LCDG4 and DigiSim

Simulation activities at NIU

(talks for Calice Meeting, LCSimWS and LCWS'05)

Guilherme Lima

DHCal Meeting at NIU
March 09, 2005

Which talks go where?

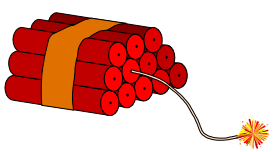
- 20' at Calice meeting: DigiSim
- 30' at LC Simulations Workshop
  Latest developments on LCDG4
- 10' for LCWS:
  LCDG4 and DigiSim (suggestions welcome!)

LCDG4 talk: Outline

- LCDG4
  - Introduction
  - Simple analyses
  - Status and plans

LCDG4: Full detector simulation

- A Geant4-based detector simulator to support detector R&D for the Linear Collider
- Alternatives: Mokka (Europe), LCS or SLIC (new, SLAC)
- LCDG4 features
  - Input format: binary STDHEP
  - Output format: ***.lcio*** (standard) or ***.sio*** (Gismo compatible)
  - Some detector geometries are implemented via XML geometry files (e.g. SiD, LD and variants)



Simplistic geometry: cylinders, disks and cones only, no cracks, limited representation of support structure

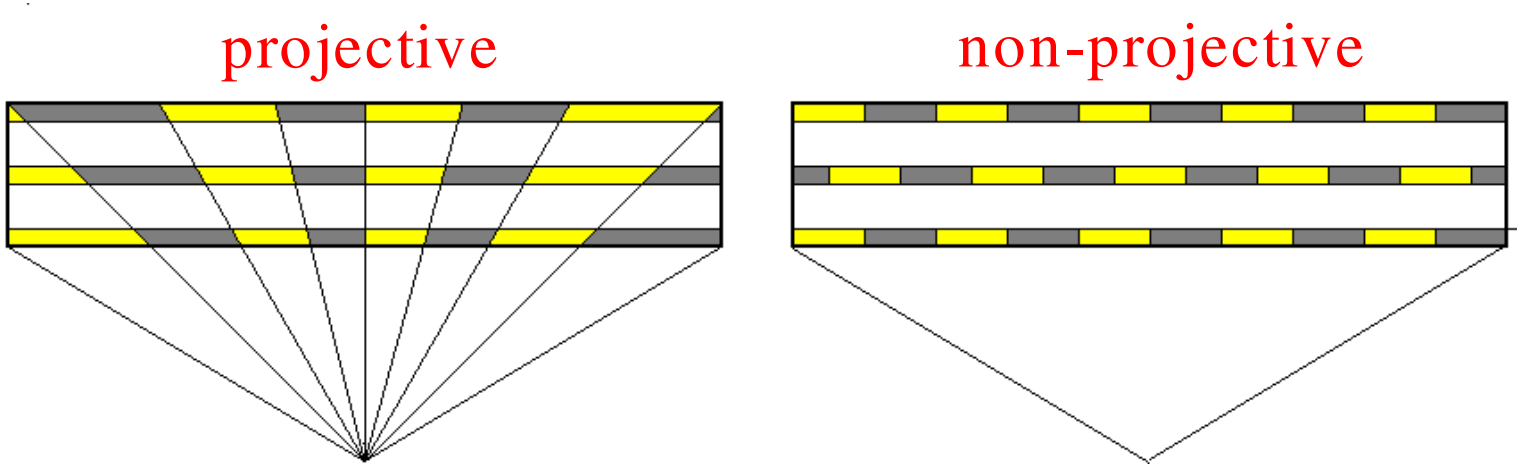
Some LCDG4-specific features

- Development bias towards calorimeters
  other subdetectors probably work fine, but no active studies have been done by the developers
- Gismo compliant *by design* (transparent change for Gismo users)
- Correct MC particle hierarchy, even when V^0s and hyperon decays are forced at the event generation stage
- Energies deposited in absorbers are easily available for cross-check analysis (ASCII format only)
- Non-projective calorimeter geometries also available via XML (see HAD_Barrel in ParFiles/SDNPHJun04.xml for an example)
- Simple analysis code and documentation available from CVS and http://nicadd.niu.edu/~lima/lcdg4/.

Projective or non-projective

Projective or non-projective virtual calorimeter cells available

- barrel only for now (see e.g. SDNPHJun04.xml for a version with Non-Projective Hcal), but nonproj endcaps may be available later
- Rectangular virtual cells with fixed size, controlled at run time (XML)
- ECal or HCal barrels can be made projective or NP independently



Geometry description in XML

An example: Hadronic calorimeter barrel.

Dimensions are in centimeters.

```
...
<volume id="HAD_BARREL" rad_len_cm="1.133" inter_len_cm="0.1193">
  <tube>
    <barrel_dimensions inner_r="144.0" outer_z="286.0" />
    <layering n="34">
      <slice material="Stainless_steel" width="2.0" />
      <slice material="Polystyrene" width="1.0" sensitive="yes“ />
    </layering>
    <!--segmentation cos_theta = "600" phi = "1200" /-->
    <cell_info length="1.0" height="1.0" offset="0." outer_r="246.5"/>
  </tube>
  <calorimeter type="had" />
</volume>
...
```

Proj or NP selection: either one, not both.

Most of detector changes are very easy to make!

More LCDG4 features

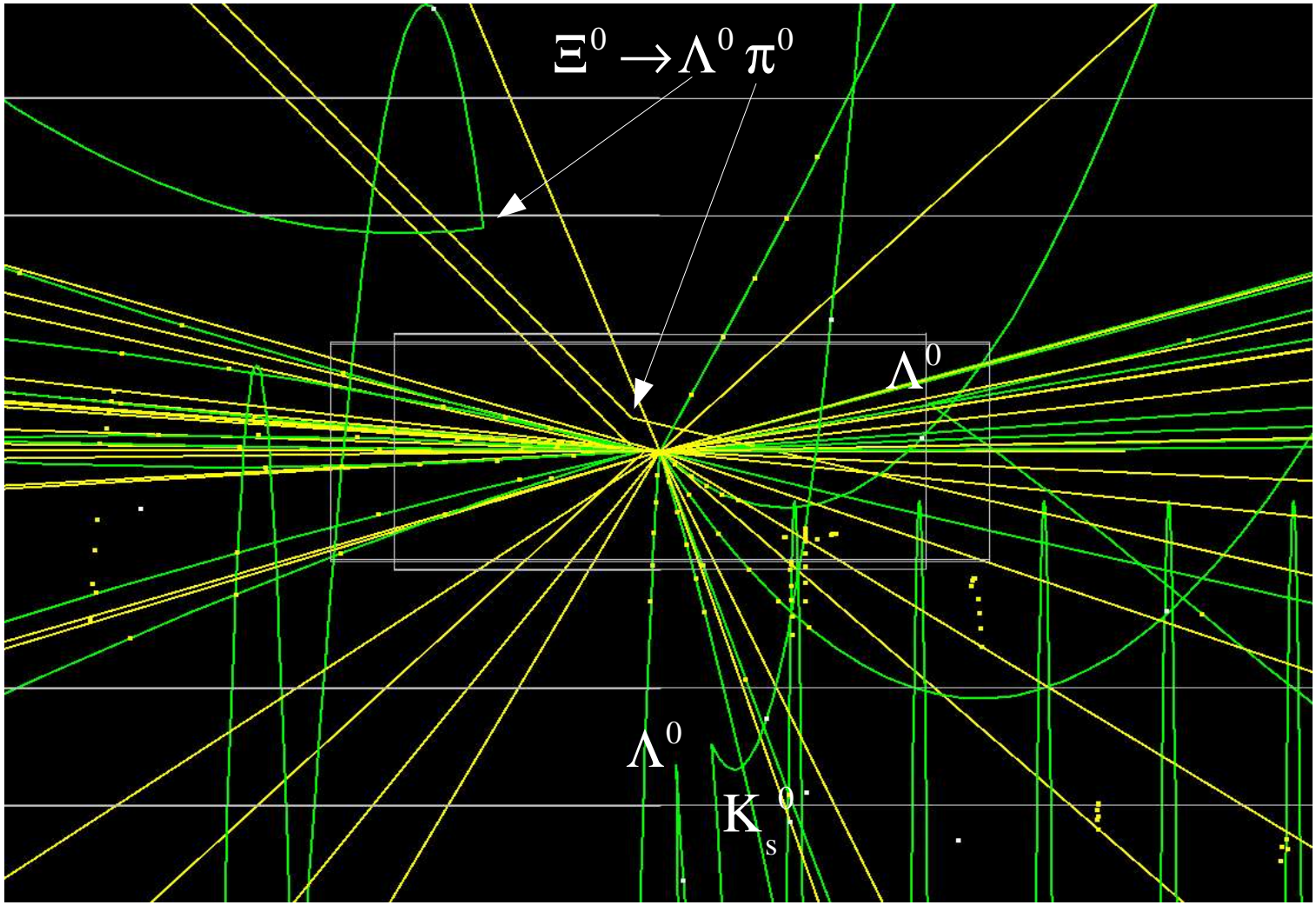
- Uses Geant4 version 7.0 and most hadronic physics lists available, including the LC-oriented physics list LCPhysics.
- Correct handling of pre-assigned decays (forced in generator), even for decays of short-lived particles or hyperons
- Useful debugging tools are available, like:
  - Run-time control (RTC) of debugging printout
  - RTC of random seed initialization, either one-time or same seed for all events
  - RTC of skipping events from input file
  - Extra compact output file with number of hits in each subdetector, for each event

SIO / LCIO output: general features

- One particle collection and several hit collections (one hit collection per subdetector)
- Each hit points to the contributing particles (except tracker hits from calorimeter back-scatterings, as in Gismo)
  - LCIO only: (x,y,z) coordinates stored for every hit
- All secondaries above an energy threshold (now set at 1 MeV), except for shower secondaries, are saved in output with:
  - Particle id and generation/simulation status codes
  - Production momentum, production* and ending position (*LCIO only)
  - Calorimeter entrance point: position and momentum (SIO only)
  - Pointers to parent particles

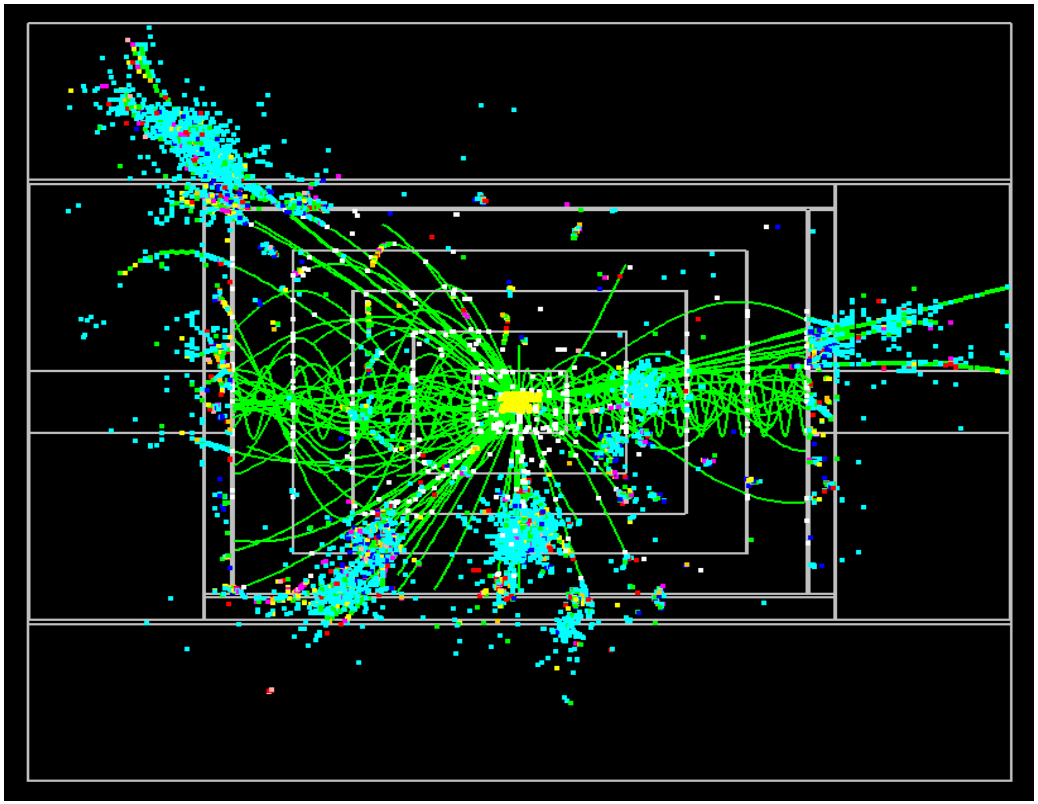
Zoom on the primary interaction

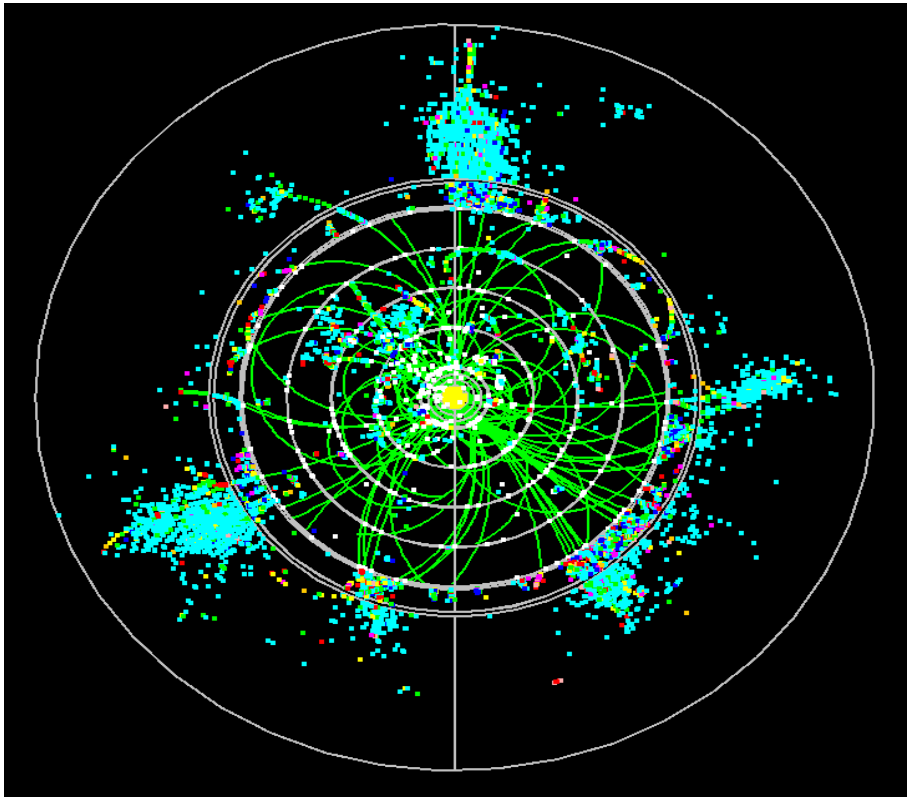
All decays forced in the generator are correctly processed by the LCDG4 simulation



Green: charged
Yellow: neutrals

e^+e^- into ttbar event (SDJan03)

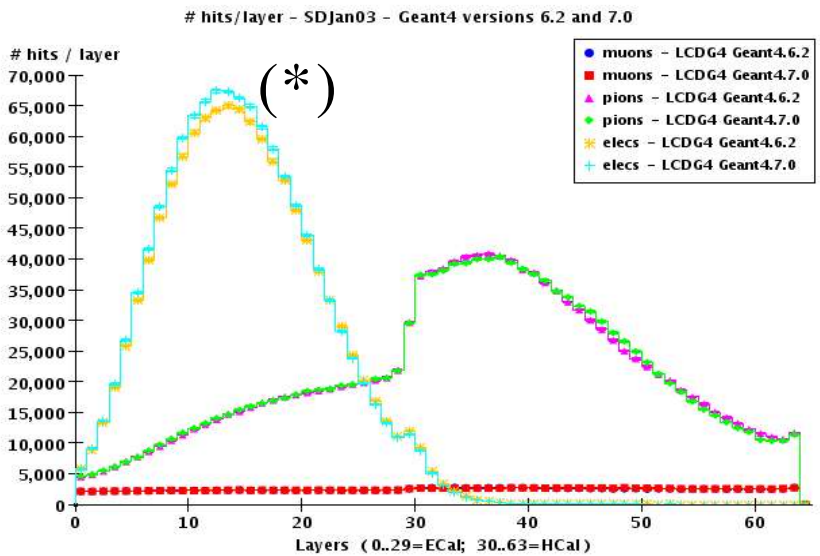


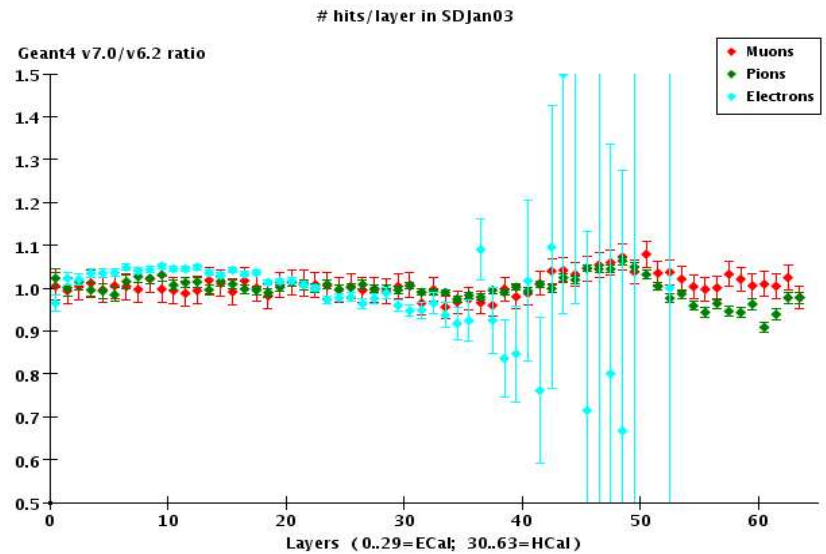


Simple analyses

- A comparison between Geant4.6.2 and Geant4.7.0, based on normally incident, 50GeV muons, electrons and pions
- A comparison of LCPHYS and LHEP physics lists, using Geant4.7.0 and the same single particle samples above

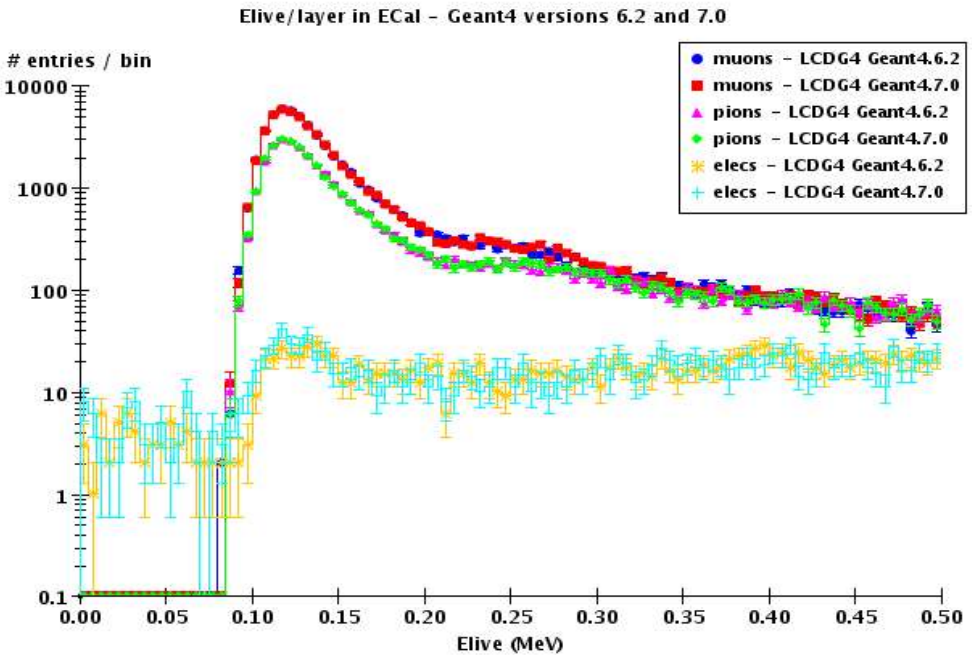
Shower profiles

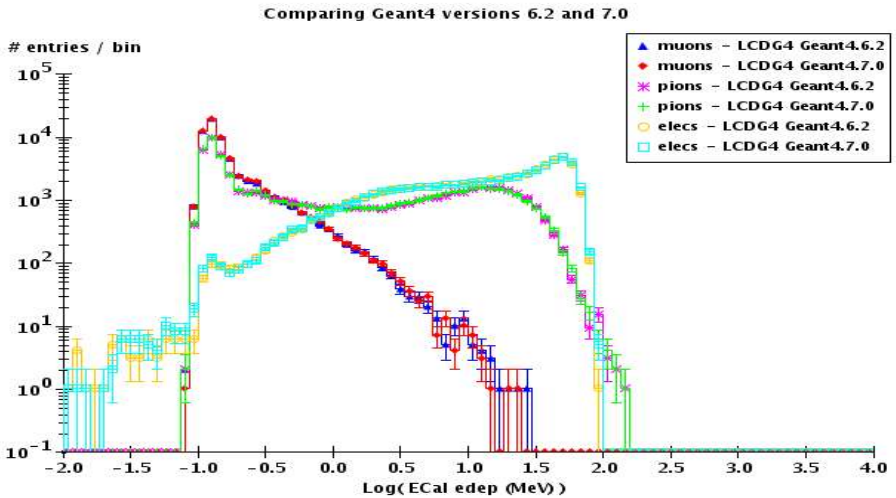


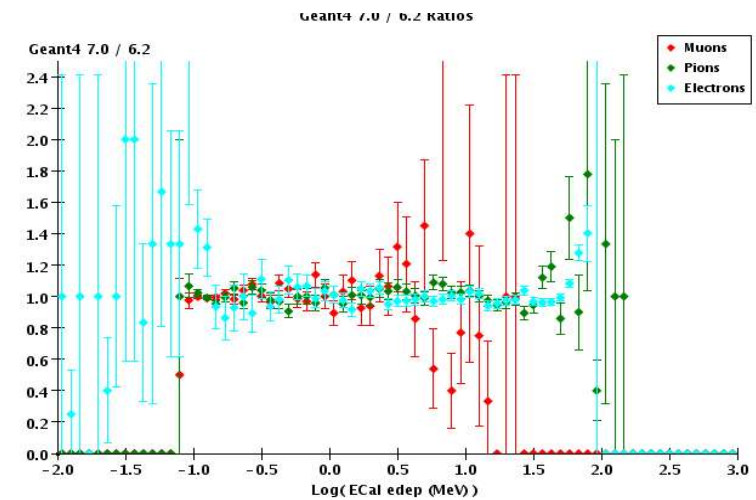


Geant4.6.2 and Geant4.7.0 are basically compatible within statistical uncertainties, with some exceptions (*)

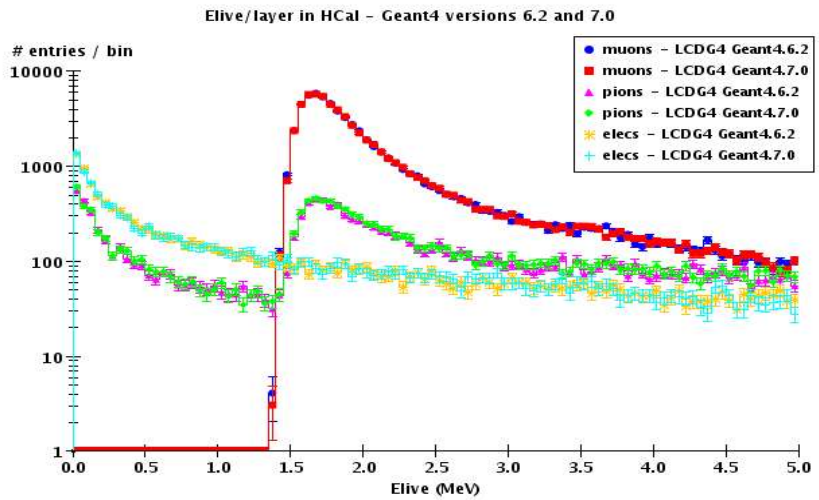
ECAL: Elive/layer

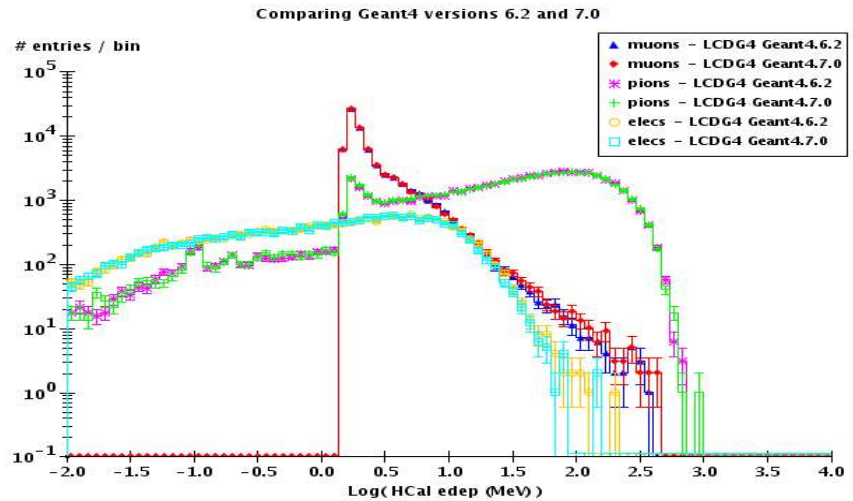


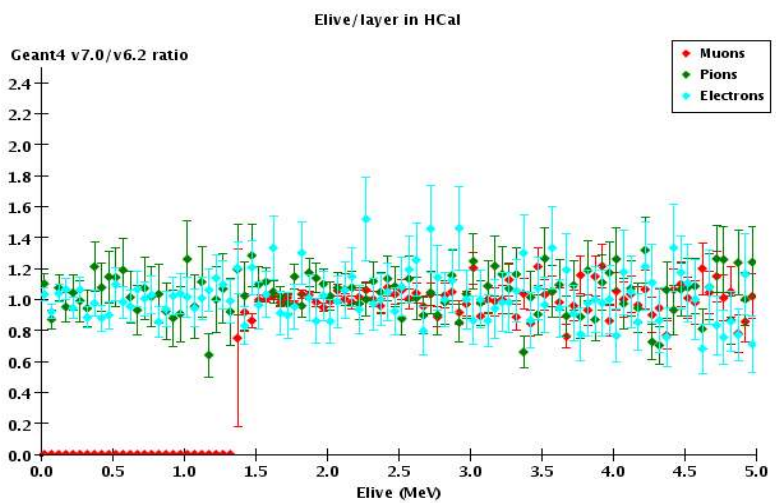


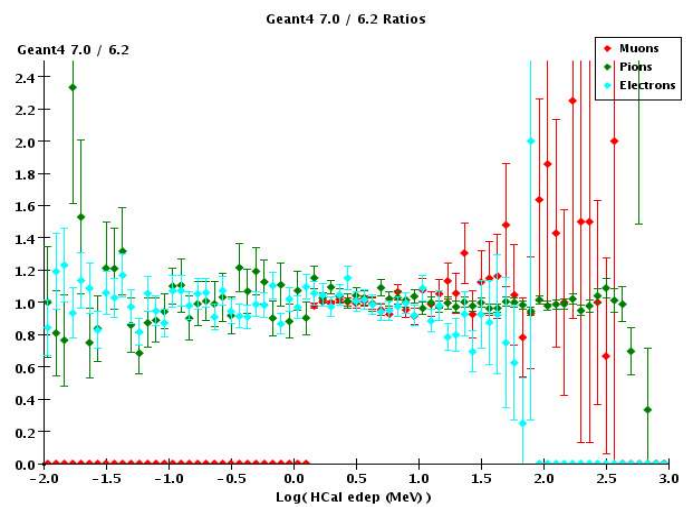


Hcal Elive/layer

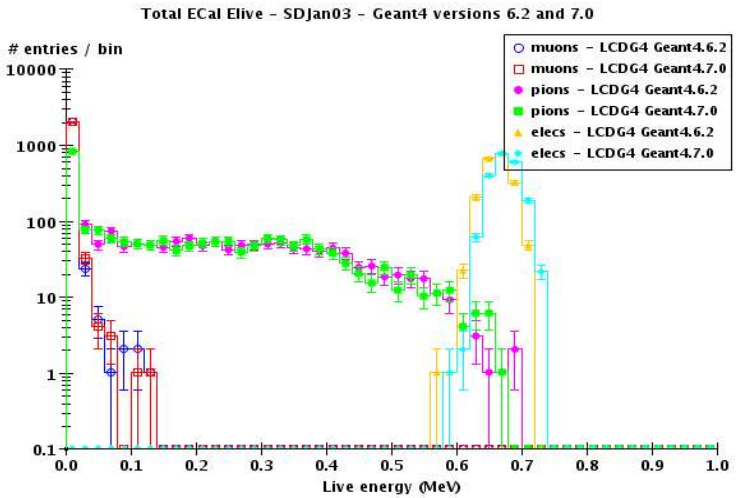


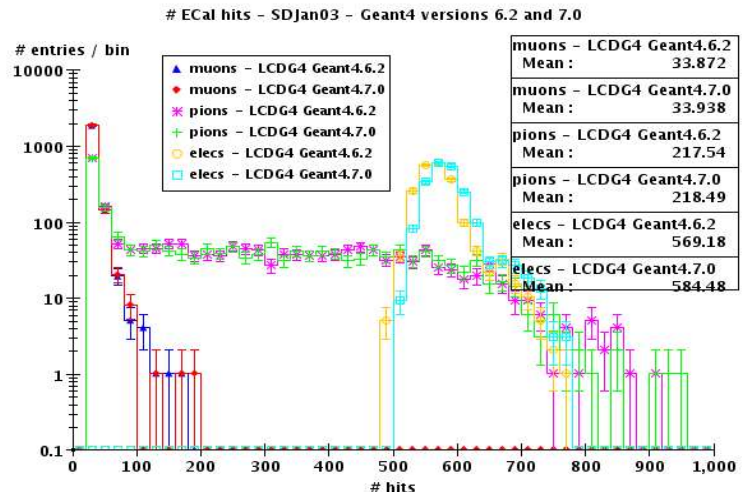


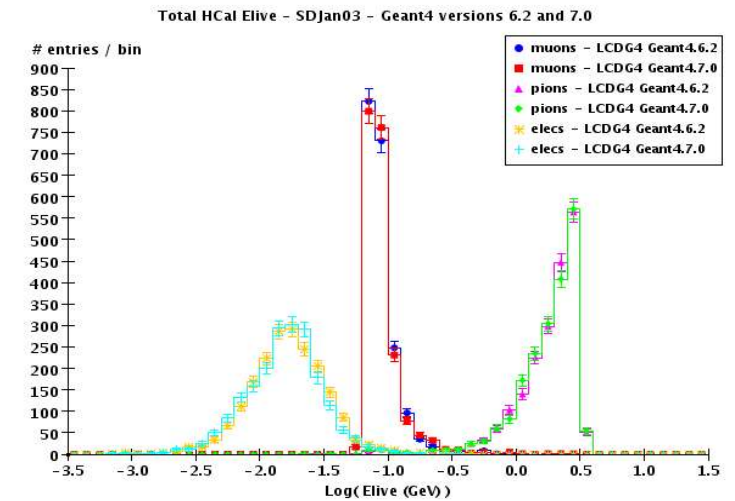


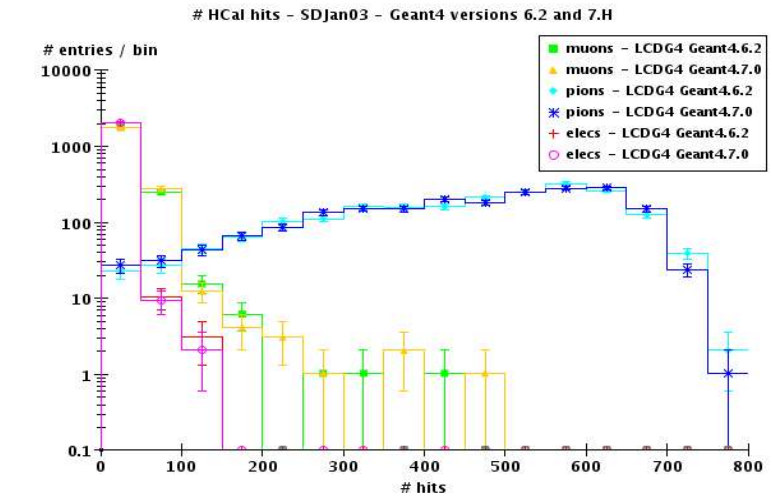


Ecal: Total Elive

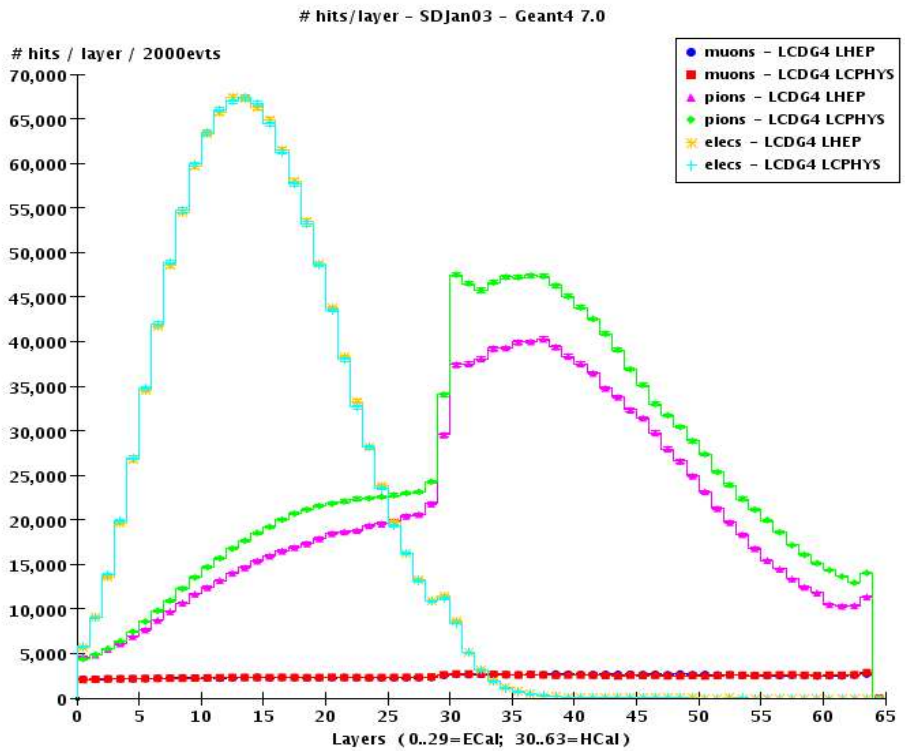


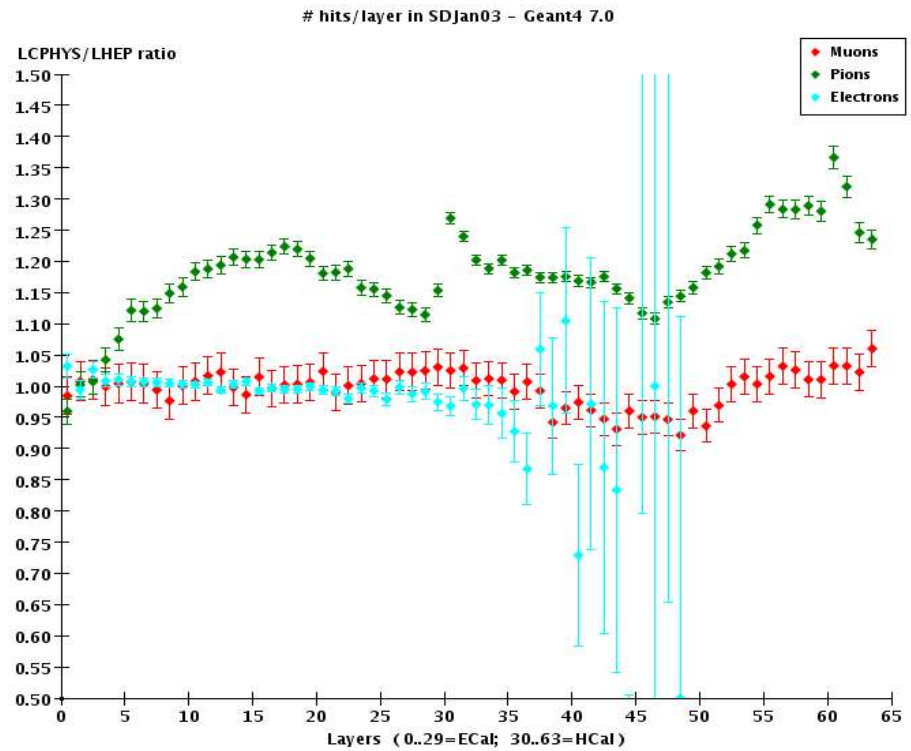




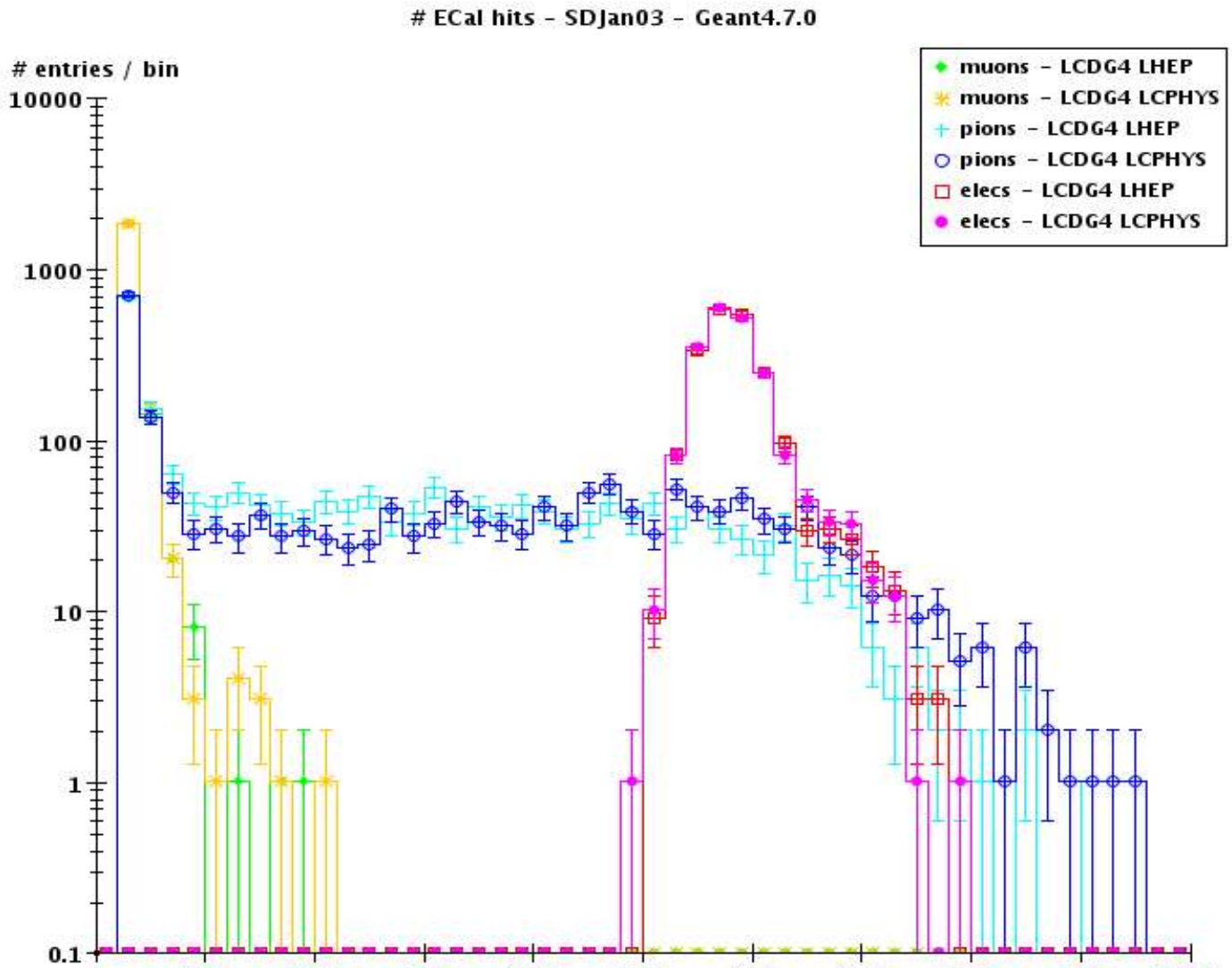


Shower profiles

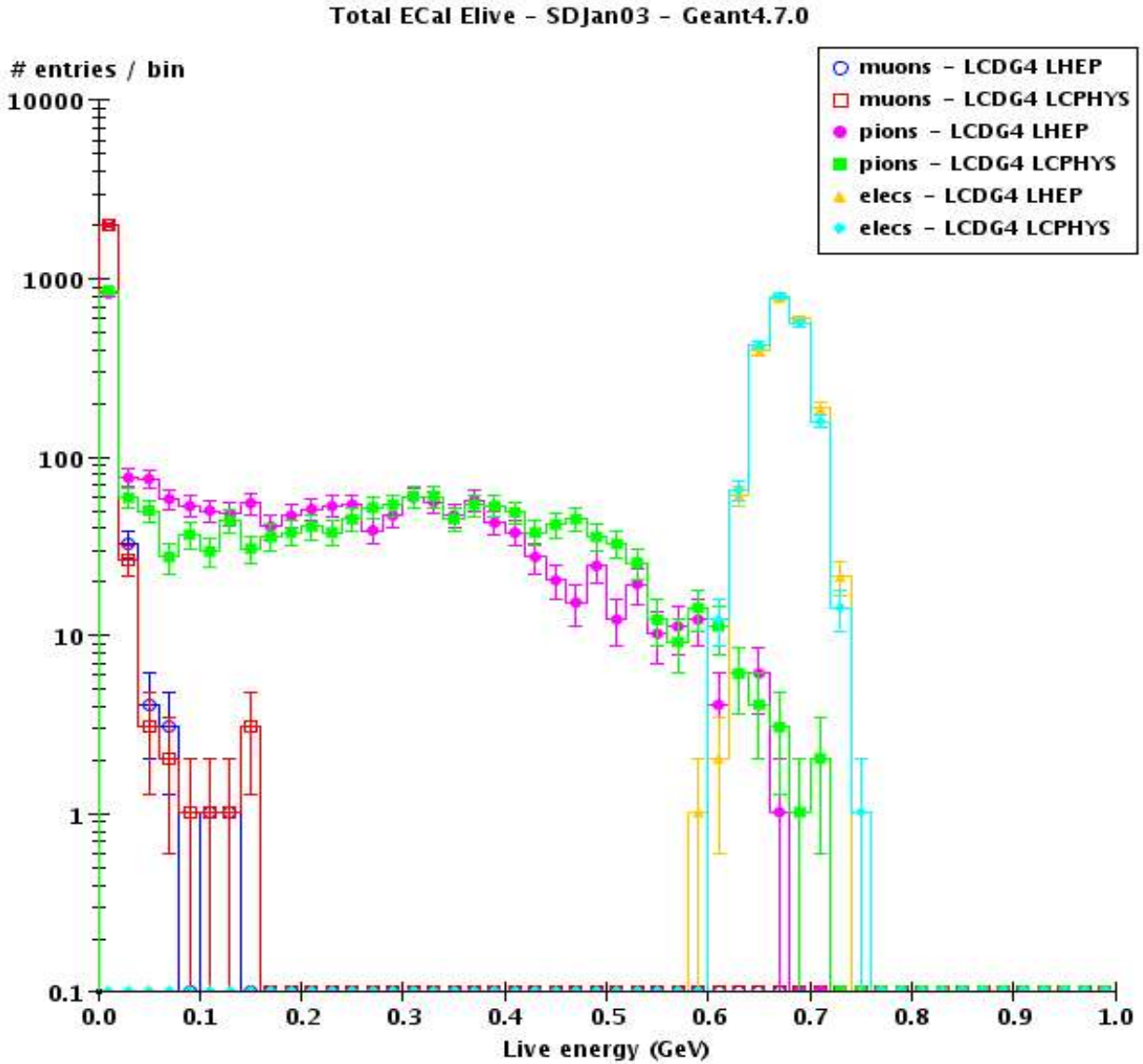




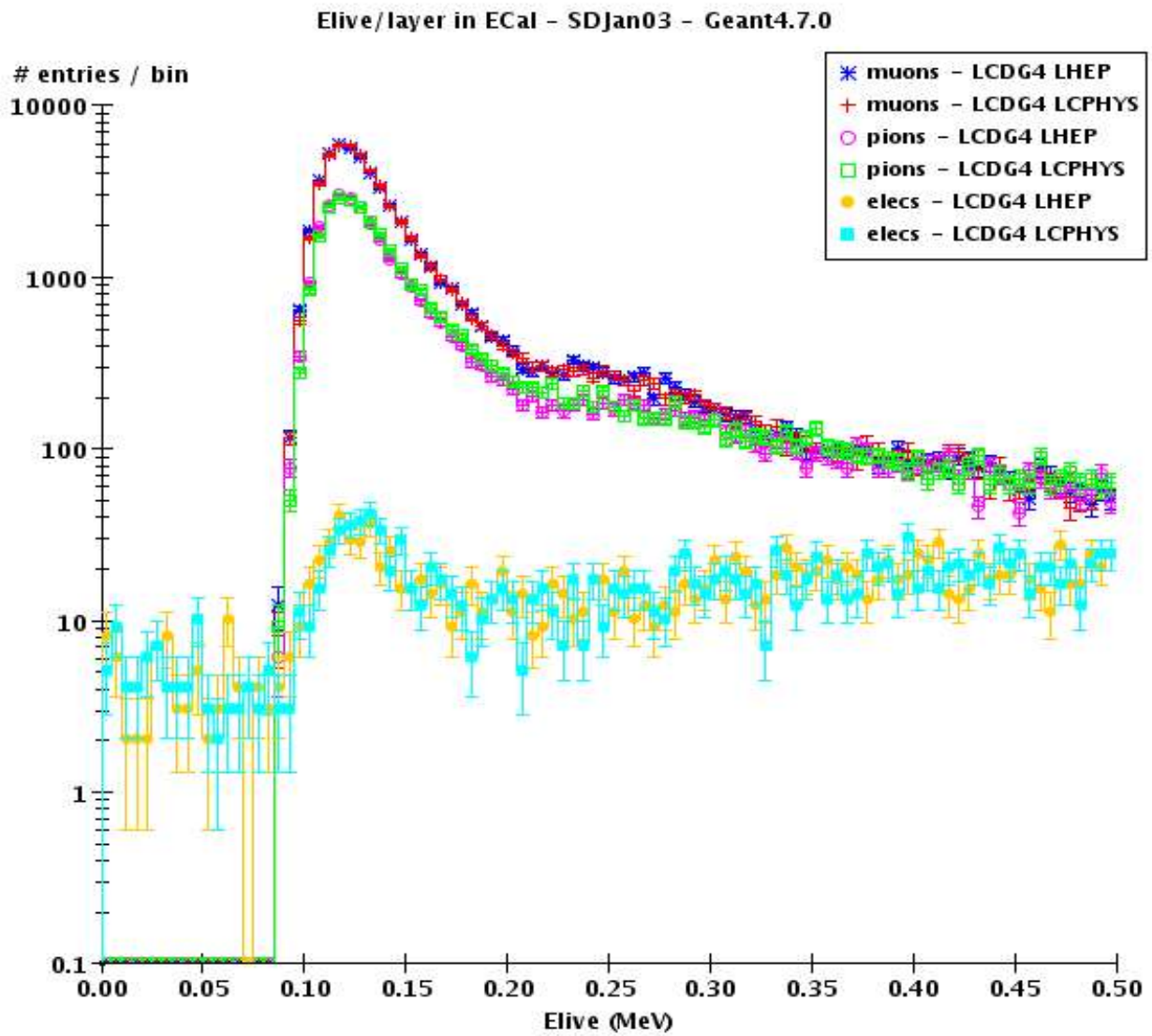
Ecal Nhits


ECal hits - SDJan03 - Geant4.7.0
entries / bin
10000
1000
100
10
1
0.1
muons - LCDG4 LHEP
muons - LCDG4 LCPHYS
pions - LCDG4 LHEP
pions - LCDG4 LCPHYS
elecs - LCDG4 LHEP
elecs - LCDG4 LCPHYS

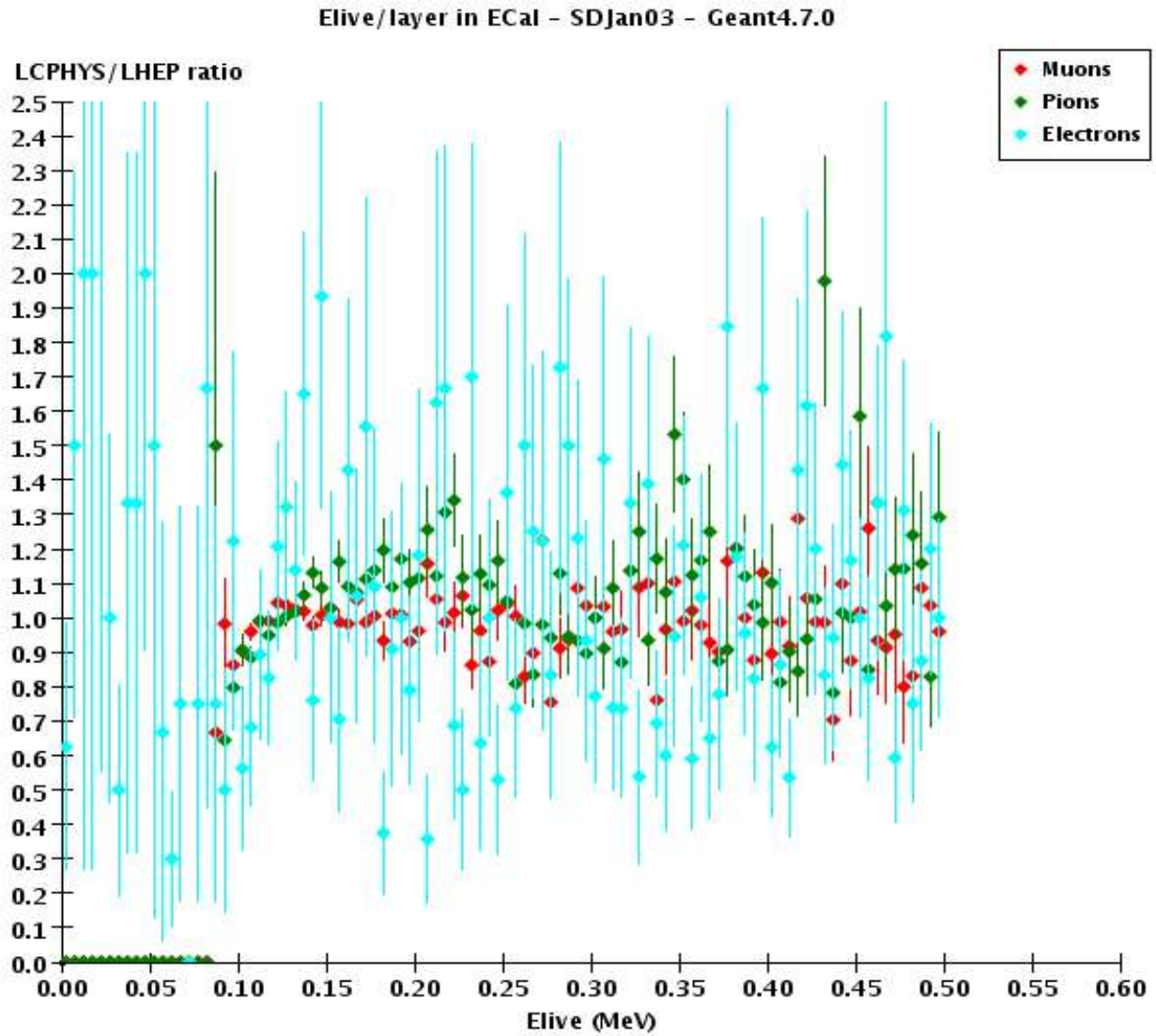
Ecal: total Elive



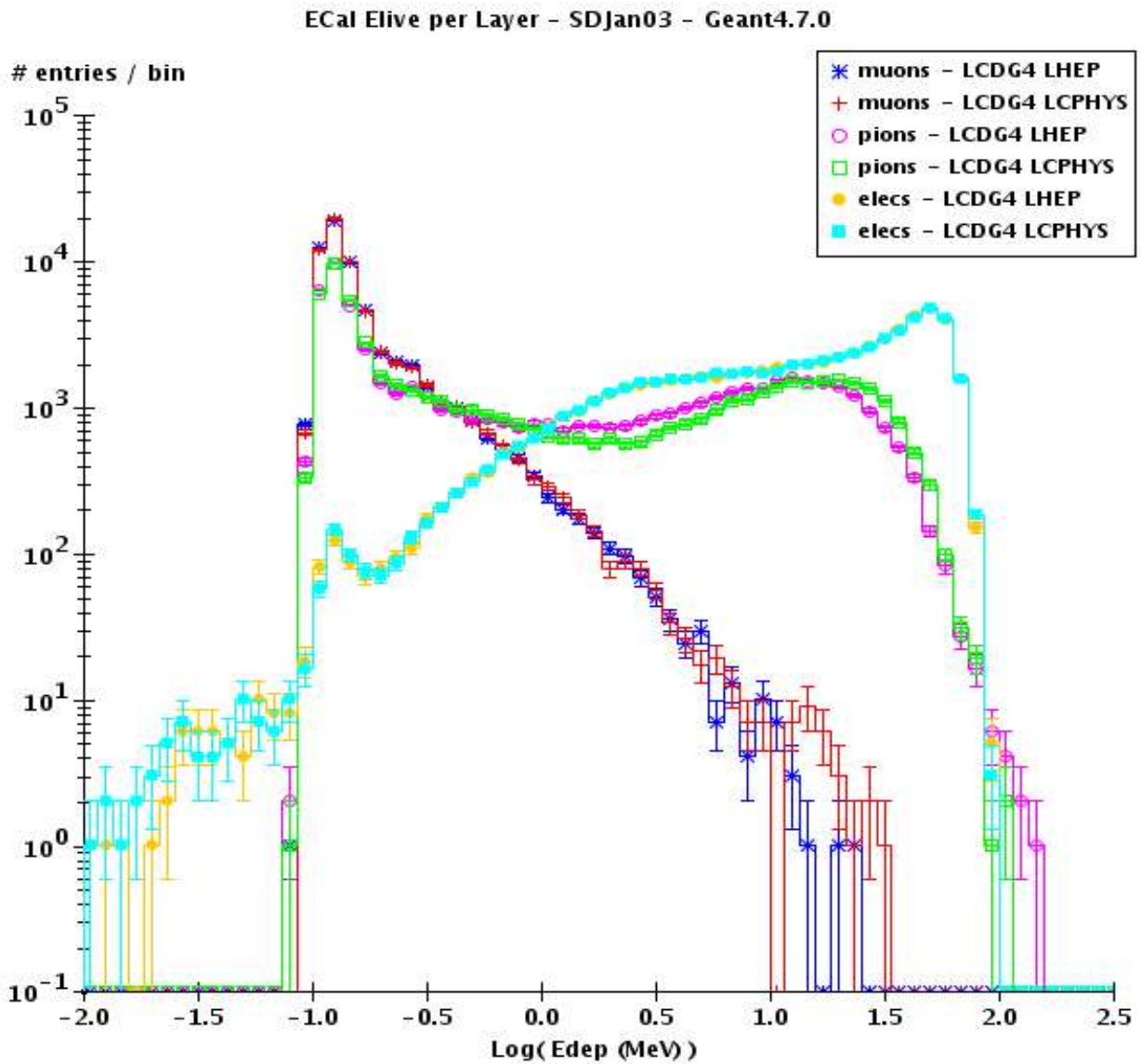
Ecal: Elive/layer



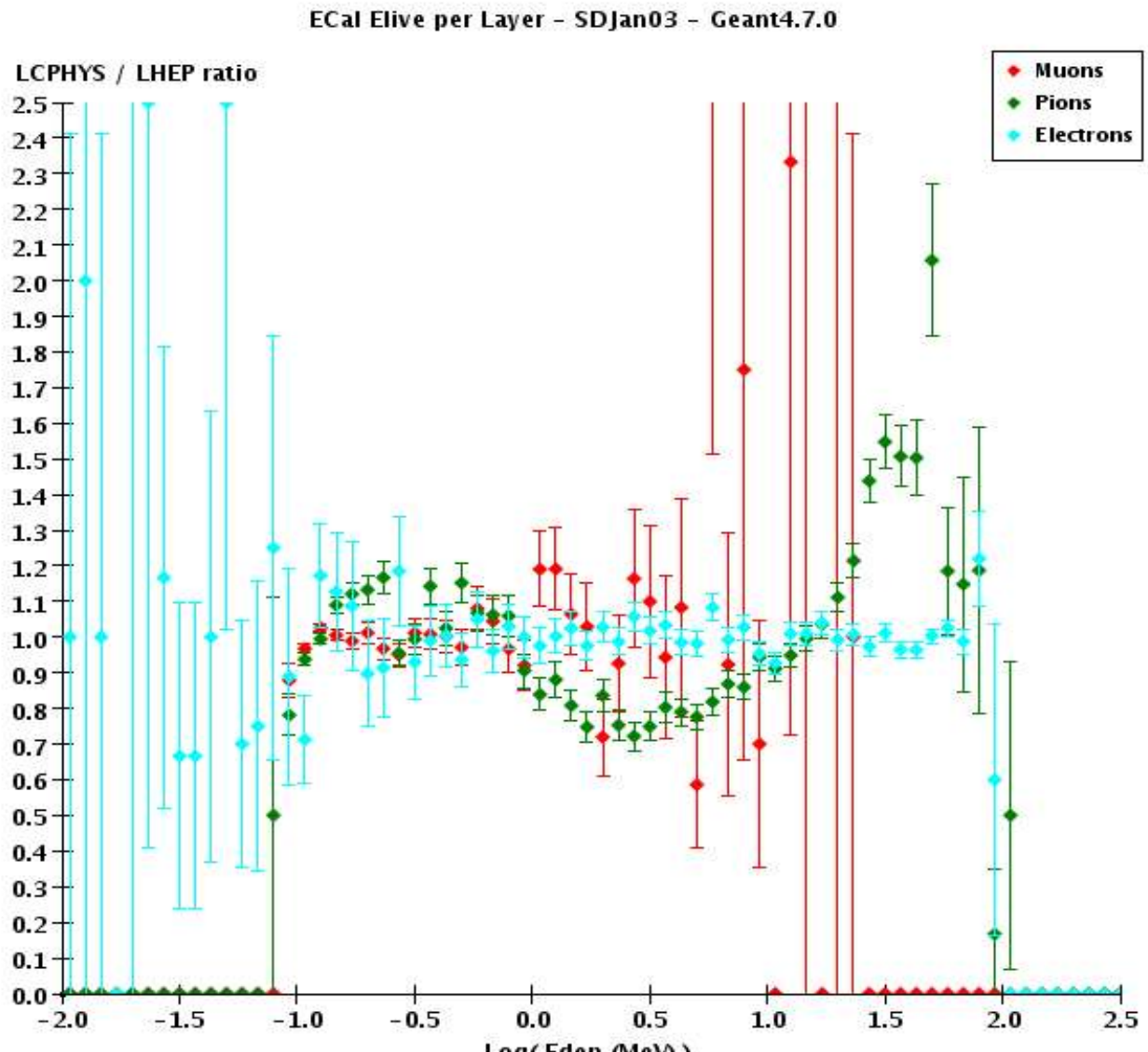
Ecal: Elive/layer



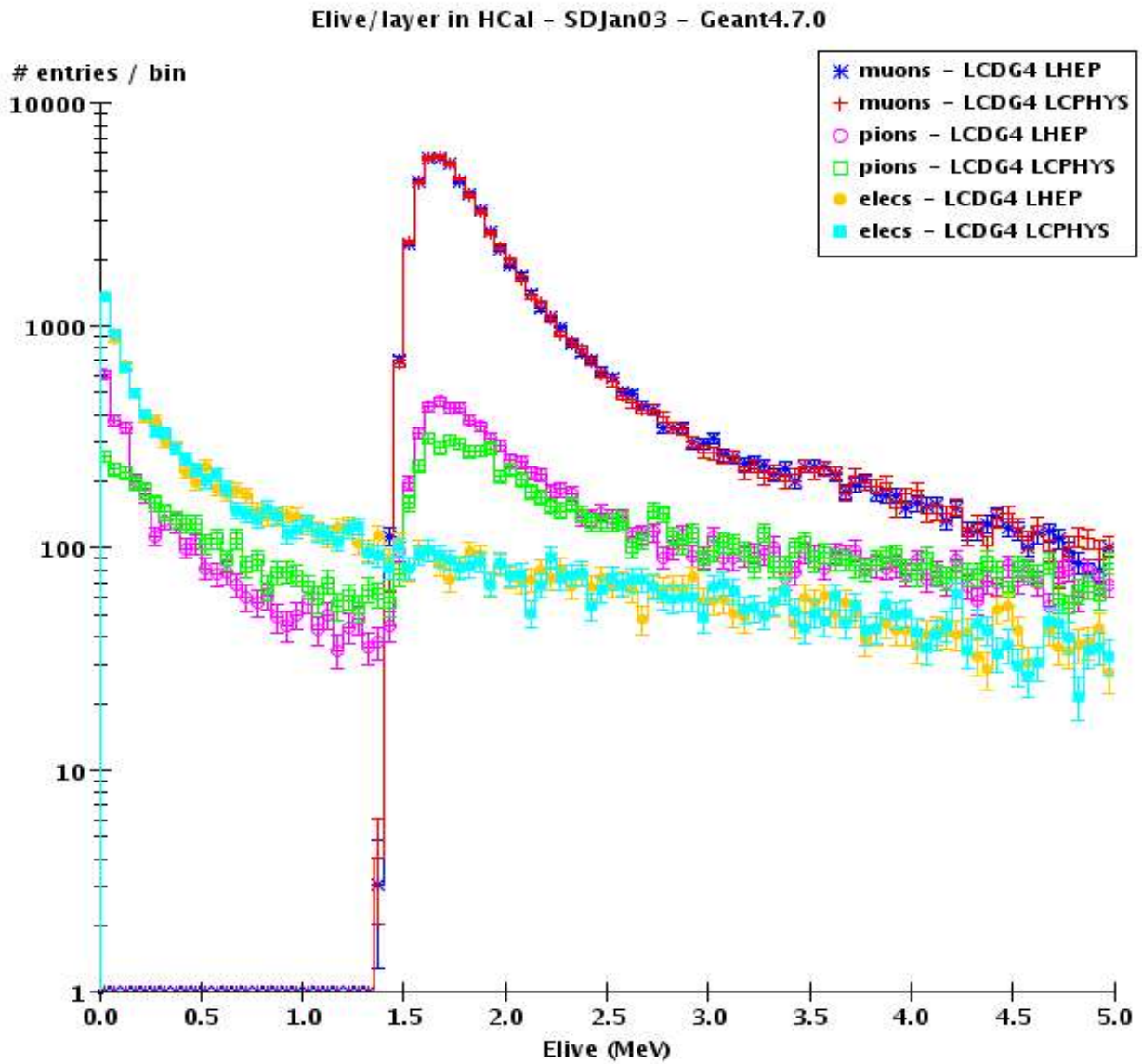
Ecal Elive/layer (log)



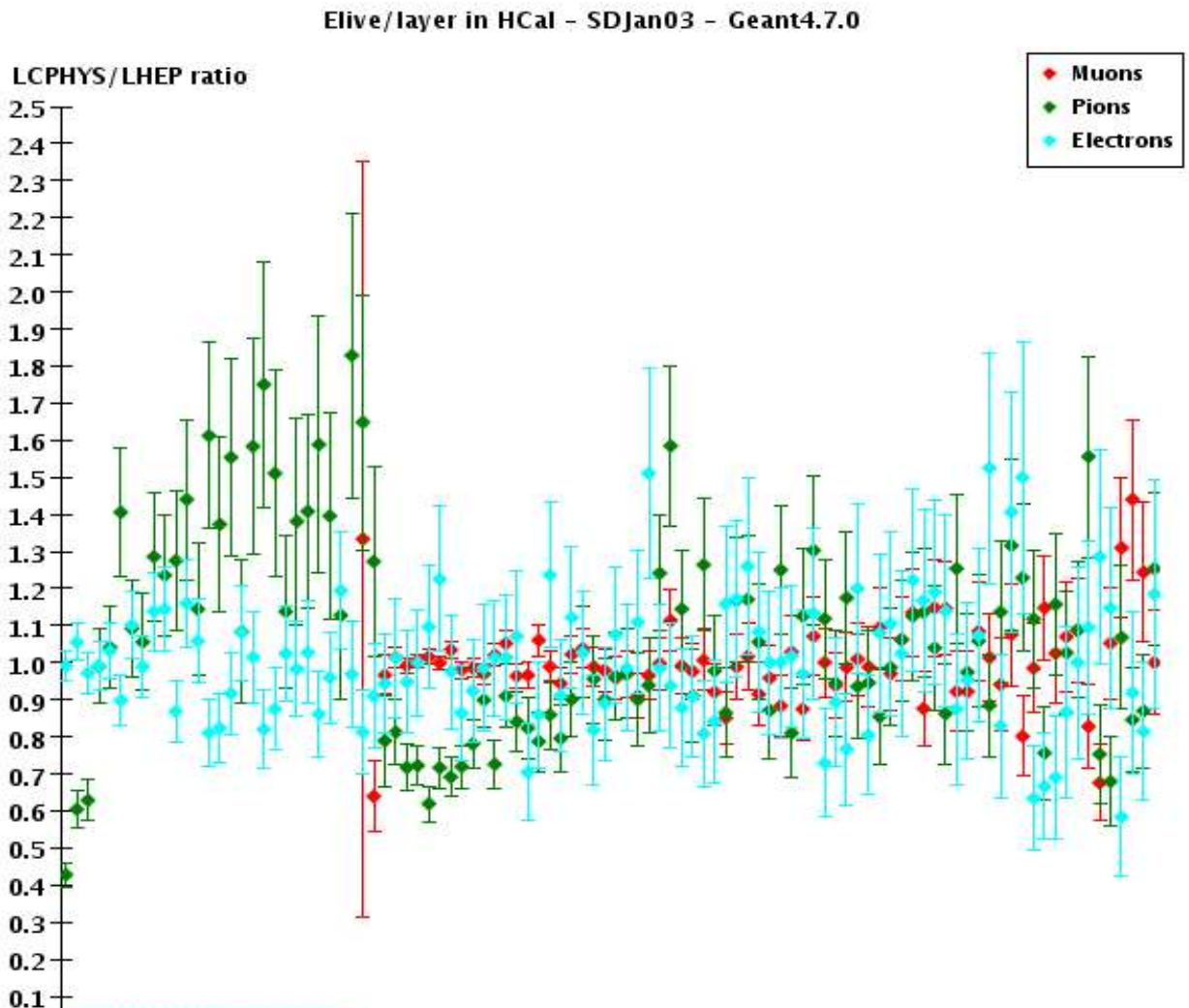
Ecal Elive/layer lcphys/lhep



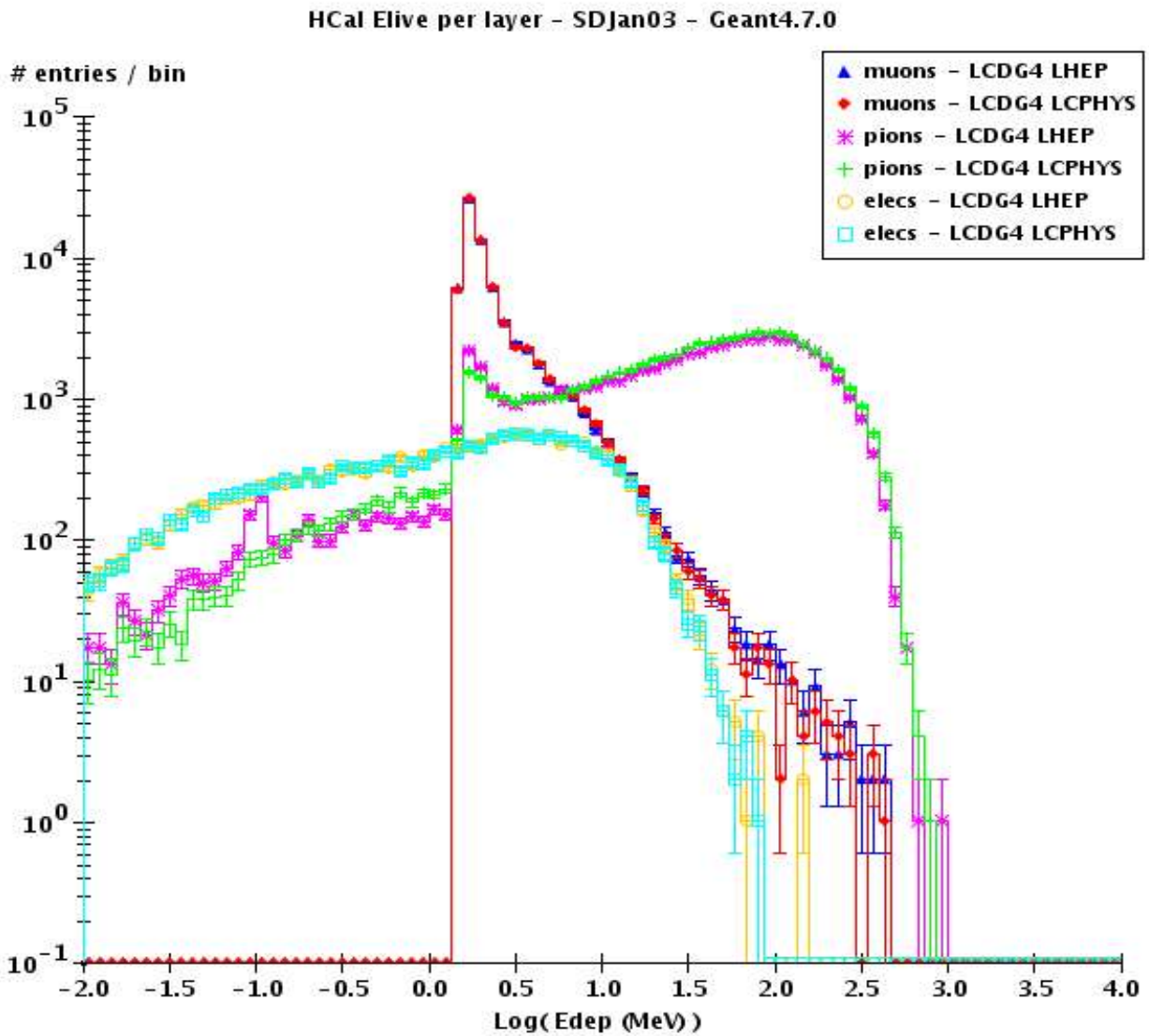
Hcal: elive/layer



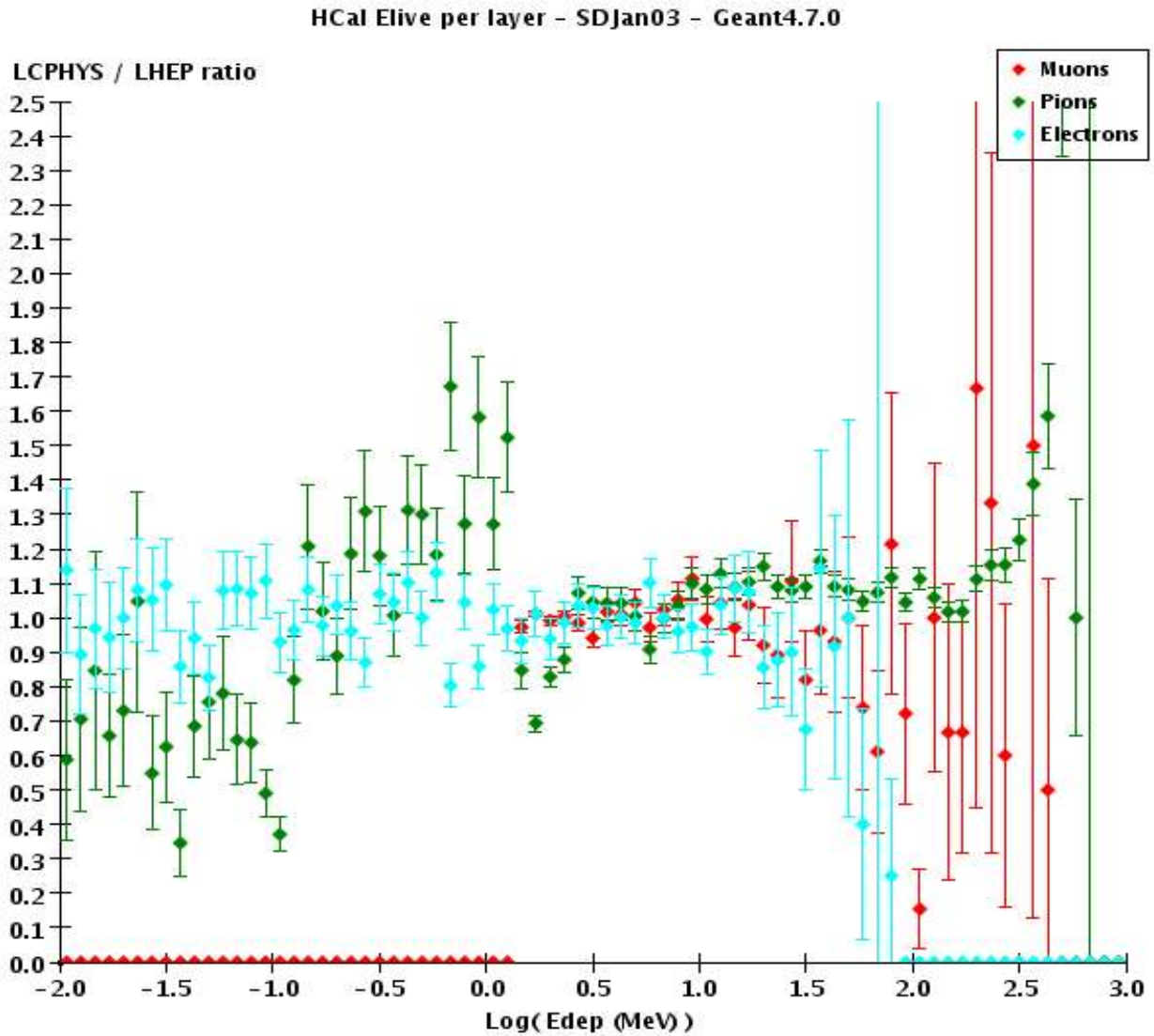
Hcal elive/layer



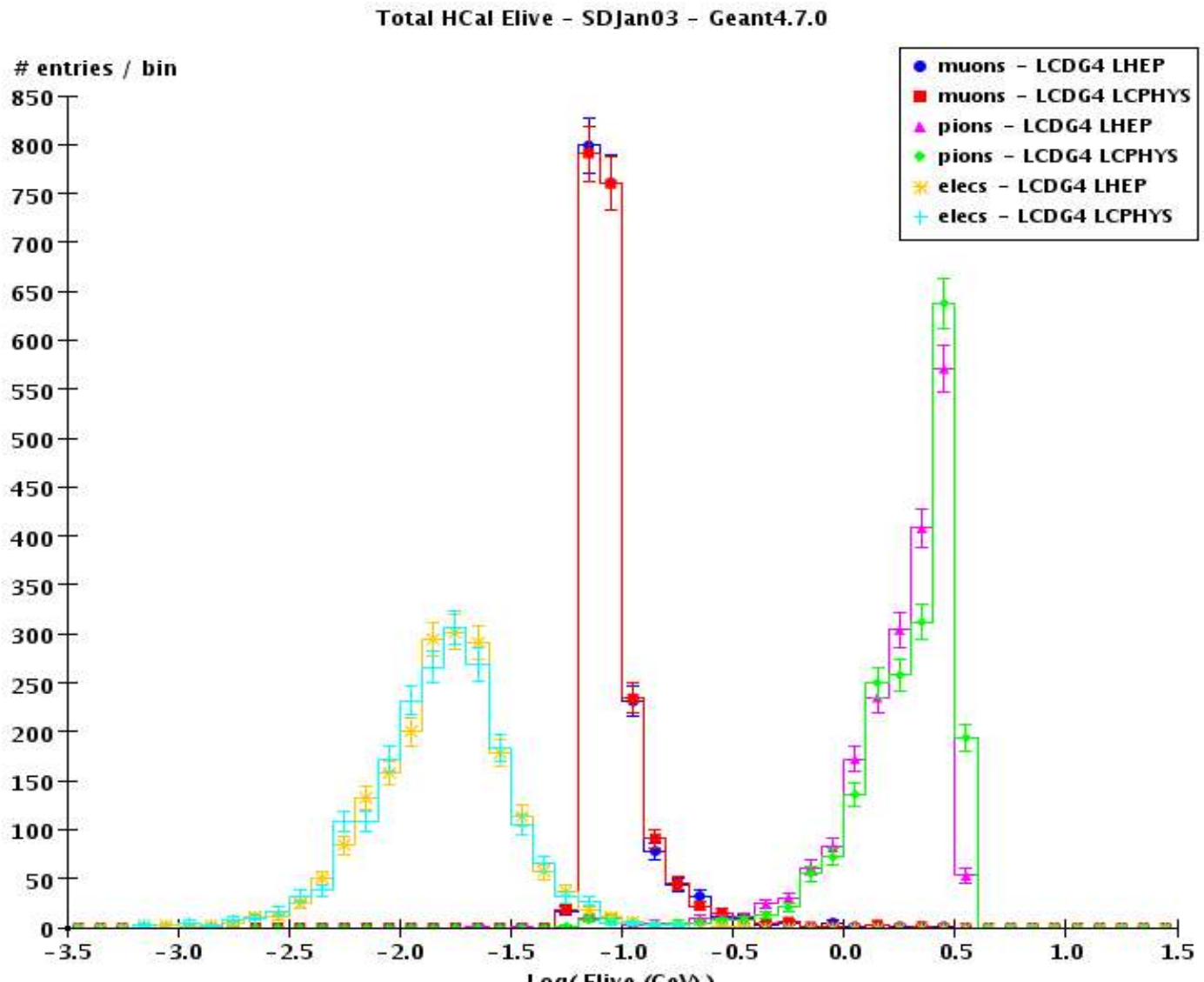
Hcal Elive/layer



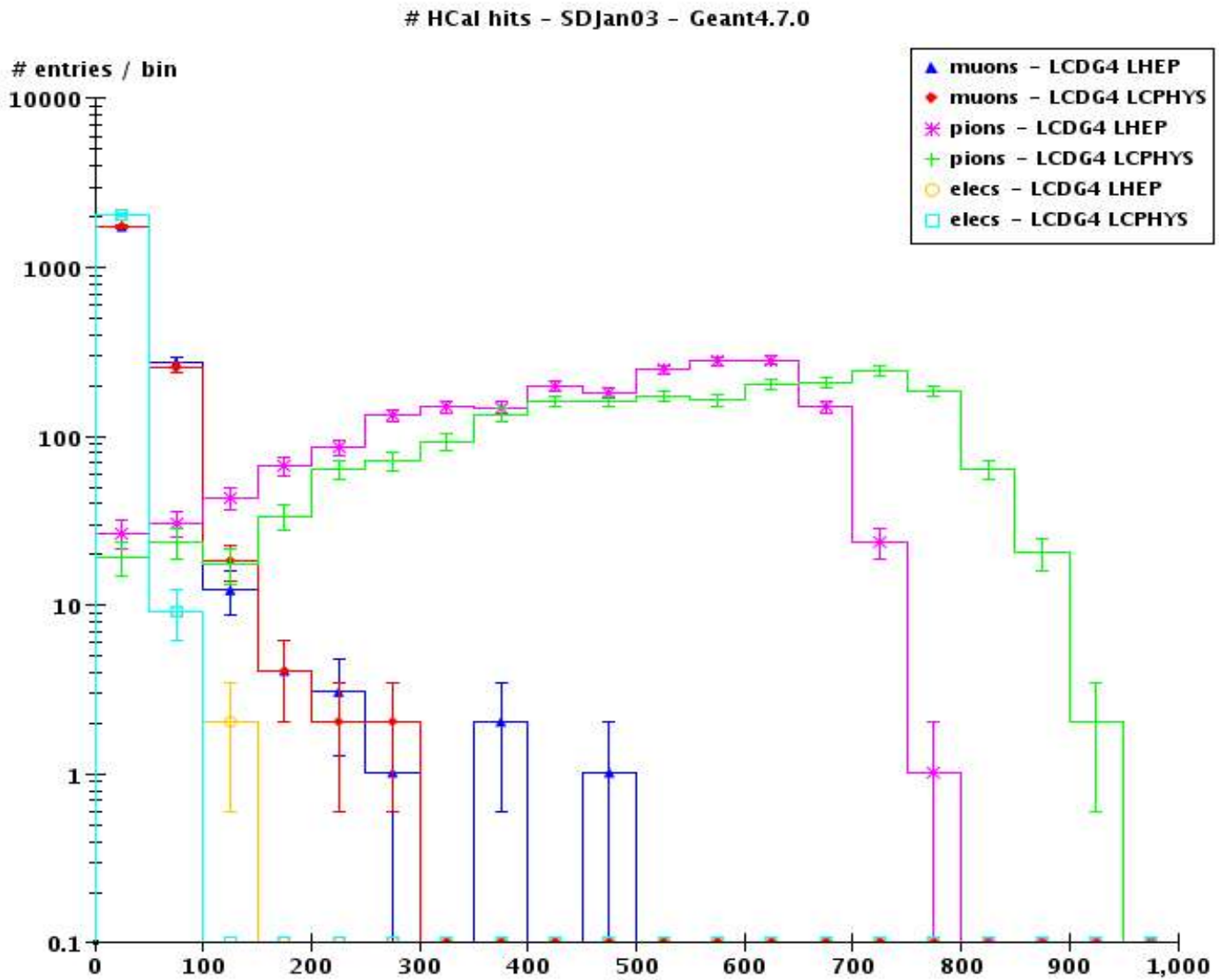
Hcal elive/layer ratio



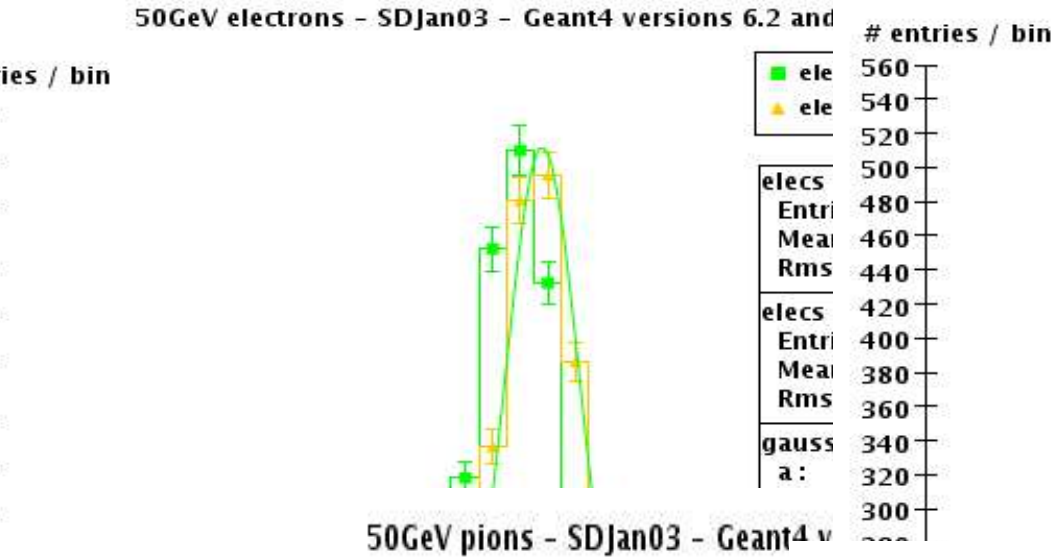
Hcal: total elive

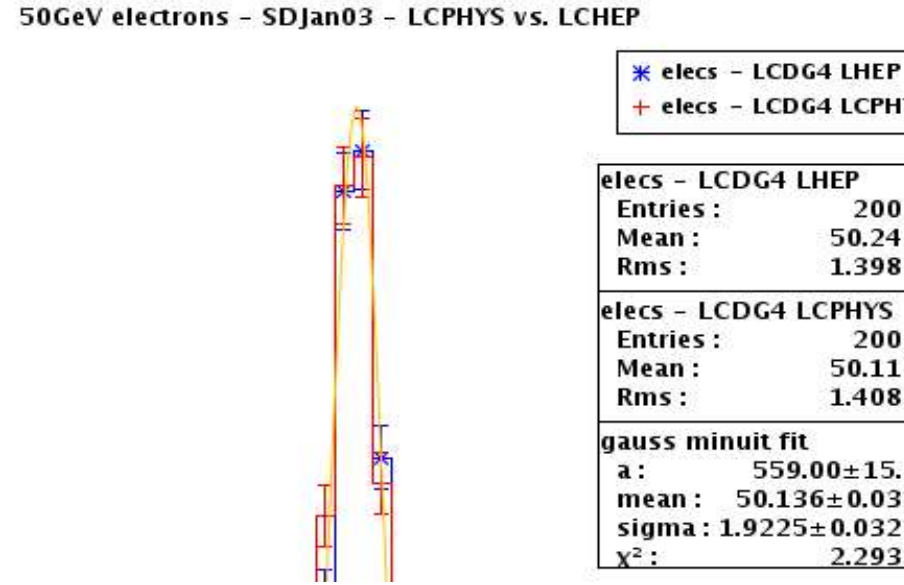


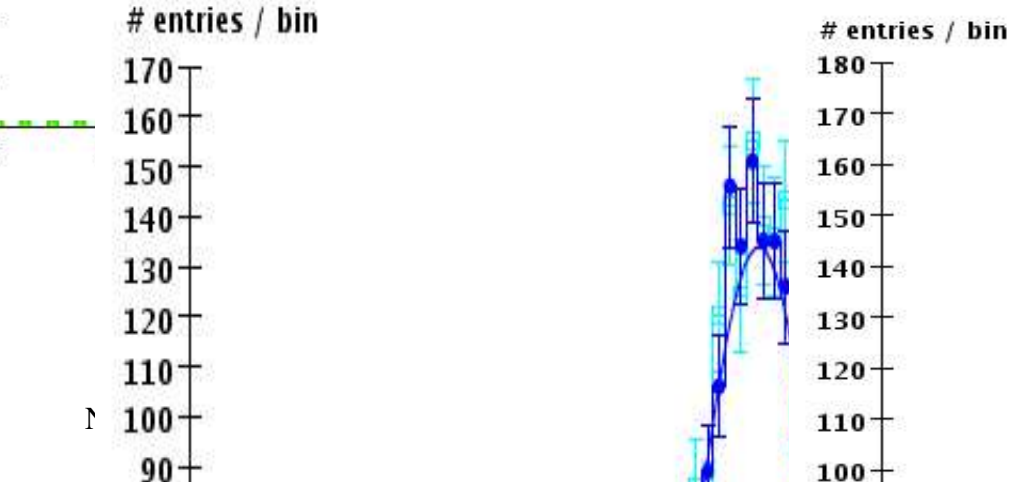
Hcal Nhits

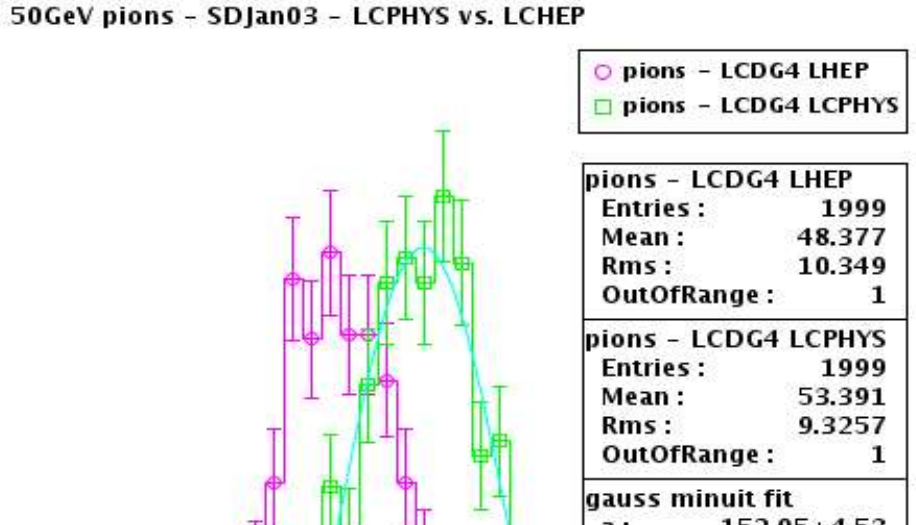

HCal hits - SDJan03 - Geant4.7.0
entries / bin
10000
1000
100
10
1
0.1
0
100
200
300
400
500
600
700
800
900
1,000
muons - LCDG4 LHEP
muons - LCDG4 LCPHYS
pions - LCDG4 LHEP
pions - LCDG4 LCPHYS
elecs - LCDG4 LHEP
elecs - LCDG4 LCPHYS

Reco Energy


50GeV electrons - SDJan03 - Geant4 versions 6.2 and
entries / bin
560
540
520
500
480
460
440
420
400
380
360
340
320
300
elecs
Entri
Mea
Rms
elecs
Entri
Mea
Rms
gauss
a :
50GeV pions - SDJan03 - Geant4 v


50GeV electrons - SDJan03 - LCPHYS vs. LCHEP
elecs - LCDG4 LHEP
elecs - LCDG4 LCPH
elecs - LCDG4 LHEP
Entries : 200
Mean : 50.24
Rms : 1.398
elecs - LCDG4 LCPHYS
Entries : 200
Mean : 50.11
Rms : 1.408
gauss minuit fit
a : 559.00±15.
mean : 50.136±0.03
sigma : 1.9225±0.032
χ² : 2.293


entries / bin
170
160
150
140
130
120
110
100
90
entries / bin
180
170
160
150
140
130
120
110
100


50GeV pions - SDJan03 - LCPHYS vs. LCHEP
pions - LCDG4 LHEP
pions - LCDG4 LCPHYS
pions - LCDG4 LHEP
Entries : 1999
Mean : 48.377
Rms : 10.349
OutOfRange : 1
pions - LCDG4 LCPHYS
Entries : 1999
Mean : 53.391
Rms : 9.3257
OutOfRange : 1
gauss minuit fit

MC Samples for general use

- Samples currently available at NIU through sftp:
  scpuser@k2.nicadd.niu.edu (lcd_2004): /pub/lima/lcdg4/v03-19
  - 2K each of $e^{\pm}$, $\mu^{\pm}$, $\pi^{\pm}$, γ, n at $\theta = 90^{o}$ and flat in φ
    energies = 2, 3, 5, 10, 15, 20, 30, 50 GeV
  - 5K 10 GeV K_S^0 into $\pi^0\pi^0$
  - 10K 10 GeV K_S^0 into $\pi^+\pi^-$
  - 5K 10 GeV Σ^+, inclusive decays
  - 5K 1..10GeV Λ^0, inclusive decays
  - 10K Z into (hadrons) at 91 GeV
  - 5K ttbar inclusive at 350 GeV
  - 10K WW into (hadrons)(any) at 500 GeV
  - 10K $\tau^+\tau^-$ at 500 GeV
  - 4K ZH into (any)(bbbar) at 500 GeV and M_H=120, 160 GeV (2K events each)

Both SDJan03 and SDNPHOct04 geometries, plus some variants

- Other samples can be requested to lima at fnal.gov. Please read http://nicadd.niu.edu/~jeremy/lcd/simreq/ for guidelines.

Other interesting samples available

- Some interesting samples have been produced with alternative geometries, and are also available through sftp:
- SDJan03-ANL in v03-06:
  Replacing the 2.cm steel absorbers (34 layers) with 0.7cm tungsten absorbers (60 layers)
  - 2K each of $\mu^{\pm}$, $\pi^{\pm}$ at $\theta = 90^{o}$ and flat in φ
    energies = 2, 3, 5, 10, 15, 20, 30, 50 GeV
  - 10K Z into (hadrons) at 91 GeV
- SDNPHOct04-BXX-EcalRminYYY in v03-16:
  Non-projective HCal barrels with Ecal inner radius varying from 127 to 164cm, and magnetic fields (B) ranging from 3.0 to 5.0 GeV, while keeping BR^2 = constant
  - 2K $\mu^{\pm}$ at $\theta = 90^{o}$ and flat in φ, energies = 2, 3, 5, 10, 15, 20, 30, 50 GeV
  - 2K e+e- into ZH into (qqbar)(bbbar), at 500 or 1000 GeV

How to access the MC samples

Several single-particle and physics data samples available from NIU data server using secure ftp:

```
% sftp scpuser@k2.nicadd.niu.edu
password: lcd_2004
sftp> cd pub/lima/lcdg4/
Sftp> ls   (to see a list of versions available)
Sftp> cd <some-version>
sftp> ls   (to see a list of files available)
sftp> mget muons-10gev*.slcio    (for example)
sftp> quit
%
```

See http://nicadd.niu.edu/~jeremy/admin/scp/index.html for more detailed access instructions, including instructions for windows winscp utility.

Some things to do

- Fixes to LCIO SimStatusCode (pending on LCIO to make more flags available)
- Non-projective fixed-size calorimeter cells for endcaps
- Simplify MCParticle handling (more portable)

LCDG4 status summary

- Only cylinders, disks and cones supported by current LCDG4 version (like in Gismo).
  More realistic geometries to be implemented in another package (SLIC). Work is already under way.
- Visualization framework within JAS for .sio or .slcio files
- Source code available from SLAC or NIU CVS repositories
- Several MC physics samples have been generated for algorithm development and studies (both SIO or LCIO formats are available)
- For more information please check the LCDG4 documentation web page: http://nicadd.niu.edu/~lima/lcdg4/, or under subdirectory doc of CVS module

Talk Outline

- LCDG4
  - Status
  - Simple studies
  - Plans

Summary

- Geant4-based full detector simulation tools are in place
- LCDG4 replaces Gismo for output in SIO format (ALCPG)
- LCDG4 and LCS (XML-based), as well as Mokka (MySQL-based) can be used for standard LCIO format
- Many geometry options can be easily checked with LCDG4
- Data samples available for general use at NIU server

  Anonymous access via ssh / scp:

  scpuser@k2.nicadd.niu.edu (pwd=lcd_2004): /pub/lima/lcdg4/v03-06

  See http://nicadd.niu.edu/~jeremy/admin/scp/ for detailed access instructions.

DigiSim
A digitization simulation package for the ILC

Guilherme Lima
for the ILC group at NIU

LC-Cal Meeting
December 20, 2004

Talk Outline

- DigiSim: Purpose and requirements
- Package description
- Usage and configuration
- Developing new functionality
- Non-physics results
- Current status and summary

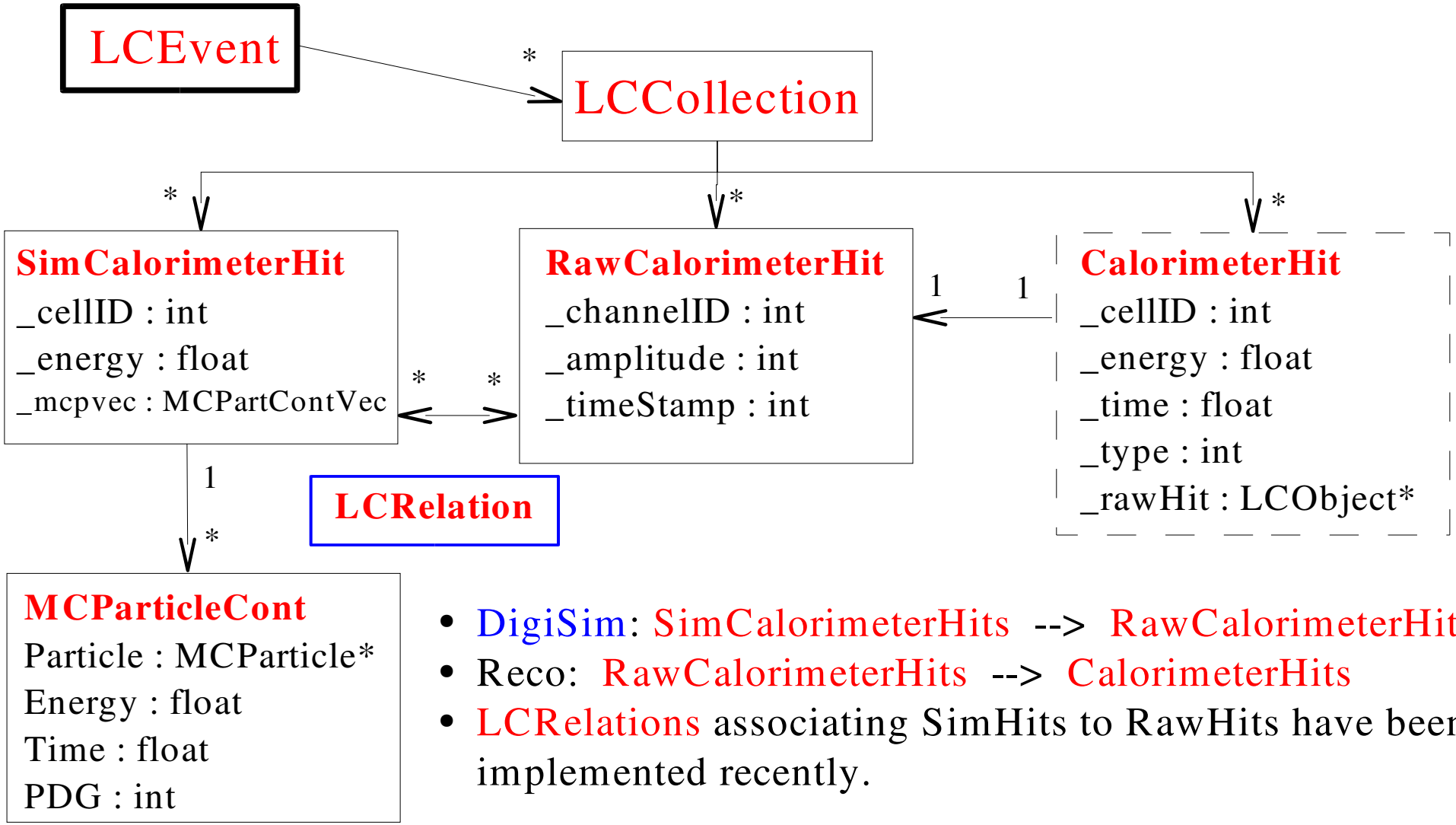
DigiSim

- Goal: a program to simulate the digitization process for the ILC detector simulation
- DigiSim role is to convert the simulated data (energy depositions and hit timings) into the same format AND *as close as possible* to real data from readout channels.
- Same reco / analysis software can be used for MC and real data.
- *As close as possible* means that all known effects from digitization process should be taken into account, if possible cell ganging, inefficiencies, noise, crosstalks, hot and dead channels, non-uniformities, etc.

Requirements and choices

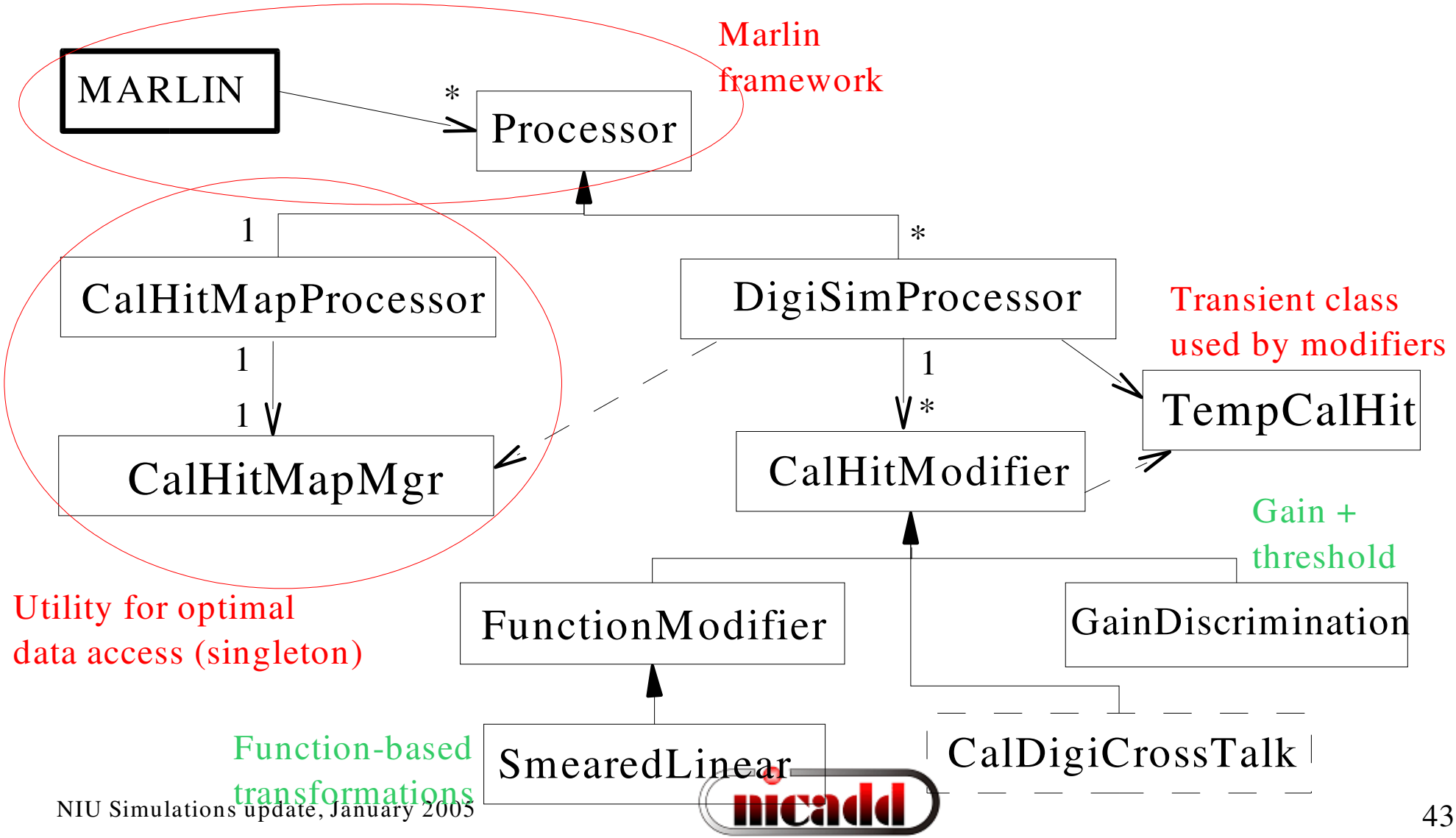
- Basic requirements:
  - Object oriented design to simplify maintenance and implementation of new functionality
  - Should be used within the CALICE test beam project, as a testbed for the reconstruction framework
  - All test beam code based on C++ and LCIO
- Gaede's Marlin (v0.6) chosen as the C++ framework

LCIO event model

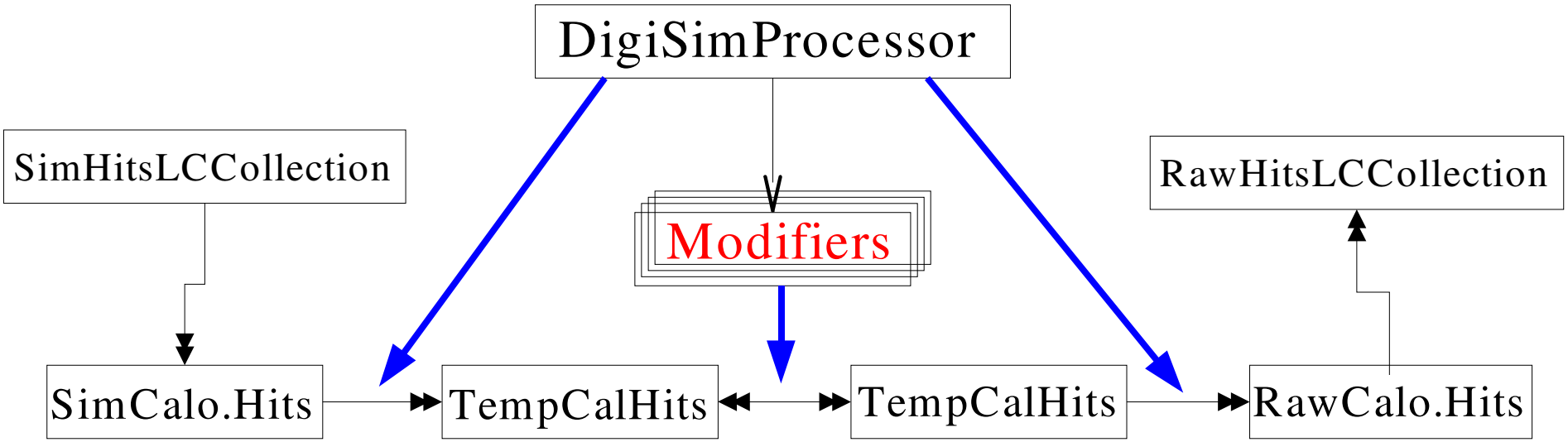


- DigiSim: SimCalorimeterHits --> RawCalorimeterHits
- Reco: RawCalorimeterHits --> CalorimeterHits
- LCRelations associating SimHits to RawHits have been implemented recently.

DigiSim class diagrams



DigiSim event loop



- Calorimeter hits are shown here, but the same structure could also be used for tracker hits
- TempCalHits are both input and output to each modifier
- All processing is controlled by a DigiSimProcessor (one per subdetector)
- Modifiers are configured at run time, via the Marlin steering file
- New modifiers can be easily created for new functionality (more info later)

Steering file example

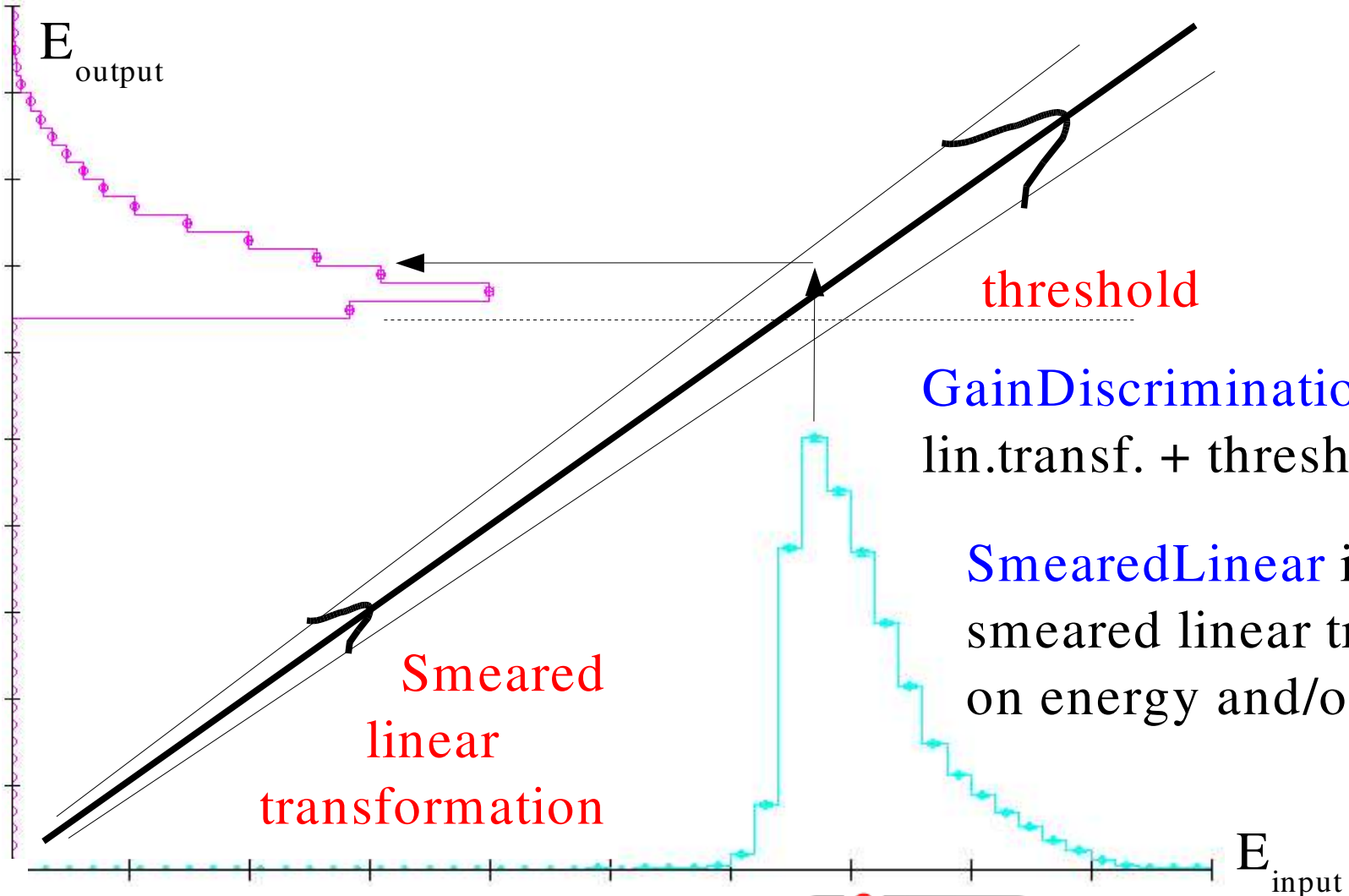
```
##########################################
#  Example steering file for DigiSim
##########################################
.begin Global  ----------------------------------------
# specify one or more input files (in one or more lines)
LCIOInputFiles inputfile

# the active processors that are called in the given order
ActiveProcessors CalHitMapProcessor
ActiveProcessors EMDigiSimProcessor
ActiveProcessors HADDigiSimProcessor
ActiveProcessors TCDigiSimProcessor
ActiveProcessors OutputProcessor

# limit the number of processed records (run+evt):
MaxRecordNumber 500
.end Global  ----------------------------------------
```

Configuring processors and modifiers

```
##################################################################
# A DigiSim processor.  It instantiates one or more calorimeter hit
#  "modifiers", which together represent the full digitization
process
.begin EMDigiSimProcessor
-----------------------------------------------
ProcessorType     DigiSimProcessor
InputCollection   EMcalCollection
OutputCollection  EMrawCollection

ModifierNames     EMFixedGain EMThreshOnly EMDigiIdent

# Parameters:  type                 gainNom gainSig  thresh   thrSig
EMFixedGain    GainDiscrimination  1000000       0       0        0
EMThreshOnly   GainDiscrimination        1       0      30        0
EMGainThresh   GainDiscrimination  1000000   50000      30      1.5

# A function-based modifier        ElinNom ElinSig TlinNom  TlinSig
EMDigiIdent    SmearedLinear             1       0       1        0
.end ------------------------------------------------------------------
```

(As many as needed)

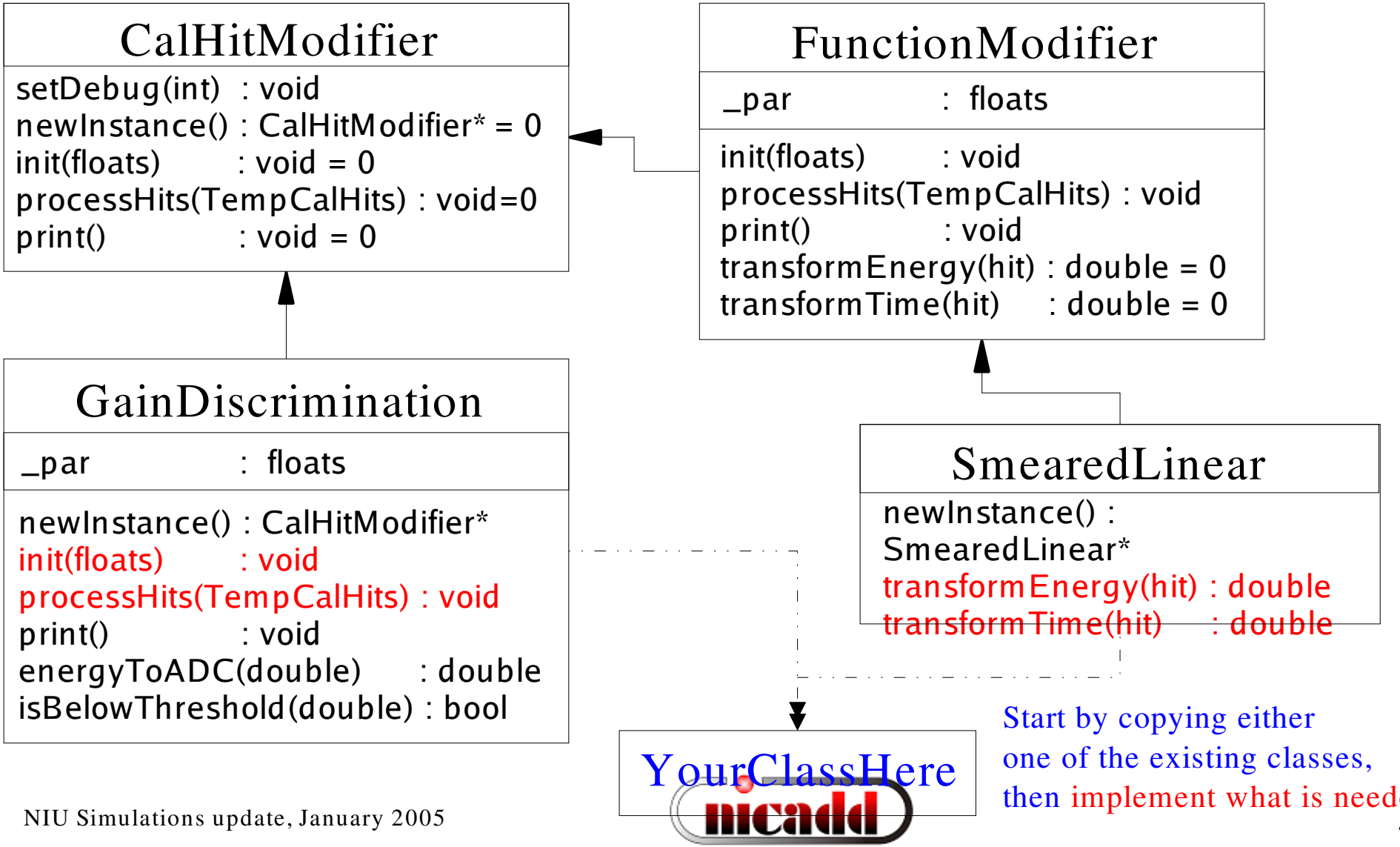
Existing modifiers



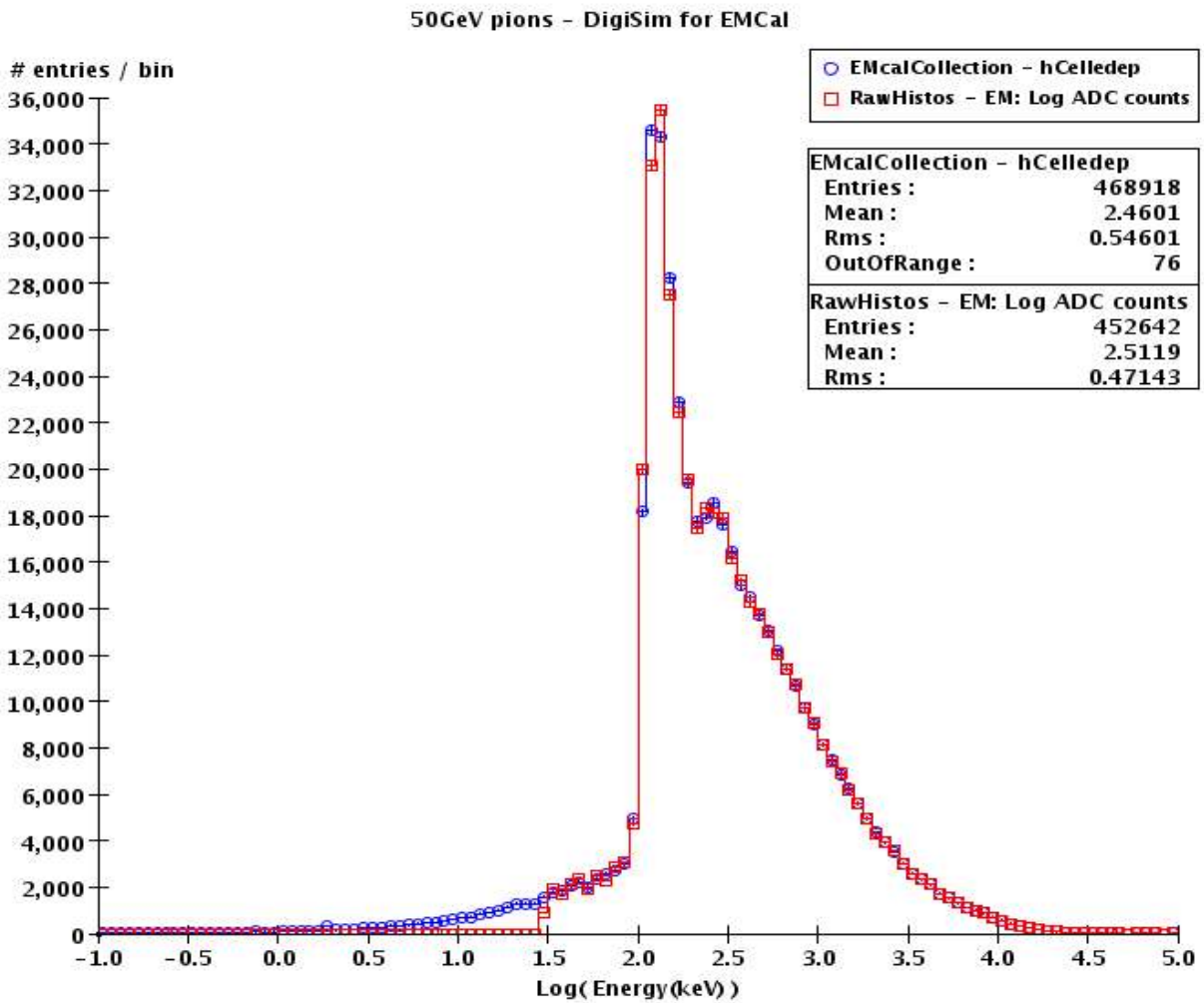
GainDiscrimination is a smeared lin.transf. + threshold on energy

SmearedLinear is a func-based smeared linear transformation on energy and/or timing

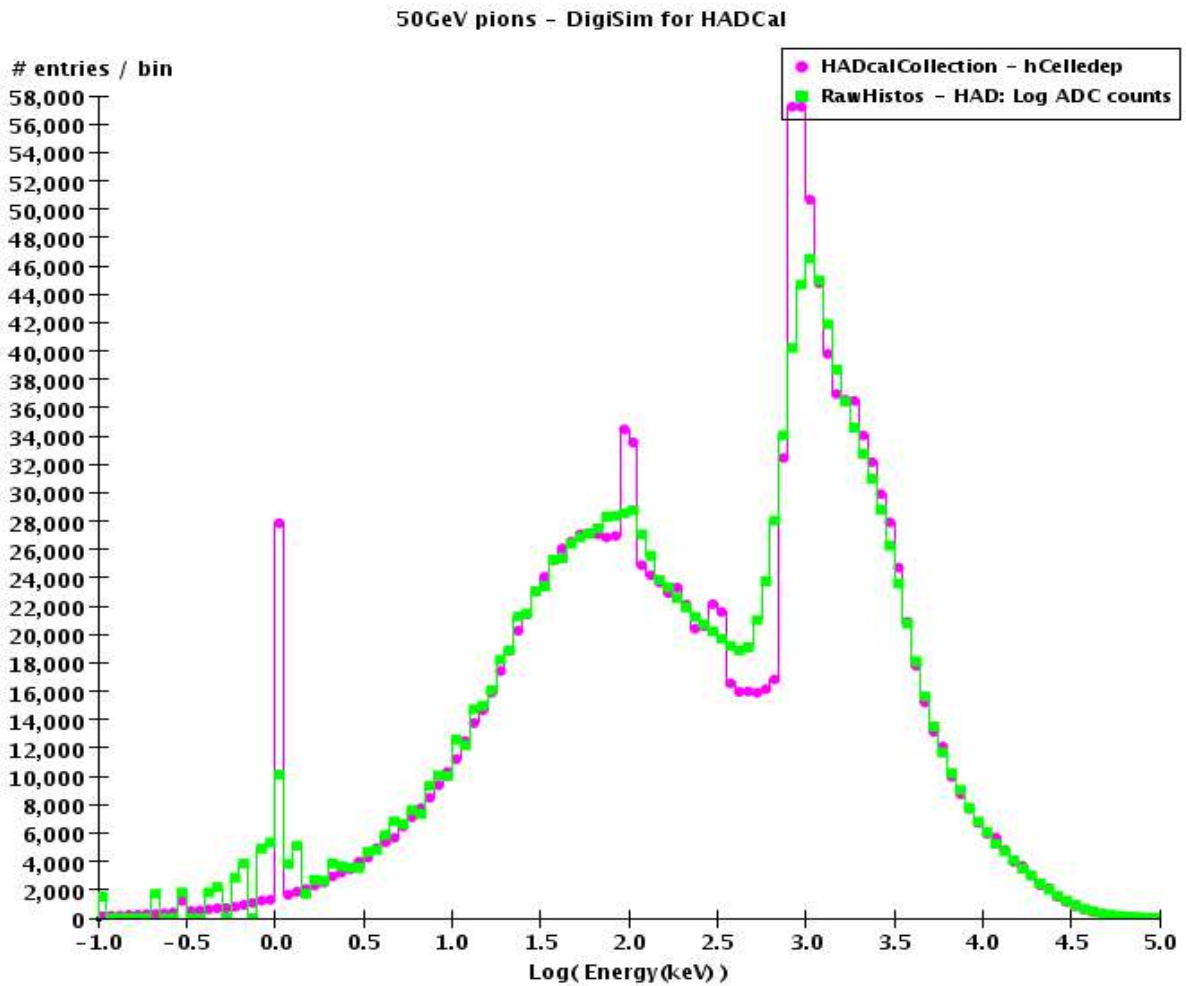
Creating new modifiers



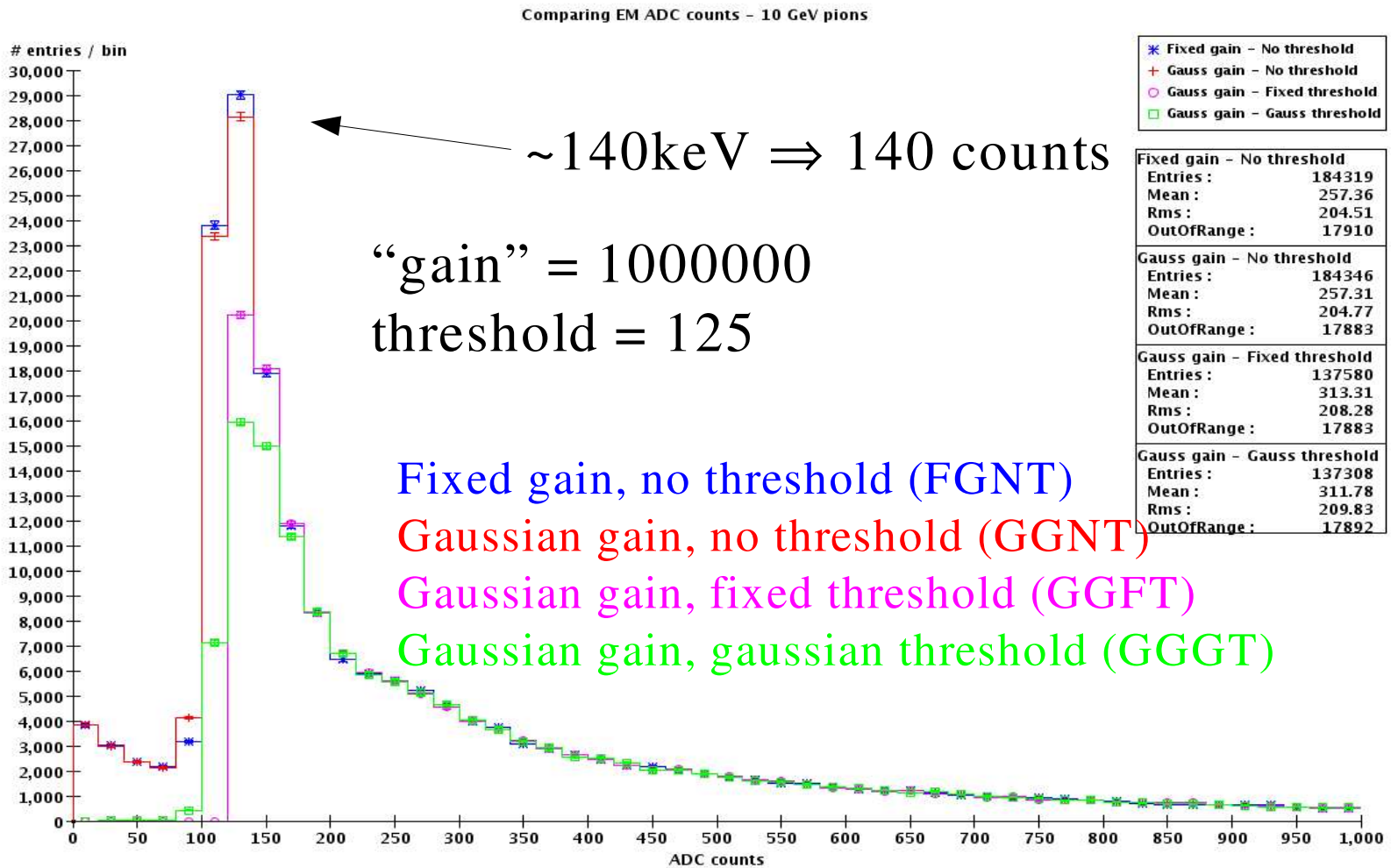
Digisim for EMCal (not real)



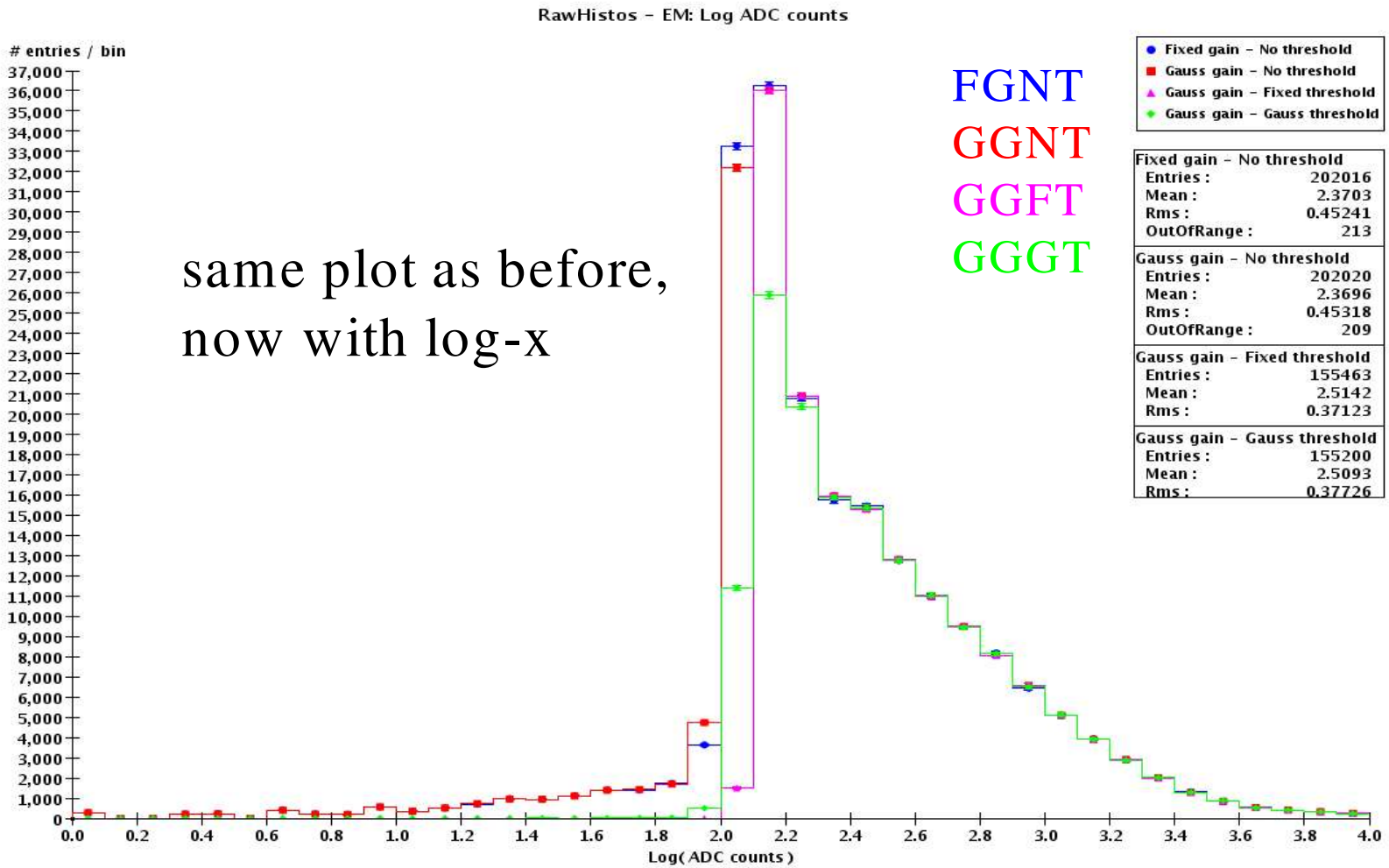
DigiSim for Hcal (0-th order)



10 GeV $\pi^{\pm}$ on EM: gain + threshold

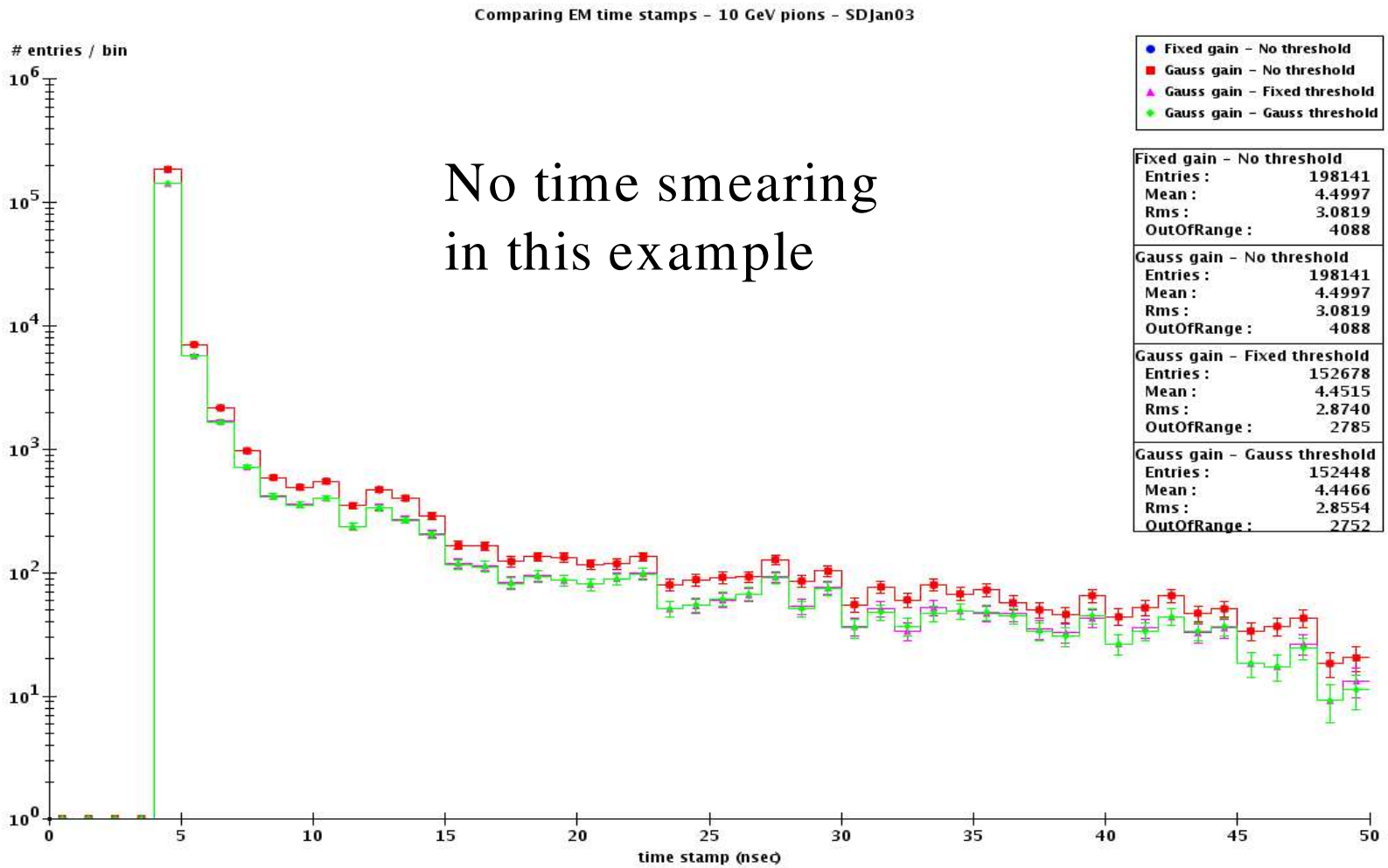


10 GeV $\pi^{\pm}$ on EM: gain + threshold



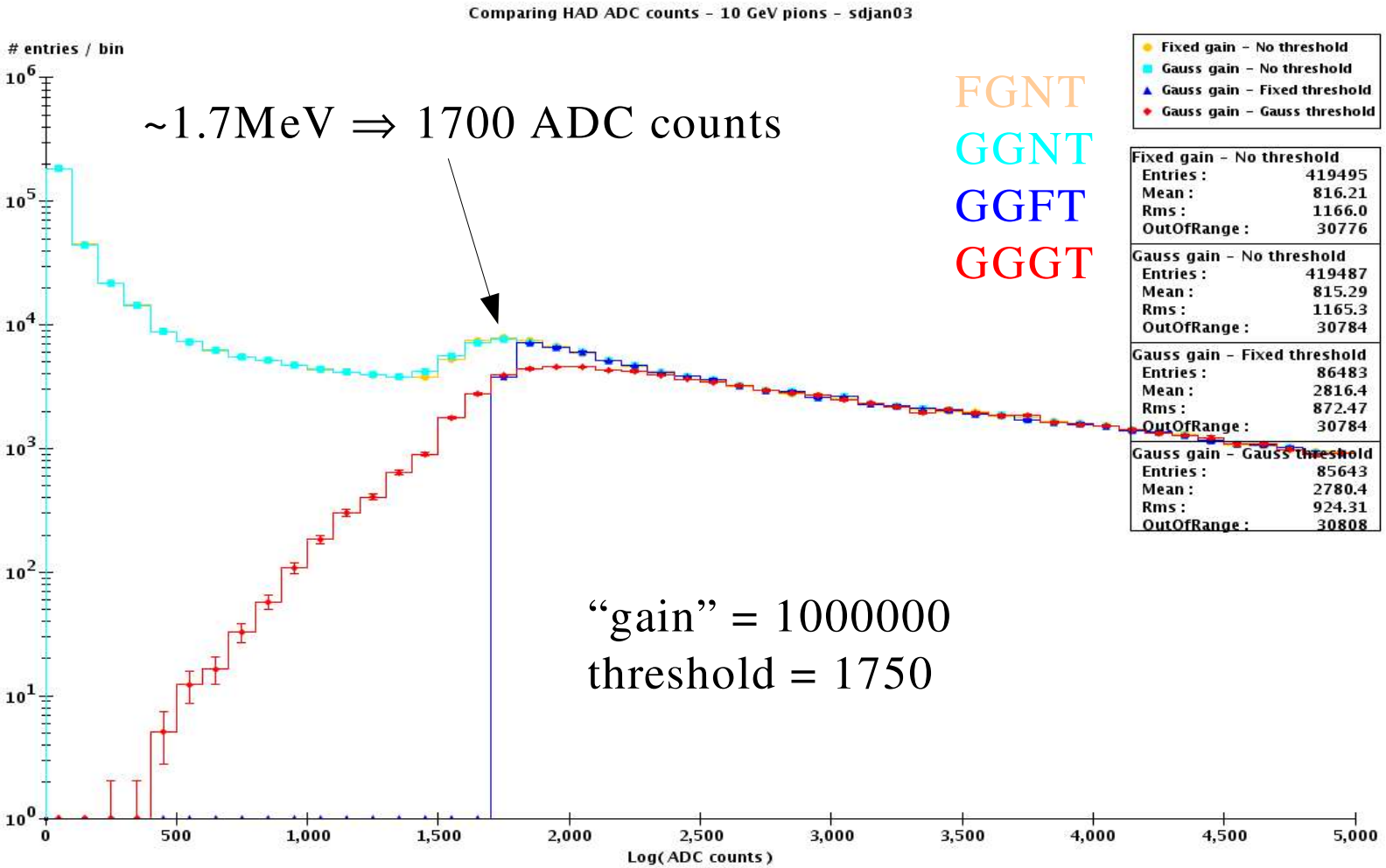
same plot as before,
now with log-x

10 GeV $\pi^{\pm}$ on EM: time stamps

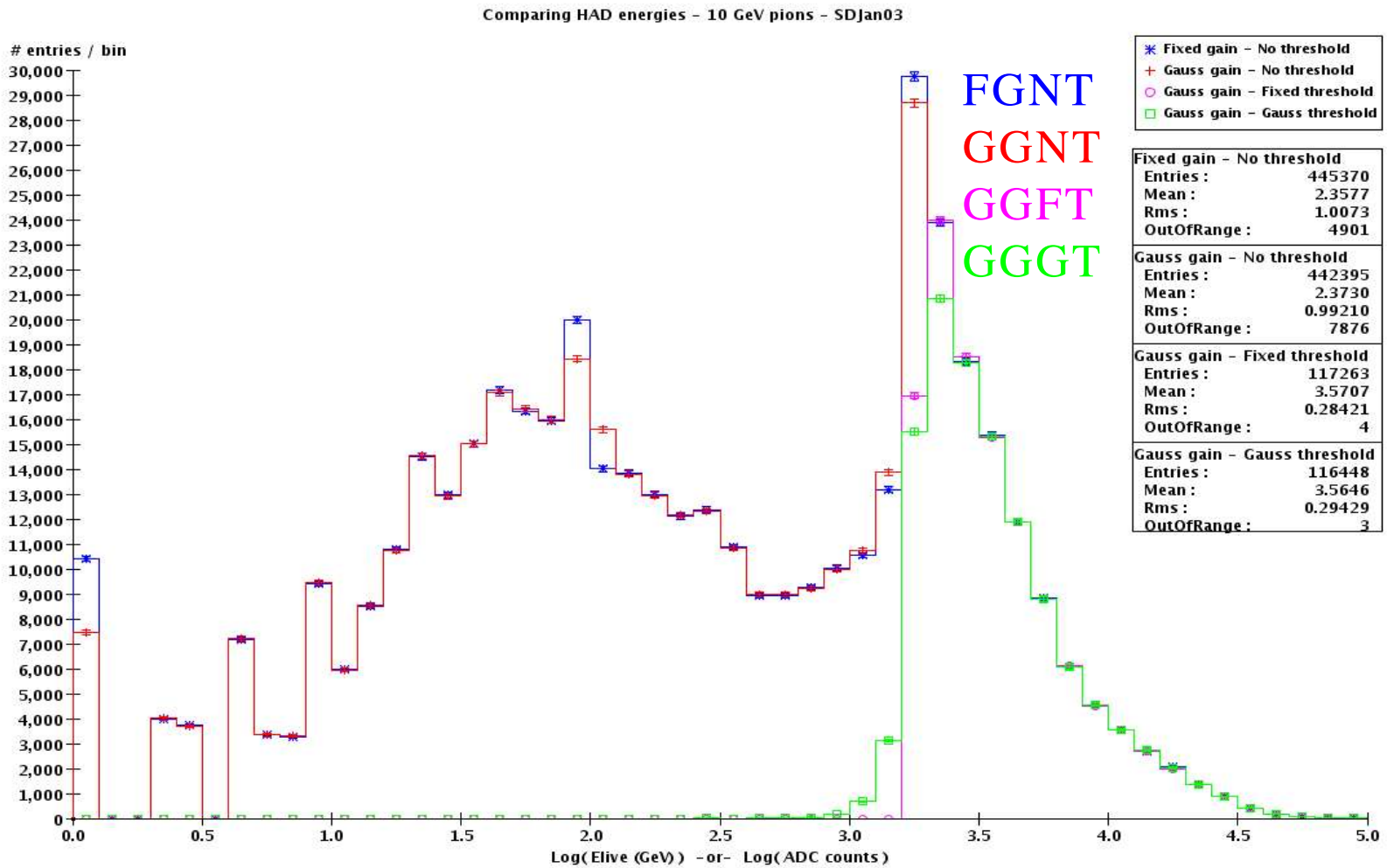


No time smearing
in this example

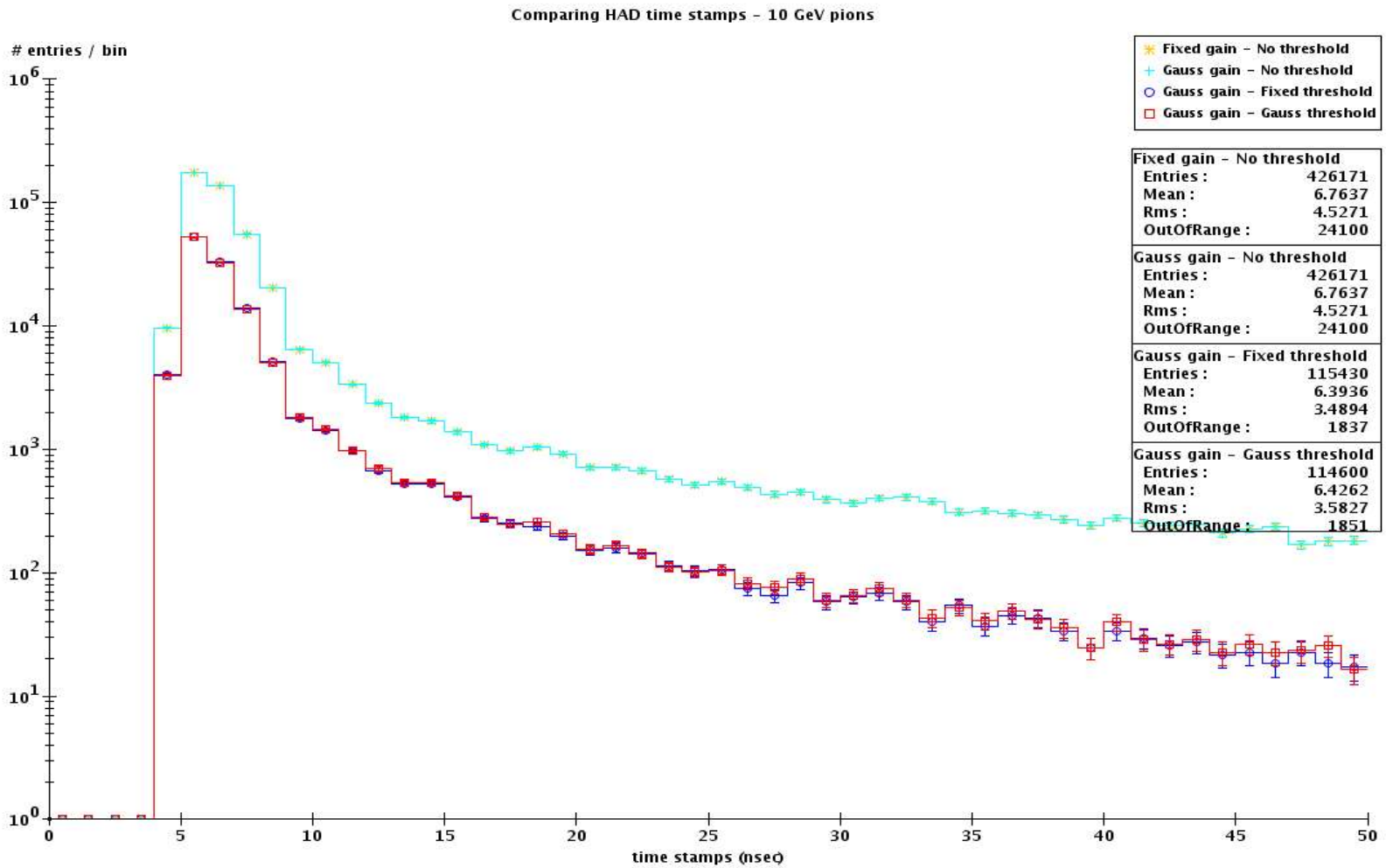
10 GeV $\pi^{\pm}$ on HAD: gain + threshold



10 GeV $\pi^{\pm}$ on HAD: gain + threshold



10 GeV $\pi^{\pm}$ on HAD: time stamps



Status

- A first version of DigiSim (proof of concept) is implemented
  - Two real modifiers implemented:
    GainDiscrimination and function-based SmearedLinear
  - LCRelation: to be used (soon?), example code at hand (thanks to F.Gaede)
- Output LCIO files contain EM and HAD RawCalorimeterHit collections, while keeping simulation collections untouched.
- Analysis code for plotting raw hits, plots confirm expected behavior.
- Creation of new (simple) modifiers is quite easy, by just copying one of the existing modifier classes and implementing the desired transformation.
- Some complicated ones: crosstalk and cell ganging require neighborhood information, from geometry-aware classes. NonProj code exists (java and C++), but projective geometry only available in java.

Summary

- A first version of a digitization simulation program, DigiSim, has been developed at NIU, aiming at full digitization of ILC detector simulations
- DigiSim is object-oriented, powerful, extensible, yet very simple implementation using C++
- Digitization of tracking detectors can probably be implemented as easily as the calorimeter hits
- CALICE test beam can use DigiSim as a testbed, for evaluation and improvement suggestions. Please use it for your digitization studies.
- Send any questions or comments to me: lima at nicadd.niu.edu
- Documentation available at http://nicadd.niu.edu/digisim/DigiSim.html, including download and building instructions

Thank you!