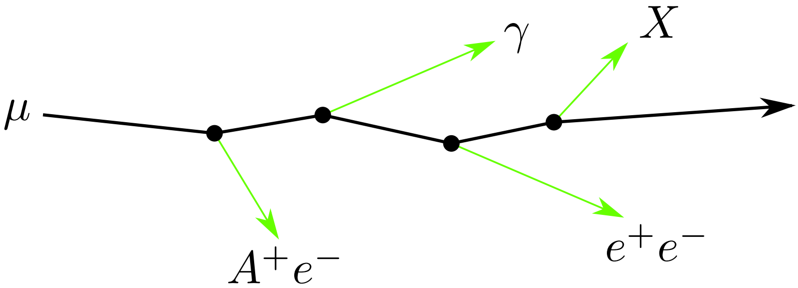
technische universität
dortmund



New issues

Jean-Marco Alameddine

3/30/2023

CORSIKA technical call

Overview

I've reviewed the current status of `Cascade.inl` and EM modules, and opened some new issues...

- ParticleWriterParquet counts kinetic energies instead of total energies
- Cascade.inl does not take into account decreasing cross sections anymore (possibly?)
- Excessive amount of PROPOSAL::getMaxStepLength calculated a negative step length warnings
- Stochastic photon propagation (MR)
  - → Not an issue, but supposed to fix some warnings
- Harmonize calculation of sqrtSNN in corsika_proposal::HadronicPhotonModel with corsika::sophia::InteractionModel (MR)
  - → Also not an issue, but fixes crashes with PROPOSAL + SOPHIA

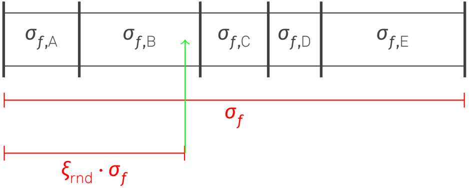
ParticleWriterParquet counts kinetic energies instead of total energies

- We realized that `ParticleWriterParquet` writes kinetic particle energies instead of total particle energies to the output
  - → This is to be consistent with CORSIKA 7 and AIRES, which have the same behaviour
  - → Label in output been changed from `energy` to `kinetic_energy` in PR !490
- A functionality of `ParticleWriterParquet` is the method `getEnergyGround()`, which returns the energy that has been absorbed in the observation plane
  - → However, this method returns the sum of all kinetic energies
  - → Could be fixed by just tracking the total instead of kinetic energy in `ParticleWriterParquet`
- Similar problem with the `EnergyLossWriter`, which provides a method `getEnergyLost()`
  - → `EnergyLossWriter` keeps track of all ”energy losses” (e.g. continuous energy losses, or particles erased by the `ParticleCut`)
  - → For `ParticleCut`, we track the *kinetic* energies of the particles that were cut
- At the end of our example scripts, we compare `Efinal = dEdX.getEnergyLost() + obsLevel.getEnergyGround()` to the total energy of the shower-inducing particle
  - → This is meant as a validation
  - → However, we don’t expect these quantities to be identical with the current accounting...

Cascade.inl does not take into account decreasing cross sections anymore (possibly?)

1. Take particle from stack, with total energy E_i
2. Calculate total inelastic cross section σ_i

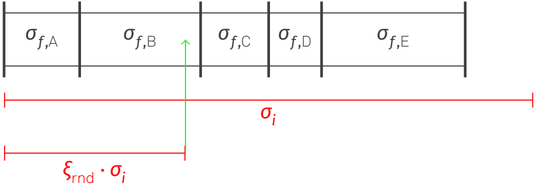
   → Use σ_i to sample distance to next interaction, λ
3. Apply continuous energy losses to our particle: $E_i \rightarrow E_f$
4. Re-calculate the total inelastic cross section with the updated energy E_f, we get σ_f
5. Sample which interaction is actually executed by sampling a $\xi_{\text{rnd}} \cdot \sigma_f$, with $\xi_{\text{rnd}} \in [0, 1)$



- This way, we *always* select an interaction process
- However, we have seen that σ has changed due to the continuous energy losses...

Cascade.inl does not take into account decreasing cross sections anymore (possibly?)

- In reality, I believe the sampling process should look something like this
- To sample which process we end up with, we shoudn't use $\boldsymbol{\sigma_f}$ (calculated after applying the continuous losses), but rather $\boldsymbol{\sigma_i}$ (calculated before applying the continuous losses)
  - → This way, we might end up with NO interaction selected
  - → This is how we would account for the change of the cross section due to the continuous energy losses


$\sigma_{f,\mathrm{A}}$
$\sigma_{f,\mathrm{B}}$
$\sigma_{f,\mathrm{C}}$
$\sigma_{f,\mathrm{D}}$
$\sigma_{f,\mathrm{E}}$
σ_i
$\xi_{\mathrm{rnd}} \cdot \sigma_i$