If results are "red" human intervention is needed:
     - If new results are considered improvements or expected, **copy from latest tag to branch in CurrentReference**
     - Next time test is run it will take new reference
- Steps 1-4 can be automatized and integrated in ctest
- Can we save all files? Do we have space limits?

Conslusions

- We can solve the two main issues currently preventing wider use of automatic testing for physics
- Results plots can be uploaded to CDash
- Development of reference handling is needed
- **However the most important point:**
  - Developers should provide tests/physics quantities to check!
    - Using g4analysis, you do not have anymore excuses!