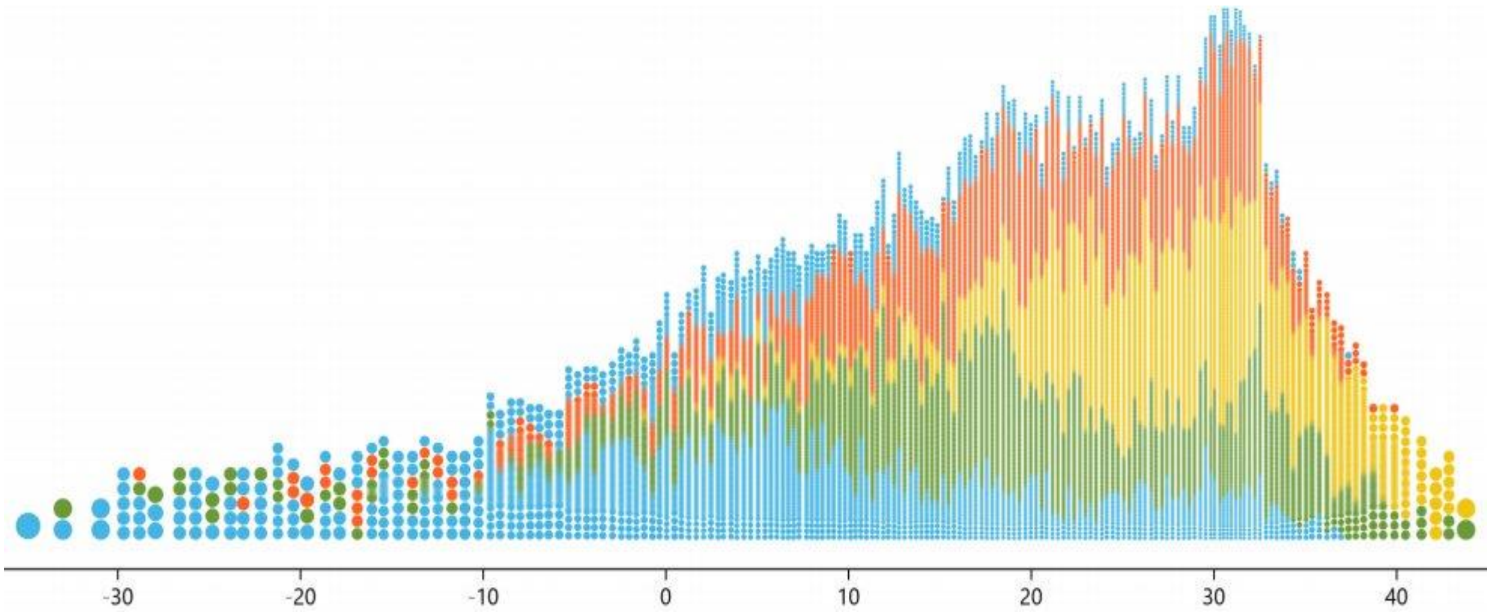
Tutorial category: Expert mode

Histograms & Cuts



Version 1.0

Date 10/02/2020

Official MadAnalysis 5 website : http://madanalysis.irmp.ucl.ac.be/

Goals of this tutorial

- Applying cuts in the analysis
- Plotting some observables in the analysis
- Extracting the cut efficiencies and the plot data.

Requirements

- MadAnalysis 5 is installed on your system and has been launched successfully at least one time. The collection of example samples is installed too.

- A first experience with the expert mode (see the tutorial “First steps in the expert mode”).

- Basic skills in C++ programing.

- You have chosen which text editor is your favorite ☺

Part 1

Implementation in the analysis

Prerequisites to this tutorial

1. MadAnalysis 5 has been downloaded.
   (see tutorial "Normal Mode - 2 - Installation & first launch")

2. MadAnalysis 5 has been launched at least one time and the building the SampleAnalyzer library has worked properly.
   (see tutorial "Normal Mode - 2 - Installation & first launch")

3. The collection of example samples has been installed through the MadAnalysis 5 console:

```
ma5> install samples
```

   (see tutorial "Normal Mode - 2 - Installation & first launch")

4. Then you have created an empty analysis within the expert mode of madanalysis.
   (see tutorial "Export Mode - 1 - First steps in expert mode")

5. You have uncommented an example in the Execute function.
   (see tutorial "Export Mode - 2 – Understanding the predefined analysis")

Declaring histograms and cuts

- For declaring an histogram or a selection cut, you need to edit the source file `MyAnalysis.cpp` and to change the content of the function `Initialize()`.

```
MAbool MyAnalysis::Initialize(const MA5::Configuration& cfg,
const std::map<std::string,std::string>& parameters)
{
  ........
}
```

- The first step consists in defining a selection region for the analysis. By default, we call this region “myregion”.

```
Manager()->AddRegionSelection("myregion");
```

Declaring cuts

- For declaring a cut called “My first cut”, you have just to implement the following line:

```
Manager()->AddCut("My first cut");
```

Of course, the name “My first cut” is not very meaningful and usually people choose a name which describes the criterion on which you cut. But this name must be unique in order to identify the cut.

- This an example with several cuts:

```
Manager()->AddCut("MET > 20 GeV");
Manager()->AddCut("I would like mu with PT > 50 GeV“)
```

The order of the cuts in the declaration is important because this order will be used for presenting the final result in a cut-flow chart.

Declaring histograms

- For declaring a 2D histogram called “My plot” with a number of bins equal to 200 and x-axis going from 10 to 1000, the corresponding implementation is the following:

```
Manager()->AddHisto("My plot",200,10.,1000.);
```

Similarly to the selection cut, the name is used for identifying the histogram and must be unique. The three last arguments are also mandatory.

- This an example with several plots:

```
Manager()->AddHisto("MET", 100,0.0,1000.0);
Manager()->AddHisto("PT(mu)", 100,0.0,5000.0);
```

The declaration order of the plots is not relevant.

- Only 2D histograms are available if you use only the SampleAnalyzer architecture. To settle this limitation, an alternative will be presented in the last part of this tutorial.

Filling histograms or applying cuts

- For filling an histogram or for applying a selection cut, you need to edit the source file `MyAnalysis.cpp` and to change the content of the function `Execute()`.

```
MAbool MyAnalysis::Execute(SampleFormat& sample, const EventFormat& event)
{
  ........
}
```

- By default, one event has a weight of 1 in a the histogram or in a counter. It is usually convenient to specify (at the beginning of the function Execute) the weight of the event for getting at the end the proper normalization of the histogram. It can be done by the following lines:

```
MAfloat32 __event_weight__ = 1.0;
if (weighted_events_ && event.mc()!=0) __event_weight__ = event.mc()->weight();
Manager()->InitializeForNewEvent(__event_weight__);
```

Applying a cut

- We assume that we would like to remove all events for which the missing transverse energy is less than 20 GeV. The corresponding lines are given below:

```
MAbool toRemove = PHYSICS->Transverse->EventMET(event.mc()) < 20.0;
if (!Manager()->ApplyCut(toRemove, "MET > 20.0")) return true;
```

The first line checks if the MET of the event is less than 20 GeV. The boolean `toRemove` is equal to true if the event should be removed or false in the opposite case.

The second line applies this criterion to the event with the help of the command `ApplyCut`. The first argument corresponds to the condition of rejection: if it is equal to true, it means the event must be removed. The second argument correspond to the name of cut given in the declaration.

`ApplyCut` returns false if the event must be removed. In this case, the analysis of the current event must be stopped and we must move to the next event. That's why the meaning of the “`return true`” at the end of the line.

Histograms & Cuts

Filling an histogram

- We assume that we would like to plot the transverse missing energy of the event. So we must fill the histogram with this observable like that:

```
Manager()->FillHisto("MET", PHYSICS->Transverse->EventMET(event.mc()) );
```

The argument of `FillHisto` is the name of the histogram given at the declaration. The second argument is the value of the observable i.e. in our case the MET.

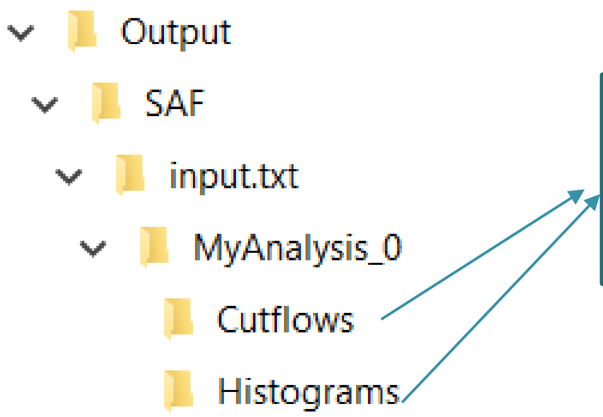
Part 2

Getting the results with the SAF files

Focus on the `Output/SAF` folder

- We assume that you have launched an analysis which contains histograms and selection cuts.

- Results of the analyses can be found in the folder `Output/SAF`, particularly in the sub-folders `Histograms` and `Cutflows`.



Theses folders are now NOT empty because cuts or histograms has been defined in your analysis.

Look on the histograms

- This is an extract of the file `Output/SAF/input.txt/MyAnalysis_0/Histograms/histos.saf`:

```
<Histo>
  <Description>
    "MET"
    # nbins   xmin           xmax
      100     0.000000e+00   1.000000e+03
    # Defined regions
      myregion    # Region nr. 1
  </Description>
  <Statistics>
      2000         0                # nevents
      1.285186e+2  0.000000e+00     # sum of event-weights over events
      2000         0                # nentries
      2.570373e+2  0.000000e+00     # sum of event-weights over entries
      3.435435e+2  0.000000e+00     # sum weights^2
      4.172962e+2  0.000000e+00     # sum value*weight
      1.390071e+2  0.000000e+00     # sum value^2*weight
  </Statistics>
```

Look on the histograms

- This is an extract of the file `Output/SAF/input.txt/MyAnalysis_0/Histograms/histos.saf:`

```
<Histo>
  <Description>
    "MET"
    # nbins    xmin            xmax
      100      0.000000e+00    1.000000e+03
    # Defined regions
      myregion     # Region nr. 1
  </Description>
  <Statistics>
      2000          0               # nevents
      1.285186e+2   0.000000e+00    # sum of event-weights over events
      2000          0               # nentries
      2.570373e+2   0.000000e+00    # sum of event-weights over entries
      3.435435e+2   0.000000e+00    # sum weights^2
      4.172962e+2   0.000000e+00    # sum value*weight
      1.390071e+2   0.000000e+00    # sum value^2*weight
  </Statistics>
```

2 columns:

- Data with positive event-weight on the left
- Data with negative event-weight on the right

Look on the histograms

- This is an extract of the file `Output/SAF/input.txt/MyAnalysis_0/Histograms/histos.saf`:

```
<Data>
      0.000000e+00   0.000000e+00    # underflow
      2.570373e+222  0.000000e+00    # bin 1 / 100
      0.000000e+00   0.000000e+00    # bin 2 / 100
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
      0.000000e+00   0.000000e+00
```

Look on the histograms

- This is an extract of the file `Output/SAF/input.txt/MyAnalysis_0/Histograms/histos.saf`:

```
<Data>
    0.000000e+00   0.000000e+00    # underflow
    2.570373e+222  0.000000e+00    # bin 1 / 100
    0.000000e+00   0.000000e+00    # bin 2 / 100
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
    0.000000e+00   0.000000e+00
```

2 columns:

- Data with positive event-weight on the left
- Data with negative event-weight on the right

Look on the cut-flow

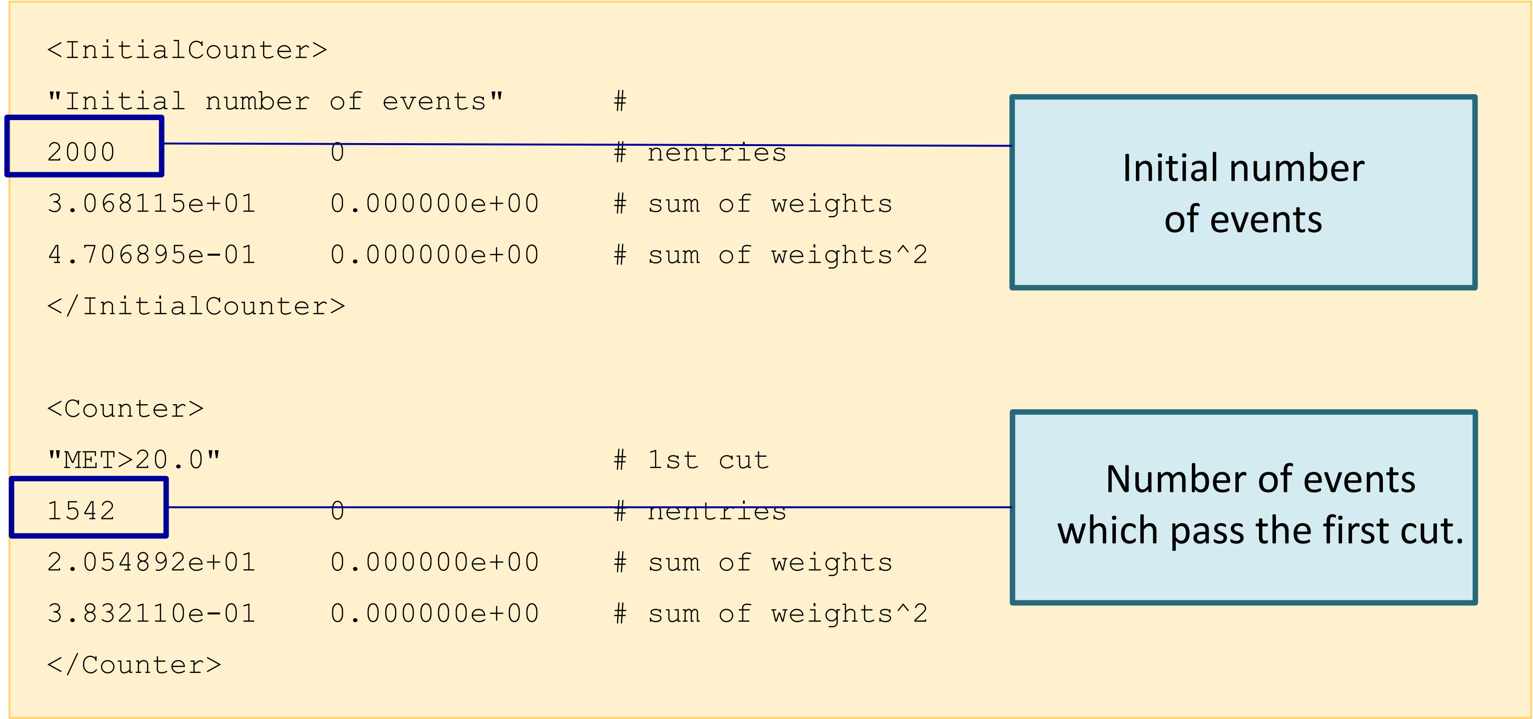
- This is an extract of the file `Output/SAF/input.txt/MyAnalysis_0/Cutflows/myregion.saf:`

```
<InitialCounter>
"Initial number of events"      #
2000            0               # nentries
3.068115e+01    0.000000e+00    # sum of weights
4.706895e-01    0.000000e+00    # sum of weights^2
</InitialCounter>

<Counter>
"MET>20.0"                      # 1st cut
1542            0               # nentries
2.054892e+01    0.000000e+00    # sum of weights
3.832110e-01    0.000000e+00    # sum of weights^2
</Counter>
```

Look on the cut-flow

- This is an extract of the file `Output/SAF/input.txt/MyAnalysis_0/Cutflows/myregion.saf:`



```
<InitialCounter>
"Initial number of events"      #
2000            0               # nentries
3.068115e+01    0.000000e+00    # sum of weights
4.706895e-01    0.000000e+00    # sum of weights^2
</InitialCounter>

<Counter>
"MET>20.0"                      # 1st cut
1542            0               # nentries
2.054892e+01    0.000000e+00    # sum of weights
3.832110e-01    0.000000e+00    # sum of weights^2
</Counter>
```

Initial number of events

Number of events which pass the first cut.

How to analyze these results?

- A ReportGenerator program will be available soon for producing automatically a PDF report from the SAF files.
- Up to now the user must writes a script in order to extract the results and to plot the histograms.

Part 3 (optional)

Alternative way to implement histograms and cuts

ROOT histograming

- If ROOT is installed on your machine and if it is detected by MadAnalysis 5, all ROOT methods can be implemented in your analysis, in particular for histograming.

- The first step is to declare in to declare in the header file `MyAnalysis.h` the different histograms. The ROOT equivalents of the plots performed by MadAnalysis 5 are the TH1F. PS: it is not necessary to include the header file `<TH1F.h>`

```
TH1F* plot_MET;
```

- In the source file `MyAnalysis.cpp`, adding the definition of the histogram in the `Initialize` function.

```
plot_MET = new TH1F(“MET”, ”MET”, 100, 0, 1000);
```

- Filling the histogram in the `Execute` function.

```
plot_MET -> Fill( PHYSICS->Transverse->EventMET(event.mc()) );
```

Histograms & Cuts

ROOT histograming

- Finally, the histograms can be saved in a ROOT file at the step of the `Finalize` function.

```
TFile* output = new Tfile(“output.root”, ”RECREATE”);
plot_MET -> Write();
Output -> Close();
```

GREAT
JOB!

About this document

- The present document is a part of the tutorial collection of the package MadAnalysis 5 (MA5 in abbreviated form). It has to be conceived to explain in a practical and graphical way the functionalities and the various options available in the last public release of MA5.

- The up-to-date version of this document, also the complete collection of tutorials, can be found on the MadAnalysis 5 website :

  http://madanalysis.irmp.ucl.ac.be/

- Your feedback interests ourselves (bug reports, questions, comments, suggestions). You can contact the MadAnalysis 5 team by the email address : ma5team@iphc.cnrs.fr

Change log

Version	Date	Update
1.0	10/02/2020	Creation