A procedure for extracting the efficiency of the trigger chambers from data + simulations

Version 2

9.07.2010

Data versus simulations

From the experimental data two kind of results can be extracted:

- relevant for the physics analysis: triggers, tracks, scalers (they contain information on the nature of the investigated physics phenomena) ---> go in the Events Summary Data (ESD) after the reconstruction

- relevant for the detector functioning/calibration: pedestals, noise, dead channels, alignment (they depend on the current status/configuration of the detection system) ---> go in the Offline Conditions Data Base (OCDB)

The Monte Carlo simulations can do:

- a model for the collisions ---> Pythia + ... (try to guess/anticipate as well as possible some of the physics characteristics of the investigated phenomena)
- a model for the detector ---> try to implement as many as possible of the detector features and tune them to the data according to the OCDB

The efficiency correction on physics results

- simulations: efficiency = “reconstructed tracks”/“reconstructible tracks”

- data: we have the “reconstructed tracks” and the efficiency (calculated with the help of the simulations) and we want to get the “reconstructible tracks” in a certain region of the phase space where we want to do our physics

This is correct, if the calculation of the efficiency takes into account all the effects of the detector!

The case of the muon trigger chambers

- what do we have in the simulations?
  - geometry (alignment)/segmentation
  - trigger algorithm
  - masks (dead/noisy local/global channels)
  - RPC efficiency = 100% (with a granularity per local board)

What we don't have in the simulations?

- “real” RPC efficiencies: probability to induce a charge on a pad (strip) when a charged particle (muon) is crossing the gas of the chamber (there are “hits”)
- other factors affecting the trigger results, like the time alignment of the four trigger planes, which could eventually be included in the RPC efficiency

How to determine these efficiencies?

- make a ration between data and simulations, under certain conditions

- use data and do simulations with ¾ trigger coincidence; this means that we can “see” in the data events where one chamber is not efficient (does not give a signal) where the track crosses it. If it is a matter of segmentation, geometry (alignment) or dead channels and if it was correspondingly considered in the simulations, the ratio data/simulation will cancel this source of efficiency loss; what is left, is what is not present in the simulations

What is the highest granularity with which the “residual” efficiency can vary?

- possibilities: trigger chamber (1, 2, ... 4), chamber slat (1, 2, ... 18)
  or local board (1, 2, ... 234) and for each of the two cathode planes (eficiencies for the two cathode planes, though different, are not statistically independent...)

Once the “residual” efficiencies determined and introduced in the simulations, these can be used to determine the trigger efficiency per reconstructed track in coordinates of the working phase space.

We thus have two kinds of efficiency: the "residual" efficiency called R (calculated per local board) and the "a posteriori" trigger efficiency (calculated per phase space element).

Coordinates of the working space: (p, phi, theta, dca)

Goal of the efficiency calculation:

This point is unclear to me: it may be the reason why (and where) to use track extrapolation.

- if we have N' reconstructed (muon) tracks matched with a trigger in the element of the phase space (delta_p, delta_phi, delta_theta, delta_dca), what was the most probable number of reconstructed (muon) tracks matching or not with a trigger.

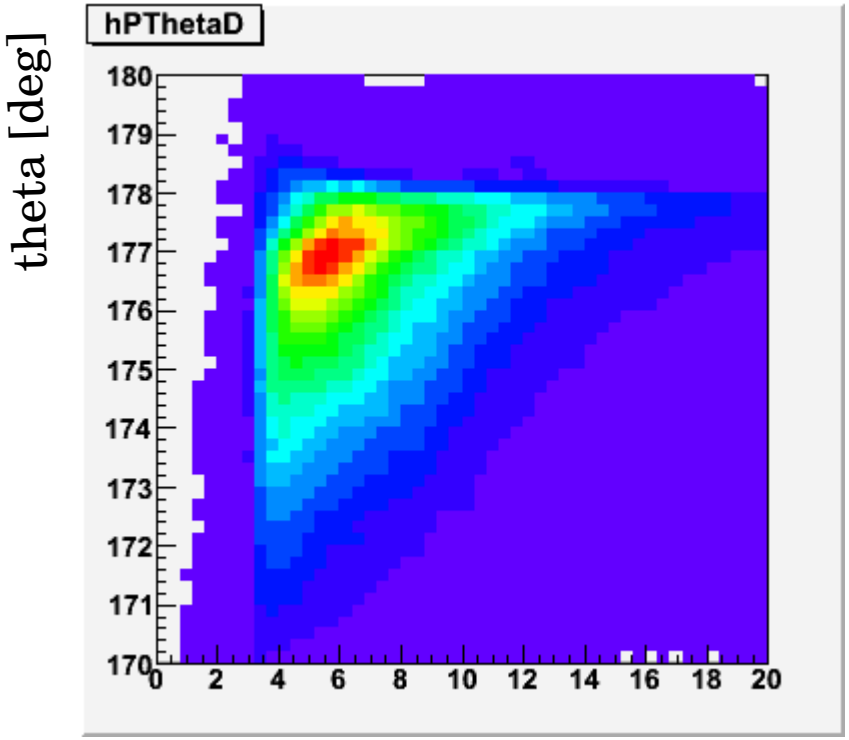
The direct efficiency determination (in the phase space of the reconstructed tracks)

I don't understand which "efficiency" is calculated here and how was it calculated!

Data runs 119156 to 120079 (only longer runs selected)
Simulation PDC09 113021 to 113699

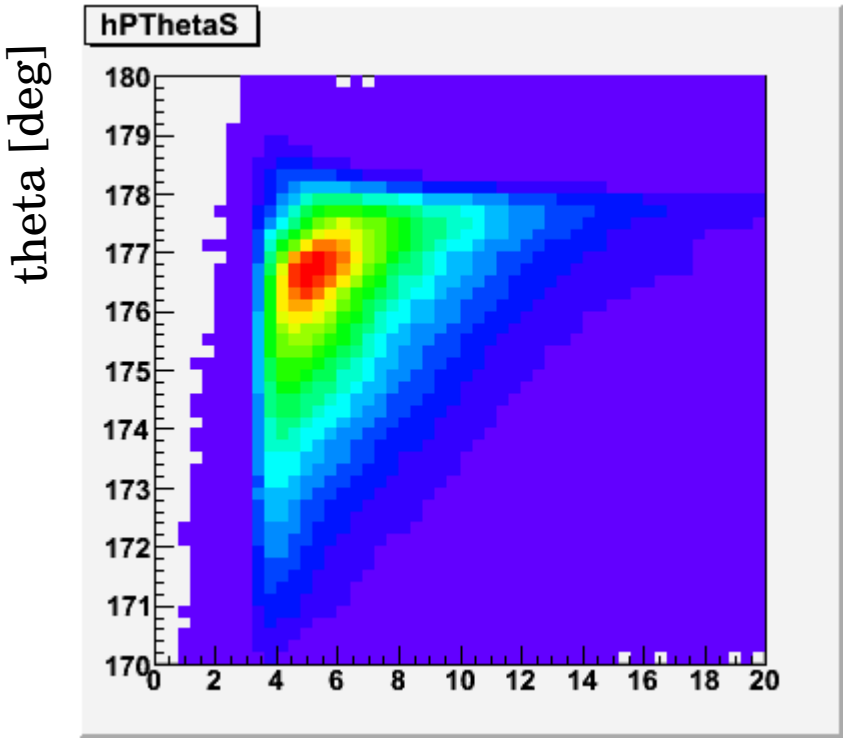
all reconstructed tracks

data



(from AliESDs.root)

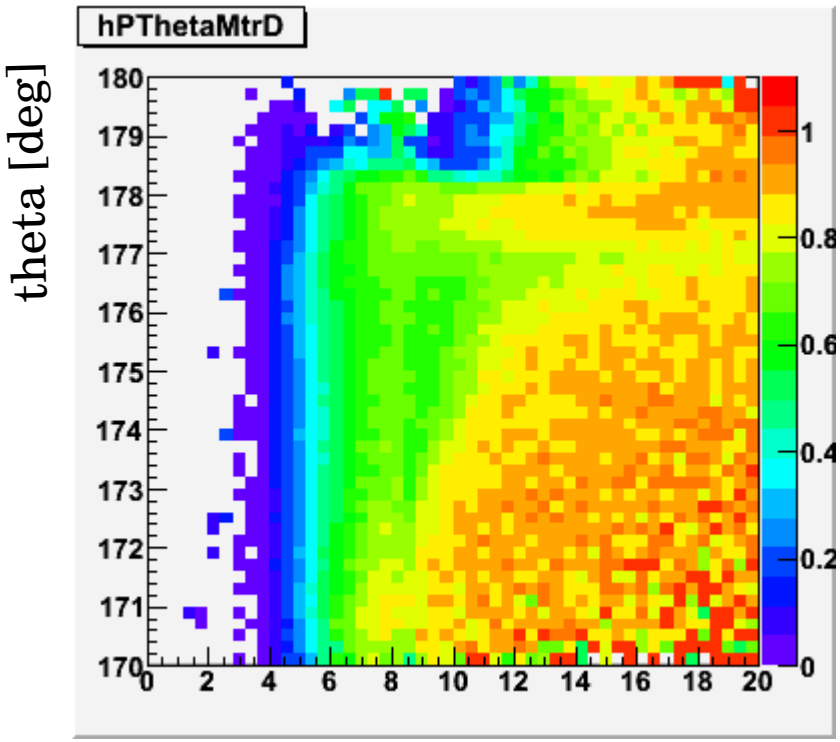
simulation



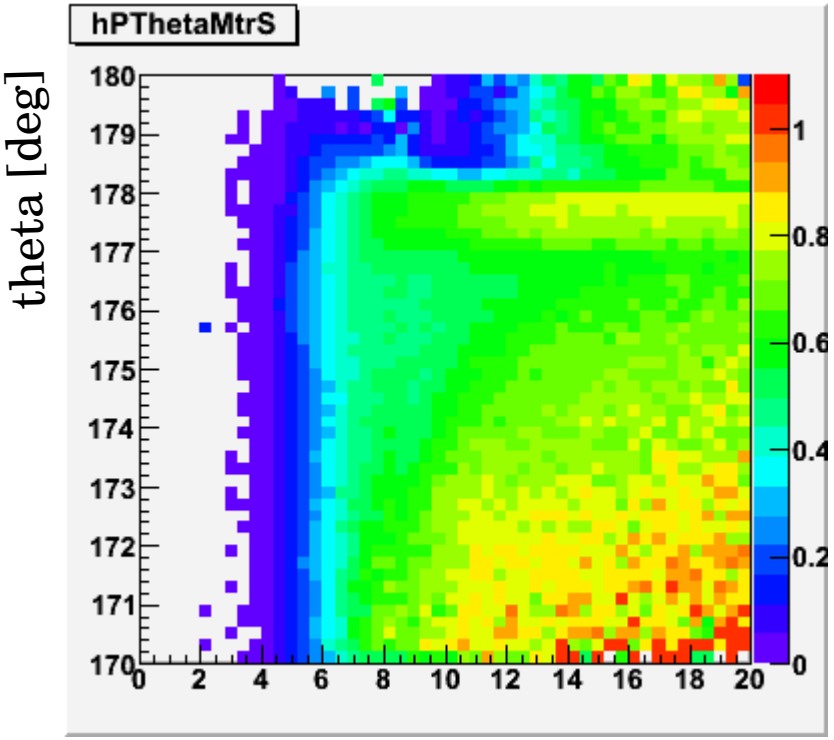
Tracks matched with a trigger, divided by all tracks

- all tracks are muons + hadrons, matched tracks are only muons
- proportion hadrons/muons in the simulation is in principle different from that in the data ---> if we divide in this picture data by the simulation, we will get for the efficiency values larger than 1 !

data



simulations

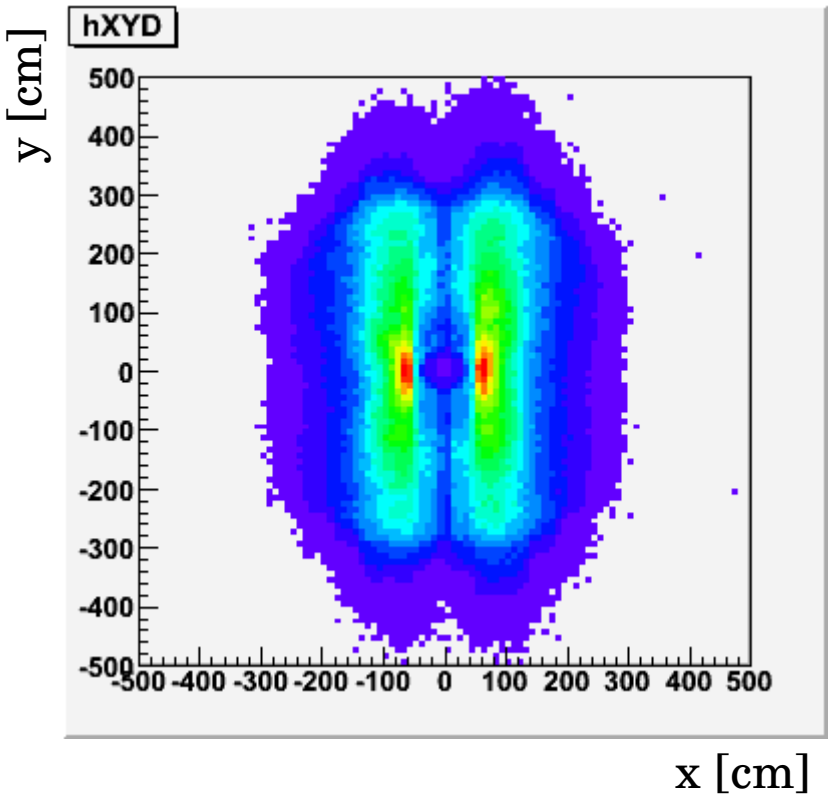


So, we must pass by the RPC efficiency

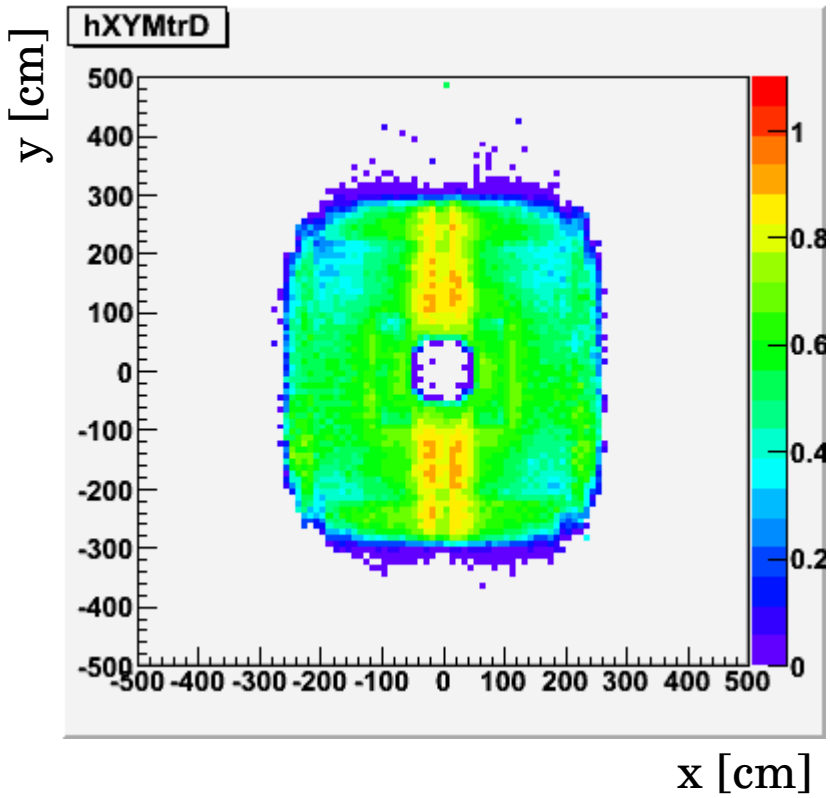
- we have to use coordinates in which the structure of the trigger plane becomes visible: the (x,y) impact point of the track extrapolation will be considered in the next plots

(data)

all tracks



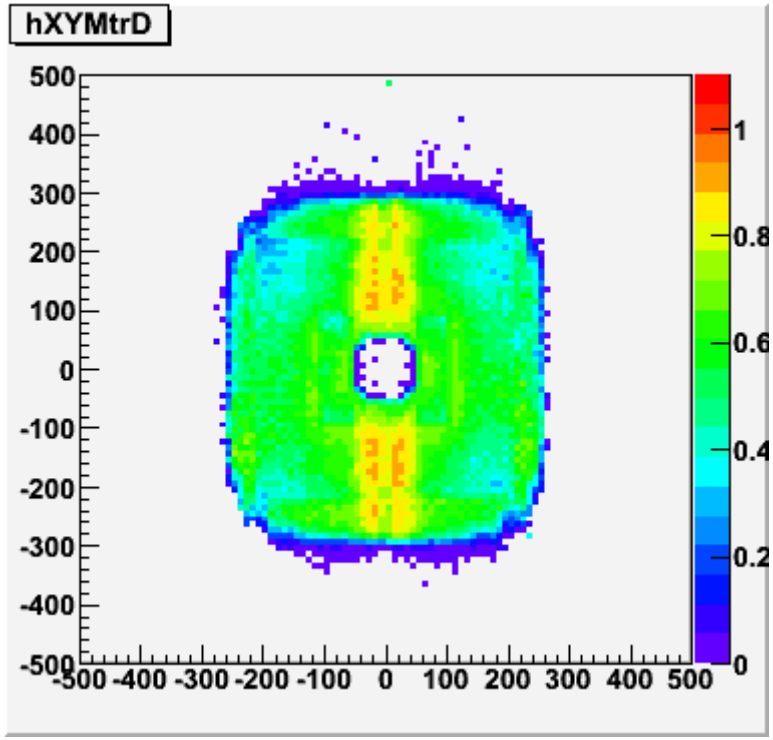
trigger/all



We need the same reference in data as in the simulation, so we can chose the reconstructed tracks matched with the trigger (i.e. muons).

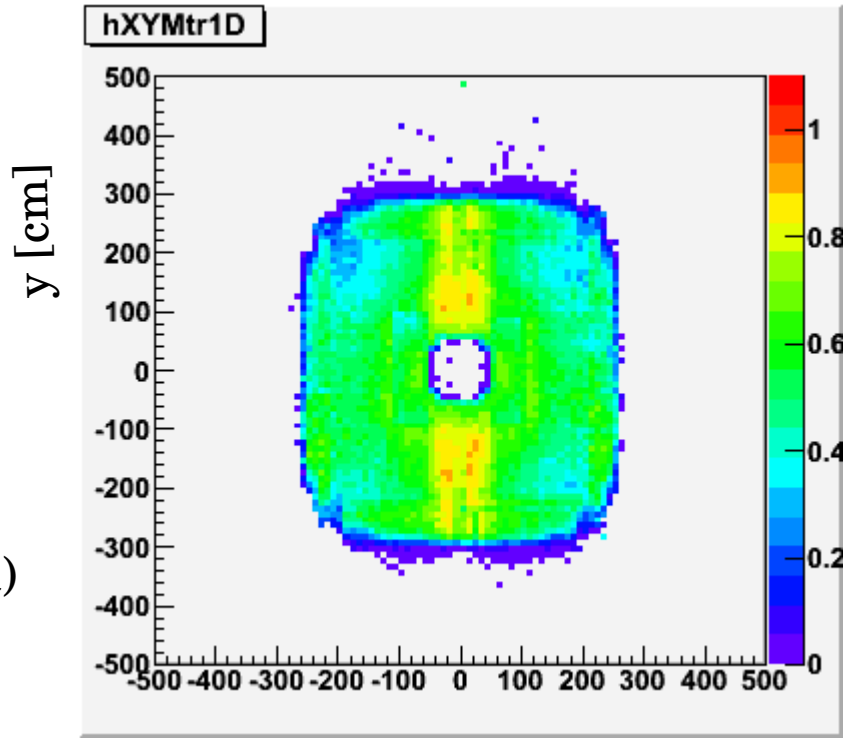
In the condition that the track matches a trigger, we can separate four cases, in which we ask that a given plane takes part in the trigger decision, that is it is “efficient”.

trigger/all

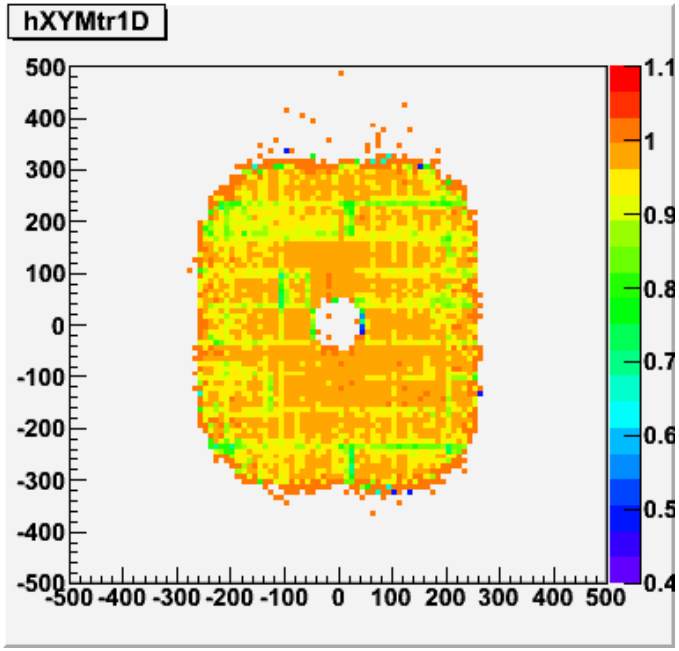


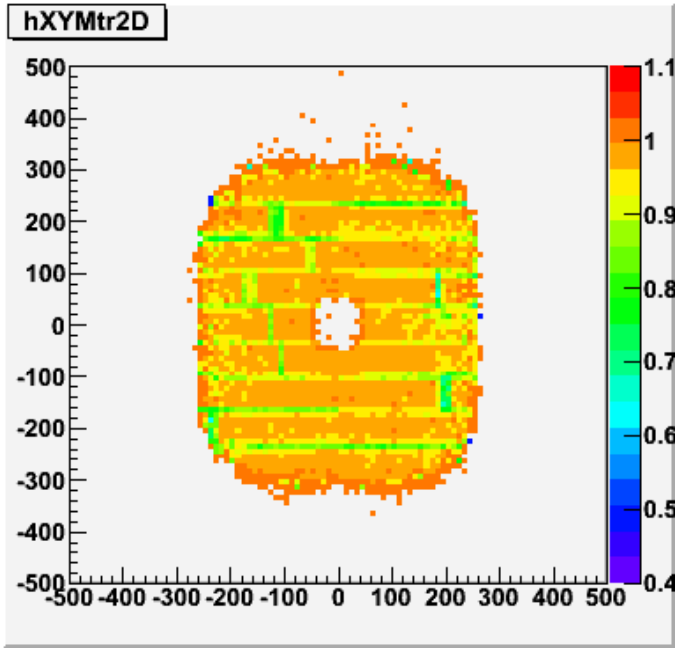
(data)

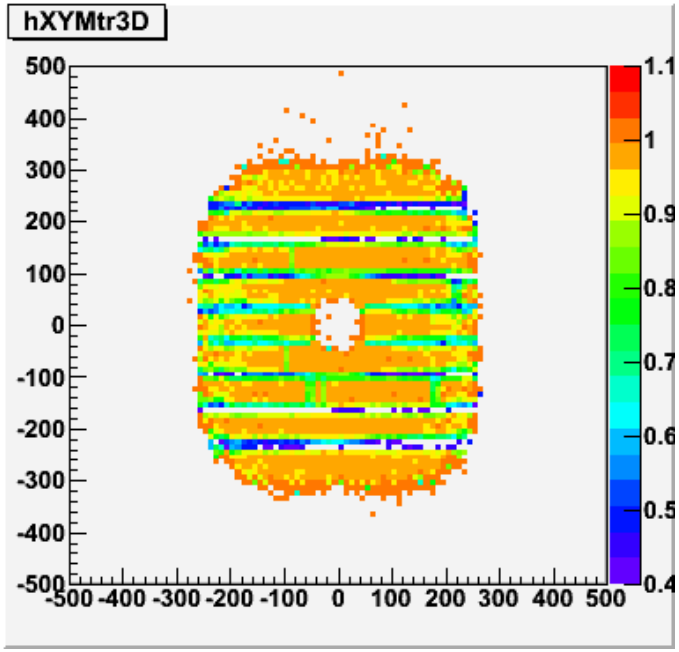
+ 1st trigger plane in the trigger

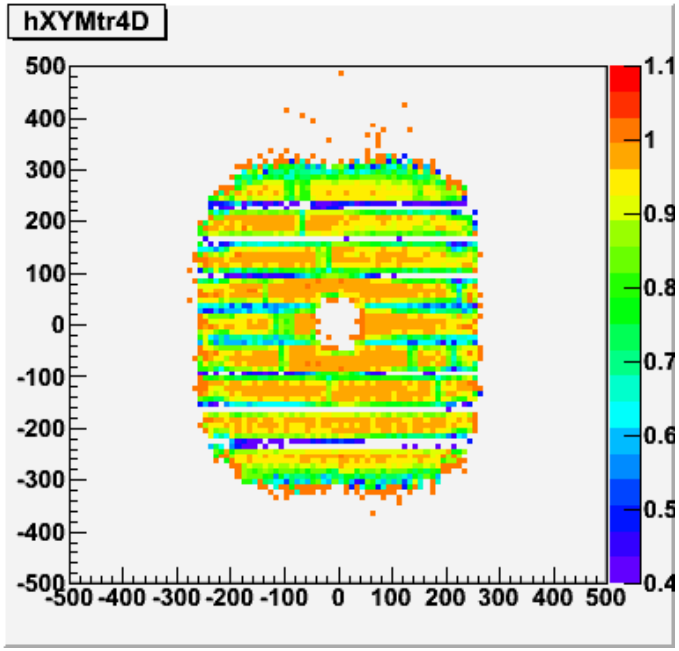


The ratio of the plots from the previous page (data)

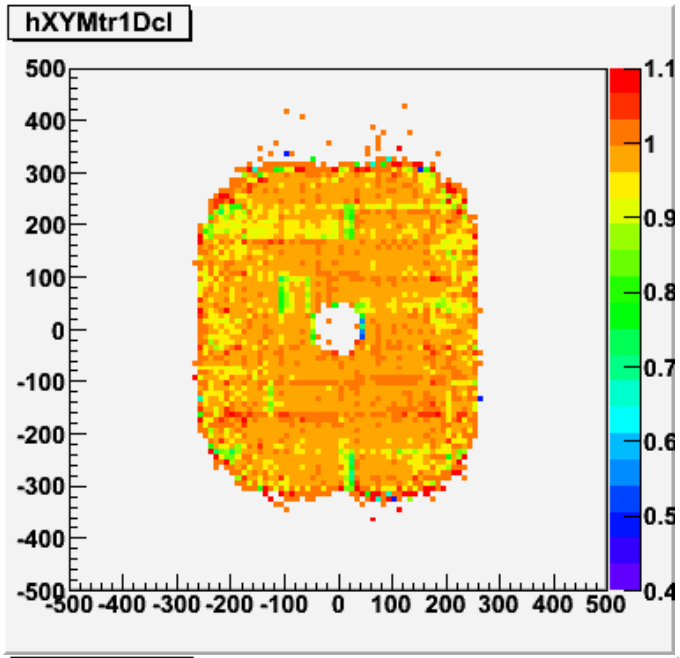


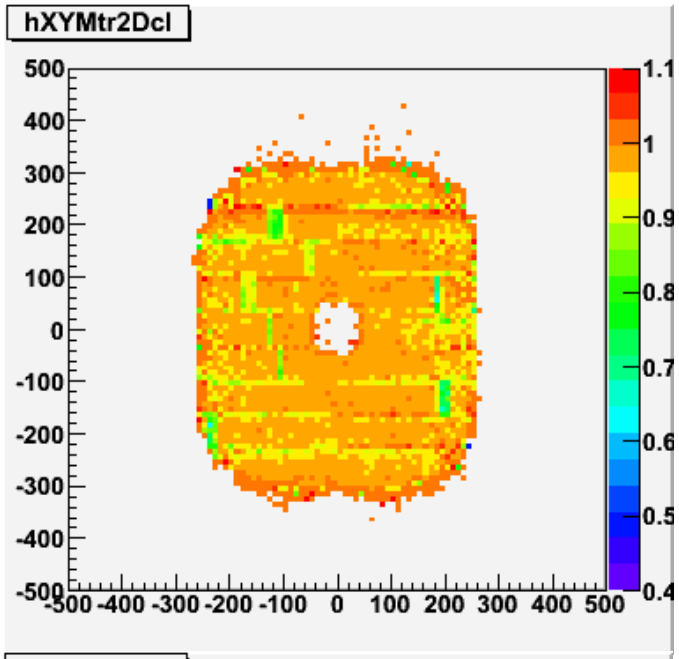


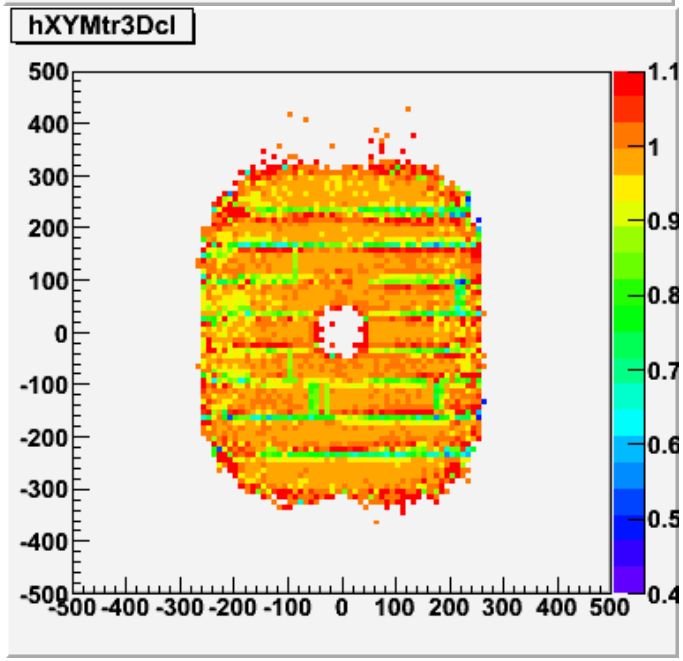


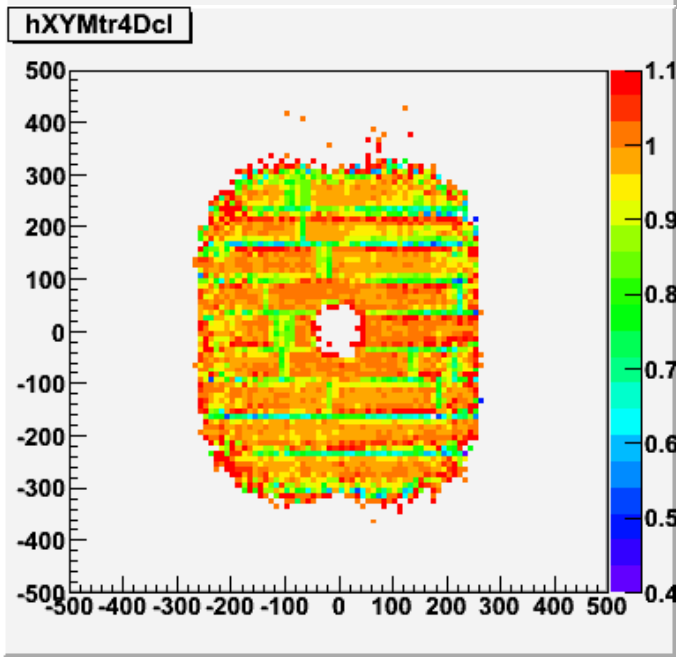


Create the same plots for the simulation and do the ratio data/simulation









Problems/comments

- do we see any structure on RPCs or local boards level ?

- was the choice of the (x,y) impact point (on the first plane of the track extrapolation) a good one ? ... yes

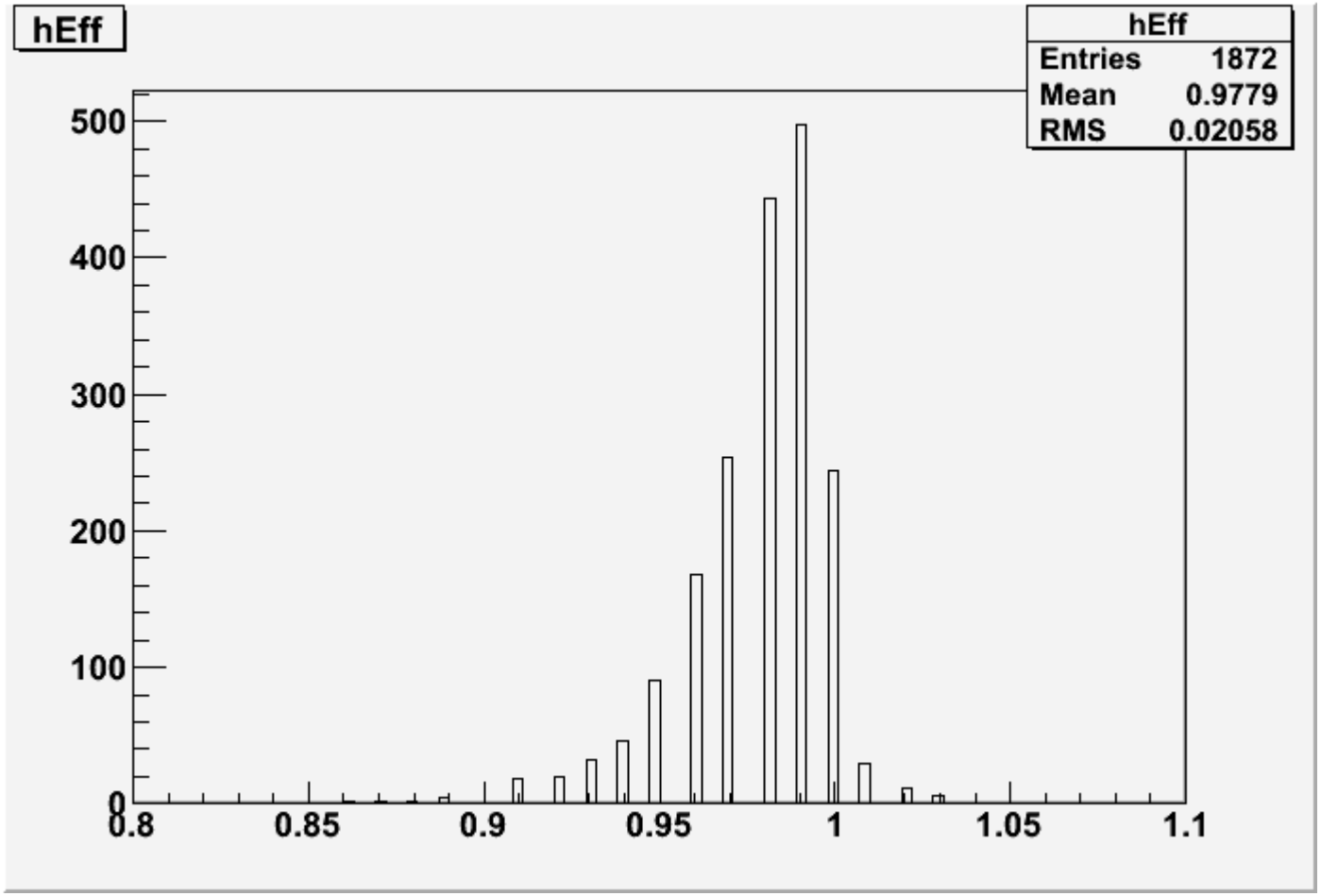
- there is another information on the planes contributing to the trigger (the field "TriggerWithoutChamber(<nr>)") which was not yet calculated in the PDC09 simulation ... actually this is not quite useful for the moment ...

- the PDC09 was done with all channels working, while the data has some local channels which are masked and we see this in the efficiency

- one track may not stay within the same local board when crossing the four planes, or it may even not stay within the same RPC (border effects); there is also a flag attached to the ESD muon track ("GetEffFlag") which identifies such cases

Results from data 119156-120079 and simulations PDC09 LHC09a18 (MB)

distribution of efficiency values (values > 1 are explained by statistics...)
(234 boards x 2 cathodes x 4 chambers = 1872 values)

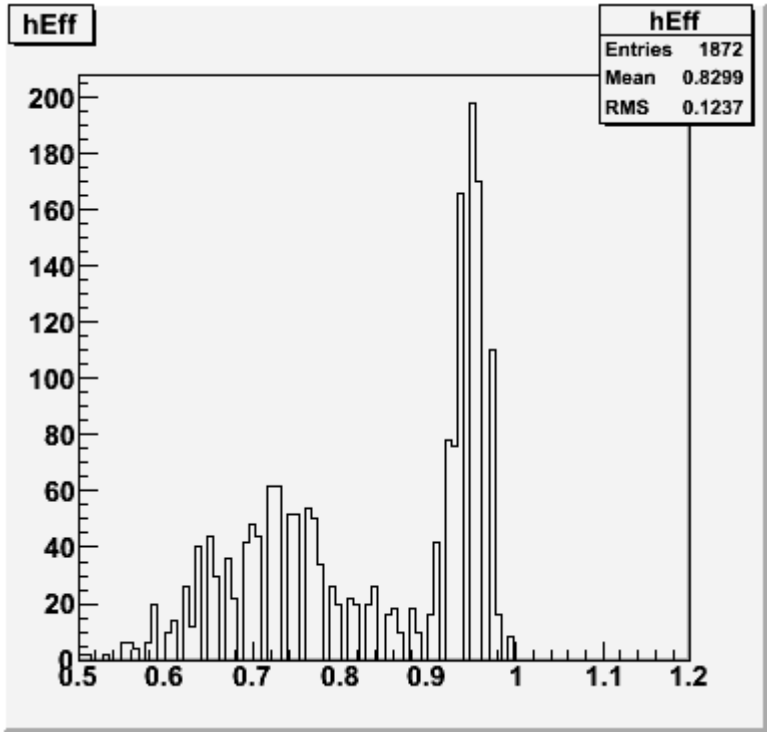


Obs: efficiency is chosen the same for the two cathode planes

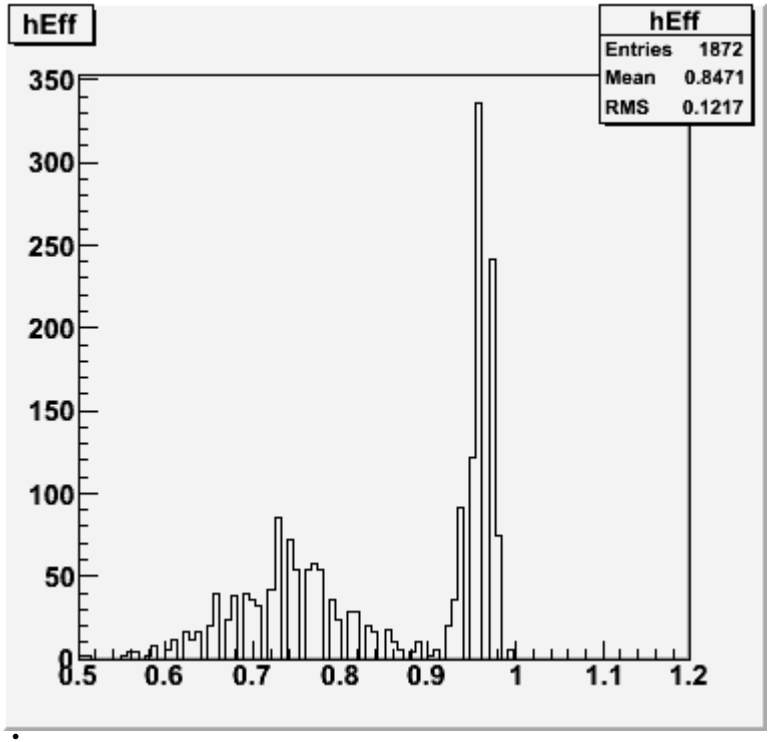
The values from the previous plot are obtained from the ratio of the efficiencies resulted from the data analysis (D) and from the simulations (S)

The efficiency values from the simulations (all < 1) include algorithm efficiency and other acceptance/geometry effects.
The same values for the data represent the efficiencies multiplied with the RPC efficiency which is present in the data. The factorization hypothesis of this type efficiency is essential for this method of determining it.

data



simulations



single plane efficiency

Code implementation details

Trigger efficiency implementation in AliRoot (simulations)

Configuration file for simulations: `Config.C`

```
AliMUON *muon = new AliMUONv1("MUON", "default");
...
muon->SetTriggerEffCells(1);
...
```

Is it still there? Francesco Bossù told me something about old and new methods... but looking at the code of the latest trunk (42360) it seems that it is still used.

(not used in the official productions until now)

to apply trigger chamber efficiency according to the OCDB entry:
`OCDB/MUON/Calib/TriggerEfficiency`

implemented by the class:
`AliMUONTriggerEfficiencyCells`

AliMUONTriggerEfficiencyCells is the OCDB class that contains the information, while AliMUONTriggerChamberEfficiency is the class now used to fetch data from the OCDB muon trigger efficiency

How is it done? See next page...

Trigger efficiency implementation in AliRoot (cont.)

The “digitizer” executing function (transform hits to digits)

`AliMUONSDigitizerV2::Exec` OK, verified

checks for `muon->GetTriggereffCells()` (which was set in `Config.C`)

and initializes the trigger response

`AliMUONResponseTrigger::InitTriggerEfficiency`

from the OCDB object in the folder

`MUON/Calib/TriggerEfficiency/Run<validity>.root`

via the class `AliMUONTriggerChamberEfficiency`.

The DisIntegrate() function inside AliMUONResponseTrigger creates at most two digits (one for each cathode) for each hit, by asking the IsTriggered() function. If it is kFALSE for a certain cathode, the digit for that cathode and local board will not be added to the digits list.

The transformation from hits to digits in the trigger chambers is done in:

`AliMUONResponseTrigger::DisIntegrate`

where the digit is created only if the function

`AliMUONTriggerChamberEfficiency::IsTriggered`

returns “true” for a given detection element, local board and cathode plane, by calling

`AliMUONTriggerChamberEfficiency::GetCellEfficiency`

The trigger efficiency object: `AliMUONTriggerEfficiencyCells`

is initialized from
`$ALICE_ROOT/MUON/data/efficiencyCells.dat`
containing efficiency as "floats" (currently all at 0.97) by chamber, by local board and by cathode (0 or 1):

The efficiencyCells.dat file is still there, but it seems that the macro that creates it does not rely on this text file anymore, but only on the list of histograms, as explained in Slide 19.

```
localBoards


detElemId:      1100
 cathode:       0
 0.97 0.97 0.97 0.97 0.97 0.97 ...

 cathode:       1
 0.97 0.97 0.97 0.97 0.97 0.97 ...

detElemId:      1200
 cathode:       0
 0.97 0.97 0.97 0.97 0.97 0.97 ...

 cathode:       1
 0.97 0.97 0.97 0.97 0.97 0.97 ...
```

Obs: the local boards are arranged per regional crate

(cont.)

With the macro

`$ALICE_ROOT/MUON/MUONTriggerChamberEfficiency.C`

~~this text file is converted in the OCDB object~~.

~~Alternatively~~, this macro can use a list of histograms from the file

`MUON.TriggerEfficiencyMap.root`

which is produced by the analysis

```
PWG3/muon/AnalysisTrigChEff.C
PWG3/muon/AliAnalysisTaskTrigChEff.cxx
```

Update in code - May 3rd 2010 PWG3-MUON
"RPC efficiencies in simulation"

The file "`$ALICE_ROOT/data/efficiencyCells.dat`" is not used anymore, instead the "efficiency map" is directly use to initialize the efficiency cells.

The two cathode plane probabilities are now correlated by a third probability to fire simultaneously the two planes ("both").

If an OCDB `TriggerEfficiency` object from previous versions is used, this new type of probability does not exist and is taken as the default value of "-1", the probabilites of the two cathode planes remain thus independent.

The efficiencies extracted from data shown before are extracted with the assumption that both cathode planes fire simultaneously ("both").

Does it mean that no "both" value is saved, and bending and nonbending values are the same? Because from the tests it seems so.

One example from simulations

Select a kinematic region of negative muons going in the region of the boards nr. 189 and 190. Chose the efficiency 97 % for those two boards and 100 % for the rest of the boards.

```
localBoards

detElemId:       1100
 cathode:        0
 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00
...
 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 0.97 0.97 1.00 1.00
 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00
 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00
 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00

 cathode:        1
...
```

After two set of simulations (one ideal with 100 % efficiency for all boards), with about 4000 tracks per board (189 and 190), one “reconstructs” the efficiency of 97.4+/-0.5 %.

Which efficiency?! This is not clear at all.