The interface to EvtGen in CMS

Roberto Covarelli

University of Rochester

EvtGen workshop, 06/12/2010

Outline

- Introduction to the CMS framework
- The interface to EvtGenLHC in CMS:
  - General structure
    - The Pythia-EvtGen generation chain
  - Technical details and documentation
  - Peculiar features of implementation:
    - The usage of "alias" particles
    - The use of external generators (Pythia, PHOTOS)
- Some tests of EvtGen for inclusive productions
- Conclusions
- Remaining issues

The CMS framework (CMSSW)

- Code compilation/linking (C++):
  - Automatic generation of MakeFile and compilation (scram)
  - Library linking (in order):
    - User-defined libraries in local areas
    - Standard CMSSW packages (a set of compatible package versions is altogether referred to as a release)
    - "External" (non-CMSSW) libraries, e.g. MC generators
  - Single executable application as output (cmsRun)
- Configuration and running (Python):
  - cmsRun driven by a configuration file
    - It contains a schedule of modules to be run in the specified order
  - Output information stored in ROOT file format

EvtGenLHC in CMSSW

- Event generation flow:
  1) Run e.g. Pythia6 as the event source
     - Particle types known from EvtGen tables artificially made stable
  2) Run EvtGenLHC as an "external decay driver"
     - Decay "undecayed" particles that are in EvtGen tables
     - Inclusive B decays (i.e. those whose BR's are not specified) are generated via external interface to Pythia6
     - Radiative corrections calculated via interface to PHOTOS
  3) Output stored as CMS HepMCProducts
     - Decay products are translated from standard HEP to HepMC format

Technical details/documentation

- Technical details:
  - Tested in CMSSW up to 3.9.0 releases
  - External packages needed:
    - EvtGenLHC 9.1 (officially maintained in GENSER)
    - PHOTOS 2.15.5 (officially maintained in GENSER)
      - Upgrade of 2.15.3, required by ATLAS to inhibit FORTRAN "stop" statements
      - Required by CMS in order to fix mess with internal /HEPEVT/-like common blocks (see next slides)
- Documentation:
  - On TWiki:

https://twiki.cern.ch/twiki/bin/view/CMS/EvtGenInterface

"Alias" particles: implementation

- A bit involved because three kinds of aliases are possible:
  1) Aliases that are part of the decay tree of another EvtGen alias
     - Treated normally by EvtGen
  2) Alias particles originally produced by PYTHIA
     - If more than one in the event, one is randomly chosen to be the alias, others are decayed normally
  3) Aliases that are part of an EvtGen decay tree where the mother is NOT an alias (e.g. $B^{0}* \rightarrow B^{0}\gamma$, $B^{0} \rightarrow$ signal mode)
     - This is a particular case, since EvtGen generates the initial decay tree in one go: so daughters must be scanned and, if aliases are found, their products must be dropped and re-generated → see timing

No input required by the user (besides decay file)

"Top-level" aliases: names must be provided by the user in a separate `"forced_decay"` string

External generators

- PYTHIA:
  - PYTHIA called inside EvtGen itself to generate inclusive B decays, e.g. multi-body with unknown BRs and simple phase space distribution
  - OK, but initialization forced within the interface to avoid overwriting of PYTHIA event source parameters (like stable status for B hadrons) → some weird effects observed!!!
- PHOTOS:
  - Originally using /HEPEVT/ common block for C++/Fortran data exchanging
  - Conflicts with CMSSW simulation-reco chain → not understood
  - Solved with:
    - PHOTOS 2.15.5 (currently default in CMSSW)
    - In the interface, with a new EvtGen-PHOTOS data exchange method which completely removes use of common blocks (overrides standard EvtPHOTOS)

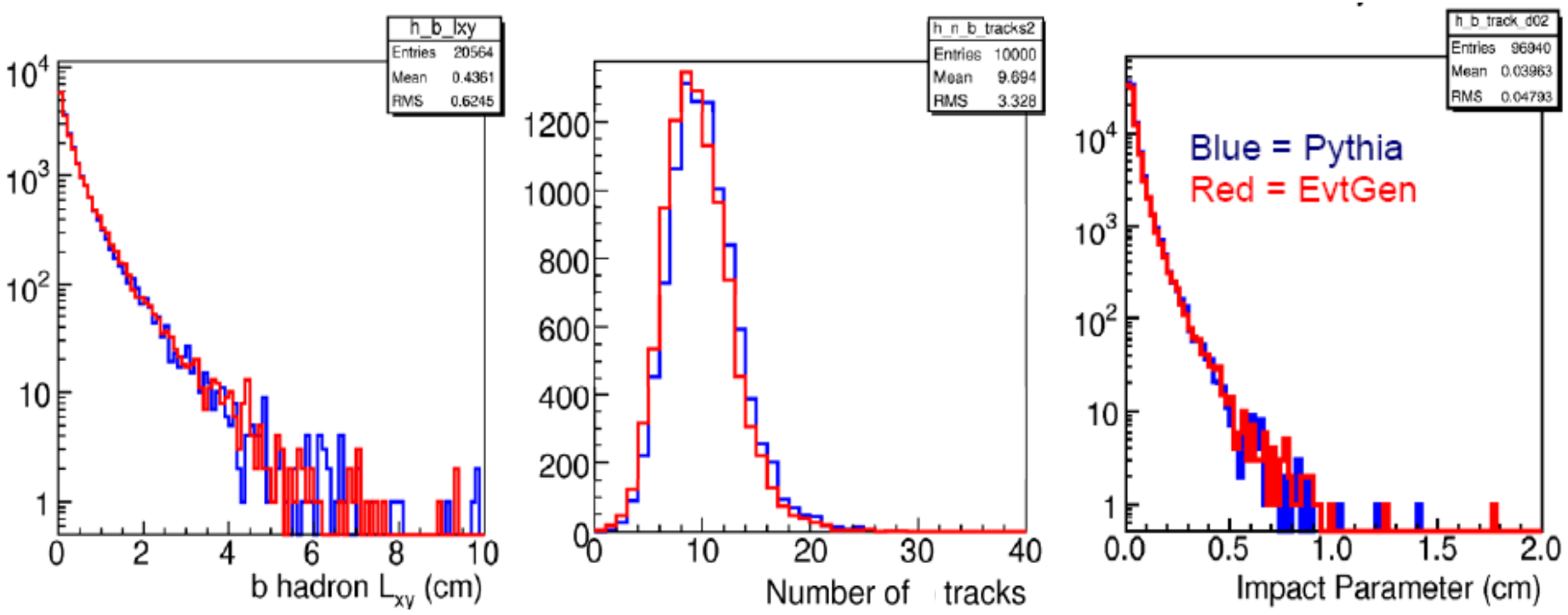
EvtGen for inclusive samples

- Is EvtGen suitable/better for inclusive sample generation?
- Test comparisons with PYTHIA 6:
  - Timing
  - Key variables in ttbar samples (used in b-tagging)
  - Key variables in QCD samples (all EvtGen decays activated, including light mesons… etc.) → ongoing

- EvtGen generation adds 5.5 ms CPU-time per event w.r.t. PYTHIA
  - Observable in a standalone generation task (1.5h for 10^6 events) or in productions with high rejection in generator-level filters
  - Dominated by:
    - recursive search for alias particles
    - light meson decays → optimization?

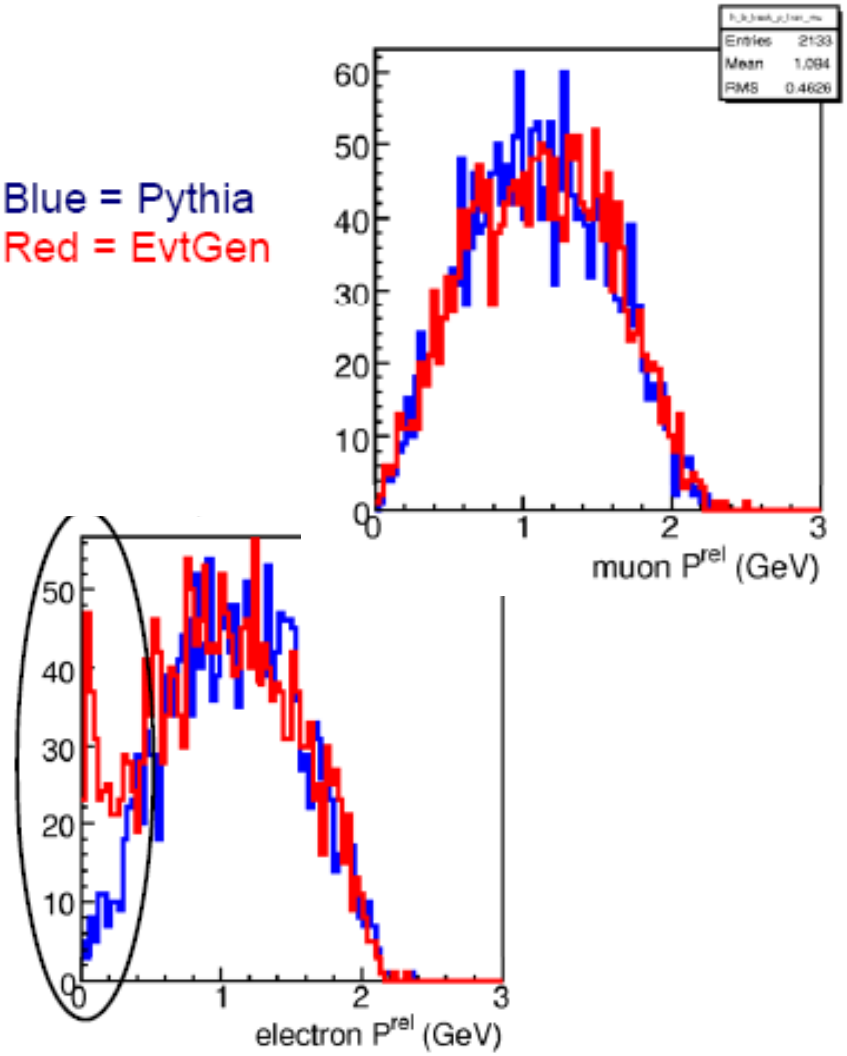
ttbar samples (I)

- Study by G. Giurgiu *et al.*: comparison of MC truth variables relevant for b-tagging
  - PYTHIA vs. EvtGen with long-lived particle decays inhibited



ttbar samples (II)

Blue = Pythia
Red = EvtGen



- Good agreement found in most variables
  - EvtGen predicts a slightly smaller number of tracks per B vertex
  - Discrepancy observed in $b \rightarrow e$ spectrum at very low values of p_T^{rel} (lepton trasverse momentum relative to B hadron direction): under investigation

Attempt to use "merged" version

- Original version has compilation problems inside the CMS framework
- No problem using the "merged" version as implemented in LHCb
  - No big changes required to use it in CMSSW (e.g. PHOTOS, see previous slides)
  - Seemingly big difference in how CPV is implemented w.r.t. the GENSER version

Conclusions

- The CMSSW interface to EvtGen is working fine:
  - Most delicate tasks to be dealt with:
    - "Alias" particle handling
    - Interface to other generators (Pythia, PHOTOS)
  - Already widely used for B-physics signal production
- Set of validation plots provided with the package
  - Generator information for relevant particles
  - Generator-level observables of most important decay modes
- Tests ongoing of use in inclusive event production

Remaining issues

- Software-related:
  - CMS interface:
    - most issues solved, new implementation requires understanding:
      - CP violation
    - timing optimization?
  - EvtGenLHC code:
    - CMS users mostly complaining about lack of documentation:
- Physics-related:
  - Particle properties (including not just PDT standard information, like masses and lifetimes, but also B-specific, e.g. helicity amplitudes in P → VV decays):
    - How often are they updated (LHC-wide versioning)?
    - Compatibility with other MC generators?
  - Tuning issues:
    - Decision of using/not using EvtGen as a general decay tool also depends on experience by other experiments:
      - Status of comparison with other generators
      - Effort needed to adjust MC tunes after use of EvtGen