EVNDISP Manual
IACT event analysis and display
v4.00

Gernot Maier (DESY)
.....

SVN *Rev* : 361
SVN *Date* : $2011-09-2916:25:12+0200(Thu, 29Sep2011)$
SVN *Author* : *maierg*

Abstract

eventdisplay is a complete package for VERITAS and CTA analysis (whatever 'complete' means...) The package consists of several analysis steps and tools:

1. evndisp (calibrate and parametrize images, event reconstruction, stereo analysis)
2. mscw_energy (use lookup tables to produce msw, msl, and energies)
3. anasum (produce maps and calculate analysis results)
4. shared library tools and macros (produce the energy spectrum and integral fluxes, plot maps, plot sensitivities, etc.)
5. makeEffectiveArea (calculate effective areas)
6. trainTMVAforGammaHadronSeparation (tools to train MVA and optimize cuts)
7. ...

This is a very incomplete manual, started in September 2009. Please help updating it.

Contents

Chapter 1

Documentation

EVNDISP is work-in-progress and the documentation is not in the state it is supposed to be. Apart from the information in this manual, other sources for help are:

README files

INSTALL: information on installation the analysis package, dependencies, environmental variables

README.CTA : short description of a typical CTA analysis

README.VTS: description of a typical VERITAS analysis

AUTHORS: author description

Description and command line options for the different software parts:

README.EVNDISP:

README.MSCW_ENERGY:

README.ANASUM:

README.EFFECTIVEAREA:

README.ANALYSISLIBRARY:

README.SCRIPTS:

README.MACROS:

WIKI pages

The EVNDISP manual for VERITAS users:
`http://veritas.sao.arizona.edu/wiki/index.php/Eventdisplay_Manual`

Chapter 2

Introduction

Chapter 3

Installation and auxiliary data files

3.1 Auxiliary data files

The auxiliary data files contain information needed for the analysis. This might be files describing the detector geometry, lookup tables, effective areas file, etc. We assume in the following that these files are located in the directory *$EVNDISPDATA*

3.1.1 Detector description

VERITAS

CTA

No detector description is needed, since the telescope and array description is read directly from the DST file (*telconfig* tree). The converter from hessio to EVNDISP DST format needs a subarray file, a simple list of telescopes to be selected from the corresponding hyper array. The subarray files for prod1 can be found in *$EVNDISPDATA/DetectorGeometry/*.lis*.

Chapter 4

eventdisplay - calibration, image analysis and stereo reconstruction

Chapter 5

CTA analysis

5.1 General concept

To use Eventdisplay for CTA analyses, you first have to convert the *simtel.gz* file into ROOT format. This you have to do for each subarray separately. Afterwards you can analyze this subarray file running *eventdisplay* in the standard way, which means:

CTA.convert_hessio_to_VDST convert simtel.gz files into ROOT format

eventdisplay image cleaning & calculation of telescope parameters & reconstruction of the direction and core; display

mscw_energy train and use lookup tables to estimate the energy and mean scaled parameters

trainTMVAforGammaHadronSeparation optimize cuts or train MVA

makeEffectiveArea make effective areas

$EVNDISPSYS/macros/plot_sensitivity.C plot sensitivity curves

(For most of these steps there are scripts to run the analysis on the DESY batch system. Use them to simplify your life.)

5.2 Analysis

5.2.1 Step 1: Converter

In this step a simulation data file in hessio format is converted to the EVNDISP DST format (see 6.2). These are the possible options to run the converter:

```
$EVNDISPSYS/bin/CTA.convert_hessio_to_VDST

c_DST: A program to convert hessio data to EVNDISP DST files (v.4.00)

Syntax: ./bin/CTA.convert_hessio_to_VDST [ options ] [ - | input_fname ... ]
```

```
Options:
   -v                (More verbose output)
   -q                (Much more quiet output)
   -s                (Show data explained)
   -S                (Show data explained, including raw data)
   --history (-h)    (Show contents of history data block)
   -i                (Ignore unknown data block types)
   --max-events n    (Skip remaining data after so many triggered events.)
   -a subarray file  (list of telescopes to read with FOV.)
   -o dst filename   (name of dst output file)
   -f on=1/off=0     (write FADC samples to DST file; default=0)
```

The following options are necessary: -a and -o. Note the limitations of the DST format (6.2.1).
Note that many of the CTA simulation files contain data for a so called hyper array. To select the actual array, the corresponding subarray has to be specified in a ASCII text file (called *subarray* file in the following). The following example file selects an array consisting of telescopes 63, 19, 67, and 33, each telescope with a field of view of 8 degrees:

```
63 8
19 8
67 8
33 8
```

Subarray files for most of the typical arrays used in the CTA sensitivities studies are part of the analysis file package (see 3.1). Note that a subarray file is always needed, even if all telescopes from the hessio file are read out.
For a typical run, the following command line should be used:

```
./CTA.convert_hessio_to_VDST -a subArray.list -o dstfile.dst.root \
        gamma_run12241.simhess.gz
```

For FADC analysis, add the option *-f 1*. Note again the limitations of the DST format (6.2.1).

5.2.2 Step 2: Display (event-by-event)

It is always useful to look at events in the display. To do this, it is best to select a small subarray with the *-teltoana* option in evndisp. The display might otherwise be quite slow in responding to your input due to the large number of objects to be drawn.
A typical command line to look at events might be (remove the *-traceanalysis=1* command for dst files without trace information):

```
$EVNDISPSYS/bin/evndisp -display=1 \
-reconstructionparameter ./EVNDISP.reconstruction.runparameter \
-useFixedThresholds -imagethresh=10.0 -borderthresh=5.0 \
-l2setspecialchannels nofile \
-sourcefile tt.v2.root -traceanalysis=1
```

5.2.3 Step 3: Calibration, image analysis and stereo reconstruction

5.2.4 Step 1 & 3 combined [USE]

Run the converter and Eventdisplay for a specific subarray: use script $EVNDISPSYS/scripts/CTA/CTA.EVNDISP.sub_convert_and_analyse_MC_VDST.sh
NOTE: The image cleaning thresholds can be specified in the file $EVNDISPDATA/ParameterFiles/EVNDISP.reconstruction.runparameter file either for all telescopes to the same values or for each telescope type seperately.

5.2.5 Step 4: mscw_energy

If you use standard configurations maybe some lookup tables already exist (ask Gernot or Heike where you could find them).
If not, you have to create them yourself with
$EVNDISPSYS/scripts/CTA/CTA.MSCW_ENERGY.sub_make_tables.sh
If you have your lookup tables you have to run mscw_energy for estimating the energy of each event:
$EVNDISPSYS/scripts/CTA/CTA.MSCW_ENERGY.sub_analyse_MC.sh

5.2.6 Step 5: Optimize cuts or train MVA

5.2.7 Step 6: Effective Areas

To calculate effective areas, look at:
$EVNDISPSYS/scripts/CTA/CTA.EFFAREA.sub_analyse.sh
For the first time you have to use mscw files as input since MCpars has to be analysed once. If you change your cuts afterwards (but not the number of input files) you can use the faster version by using the effective area file as input (helps a lot for protons).

Chapter 6

Input data format

6.1 VBF - VERITAS Bank Format

6.2 DST - data summary tree

The DST format is a simple ROOT tree containing standard C++ variables only (no class data).

6.2.1 Limitations

The implementation requires the hardwiring of the maximum number of telescopes, channels, etc. These values can be found in `inc/VGlobalRunParameter`, for example:

```
// HARDWIRED MAXIMUM NUMBER OF TELESCOPES AND CHANNELS, etc.
// maximum number of telescopes
#define VDST_MAXTELESCOPES  100
// maximum number of channels per telescopes
#define VDST_MAXCHANNELS    12000
// maximum number of summation windows
// (=maximum number of samples per FADC trace)
#define VDST_MAXSUMWINDOW   64
// maximum number of time slices for pedestal calculation
#define VDST_PEDTIMESLICES 5000
// maximum number of arrayreconstruction method
#define VDST_MAXRECMETHODS  100
// maximum number of timing levels
#define VDST_MAXTIMINGLEVELS 10
```

NOTE: These numbers determine the memory requirements of *evndisp* and *CTA.convert_hessio_to_VDST*.
NOTE: *evndisp* must be compiled with the same settings as the writing program.

Chapter 7

Detector Setup

7.1 Telescope types

Different telescope types (e.g. mid-size and small-size telescopes, telescopes with different FOV, etc) are assigned a telescope type number in the code, this number is as well written to the data trees. The telescope type contains the mirror shape (DC, Parabolic, SC), the mirror area (m^2), the field of view ([deg]) and the pixel size ([deg]). For VERITAS, the telescope type correspond simply to the different telescope numbers (and are therefore 0,1,2,3).
For clarification, this is the corresponding code bit from src/CTA.convert_hessio_to_VDST.cpp:

```
fTelescope_type  = TMath::Nint(pix_size*100.);
fTelescope_type += TMath::Nint(fFOV*10.)*100;
fTelescope_type += TMath::Nint(fMirrorArea)*100*10*100;
// all large telescopes are parabolic, all others are Davies-Cotton (hardwired)
if( fMirrorArea > fParabolic_mirrorArea )      fTelescope_type += 100000000;
// Schwarzschild-Couder: check number of mirrors
else if( fNMirrors == fSC_number_of_mirrors )  fTelescope_type += 200000000;
```

Note: There is currently no way to determine the mirror/telescope shape (parabolic, Davies-Cotton, etc) from the hessio file. This is why the mirror area and the number of mirrors is used. The parabolic shape is assigned to all telescopes with a mirror area > 400 m^2. Schwarzschild-Couder Design are all telescopes with 2 mirrors only.