Generating Initial Particle Kinematics with GenTools

Brett Viren

Physics Department



Daya Bay Offline Workshop @ BNL, 2008/04/7

Contents

1. Overview
2. GenTools Design
3. Kinematics Data
4. Suite of Existing Tools
   - GtHepEvtGenTool
   - GtGunGenTool
   - GtPositionerTool
   - GtTimeratorTool
5. GenTools Algorithms
   - GtGenerator
   - GtHepMCDumper
6. Some Example Configurations
7. Writing Your Own GenTool
8. Example
9. Summary

Generation of Initial Kinematics

The basics:

- Kinematics = spatial/temporal vertex, particle types and 4-momenta.
- Models all initial interactions that we don't want to leave for Geant4.
- Examples:
  - $\bar{\nu}_e + p \rightarrow e^+ + n$
  - Cosmic-μs
  - Radioactive decays
  - Calibration sources
  - Bare particles for testing

Some requirements

- Want to support existing, "legacy" stand-alone programs.
- Easily write new generators.
- Position vertices in volumes following various distributions.
- Choose event times based on some rate.
- Parameterized generation for testing, simple calibration source modeling.
- Decouple unrelated quantities (eg, time, vertex, momentum)
- Use Gaudi TES, standardize object format.
- User-accessible configurations.
- Support multiple processing models.

GenTools Design

Central idea: a factory. One tool for each job, an assembly line connects the tools.

1. Start with an empty event object.
2. Sequentially pass it through a series of tool objects.
3. Each tool may add new or modify existing data in the event.
4. Results are then based on which tools are used and how they are configured by the user.

Features:

- Marshalling of data decoupled from actual generation.
  - It is up to an algorithm which uses the generators to impose the processing model by how it marshals the resulting output.
- Modular components: eg. reuse same vertex positioning tool with different momentum generators.

Try to use this model for all stages of simulation!

Kinematics Data

Intermediate and final TES data are in form of:

HepMC::GenEvent : vertex, an event number, process ID

HepMC::GenVertex : 4-vertex, type, incoming/outgoing particles

HepMC::GenParticle : static and dynamic particle quantities

Default location in TES: `/Event/Gen/HepMCEvents`, a simple collection of `HepMC::GenEvents`.

More on HepMC at: http://lcgapp.cern.ch/project/simu/HepMC/

GtHepEvtGenTool

Converts sources of HEPEvt formatted data

- Can run "legacy" generator executables, read their output.
- Can read HEPEvt formatted files.
- Newly written HEPEvt parser class (simpler than G4dyb's)
- Caveats (same old ones as with G4dyb):
  - Must assume enough input events provided but will re-run to get more.
  - Existing generators do not always produce useful 4-vertex info, need to rely on other tools for this.

Configuration parameter:

HepEvtDataSource command line to execute or filename to run. Uses the same hint as G4dyb to decide which (first character of command line must be a pipe symbol ("|").

GtGunGenTool

Parameterized particle generation

- Only handles particle type and 4-momentum
- Number and type of particle, momentum and direction distributions.
- Directions generated in volume-local or global reference frames.

Configuration parameters:

ParticleName Canonical PDG name of particle to generate

ParticlesPerEvent # of particles to generate each event

MomentumMode "Fixed", "Uniform", "Smeared" (ie. Gaussian)

Momentum Value for Fixed or center for Uniform & Smeared

MomentumSpread Momentum distribution width for Uniform & Smeared.

DirectionMode "Fixed", "Uniform", "Smeared" (ie. Gaussian)

Direction Value for Fixed or center for Uniform & Smeared

DirectionSpread Direction distribution width for Uniform & Smeared

Volume Detector Element to use for coordinate system.

GtPositionerTool

Generate a vertex.

- Only generates an event's 3-vertex.
- Uses physical volume to generate position in local coordinates.
  - You do not need to guess a global point that is inside the volume as with G4dyb.
- Transforms to global coordinates to make Geant4 happy.
- Not all parameterizations are yet implemented, **volunteer?**

Configuration parameters:

Mode	"Fixed", "Uniform", "Smeared" (ie. Gaussian)
Strategy	"FullVolume", "AvoidDaughters", "Surface"
Volume	Detector Element to use for coordinate system.
Position	Value for Fixed or center for Uniform & Smeared
Spread	Distribution width for Uniform & Smeared

GtTimeratorTool

Set event time based on an average rate.

- Sets time for each event based on previous time and average rate.

Configuration parameters:

StartTime Initial starting time.

LifeTime The "lifetime" for the event to occur or 1/rate.

GtGenerator

Driving algorithm for a simple processing model (single event type push).

- Manages the factory assembly line of tools
- Writes results to TES.

Configuration parameters:

GenTools	List of names of GenTools to use
Location	TES path in which to place the results.
GenName	Label to identify what this algorithm is generating.

A different variant of this algorithm is needed to support "15 minutes of fame" style processing (Zhe's talk).

GtHepMCDumper

Dump generator information to terminal.

- Simply checks that kinematics can be written/read to/from TES.
- Gives examples of how to read TES and unpack HepMCEvents.

Configuration parameters:

Location TES path. Defaults to /Event/Gen/HepMCEvents.

Configuration - Basics

```
1  from  GaudiPython import AppMgr
2  import GaudiKernel.SystemOfUnits as units
3  app = AppMgr()
4  app.TopAlg += [ "GaudiSequencer/GenSeq" ]
5  genseq = app.algorithm("GenSeq")
6  genseq.Members = \
7       [ "GtGenerator/GenAlg", "GtHepMCDumper/GenDump" ]
8  gen = app.algorithm("GenAlg")
9  gen.GenTools = \
10      [ "GtGunGenTool", "GtPositionerTool", "GtTimeratorTool" ]
11 gen.GenName = "Bang Bang"
12 gen.Location = "/Event/Gen/HepMCEvents" # this is default
```

- Bring in units module for later.
- Put `GtGenerator` and dumper as sub-algorithms of a sequencer.
- Add sequencer as a top-level algorithm.
- Set generator tool list, give it a name and an output location.

Configuration - Positioner and Timerator

```
1  poser = app.property("ToolSvc.GtPositionerTool")
2  poser.Volume = "/dd/Structure/AD/la-oil2"
3  poser.Mode = "Fixed"
4  poser.Position = [0,0,5.0/16.0*units.meter]
5
6  tim = app.property("ToolSvc.GtTimeratorTool")
7  tim.LifeTime = int(1*units.second)
```

- Generate **Fixed** positions in LingAo AD#2 from the center line of the volume and up at the level of the 5^{th} PMT ring.
- Set the mean event rate to 1 Hz.

Configuration - Particle Gun

Sets particle and momentum from parameters.

```
1  gun = app.property("ToolSvc.GtGunGenTool")
2  gun.Volume = "/dd/Structure/AD/la-oil2"
3  gun.ParticlesPerEvent = 1
4  gun.ParticleName = "opticalphoton"
5  gun.Momentum = 2.5*units.eV
6  gun.MomentumMode = "Fixed"
7  gun.DirectionMode = "Fixed"
8  from math import sin, cos, pi
9  pmt_column_number = 9
10 angle = (2*pmt_column_number - 1)*pi/24.0;
11 gun.Direction = [ cos(angle),sin(angle),0 ] # aim for PMT
```

- Set volume in which directions are generated.
- # particles, name, momentum and direction and their modes.
- This with the positioner hits PMT at ring #5, column #9.

Configuration - HEPEvt source

Uses HEPEvt generator source (file or executable)

```
1  hepevt = self.app.property("ToolSvc.GtHepEvtGenTool")
2  hepevt.HepEvtDataSource = "|uranium.exe 100 987654321"
3  #hepevt.HepEvtDataSource = "/path/to/uranium-output.hepevt"
```

- Set HEPEvt data source is executable command or file.
- Commands must start with a pipe symbol "|"
- Both should generate at least as many events as the job intends to consume.
  - If these are exhausted the source is re-opened and read again.

How to write your own GenTool

1. Inherit from GaudiTool and IHepMCEventMutator
2. Implement the method:
   `IHepMCEventMutator::mutate(HepMC::GenEvent& event)`
3. Add usual Tool and Factory declaration and load CPP macros
4. Generic tool writing help here:
   https://wiki.bnl.gov/dayabay/index.php?title=GaudiTool

Karen is working on providing such native GenTools for the Uranium and Thorium generators.

Example

Example algorithm to histogram kinematics quantities:

- `tutorial/Simulation/SimHistsExample/src/GenHists.cc`

Driver script:

- `tutorial/Simulation/SimHistsExample/python/genhists.py`

Summary

- GenTools provides a full featured kinematic generator package.
- Supports writing to Gaudi TES, Gaudi configuration mechanism.
- Exploits DetDesc to simplify local→global transformations.
- Expected generator features from G4dyb are reproduced or improved.
- "Legacy" HEPEvt executables and files still supported.
  - Work underway (Karen) to move "legacy" U/Th to native GenTools.
- Some loose ends need tieing up, volunteers welcome.