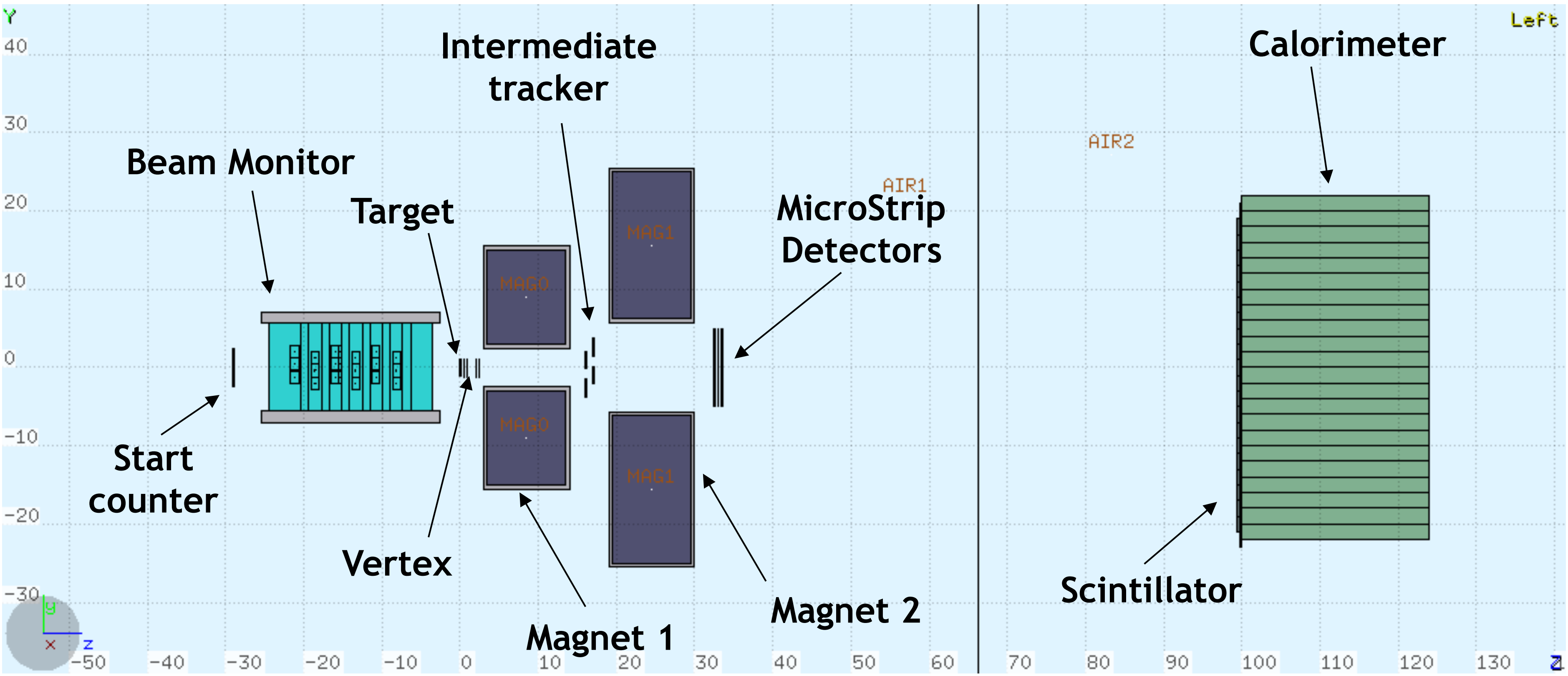
A very short description of the ROOT file of Simulation Output

September 2019

G. Battistoni & S.M. Valle

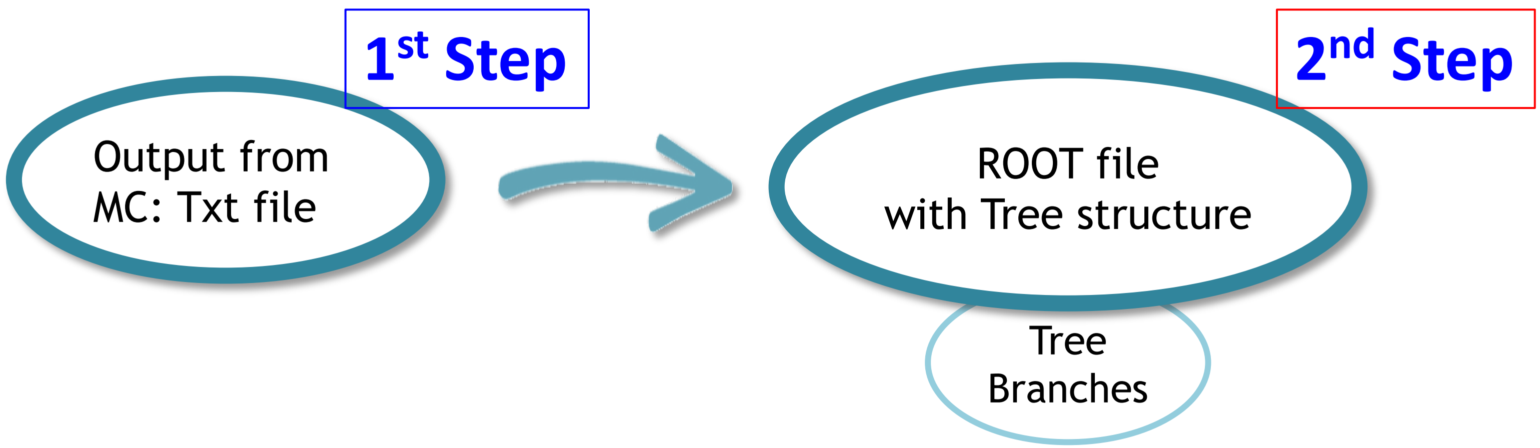
The FOOT simulation setup in newgeom branch



The MC Output for FOOT

We have configured some user routines of FLUKA to produce an "ad hoc" event-by-event output written as an ASCII file (*TXT.dat)

Those ASCII files contain information about all the particles and interactions simulated. A simple and portable code reads these txt's and outputs ROOT files



Example of a root output file

For instance the Root file: 16O_C2H4_200_1.root

(also in /gpfs_data/local/foot/Simulation/V14.0.1 on Tier3)

2 10^7 events of ^{16}O on a 2 mm C_2H_4 target

Usually only events with inelastic interaction in the target were written on output, for compactness: ~1.1% of total no. of simulated events. However full simulation can be produced.

The root data by FLUKA

The data are stored in a root file with several blocks in the structure EVENT_STRUCT *(defined in the file EventStruct.h)*:

- The particle block: kinematics information of the produced particles
- The detector block: information about the detector outputs of the event and namely about energy releases and hits + links to "MC truth".
- The crossing block: information about the particle that cross different regions of the setup (both inactive and active)

The particle structure

for each of the produced particles we register the info in arrays: i.e. TRmass[2] is the mass of the 3rd produced particle

EventNumber = FLUKA event number:

TRn= number of particles produced: max equal to **MAXTR**

TRpaid = index in the part common of the particle parent

TRcha = charge

TRbar = barionic number

TRfid = FLUKA code for the particle (es: photon, jpa=7)

TRgen = generation number

TRdead = number of the region where the particle dies

TRix, TRiy, TRiz = production position of the particle

TRfx, TRfy, TRfz = final position of the particle

TRipx,TRipy,TRipz = production momentum of the particle

TRifx,TRfpy,TRfpz = final momentum of the particle

TRmass = particle mass

TRtime = production time of the particle

TRtrlen = Track lenght of the particle

```
Int_t EventNumber;
Int_t TRn;
Int_t TRpaid[MAXTR];
Int_t TRgen[MAXTR];
Int_t TRcha[MAXTR];
Int_t TRreg[MAXTR];
Int_t TRbar[MAXTR];
Int_t TRdead[MAXTR];
Int_t TRfid[MAXTR];
Double_t TRix[MAXTR];
Double_t TRiy[MAXTR];
Double_t TRiz[MAXTR];
Double_t TRfx[MAXTR];
Double_t TRfy[MAXTR];
Double_t TRfz[MAXTR];
Double_t TRipx[MAXTR];
Double_t TRipy[MAXTR];
Double_t TRipz[MAXTR];
Double_t TRfpx[MAXTR];
Double_t TRfpy[MAXTR];
Double_t TRfpz[MAXTR];
Double_t TRmass[MAXTR];
Double_t TRtime [MAXTR];
Double_t TRtof[MAXTR];
Double_t TRlen[MAXTR];
```

The individual detectors structures

For each detector with n energy releases the info are stored in arrays (x, p, De, time, etc...) with the i-th component related to the i-th release . Same syntax for all scint detector: "info""NAMEDETECTOR”[index of the release]

DETn = number of energy release in the detector DET

DETid = position of the particle responsible of the release in the particle block

DETixin, DETyin, DETzin = inizial position of energy release

DETxout, DETyout, DETzout = final position ” “

DETpxin, DETpyin, DETpzin = inizial momentum “ ”

DETpxout, DETpyout, DETpzout = final momentum “ “

DETde = energy release

DETtim = initial time of the energy release

Start Counter: STC

```
Int_t STCn;
Int_t STCid[MAXSTC];
Double_t STCxin[MAXSTC];
Double_t STCyin[MAXSTC];
Double_t STCzin[MAXSTC];
Double_t STCxout[MAXSTC];
Double_t STCyout[MAXSTC];
Double_t STCzout[MAXSTC];
Double_t STCpxin[MAXSTC];
Double_t STCpyin[MAXSTC];
Double_t STCpzin[MAXSTC];
Double_t STCpxout[MAXSTC];
Double_t STCpyout[MAXSTC];
Double_t STCpzout[MAXSTC];
Double_t STCde[MAXSTC];
Double_t STCal[MAXSTC];
Double_t STCtim[MAXSTC];


MAXSTC = 200
```

Simple case of non-segmented detector

Vertex: VTX

```
Int_t ITRn; ... MAXVTX = 300
Int_t VTXilay[MAXITR]; → plane number
```

Intermediate Tracker: ITR

```
Int_t ITRn; ... MAXITR = 300
Int_t ITRisens[MAXITR]; →sensor number
```

beam monitor BMN

```
Int_t BMNn; ... MAXBMN = 1000
Int_t BMNilay[MAXBMN];   → layer #
Int_t BMNicell[MAXBMN]; → cell #
Int_t BMNiview[MAXBMN]; → view (-1:x 1:y)
```

Microstrips: MSD

```
Int_t MSDn; ... MAXMSD = 1000
Int_t MSDilay[MAXDCH]; → layer #
```

scintillator: SCN

```
Int_t SCNn; ... MAXSCN = 5000
Int_t  SCNibar[MAXSCN];
Int_t  SCNiview[MAXSCN];
```

crystal calorimeter: CAL

```
Int_t CALn; ...  MAXCAL = 6000
Int_t CALicry[MAXCAL];
```

The crossing data structure

Not yet inherited in SHOE

This structure registers the info on the particles that cross the boundaries between the different regions of the setup (detector elements, air, target). At each crossing the info are stored in arrays

CROSSn = number of boundary crossing

CROSSid = position of the crossing particle in the particle block

CROSSnreg = no. of region in which the particle is entering

CROSSnregold = no. of region the particle is leaving

CROSSpx, CROSSpy, CROSSpz = momentum at the boundary crossing

CROSSx, CROSSy, CROSSz = position of the boundary crossing

CROSSt = time of the boundary crossi

CROSSch = charge of crossing particle

CROSSm = mass of the crossing particle

```
Int_t CROSSn;
Int_t CROSSid[MAXCROSS];
Int_t CROSSnreg[MAXCROSS];
Int_t CROSSnregold[MAXCROSS];
Double_t CROSSx[MAXCROSS];
Double_t CROSSy[MAXCROSS];
Double_t CROSSzMAXCROSS];
Double_t CROSSpx[MAXCROSS];
Double_t CROSSpy[MAXCROSS];
Double_t CROSSpz[MAXCROSS];
Double_t CROSSm[MAXCROSS];
Double_t CROSSch[MAXCROSS];
Double_t CROSSt[MAXCROSS];
```

MAXCROSS = 10000

Energy releases and hits connection to particles

To find which particle released energy in a detector we need to build a pointer to the particle block. Given the j–th energy release in the detector DET, then we build:

```
pointer= pevstr->DETid[j]–1;
```

Then the features of the particles responsible of the release (for example the mass and the x coord of production) can be retrieved from the particle block as:

```
Massa = pevstr->TRmass[pointer];
Xprod = pevstr->TRix[pointer];
```

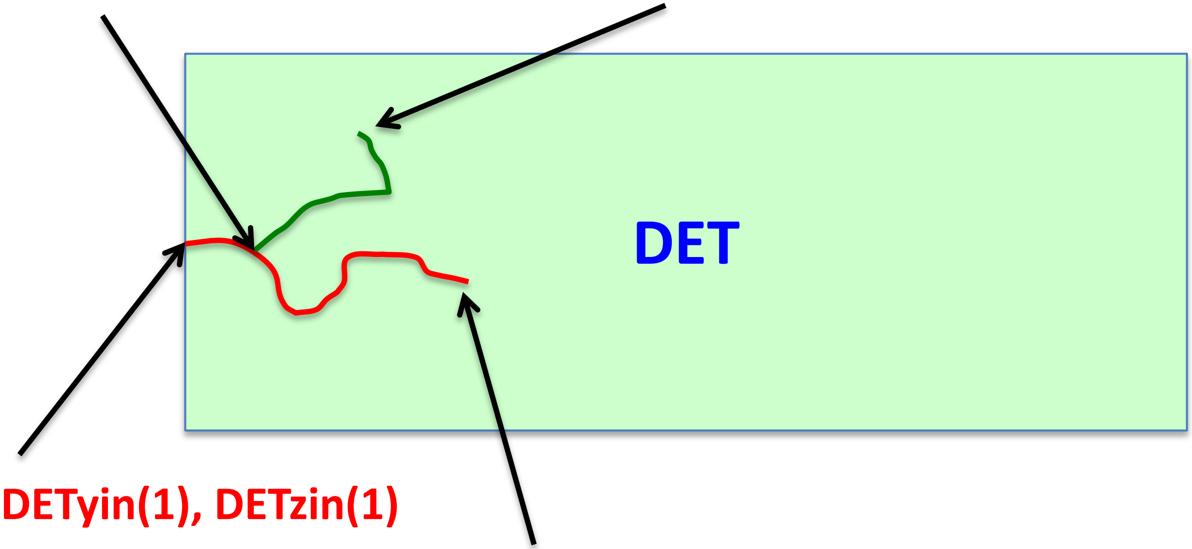
DETid(2)-1 = pointer to the particle in Particle Structure that originated hit=2
to access all infos (id, quantum numbers + kinematics) about that particle

DETn = 2

DETde(2) = Sum of energy releases by that “particle”
in a given region of detector DET

DETxin(2), DETyin(2), DETzin(2)

DETxout(2), DETyout(2), DETzout(2)



DETxin(1), DETyin(1), DETzin(1)

DETxout(1), DETyout(1), DETzout(1)

DETn = 1

DETde(1) = Sum of energy releases by that “particle”
in DET

DETid(1)-1 = pointer to the particle in Particle Structure that originated hit=1
to access all infos (id, quantum numbers + kinematics) about that particle