The POLAR Reduced Data Format

in a Nutshell

Neal Gauvin
neal.gauvin@unige.ch

November 29th 2018

The Idea

- Minimal information for end-user analysis.
- Lightweight : easy to distribute.
- No pre-computed values. Done by the end-user.
  Easy nowadays with data framework such as ROOT's RDataFrame.
- stand-alone and self-explanatory.

The File

Filename syntax is polar_xxxxx.yyy.root, where xxxxx.yyy is a float representing the number of revolutions of TG-2 since launch time of the first event.
For instance : polar_02577.674.root
Unique names. No end time. Processing pipeline ensures absence of time overlap.
Contains:

- version : version number.
- sci_data : TTree of scientific data.
- aux_data : TTree of auxiliary data.
- gti : TTree of Great Time Intervals (GTI).

The sci_data Format

pedestal, bad and pre-scaled cosmics events are filtered out.

- unix_time : unix time (double)
- post_cosmic : post_cosmic tag (bool).
- spike : spike tag (bool)
- switched_trigger : event with switched trigger (bool).
- source : event from source (bool)
- energies : list of photons with energy above 3 keV (*vector* $<$ *float* $>$) or triggering bar, sorted by decreasing energy and rounded to 0.1 keV.
- positions : corresponding positions (*vector* $<$ *int* $>$) following [0,1599] convention.
- weights : corr. weights based on dead time: 1/(1-dead_time) (*vector* $<$ *float* $>$, 3 digits precision).
- weight_event : weight for whole event, computed as the multiplication of the weights for each triggering module.
- overflows : true if bar is overflow (*vector* $<$ *bool* $>$).
- scatter_angle : scattering angle used to build modulation curves.
- accumulated_energy : sum of all energies from the triggering modules, including negatives.
- aux_entry : corresponding entry (closest in time) in aux_data (Long64_t). No replication of the auxiliary data in every sci event.

The aux_data Format

unix_time	:	unix time (double).
x,y,z	:	positions in x,y,z direction (double).
vx,vy,vz	:	speed in x,y,z direction (double).
raz,decz	:	(double).
rax,decx	:	(double).
is_orbit_up	:	is TG-2 flying in the upward direction ?
temperature	:	mean temperature of FEEs (float), rounded to $0.1^{\circ}\mathrm{C}$.
fe_rate	:	FE rate ($vector < unsigned\ short >$ (25)).
fe_rate_cosmics	:	FE cosmics rate ($vector < unsigned\ short >$ (25))
time_since_saa	:	time [s] since last SAA.

Conclusion

Processed data of the whole POLAR mission is available in a reduced format.

- Restricted to POLAR Collaboration Member for now.
- Version 1.0
- Total size 1.2Tb
- Used for light-curves and Nicolas's background studies.

Some Code

C++ Snippets to Retrieve Variables

```
// Initialize sci TTree
TTree* t_sci = (TTree*)f->Get("sci_data");
double unix_time;
t_sci->SetBranchAddress("unix_time",&unix_time);
bool post_cosmic;
t_sci->SetBranchAddress("post_cosmic",&post_cosmic);
vector<int>* energies = nullptr;
t_sci->SetBranchAddress("energies",&energies);
Long64_t aux_entry(0);
t_sci->SetBranchAddress("aux_entry",&aux_entry);

// Initialize aux TTree
TTree* t_aux = (TTree*)f->Get("aux_data");
double x;
t_aux->SetBranchAddress("x",&x);
vector<int>* fe_rate = nullptr;
t_aux->SetBranchAddress("fe_rate",&fe_rate);
```

C++ Snippets to Retrieve Variables

```
// Loop on events
Long64_t i = 0;
Long64_t pre_aux_entry=-1;
while( i<t_sci->GetEntries() ){
  t_sci->GetEntry(i++); //Load sci variables
  if(post_cosmic) continue;
  // Load corresponding aux, only if not already loaded.
  if(aux_entry != pre_aux_entry){
    t_aux->GetEntry(aux_entry);
    pre_aux_entry = aux_entry;
  }
  ...
}
// Reset branch addresses and delete vectors before leaving.
t_sci->ResetBranchAddresses();
t_aux->ResetBranchAddresses();
delete energies;
delete fe_rate;
```

How to Draw Vectors From ROOT's Prompt ?

Get your much-beloved ROOT prompt, and load a data file.

```
// Draw all elements of vector energies for all events.
sci_data->Draw("energies")
// Draw first element, ie highest energy, of vector energies.
sci_data->Draw("energies[0]")
// Draw 10th element of vector, if size>10 ! Safe to use.
sci_data->Draw("energies[10]")
// To know the number of photos in the list, ie the size of energies.
sci_data->Draw("@energies.size()")
// The following will not work :
sci_data->Draw("energies.size()")
// In ROOT's terminology, energies means ``loop on all elements''.
// The @ tells ROOT to consider energies as the vector object it is.
```