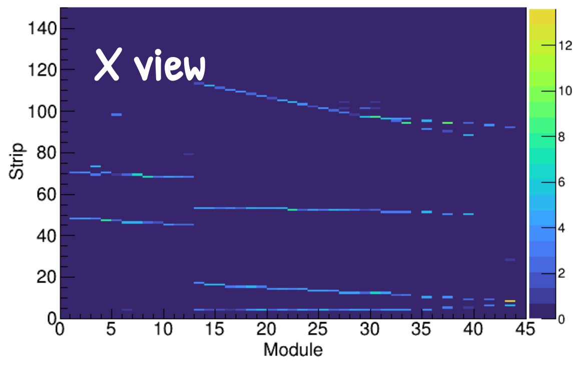
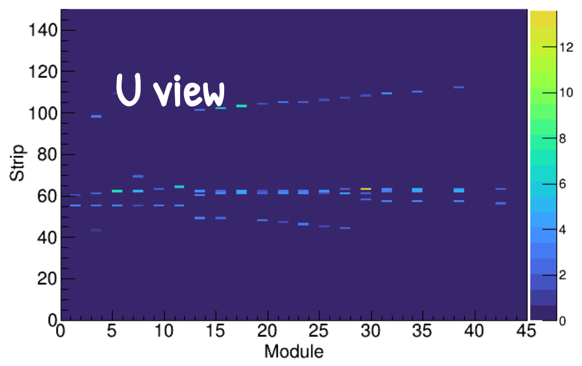
Status of MINERvA output to CAFs

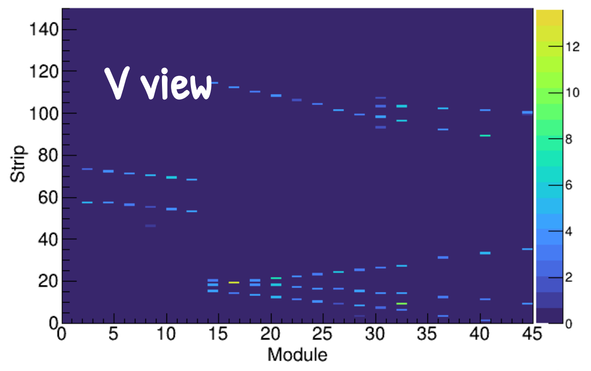
Noë Roy
on behalf of the Reco & Sim Minerva 2x2 team

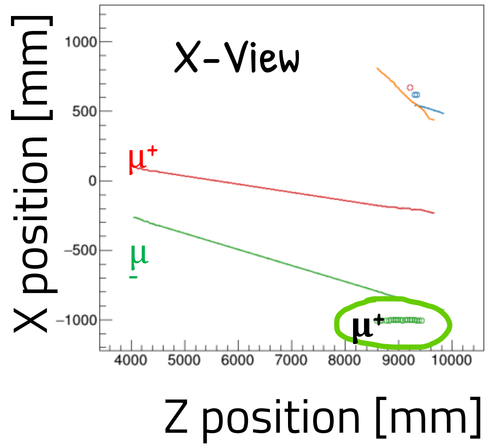
2x2 Workshop – 05/20/2023

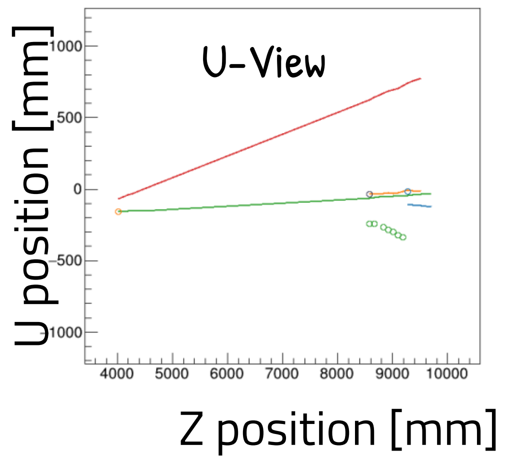
Output from MINERvA (again)

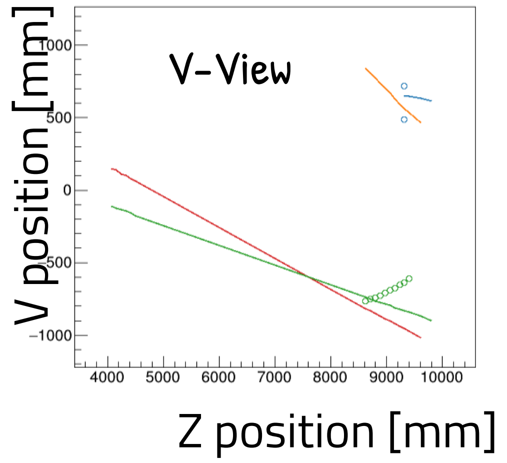












— ID 10 - (-13) - Evis 267.0 - Slice 1 - θ 0.21 - ϕ -1.89

— ID 30 - (-13) - Evis 75.5 - Slice 2 - θ 0.13 - ϕ -2.98

— ID 32 - (13) - Evis 288.8 - Slice 4 - θ 0.16 - ϕ -2.53

— ID 43 - (-13) - Evis 172.2 - Slice 6 - θ 0.42 - ϕ -2.56

○ Blob 0 - (-13) - Evis 1.2 - Size 1 - Slice 2 - ID 43 -Type 2

○ Blob 1 - (-13) - Evis 26.5 - Size 5 - Slice 2 - ID 43 -Type 3

○ Blob 2 - (-13) Evis 122.4 - Size 36 - Slice 3 - ID 32 -Type 3

○ Blob 3 - (-13) - Evis 1.9 - Size 1 - Slice 4 - ID 10 -Type 2

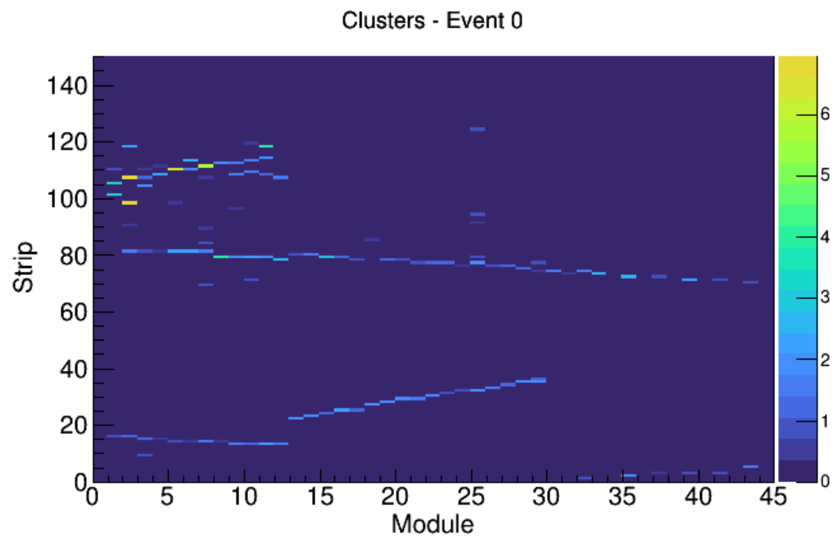
○ Blob 4 - (13) - Evis 2.3 - Size 1 - Slice 6 - ID 32 -Type 2

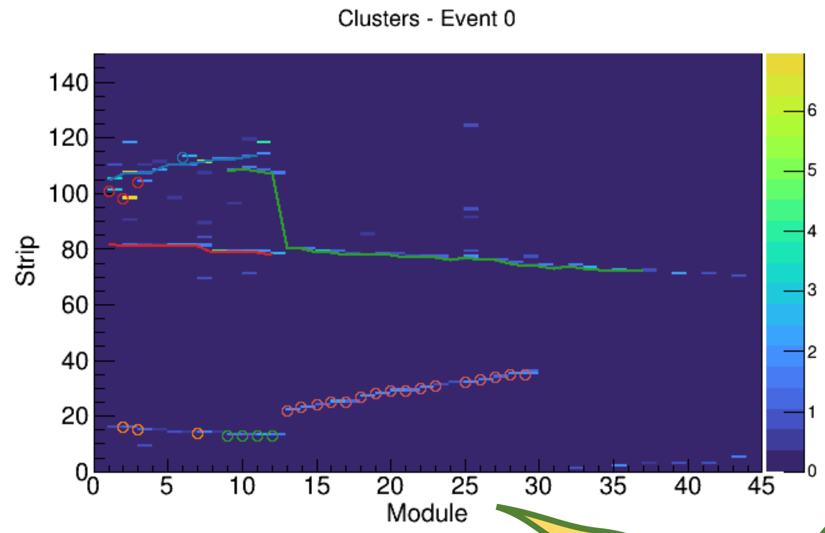
Reconstruction of tracks and blobs (shower like particle)

True particle linked to object: particle that contributed the most (energy wise)

YORK

Output from MINERvA (again but DATA)





(This is the same data as this morning but with a more recent light yield constant)

ID 0 - Evis 40.0 - Slice 1 - θ 0.18 - ϕ -2.45

ID 0 - Evis 57.2 - Slice 3 - θ 0.49 - ϕ 0.82

ID 0 - Evis 66.0 - Slice 6 - θ 0.16 - ϕ -2.47

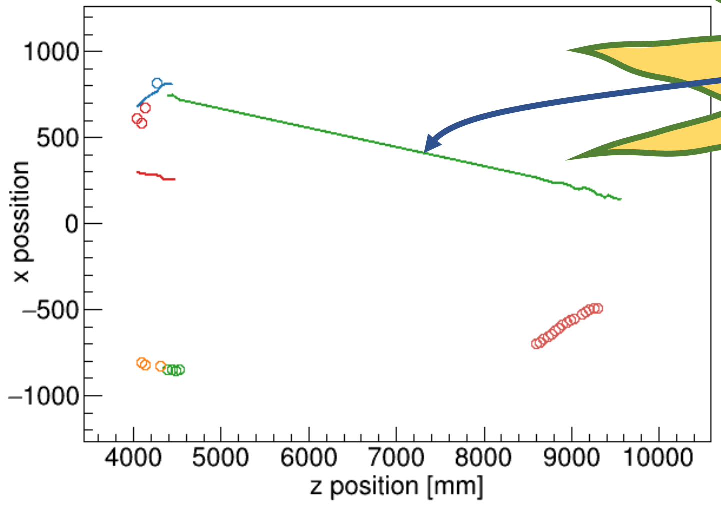
Blob 0 - Evis 20.8 - Size 6 - Slice 3 - ID 0 -Type 3

Blob 1 - Evis 17.6 - Size 5 - Slice 3 - ID 0 -Type 3

Blob 2 - Evis 11.5 - Size 7 - Slice 5 - ID 0 -Type 3

Blob 3 - Evis 13.5 - Size 9 - Slice 5 - ID 0 -Type 3

Blob 4 - Evis 44.9 - Size 27 - Slice 11 - ID 0 -Type 3



Upstream to downstream DATA track !!!

Reconstruction of tracks and blobs (shower like particle)

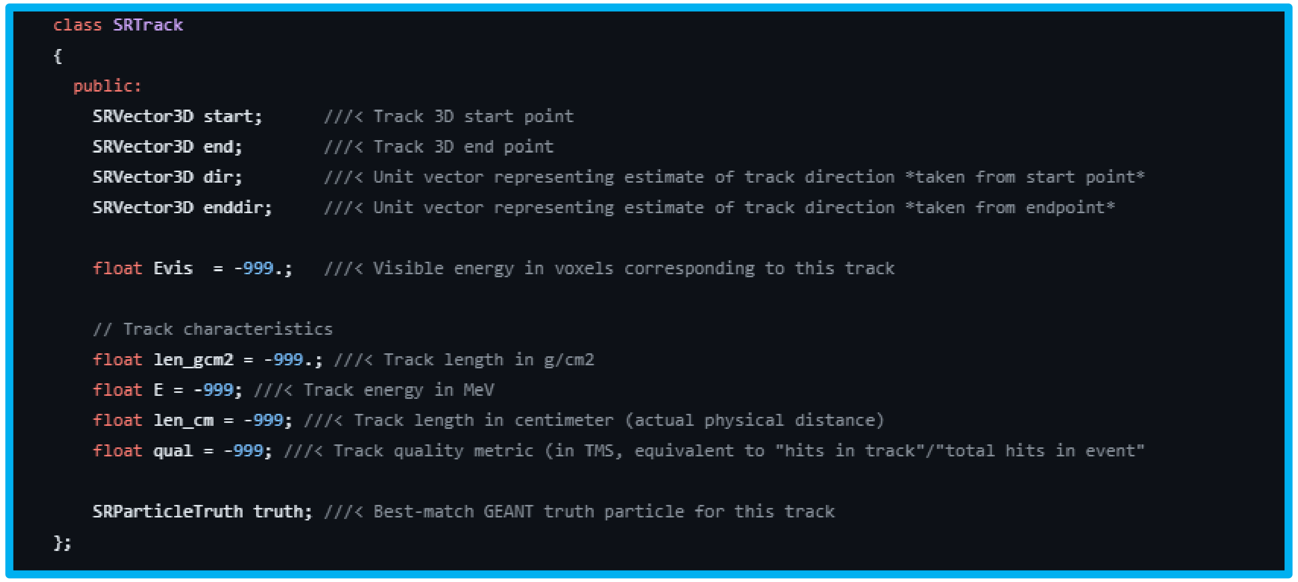
What we can add into the CAFs

We don't want to keep a list of clusters but the core parameters of the objects

1) Keep track of tracks.

Already a SRTrack object in CAFS :

```
- Start (3D)
- End (3D)
- End Direction (3D)
- Visible Energy
- Link to truth particle
```



```
class SRTrack
{
  public:
    SRVector3D start;      ///< Track 3D start point
    SRVector3D end;        ///< Track 3D end point
    SRVector3D dir;        ///< Unit vector representing estimate of track direction *taken from start point*
    SRVector3D enddir;     ///< Unit vector representing estimate of track direction *taken from endpoint*

    float Evis  = -999.;   ///< Visible energy in voxels corresponding to this track

    // Track characteristics
    float len_gcm2 = -999.; ///< Track length in g/cm2
    float E = -999; ///< Track energy in MeV
    float len_cm = -999; ///< Track length in centimeter (actual physical distance)
    float qual = -999; ///< Track quality metric (in TMS, equivalent to "hits in track"/"total hits in event"

    SRParticleTruth truth; ///< Best-match GEANT truth particle for this track
};
```

We can add angles if needed but can be computed with start & end positions

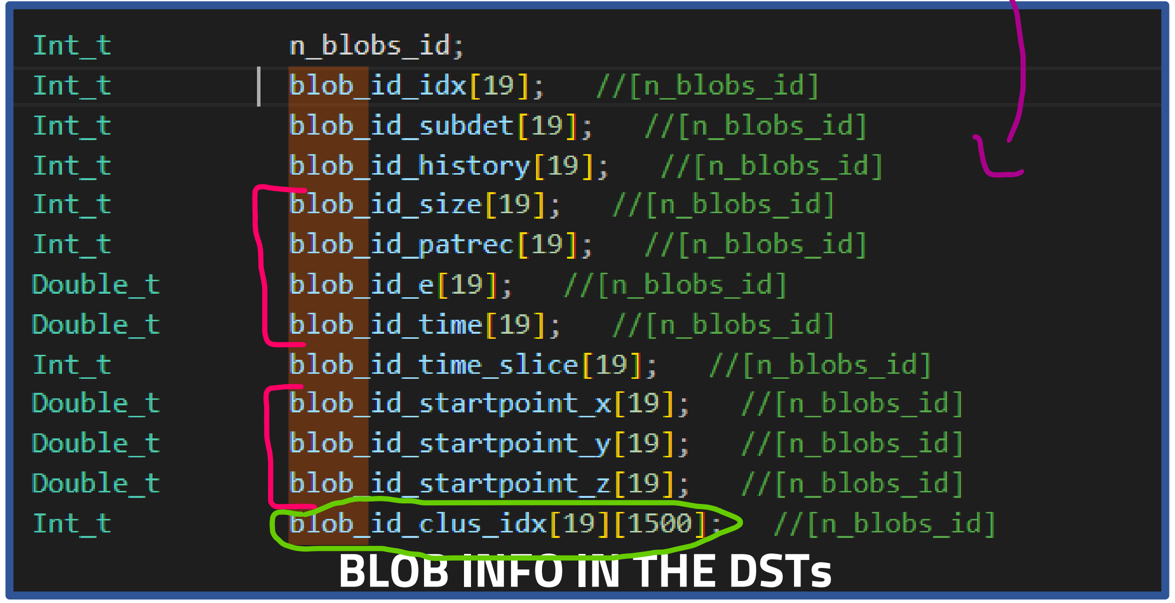
We might want to add **dE/dX at the start point and end point** of the track to have some basic PID (Proton/Muons).
At the Track level reco, no dE/dX is computed but could be added with the clusters somewhere.

Could be computed in the CAFs Maker with the cluster list: **would need to update SRTrack.**

What about the blobs ?

Keep track of BLOBS ?

Right now what do we keep in the DSTs



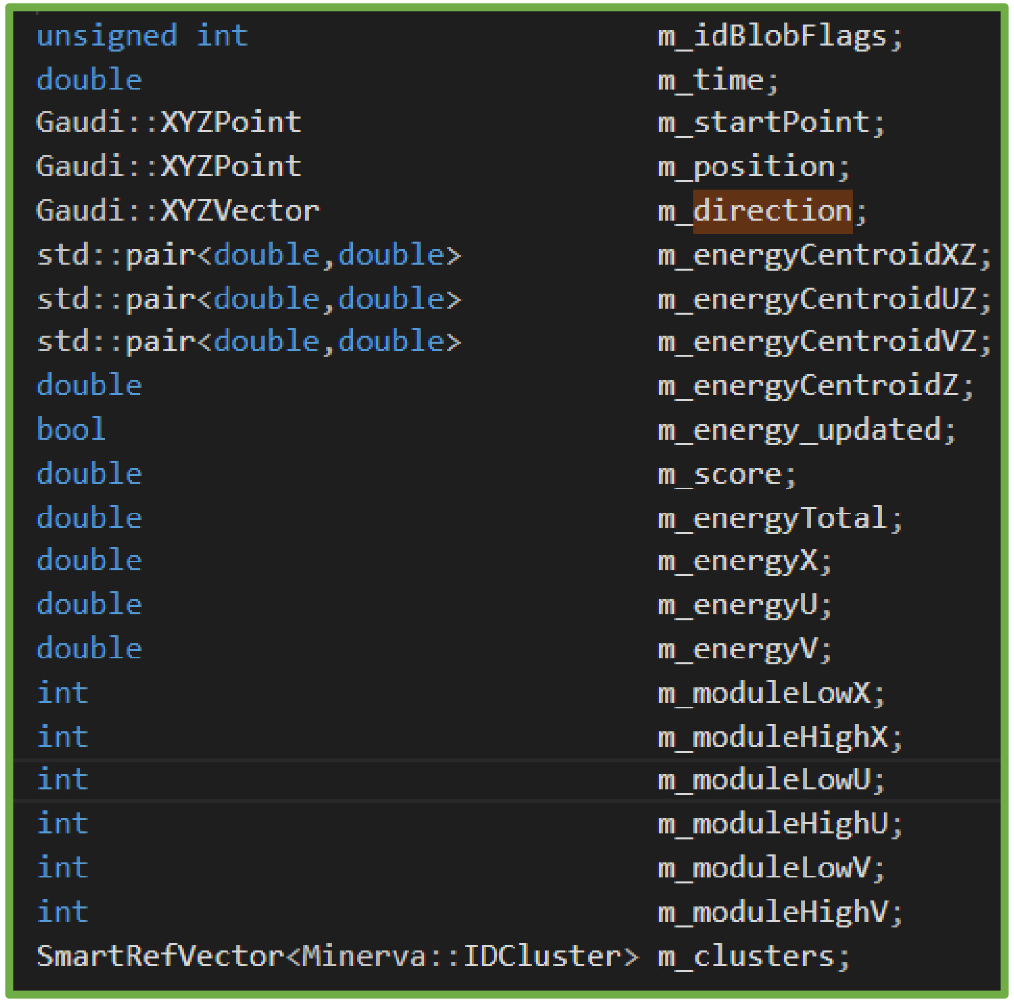
```
Int_t          n_blobs_id;
Int_t          blob_id_idx[19];   //[n_blobs_id]
Int_t          blob_id_subdet[19];   //[n_blobs_id]
Int_t          blob_id_history[19];   //[n_blobs_id]
Int_t          blob_id_size[19];   //[n_blobs_id]
Int_t          blob_id_patrec[19];   //[n_blobs_id]
Double_t       blob_id_e[19];   //[n_blobs_id]
Double_t       blob_id_time[19];   //[n_blobs_id]
Int_t          blob_id_time_slice[19];   //[n_blobs_id]
Double_t       blob_id_startpoint_x[19];   //[n_blobs_id]
Double_t       blob_id_startpoint_y[19];   //[n_blobs_id]
Double_t       blob_id_startpoint_z[19];   //[n_blobs_id]
Int_t          blob_id_clus_idx[19][1500];   //[n_blobs_id]
```

BLOB INFO IN THE DSTs

We could keep a « shower like » object with
- Start position
- Energy
- Direction

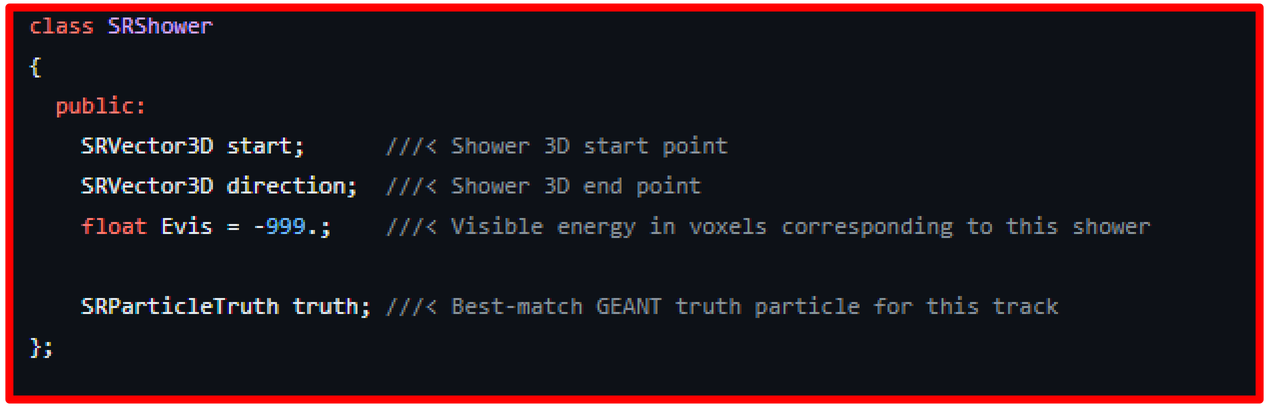
Even if we don't go further in the MNV reco

ALL the BLOB INFORMATIONS



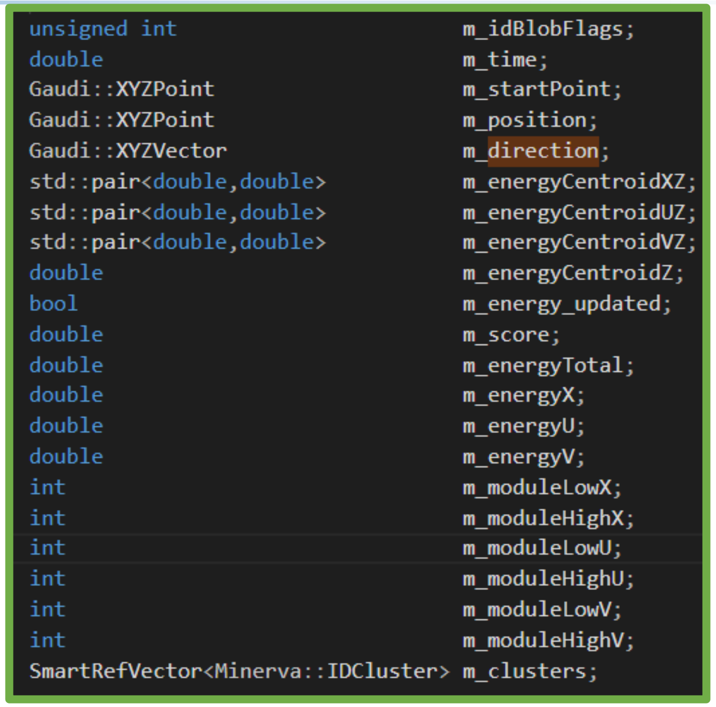
```
unsigned int                          m_idBlobFlags;
double                                m_time;
Gaudi::XYZPoint                       m_startPoint;
Gaudi::XYZPoint                       m_position;
Gaudi::XYZVector                      m_direction;
std::pair<double,double>              m_energyCentroidXZ;
std::pair<double,double>              m_energyCentroidUZ;
std::pair<double,double>              m_energyCentroidVZ;
double                                m_energyCentroidZ;
bool                                  m_energy_updated;
double                                m_score;
double                                m_energyTotal;
double                                m_energyX;
double                                m_energyU;
double                                m_energyV;
int                                   m_moduleLowX;
int                                   m_moduleHighX;
int                                   m_moduleLowU;
int                                   m_moduleHighU;
int                                   m_moduleLowV;
int                                   m_moduleHighV;
SmartRefVector<Minerva::IDCluster>    m_clusters;
```

What's already available?



```
class SRShower
{
  public:
    SRVector3D start;      ///< Shower 3D start point
    SRVector3D direction;  ///< Shower 3D end point
    float Evis = -999.;    ///< Visible energy in voxels corresponding to this shower

    SRParticleTruth truth; ///< Best-match GEANT truth particle for this track
};
```



```
unsigned int                         m_idBlobFlags;
double                               m_time;
Gaudi::XYZPoint                      m_startPoint;
Gaudi::XYZPoint                      m_position;
Gaudi::XYZVector                     m_direction;
std::pair<double,double>             m_energyCentroidXZ;
std::pair<double,double>             m_energyCentroidUZ;
std::pair<double,double>             m_energyCentroidVZ;
double                               m_energyCentroidZ;
bool                                 m_energy_updated;
double                               m_score;
double                               m_energyTotal;
double                               m_energyX;
double                               m_energyU;
double                               m_energyV;
int                                  m_moduleLowX;
int                                  m_moduleHighX;
int                                  m_moduleLowU;
int                                  m_moduleHighU;
int                                  m_moduleLowV;
int                                  m_moduleHighV;
SmartRefVector<Minerva::IDCluster>   m_clusters;
```

SRShower object

- `Start`
- `Direction`
- `Evis`
- `Link to true particle`

Basic Blob direction not computed at that level of reconstruction but a basic computation could be done at the CafsMaker level with endegy centroid + StartPoint.

We might want to add a dE/dX computation at start point + Computation of some direction using centroids + Size of the blob & Type of blob.
Direction seem to be always empty and dEdX is computed in Anatuple level but could be added before or add another step somewhere?

Typical Output format and integration plans

In StandardRecord based on the other implementations:

A **SRMINERVA.h** class with:

- Vector of <MINERVATrack>
- Size_t Number of tracks
- Vector of <MINERVABlob>
- Size_t Number of Blobs

- Use the TrajID to link to the truth particle

MINERVATrack:
Updated SRTracks/SRMinervaTrack with:
- startdEdx (double/float)
- enddEdx (double/float)
- Time of track (double/float)

MINERVABlob:
Like SRShower with:
- number of clusters (int)
- MINERVA type of blob (int) (pattern used to reco)
- dEdX of Blob (double/float)
- Time of Blob (double/float)

In ND_CAFMaker :

A **MINERVARecoBranchFiller**
-> Reads dst files
-> Fills the SMINERVA class