

# Adding Full MC Event Decay Tree to File

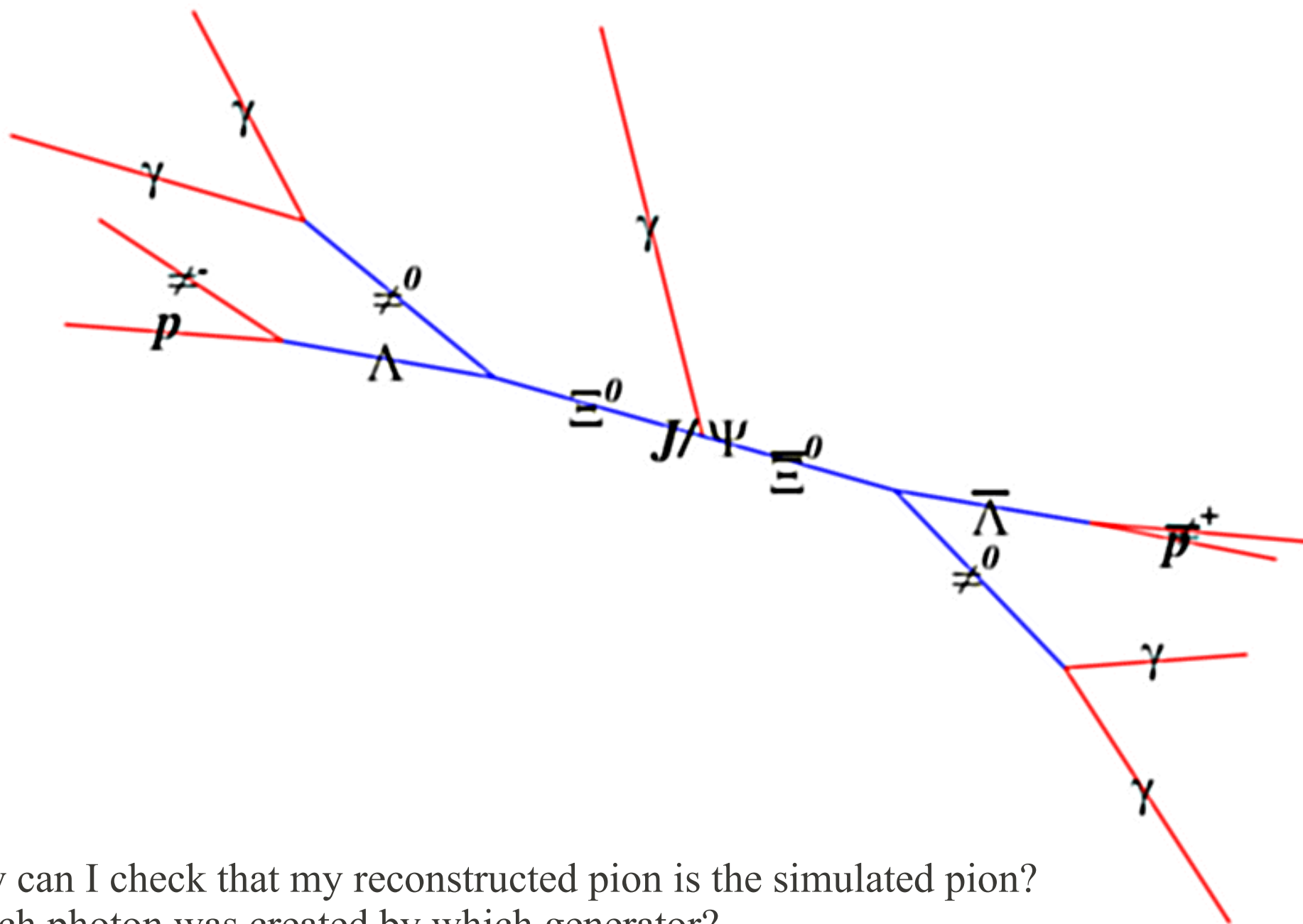
*Björn Spruck*

*II. Physikalisches Institut, Uni Gießen*

## Outline

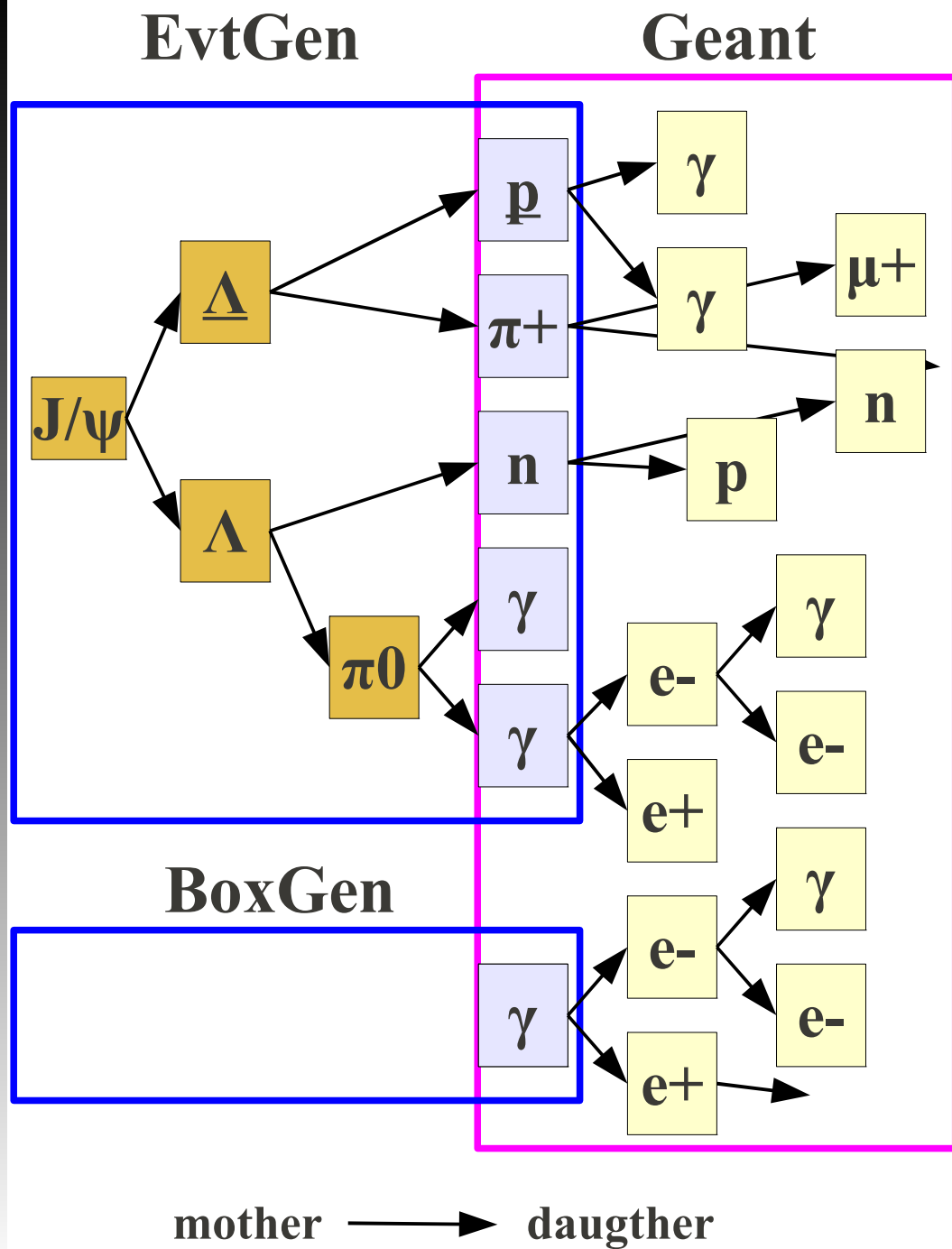
- Motivation
- Ansatz
- Technical Implementation
- User Access
- Outlook

# Decay Tree From Event Generator



How can I check that my reconstructed pion is the simulated pion?  
Which photon was created by which generator?

# Generations of Particles during Monte Carlo Simulation



- Particles are produced by Event Generators
  - EvtGen, BoxGen, DpmGen
- Several might be used for one event!
- Only “final state” particles are given to geant for further tracking
  - new particles are created by geant, setting correct relation parent/daughter
- after geant detector response is used to create tracks/hits etc ...
- Information of particle history in EvtGen is lost

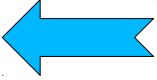
## Status up to now

- EvtGen (and other generators) deliver only “stable” (called primary) particles to MC propagator.
  - Others are explicitly rejected!
- Why?
  - adding all particles will (if geant knows them) result in double entries, because geant would try to decay them again
  - Unknown particles are ignored -> Error msg
- Until now, mother ID (index) could not be given to MC
  - always -1 (means: primary)

## ToDo

- Store decay tree (somewhere ) for later usage
  - **Particles which are already decayed have to be excluded from geant propagation**
  - **Mother (and daughter) relation has to be kept**
- KISS – keep it simple, s...
  - avoid rewriting lots of code
  - avoid changing data structures
  - do not break existing code
  - (if possible...)
- Compatible with MC Linking

## Two Possible Ways

- 1) Use an additional global data struct which contains only the decay tree
  - a) follow geant particles up to “primary”, then switch to other data struct
  - b) lot of new code to write ☹️
- 2) Store them as all other particles, but tell geant not to decay them 
  - a) simpler (as functionality exists for fastsim!)
  - b) interference with existing code?
  - c) Keeping mother ID is critical here (ID -1 used for primary)

I choosed the second Ansatz

## Technical Details

- Forbid geant to propagate particle
  - use same variable as in fastsim (dotracking in FairPrimaryGenerator)
- Keep Mother ID
  - Mother cannot be used as it interferes with IsPrimary() which checks for motherID=-1
    - but there is a second mother ID!
    - (not used in combination with first=-1)
  - if primary mother=-1 then read and use second ID
  - (Note: second mother ID has been misused by mc detector hit counting, relicts were still in the code...)

# Data Structures Involved

- **TParticle**

- used only internally for Monte Carlo (PndStack)

- ROOT class, cannot be changed easily

- **PndMCTrack**

- Saved to file for later usage
  - name: **MCTrack**
- Created from TParticles by PndStack (with opt. filtering)

- Code can be changed easily

- added second mother
- added flags (primary, generator generated / decayed, etc...)

**Filter**



**Indices can be, but do not have to be the same!  
Stored Indices are updated (by PndStack)**



## Code changes

- FairRootManager: Correct handling of Mother IDs
- FairPrimaryGenerator: Add. parameters and offset correction (if  $>1$  generator is used)
- FairGenericStack and PndStack: PushTrack with add. parameters
- PndMCTrack: Data members and access f'ons
- ... Generators, Flags?
  - `evtGen->SetStoreTree(true);`
- Some changes can also be moved to a derived class.

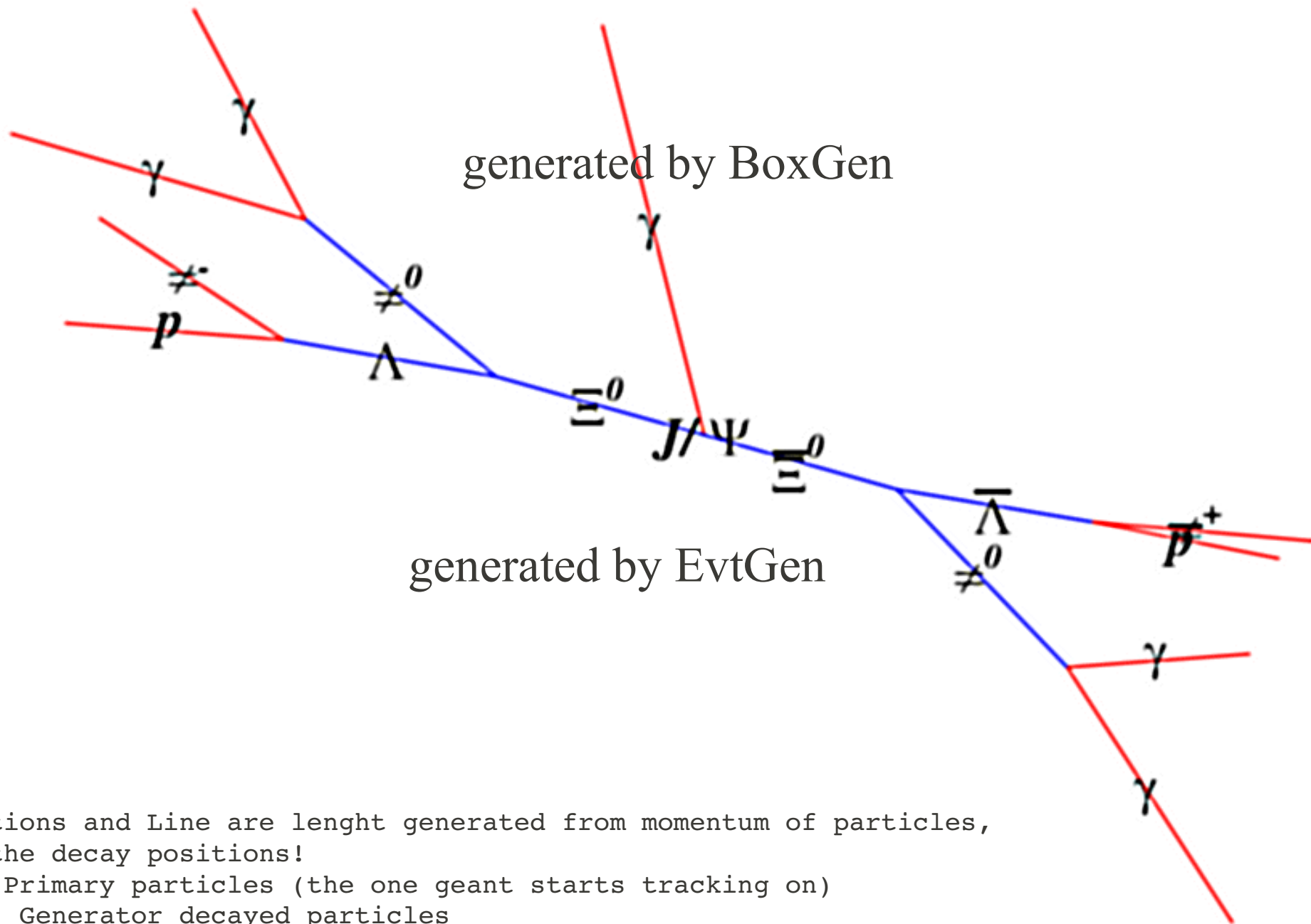
- Iterate over PndMCTrack
  - Use Mother Id and second Mother Id
- Example:

```
show_ancestors(TTree *tree=tv__tree,Int_t entry=0,Int_t id=50)
{ // tree is cbmsim, load entry and show all ancestors of id
  TClonesArray* mctrack_array=new TClonesArray("PndMCTrack");
  tree->SetBranchAddress("MCTrack",&mctrack_array);
  tree->GetEntry(entry);
  plot_parent(id,mctrack_array);
}
```

```
void plot_parent(Int_t id,TClonesArray* mctrack_array)
{
  PndMCTrack *mct=(PndMCTrack *)mctrack_array->At(id);
  cout <<id<<" ("<< mct->GetPdgCode() <<") ["<<
    mct->IsGeneratorCreated()<<mct->IsGeneratorDecayed()<<mct->IsGeneratorLast()<<"]";
  Int_t mid=mct->GetMotherID();
  if(mid<0){
    cout <<"*";
    mid=mct->GetSecondMotherID();
  }
  cout <<"=> ";
  if(mid<0){
    cout << " FIN"<<endl;
  }else{
    plot_parent(mid, mctrack_array);
  }
}
```

Output: 50(11)[000]=> 31(22)[000]=> 30(11)[000]=> 14 (22)[101]\*=> 11(111)[110]\*=> 3(-3322)[110]\*=> 1(443)[110]\*=> FIN

# Example: One Event Done With Two Event Generators




Positions and Line are lenght generated from momentum of particles,  
NOT the decay positions!

RED: Primary particles (the one geant starts tracking on)

BLUE: Generator decayed particles

I will put the macro to SVN on request.

# To Be Done I

- Handling of “unknown” particles
  - Problem in FairPrimaryGenerator
    - usage of TDatabasePDG to get mass for energy calculation
    - maybe also somewhere else, too?
  - skipping particle breaks mother ID/Index structure!
  - 1) add correct “error” handling
    - modify index
    - but: particle would be missing in tree
  - 2) add particle data to TDatabasePDG
    - impossible for user modified particles!!!
  - 3) Change code (AddTrack) to supply mass/energy from generator instead of TDatabasePDG 
    - would be even better than recalculating energy (width of particles!!)

# To Be Done II

- Fast Sim
  - Produce no Candidates for generator created and decayed particles
  - nearly done
- User Interface
  - What is needed by the User?
    - Get “real primary”
    - Get last decay tree particle before geant (“last primary”)
    - Search for intermediate states
- Put code to official repo

# Backups

# Indizes

- Indizes might change if list of particles is “cleared”
- Mother (and daughter) have to be changed accordingly!!
- Mothers are updated (after selection) in PndStack
  - Daughters not!!! But they are not used up to now.

## adding non tracked particles

- adding particle which is not known (...) results in no particle stored at all -> spoiling MC index!!! -> crash/inf loop
  - -E FairPrimaryGenerator: PDG code 40443 not found in database.  
Discarding particle.
  - can be worked around (IMO)
  - removing check or adding particles to db



# Problem with missing Particles in FairPrimaryGenerator

```
// ---> Check whether particle type is in PDG Database

TDatabasePDG* pdgBase = TDatabasePDG::Instance();

if ( ! pdgBase ) Fatal("FairPrimaryGenerator",
"No TDatabasePDG instantiated");

TParticlePDG* pdgPart = pdgBase->GetParticle(pdgid);

if ( ! pdgPart ) {

    cout << "\033[5m\033[31m -E FairPrimaryGenerator: PDG code " << pdgid << " not found in
database. Discarding particle. \033[0m " << endl;

    return;

}

// ---> Get mass and calculate energy of particle

Double_t mass = pdgBase->GetParticle(pdgid)->Mass();

Double_t e = TMath::Sqrt( px*px + py*py + pz*pz + mass*mass );
```

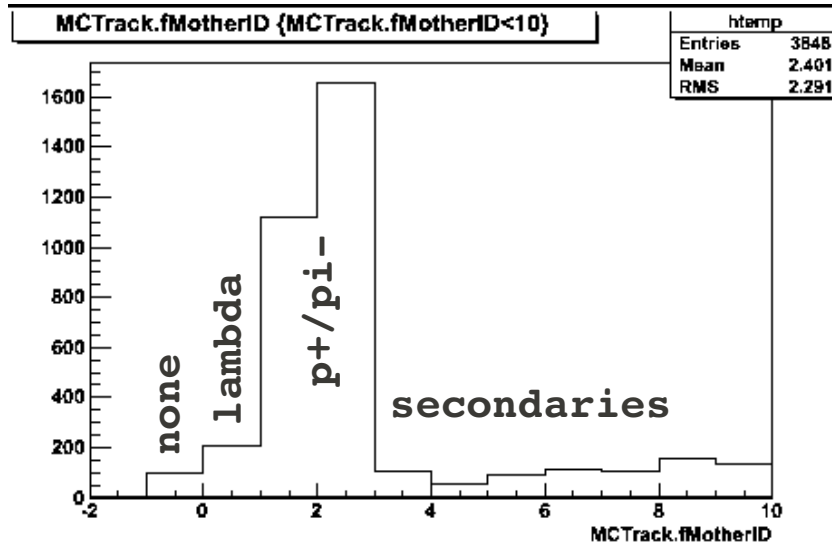
- mother id must be propagated
  - problem: index in EvtGen might be different from PndStack Index if EvtGen is not the first Generator called
    - adding an offset needed, no principle problem
    - minor change in base/ class FairBaseGenerator needed (done)
    - one more parameter to AddTrack (default is -1, keeps compatibility!)
  - big big problem:
    - a primary particle was/is detected by having mother id = -1
    - additional flag would be required! (like generated by generator or so ;-))
    - changes in base classed needed, who needs the information that it was a primary anyway? beside monte carlo truth?
  - cleanup function in PndStack
    - removes (on demand) secondaries which are not needed -> ...

## Questions...

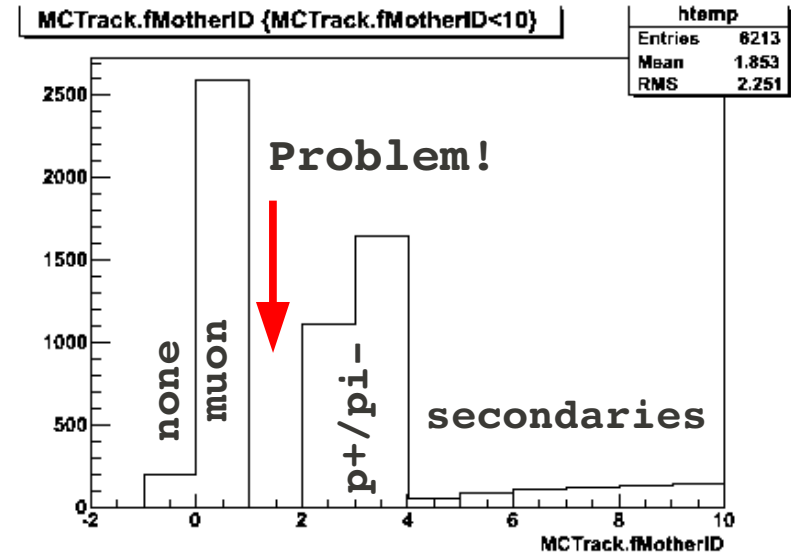
- cleanup function in PndStack
  - keep mother(s)
  - keep secondaries, energy, number points
  - updates MC Index
- Question: What do we need for MC truth
  - Talking about intermediate particles which are decayed by geant without interaction in material
  - (and from EMC to track back, one would need all tracks! esp if the shower started in front of the crystal)
- Q: What is called a primary anyway?
  - Particles which are “stable” coming out of the generator
  - Is this definition needed at all?

# Examples: parent track ID for 101 Lambdas decaying to p+ pi-

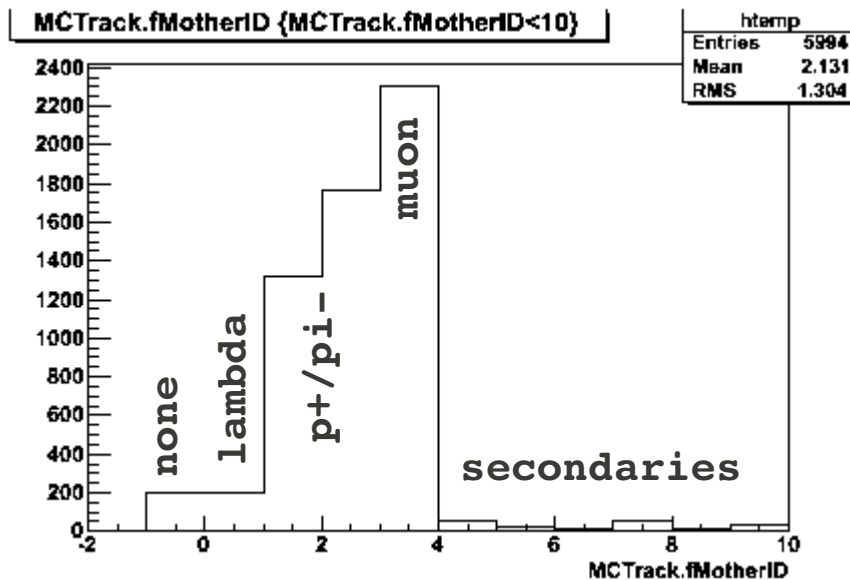
- Only EvtGen



- Box (1 muon) + EvtGen



- EvtGen + Box (1 Muon)



- Box+EvtGen (MCoffset fix)

