The ROOT2PAW package v. 3.00 README

Emiliano Mocchiutti

5th December 2005

In this README file there is a description of how to install and use the PAMELA ground data converter (root2paw).

This version converts:

- tracker LEVEL1 ntuples TO tracker LEVEL1 rootples;
- tracker LEVEL2 ntuples TO tracker LEVEL2 rootples;
- tof LEVEL1 ntuples TO tof LEVEL1 rootples;
- anticounter LEVEL1 rootples TO anticounter LEVEL1 ntuples.

INDEX:

---

1) INSTALLATION:

Please refer to the caloritmeter COMMON package (required from this package) for the installation.

---

2) STANDARD USE:

The GroundDataConvert macro has been compiled. The standard use is displayed giving the

```
GroundDataConvert --help
```

command.

To use the interpreted root2paw version cd to the macros directory and start ROOT.
If you have not changed the $HOME/.rootrc file (or you don't know what I am talking about) the PAMELA environment will be automatically loaded and you must see this printout on the terminal:

```
 Welcome to the PAMELA environment!
```

If you don't see this printout or if you start your ROOT session from another directory than the $PAM_MACROS one you must first execute the rootlogon.C file by hand:

```
  bash> root
  root[] .x /mydirectory/pamela/macros/rootlogon.C
```

At this point you should be welcomed to the PAMELA environment.
Once you are inside the PAMELA environment you just have to load the

GroundDataConvert.c file and start the macro.
The inputs needed depend on the type of data you want to convert. Read on section 3 for more informations.
The standard use to convert tracker level2 ntuples to rootples is (say for example file DW_050301_001_level2.rz):

```
  root[] .L GroundDataConvert.c
  root[] GroundDataConvert("/path/to/tracker/ntuples/DW_050301_001_level2.
rz","tracker","2","/path/to/my/output/dir/");
```

The standard use to convert tracker level1 ntuples to rootples is (say for example file DW_050301_001_level1.rz):

```
  root[] .L GroundDataConvert.c
  root[] GroundDataConvert("/path/to/tracker/ntuples/DW_050301_001_level1.
rz","tracker","1","/path/to/my/output/dir/");
```

The standard use to convert anticounter level1 rootples to ntuples is (say for example file DW_050301_001.dat):

```
  root[] .L GroundDataConvert.c
  root[] GroundDataConvert("/path/to/filesfromyoda/DW_050301_00100/",
"anticounter","1");
```

The output filename will be of the form:

dw_YYMMDD_NNNnn.Physics.LevelX.Detectorname.Event.{root/rz}

where X is the processed data level and DetectorName is “Tracker” or “Anticounter”. In the case the output directory is not given it will be placed in the YODA structure (if as filename has been given the path to the YODA unpacked file).
An example of code showing how to read the tracker rootples can be found in doc/examples.c .

---

3) FEATURES:

The GroundDataConvert macro accepts as input variables the following data:

```
int GroundDataConvert(TString filename, TString detector, TString level,
TString outDir = "", Int_t FORCE = 0)
```

  Input variables:

   * filename = AC: path to the YODA directory for a file;
                TRK: path to the ntuple filename or, if the ntuple has been moved in the YODA structure (under Physics/LevelX/ where X is the data level), path to the YODA directory.
                TOF: path to the ntuple filename or, if the ntuple has been moved in the YODA structure (under Physics/LevelX/ where X is the data level), path to the YODA directory.

   * detector = can be “anticounter”, “tracker” or “tof”.

   * level = the ntuple/rootple level data that has to be converted.

   * outDir = directory where the output data has to be stored. If as filename has been given the path to the YODA file structure this input is not mandatory and by default the program will put the output in
    /yodafilestructure/Physics/LevelX/
where X is the level of the data processed. If as input filename has been

given the name of the file the outDir input is mandatory.

* FORCE = when set to 0 (default) the program will check the existence of output data and will abort processing if the output file is found. To force the overriding set FORCE to 1.

---

4) KNOWN BUGS:

- No known bugs, for any problem contact Emiliano.Mocchiutti@ts.infn.it

---

5) CHANGELOG:

```
//
// 2.00 - 3.00  (2005/11/29): compiled.
//
// 1.05 - 2.00  (2005/10/07): added TOF and TRIGGER level1 conversion
//                            (from PAW to ROOT).
//
// 1.04 - 1.05  (2005/10/04): tracker version 2.00 conversion.
//
// 1.03 - 1.04  (2005/09/07): small bug unloading libraries fixed.
//
// 1.02 - 1.03  (2005/08/03): changes for working on 64 bit machines.
//
// 1.01 - 1.02  (2005/07/21): don't load yodaUtility.c anymore and use
//                            clone routines in CaloFunctions.h
//
// 1.00 - 1.01  (2005/07/14): small change in the call to getLEVname
//                            (changed to be compiled).
//
// 0.1 - 1.00  (2005/07/05): working, it converts AC level1 rootples to
//                           ntuples and TRK level1 and level2 ntuples to
//                           rootples.
//
// v. 0.1      (2005/06/15): created.
```