Overview of the MC generators and procedures in CMS

Alberto Sánchez-Hernández, CINVESTAV

LPCC HFWG Workshop,10th Nov 2015

The CMS Software Framework in CMS

- C++ is use in code compilation:
  - Standard tool (scram) which allow generation of makefile and compilation
  - Libraries are included in order:
    - Local user-defined
    - Standard packages (the software release)
    - non-CMSSW libraries defined as external like root or the MC generators
  - One single executable is produced (cmsRun)

- Python is used for configuration and running
  - cmsRun driven by a configuration file
    - It contains a schedule of modules to be run in the specified order
  - Output is stored in ROOT format

Monte Carlo in CMSSW

- The event generation is performed as:

  - ✓ Pythia8 is used to create the event (initial interaction, we allow to use particle guns or other generators)
  - ✓ Special mention to BCVEGPY, which is used to optimize the generation B_c^+ mesons (pythia takes too long).

  - ✓ Our main BPH generator tool: all particles known by EvtGen are made stable

    - ✓ EvtGen is called as “external decay”

      - Decay “undecayed” particles in EvtGen (standard decay tables)
      - Use external interfaces to pithier itself, PHOTOS (radiative corrections) and Tauola (τ decays)

    - ✓ output is stored as CMS HepMC

      - Decay products are translated to HepMC format

EvtGen in CMSSW

Event Generator

B-mesons

Make them stable

HepMC B-mesons

User input { PDL, Decay files }

EvtGen

HepMC decay products

HepMC Event Record

Details

- Software releases
  - CMSSW MC production branch support and use:
    - Pythia 8.212 (latest version, recently upgraded)
    - EvtGen R01-03-00 (1.3) (will upgrade soon)
    - Other packages:
      - PHOTOS 3.56
      - Tauola 1.1.4
    - Monash 2013 tune

More details

- User turn on EvtGen, by including

```
ExternalDecays = cms.PSet(
   EvtGen130 = cms.untracked.PSet(
      operates_on_particles = cms.vint32(0),           # 0 (zero) means default list
                                                       # but can be a list of particles (PDG IDs)
      decay_table = cms.string('DECAY_2010.DEC'),           # Default Decay Table
      particle_property_file = cms.FileInPath('evt.pdl'),  # Default Particle properties
      user_decay_files = cms.vstring('My_Signals.DEC'),    # user custom decay signals
      list_forced_decays = cms.vstring()               # those defined in My_Signals.DEC
   ),
   parameterSets = cms.vstring('EvtGen130')
),
```

customizable, allows to configure several signal at ones, combining all available options

Generating signals

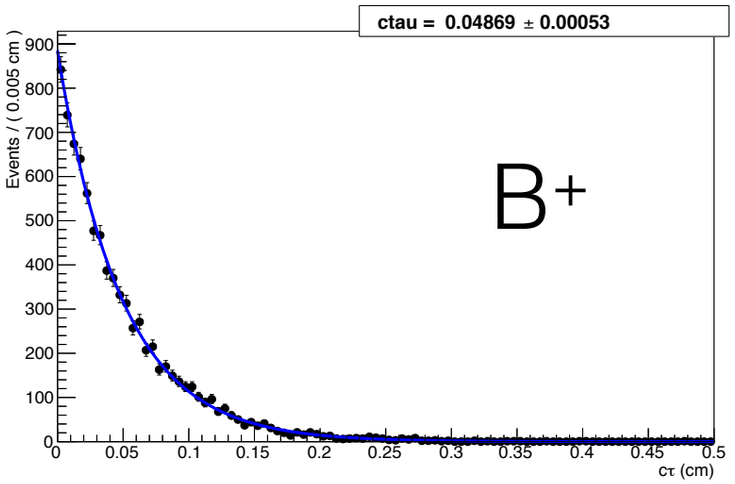
For generating signals, (a given decay = forced decay) we use the “Alias” particles, which can be:

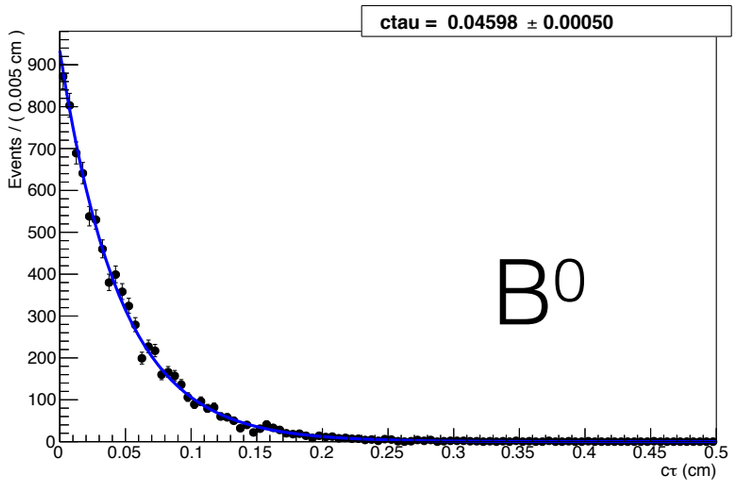
- ✓ particles part of the decay of another EvtGen alias
  - User just need to provide the decay file, no other action is required
- ✓ particles produced by PYTHIA
  - Typical only one signal per event is expected, so If more than one in the event, only one is randomly picked as the alias, the others are decayed normally
- ✓ particles part of an EvtGen decay where the mother is NOT an alias (e.g. $B^* \rightarrow B\gamma$, $B \rightarrow$ signal)
  - Since EvtGen generates the initial decay: daughters are scanned and, if aliases are found, their products are remove and re-generated

- Last two cases are adjusted by the user providing the
  - list_forced_decays = cms.vstring()

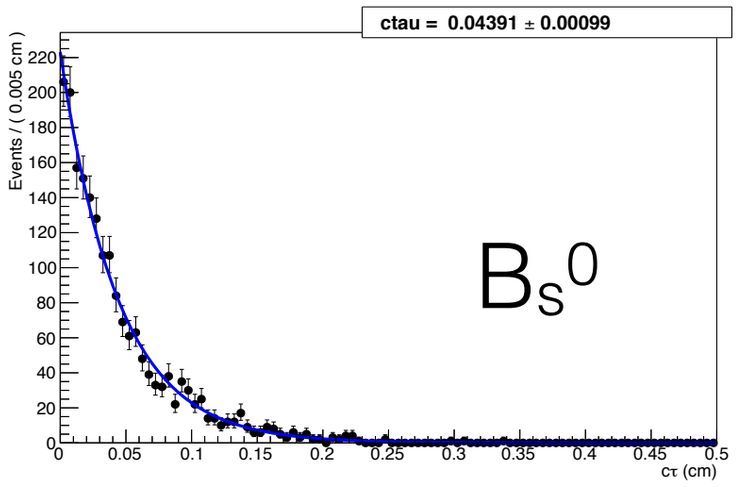
Key features

Before going into production, we made several basic test, most of the time comparing what we already have done in the past (e.g. pythia6+EvtGenLHC): We tested, rates, lifetime, mixing, CPV, …





Lifetime for B mesons:

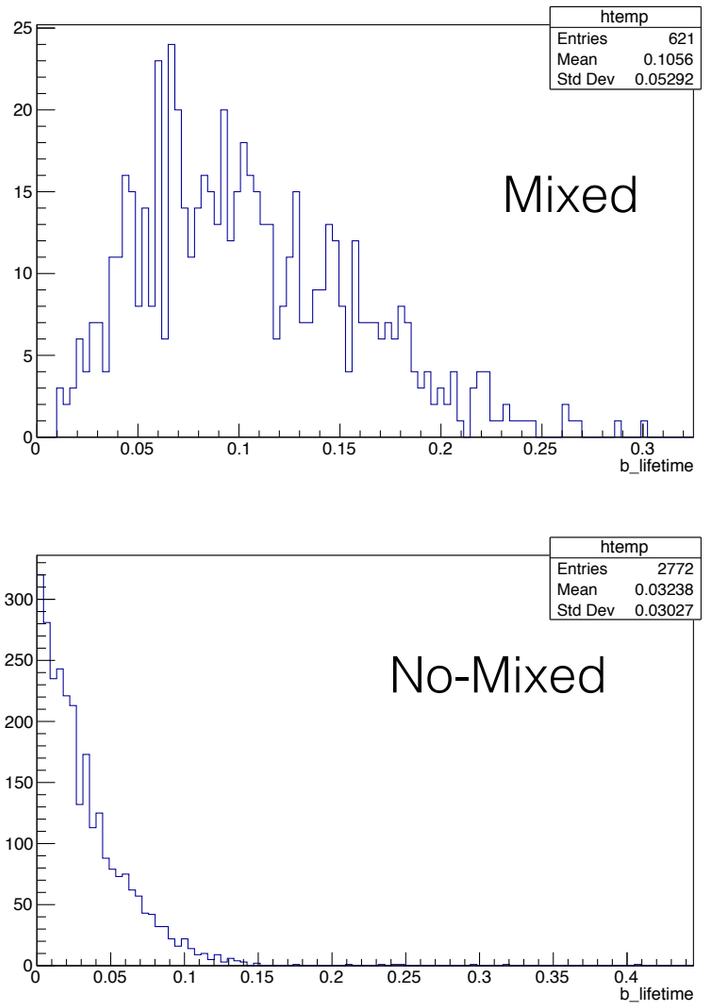


pythia8+EvtGen1.3

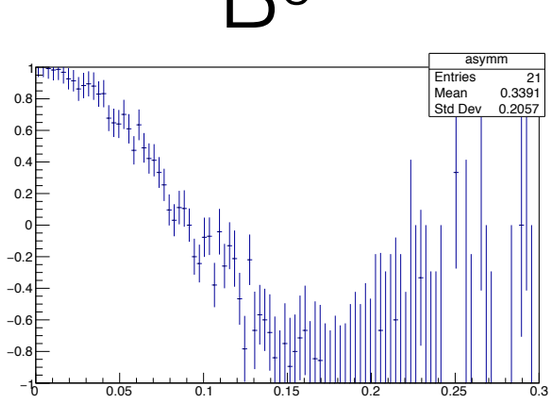
Key features

Before going into production, we made several basic test, most of the time comparing what we already have done in the past (e.g. pythia6+EvtGenLHC): We tested, rates, lifetime, mixing, CPV, …

Mixing in B mesons:

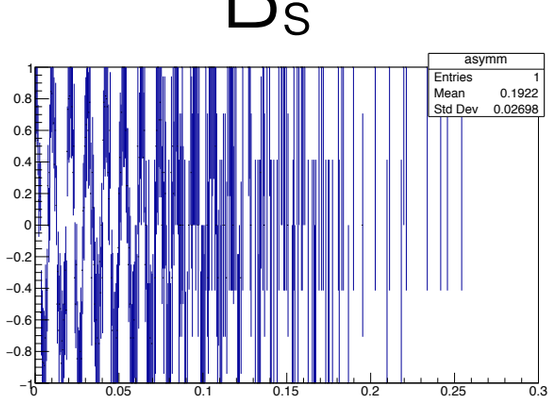


B^0



or

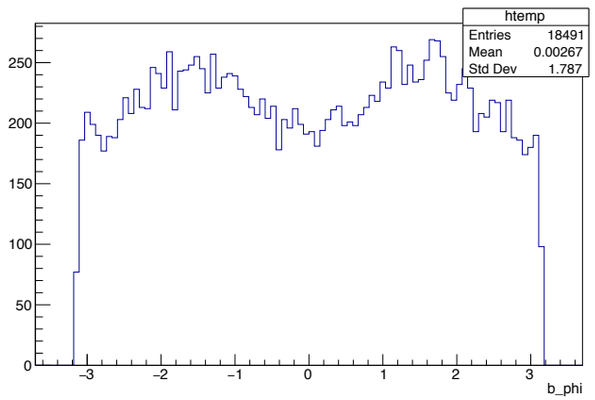
B_s

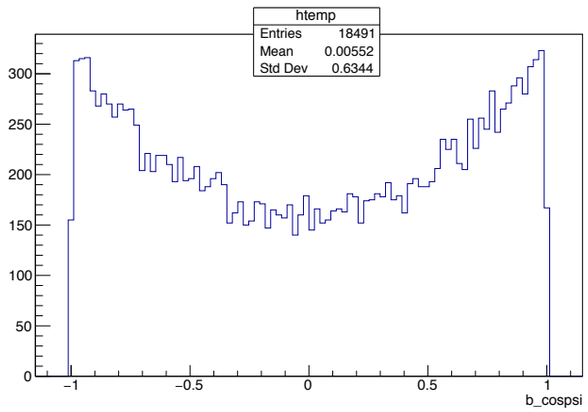


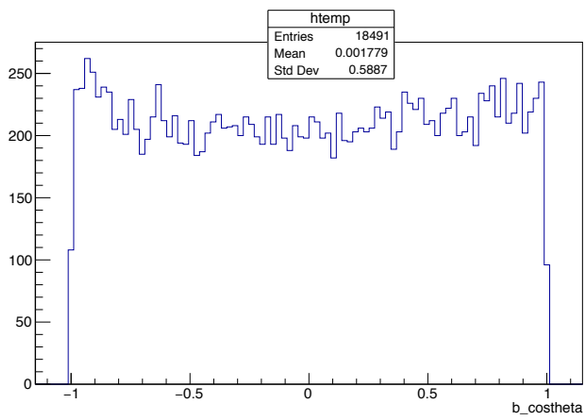
pythia8+EvtGen1.3

Key features

Before going into production, we made several basic test, most of the time comparing what we already have done in the past (e.g. pythia6+EvtGenLHC): We tested, rates, lifetime, mixing, CPV, …





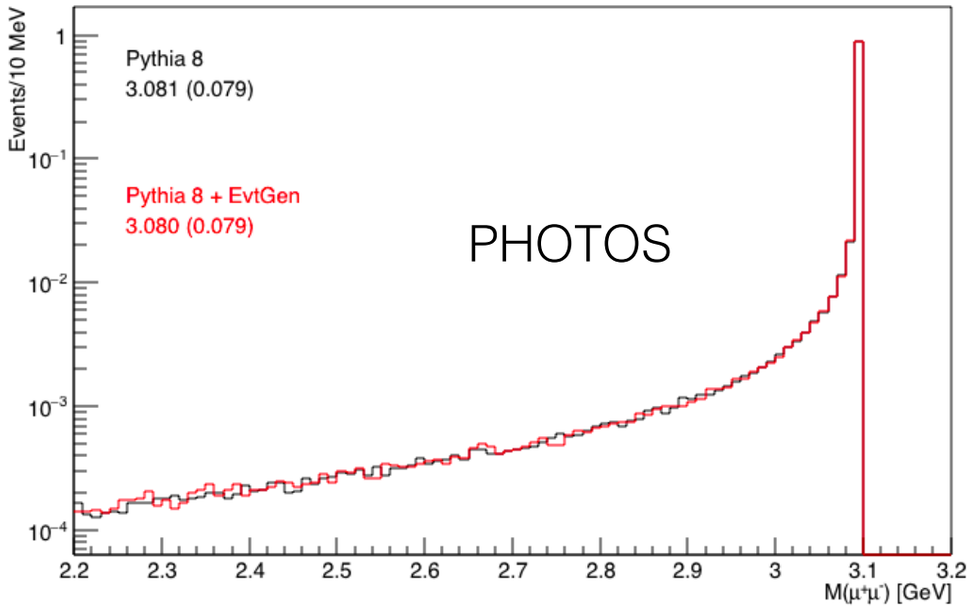


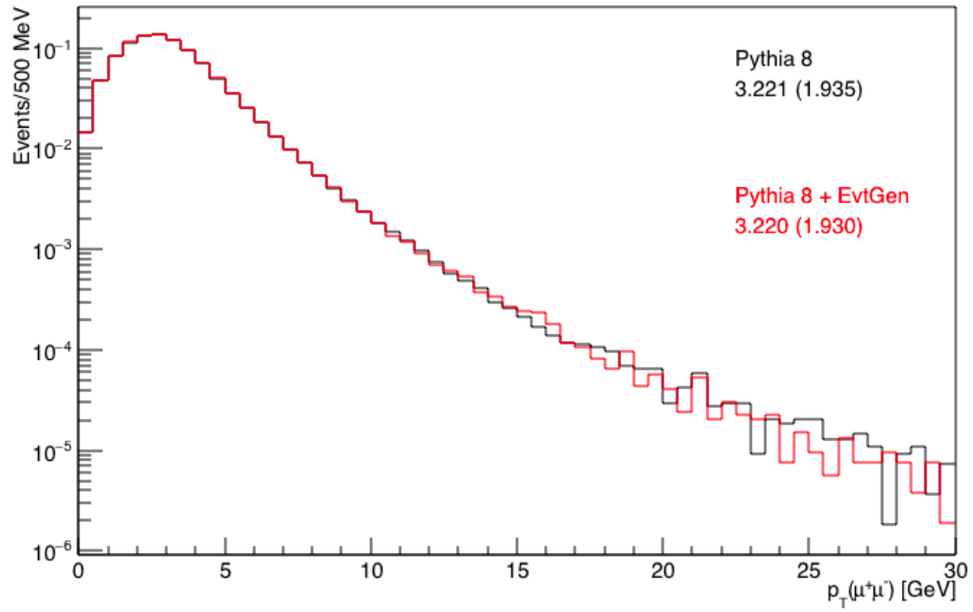
Testing of special decay models like PVV_CPLH for B_s

pythia8+EvtGen1.3

Current usage

- EvtGen used currently in CMS for massive Monte Carlo productions for current BPH analysis





Quarkonia Studies in progress for Early RunII data

Pythia8 EvtGen plugin

Recently (8.204), but specially in latest version 8.212 a EvtGen interface has being implemented. It is quite complete and allow us to use all features without worry about changes in pythia

Procedures within CMSSW

- Being CMS a multi-purposes experiment, has to deal with several requirements at once.

- Implementing a new software cycle or a patch in the generators may be not “fast” enough, since several steps need to be meet (implementation, validation, approval, etc.) before actually going into production.

- The optimistic estimate if a Br or a new decay table needs to be put into the release could take even more than a month.

Summary

- Main tool for BPH at CMS

  - Event generators: Pythia8, BCVEGPY + EvtGen
  - Standard versions, no special modifications
  - Currently using version 8.212, 2.2 and 1.3 respectively
  - For RunII CMS is using Warwick EvteGen version as default, with no further modifications

ToDo:

- No full validation of CPV has being achieved, some issues have not being resolved.

Backup

Main Tool: EvtGen

- Short history

  - Created by Anders Ryd and David Lange
  - Widely used and tuned at the B-factories (BaBar, BELLE) and CLEO
  - Also adopted as main b-physics tools at the Tevatron RunII: DZero and CDF (contributions to Bc, and b-baryons)
  - Natural adoption at LHC: LHCb, ATLAS, CMS
  - At the beginning, several version available (each experiment made its own modifications)
  - Warwick group took over in 2010 as the main developer team.