Subject: [FIXED] Mc Truth Match
Posted by Klaus Götzen on Mon, 19 Aug 2013 14:20:55 GMT
View Forum Message <> Reply to Message

Hi,

I just did a quick MC truth check with the tutorial macro. Therefore I filled three lists 'piall', 'mcpi', and 'pimatch' with

```
pimatch.Cleanup();
theAnalysis->FillList(mcpi,"PionLoose","PidAlgoIdealCharged");
theAnalysis->FillList(piall,"PionAll");

for ( int ii=0 ; ii<piall.GetLength() ; +ii )
   if ( theAnalysis->McTruthMatch(piall[ii]) )
      pimatch.Add( piall[ii] );
```

In 'mcpi' selection is based on PidAlgoIdealCharge, 'piall' contains all charged tracks with pion PID assigned, and 'pimatch' is selected from this with the McTruthMatch methode.

Then I counted the number of candidates in 'pimatch' and 'mcpi', both with and without McTruth object assigned. The result looks like this for 1000 events of the tutorial channel:

pi:1443 (with McTruth:1443) mcpi:1521 (with McTruth:1521)

As you can see, in my test all selected candidates had an attached McTruth object. Second thing is, that there is a discrepancy in the number of candidates in both methods. I think I found the reason for this.

Let's therefore take a look to the candidates in all lists for one case of mismatch, where for each candidate it is shown [track number; PDG; charge; PID probs; PDG of truth object]:

pimatch(1) : [#2; 211; C=1; P=(0;0;1;0;0;); True=211]
mcpi(2) : [#1; 211; C=1; P=(0;0;1;0;0;); True=-211] [#2; 211; C=1; P=(0;0;1;0;0;); True=211]

piall(4) : [#0; -211; C=-1; P=(0;1;0;0;0;); True=13] [#1; 211; C=1; P=(0;0;1;0;0;); True=-211]
[#2; 211; C=1; P=(0;0;1;0;0;); True=211] [#3; 211; C=1; P=(0;1;0;0;0;); True=-13]

As one can see, in 'pimatch' the PDG code of the selected object is exactly the same (including charge!) as of the corresponding McTruth object, whereas in 'mcpi' for track #1 PDG and true PDG have opposite sign.

So the reason for the difference is, that tracks with (for whatever reason) mis-reconstructed charge are accepted by ideal PidAlgo, since this one only requires certain pion PID probability without knowledge of charge, whereas the McTruthMatch performes an exact match including correct charge.

Therefore I guess, one should use the McTruthMatch methode or compare the reco PDG code to McTruth PDG code in order to do a proper counting and efficiency measurement.

Best,
Klaus

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Mon, 19 Aug 2013 15:44:09 GMT
View Forum Message <> Reply to Message

Hi Klaus,

Quote:
So the reason for the difference is, that tracks with (for whatever reason) mis-reconstructed charge are accepted by ideal PidAlgo, since this one only requires certain pion PID probability without knowledge of charge, whereas the McTruthMatch performes an exact match including correct charge.

The same seems to be true when the charge is specified in FillList

theAnalysis->FillList(mcpiplus,"PionLoosePlus","PidAlgoIdealCharged");

which was surprising to me. I used to think that when you use PidAlgoIdealCharged and asked for positively charged particles, then you would only get particles whose charge was correctly identified. However, with a quick check (also over 1000 events with the rho tutorial macros) I find 34 events with differences.

The total results are

pimatchplus = 359 piplus = 374
pimatchminus = 374 piminus = 394

and were obtained with the following code:

```
pimatchplus.Cleanup();
pimatchminus.Cleanup();

  theAnalysis->FillList(mcpiplus,"PionLoosePlus","PidAlgoIdealCharged");
  theAnalysis->FillList(mcpiminus,"PionLooseMinus","PidAlgoIdealCharged");
theAnalysis->FillList(piall,"PionAll");


for ( int ii=0 ; ii<piall.GetLength() ; ++ii ){
if ( theAnalysis->McTruthMatch(piall[ii]) ){
if ( 211 == piall[ii]->PdgCode()  ){
pimatchplus.Add( piall[ii] );
}
```

```
if ( -211 == piall[ii]->PdgCode()  ){
pimatchminus.Add( piall[ii] );
}

}
}

// consistency check (show suspicious events)
if ( ( pimatchplus.GetLength()!= mcpiplus.GetLength() ) || ( pimatchminus.GetLength() !=
mcpiminus.GetLength() ) ){
cout << "evt " << i << endl;
cout << "pimatchplus = " << pimatchplus.GetLength() << "     piplus = " <<
mcpiplus.GetLength() << endl;
cout << "pimatchminus = " << pimatchminus.GetLength() << "     piminus = " <<
mcpiminus.GetLength() << endl;
   ++nsuspiciousevents;
}

  npimatchplus += pimatchplus.GetLength();
  npimatchminus += pimatchminus.GetLength();

  nmcpiplus += mcpiplus.GetLength();
  nmcpiminus += mcpiminus.GetLength();
```

If the user is aware of these differences, then it should not be a problem, I think, even if the behaviour is a little counter-intuitive when you specify the charge in FillList.

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Mon, 19 Aug 2013 18:03:40 GMT
View Forum Message <> Reply to Message

Are you considering only fitted tracks (GetFitStatus()>0) or al the tracks?
I find quite strange that the charge is wrong. Maybe those are bad not fitted tracks, and the charge is somehow screwed?

Subject: Re: Mc Truth Match
Posted by Klaus Götzen on Mon, 19 Aug 2013 19:28:26 GMT
View Forum Message <> Reply to Message

Hi Martin,

actually it's not the case that FillList with option "XyzPlus" fills in as well negative particles. The results you shown confirm my observation I guess:

Your number piplus = 374 include all 359 pimatchplus particles, and in addition 15 negative pions, which were reconstructed wrongly as postive pions (perhaps the sign of the curvature or flight direction was fitted wrongly) and of course have as well a pion PID probability of 1.

So what you see is not negative charged particles selected by "PionLoosePlus", but reco tracks with positive charge, whose hits in reality were created by a particle with negative charge.

But how should the PID algorithm or data provider know that the true particle was negative, when tracking says, it's postive?

To clarify: The PidAlgoIdealCharged delivers perfect PID probabilities for all particle species, but it's not meant to be a McTruth match (as I just realized today...).

Best,
Klaus

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Tue, 20 Aug 2013 09:17:45 GMT
View Forum Message <> Reply to Message

Hello Klaus,

you are right, that was what I intended to say.

I believe the behaviour of FillList in combination with "PidAlgoIdealCharged" is a bit more logic when one realizes that MC truth information is only used to determine the particle species (pion, kaon, muon, etc.) and the charge is determined from tracking only (and therefore can be incorrect for high momentum particles for instance). (As you wrote already.)

To Stefano's point: I have just applied a filter to all lists using the following code (inspired by a previous posting of Klaus):

```
int SelectGoodFitStatus(RhoCandList &l) {
 int removed = 0;
 for (int ii=l.GetLength()-1;ii>=0;--ii) {
  if ( !( l[ii]->GetRecoCandidate()->GetFitStatus()>0 ) ) {
l.Remove(l[ii]);
removed++;
  }
 }
 return removed;
}
```

So in my while loop I added

```
// comment this out if you want to allow all fit status
SelectGoodFitStatus(piall);
SelectGoodFitStatus(mcpiplus);
```

SelectGoodFitStatus(mcpiminus);

before doing the McTruthMatch and the counting.

The number of events with different counts are reduced from 34 to 19 (out of 1000).
I get the following counts now:

pimatchplus = 317 piplus = 324
pimatchminus = 327 piminus = 340

I therefore believe that the mis-identified charge cannot be explained with a bad fit status.

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Tue, 20 Aug 2013 10:21:39 GMT
View Forum Message <> Reply to Message

Please use add and not remove, it is safer.
In some old thread it was demonstrated that the "remove" produces TCandList which could be problematic.

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Tue, 20 Aug 2013 10:55:55 GMT
View Forum Message <> Reply to Message

Hello Stefano,

I have tried the same with Add to a new RhoCandList instead of removing items. I get the same numbers as with remove. I attached my macro to this post for those who are interested.

Probably, remove is only dangerous if one forgets that the indices of subsequent list items are shifted when an item is removed. This needs to be accounted for in the for loop if counting up (from 0 to the end of the list) or alternatively, as Klaus suggested one can count "backwards" which circumvents the problem.

File Attachments
1) mcmatchtest.C, downloaded 517 times

Subject: Re: Mc Truth Match
Posted by Klaus Götzen on Tue, 20 Aug 2013 11:41:48 GMT
View Forum Message <> Reply to Message

Hi Martin,

I unfortunately have to disagree a bit to your statement.

The PndAnalysis::McTruthMatch methode (using the assigned McTruth candidate) takes into account the charge. I.e. it only accepts, if the charge and PDG code of the reco particle and its corresponding McTruth object agree, which implies, that the charge was correctly reconstructed by tracking.

This is not to be mixed up with filling a list with the 'PidAlgoIdealCharged' option!

Best,
Klaus

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Tue, 20 Aug 2013 11:47:02 GMT
View Forum Message <> Reply to Message

I tried to investigate the "theory" that the wrong charge showing up in lists filled with

theAnalysis->FillList(mcpiplus,"PionLoosePlus","PidAlgoIdealCharged");

might be due to high momentum tracks, but it does not seem to be the case. I printed the momenta of misidentified charged pions.

With a filter on GetFitStatus()>0 this is the result that I get:

p_total p_perp theta (p_x p_y p_z)
0.157126 0.0535453 19.9243 (0.048933 0.0217408 0.147721)
0.505125 0.0941352 10.7404 (0.0493841 -0.0801414 0.496276)
2.11389 0.891485 24.9437 (0.783278 -0.425702 1.91671)
0.396296 0.1589 23.6384 (-0.144325 -0.0664785 0.363045)
4.13429 1.62192 23.0983 (-0.296381 -1.59461 3.80285)
0.193541 0.107057 33.5834 (-0.0956728 0.0480417 0.161236)
0.163668 0.0834998 30.6758 (-0.0831942 0.00713736 0.140765)
9.32307 4.19835 26.7641 (-3.74612 1.89545 8.32427)
0.196377 0.0823667 24.7987 (0.0801052 0.0191682 0.178268)
0.35568 0.171477 28.8232 (-0.0621465 -0.159819 0.311616)
2.18964 0.724932 19.334 (-0.275077 -0.670715 2.06615)
2.20361 1.28911 35.803 (-1.14287 0.596375 1.7872)
0.237624 0.202498 58.4492 (0.115297 -0.166469 0.124338)
0.876789 0.855665 102.602 (0.416404 -0.74751 -0.191299)
0.962982 0.289888 17.5196 (0.283118 -0.0622874 0.918313)
0.275722 0.142764 31.1836 (0.12816 0.0629009 0.235883)
0.331126 0.188794 34.7612 (-0.170917 -0.080191 0.272032)
0.411353 0.143972 20.4871 (-0.0926718 0.110181 0.385335)
0.192919 0.132213 43.2614 (0.101495 0.0847284 0.14049)
0.694575 0.332611 28.6117 (0.295166 -0.153321 0.609757)

The printout is of the form:
Total momentum, perpendicular momentum, angle to z-axis in degree, (x-component, y-component, z-component)

While there are high momentum tracks present, also very low momentum charged pions seem to be misidentified.

If I leave out the filter on GetFitStatus()>0 I get the following:

0.0560073 0.0234772 24.7829 (0.000440402 0.023473 0.0508492)
0.157126 0.0535453 19.9243 (0.048933 0.0217408 0.147721)
0.505125 0.0941352 10.7404 (0.0493841 -0.0801414 0.496276)
2.11389 0.891485 24.9437 (0.783278 -0.425702 1.91671)
0.396296 0.1589 23.6384 (-0.144325 -0.0664785 0.363045)
4.13429 1.62192 23.0983 (-0.296381 -1.59461 3.80285)
2.10445 2.10366 91.5735 (-1.15011 1.76143 -0.0577857)
0.193541 0.107057 33.5834 (-0.0956728 0.0480417 0.161236)
0.0939326 0.0431508 27.3471 (-0.00740239 -0.0425111 0.0834347)
0.240746 0.107517 26.5259 (0.0884348 -0.0611497 0.215403)
0.163668 0.0834998 30.6758 (-0.0831942 0.00713736 0.140765)
9.32307 4.19835 26.7641 (-3.74612 1.89545 8.32427)
0.517521 0.159474 17.9478 (-0.0576879 0.148675 0.492337)
0.196377 0.0823667 24.7987 (0.0801052 0.0191682 0.178268)
1.23126 0.1737 8.11007 (0.150681 0.0864106 1.21894)
0.35568 0.171477 28.8232 (-0.0621465 -0.159819 0.311616)
2.18964 0.724932 19.334 (-0.275077 -0.670715 2.06615)
2.20361 1.28911 35.803 (-1.14287 0.596375 1.7872)
0.0849093 0.028615 19.6946 (0.0200407 -0.0204252 0.0799423)
0.237624 0.202498 58.4492 (0.115297 -0.166469 0.124338)
0.876789 0.855665 102.602 (0.416404 -0.74751 -0.191299)
0.0606288 0.0187204 17.9852 (0.00509841 0.0180127 0.0576663)
0.962982 0.289888 17.5196 (0.283118 -0.0622874 0.918313)
0.275722 0.142764 31.1836 (0.12816 0.0629009 0.235883)
0.837384 0.837374 89.71 (-0.561434 -0.621278 0.0042382)
2.21517 0.731518 19.2828 (0.514025 -0.520478 2.0909)
0.0869416 0.0438064 30.2557 (-0.0430866 -0.00790857 0.0750989)
0.336012 0.0657709 11.2879 (0.057233 -0.0324069 0.329512)
0.331126 0.188794 34.7612 (-0.170917 -0.080191 0.272032)
0.0412819 0.0318619 50.517 (-0.0237311 0.0212607 0.0262491)
0.411353 0.143972 20.4871 (-0.0926718 0.110181 0.385335)
0.123346 0.113654 67.1346 (0.0525635 0.100768 0.0479286)
0.0436396 0.0243122 33.8565 (-0.020283 0.0134046 0.0362399)
0.192919 0.132213 43.2614 (0.101495 0.0847284 0.14049)
0.694575 0.332611 28.6117 (0.295166 -0.153321 0.609757)

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Tue, 20 Aug 2013 11:51:15 GMT

View Forum Message <> Reply to Message

Hi Klaus,

I do agree. My statement was only about

theAnalysis->FillList(mcpiplus,"PionLoosePlus","PidAlgoIdealCharged");

Subject: Re: Mc Truth Match
Posted by Simone Esch on Tue, 20 Aug 2013 12:07:33 GMT
View Forum Message <> Reply to Message

Hello Klaus,

Quote:
To clarify: The PidAlgoIdealCharged delivers perfect PID probabilities for all particle species, but it's not meant to be a McTruth match (as I just realized today...).

I had in my simulation also particles which would be explained by the wrong determinition of the charge.

But I have also particles in my list which are by MCTruth information gammas, neutrons and particles with another PDG code
(see https://forum.gsi.de/index.php?t=tree&th=3950&start=0&rid=1562&S=9dc4d1b4a20caccf161bab4f8cbd25ba#page_top)
There seem the PID match not to work.

Have you seen something like this ?

Best regards

Simone

Subject: Re: Mc Truth Match
Posted by Klaus Götzen on Tue, 20 Aug 2013 12:40:35 GMT
View Forum Message <> Reply to Message

Hi Simone,

in the posting you have linked to I see some unusual PDG codes (say gammas, neurons, even muons etc) only in the gamma/neutral list.

I think the explanation for this is, that the EMC cluster algorithm identifies clusters created by all kinds of particles, and when no track can be associated to this cluster unambiguously, it simply is taken as a gamma candidate, i.e. setting PDG code to 22, charge and mass to 0.

When you check the PDG codes from the reco candidates in the 'Neutral' list, you should only see particles with PDG code 22. The associated McTruth objects indeed might have completely different PDG codes and even charges.

The FillList routine (as well as PidSelectors) actually set the PDG code according to the requested list type, but it of course doesn't touch the types of the McTruth objects.

Best,
Klaus

Subject: Re: Mc Truth Match
Posted by Simone Esch on Tue, 20 Aug 2013 12:56:03 GMT
View Forum Message <> Reply to Message

Hallo Klaus,

I am sorry, my link was refering to the wrong posting, here is the right one:
https://forum.gsi.de/index.php?t=tree&rid=0&S=29db68af4cc4221238
9ac9acd60b3e32&th=3950&goto=15045#msg_15045

I have in my list where i asked for positive or negative particles also neutral particles. How can that be explained?

Best Regards

Subject: Re: Mc Truth Match
Posted by Klaus Götzen on Tue, 20 Aug 2013 13:17:12 GMT
View Forum Message <> Reply to Message

Hi Simone,

indeed this is quite counter intuitive. It's quite some time ago, since you observed that behaviour. Did you try again after updating trunk?

If yes, and it looks still this weird, could you attach the .dec file and your analysis macro, so that I can take a closer look?

Best,
Klaus

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Tue, 20 Aug 2013 13:22:49 GMT

View Forum Message <> Reply to Message

Can you check using Tight instead of Loose?
The "Loose" corresponds to probability 20%, and if for some reason the ideal algorithm fails then all the pid probabilities will be at 20%... the rounding will give you the final (wrong) pid.

Moreover, checking the ideal algorithm code written by Ralf, there is no neutral id (of course, there is no neutral probability object).

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Tue, 20 Aug 2013 13:50:19 GMT
View Forum Message <> Reply to Message

Stefano Spataro wrote on Tue, 20 August 2013 15:22Can you check using Tight instead of Loose?
The "Loose" corresponds to probability 20%, and if for some reason the ideal algorithm fails then all the pid probabilities will be at 20%... the rounding will give you the final (wrong) pid.

Moreover, checking the ideal algorithm code written by Ralf, there is no neutral id (of course, there is no neutral probability object).

Hello Stefano,

I have observed what you predicted.

With

theAnalysis->FillList(mcpiplus,"PionLoosePlus","PidAlgoIdealCharged");

I get several particles with pdg code 1000010020 in that list when simulating psi(4040) to D0 anti-D0 / D+ D- and all D decays allowed.

When I change the line to

theAnalysis->FillList(mcpiplus,"PionTightPlus","PidAlgoIdealCharged");

these entries disappear!

I have also checked the GetPidInfo() array for such RhoCandidates and the first 5 entries (index 0 to 4) have 0.2 as values.

Subject: Re: Mc Truth Match
Posted by Simone Esch on Tue, 20 Aug 2013 14:09:00 GMT
View Forum Message <> Reply to Message

Hello Stefano,

for me it is the same as for Martin, my neutral and other strange wrong particles disappeard from the list. I just have some with a wrong charge, but this can be explained by Klaus explanation.

I think it is not a good idea to put everywhere the 0.2 if something at the pid stage fails.
Is there not a flag to indicate this, to be able to filter it out?

I understand that the PidAlgoIdealCharged just sets the right probability and nothing else, but still I thik that the name is then missleading. I thought i would use a somehow MC base algorythem which sets all characteristics right so that the FillList method really just pick the right ones.

If this is kept like this, this should be mentioned in the tutorial.

Thanks for the Help!!

Simone

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Tue, 20 Aug 2013 14:18:03 GMT
View Forum Message <> Reply to Message

Simone Esch wrote on Tue, 20 August 2013 16:09
I think it is not a good idea to put everywhere the 0.2 if something at the pid stage fails.
Is there not a flag to indicate this, to be able to filter it out?

We have 5 particle hypothesis. If the pid algorithm fails, whatever it is, then all the 5 particles have the same probability -> 1/5 = 0.2 -> 20%. from the statistical point of view this is correct. Those tracks should not becut away, simply you don0t have pid information for them.
The point is that one should cut with probability more than such 20%, maybe the Loose condition should be moved to 25% just to be safe.

Quote:
I understand that the PidAlgoIdealCharged just sets the right probability and nothing else, but still I thik that the name is then missleading. I thought i would use a somehow MC base algorythem which sets all characteristics right so that the FillList method really just pick the right ones.

If this is kept like this, this should be mentioned in the tutorial.

I have not understood, the PidAlgoIdealCharged assigns the mc id to each candidate, as it should be, using the PndPidProbability object -> a mcpion will have probability of being a pion 1 and prog of e/mu/k/p = 0, so that this info can be used by FillList. This is not "ideal" parameters, this is ideal identification. The FillList picks the correct particles identified by MC.

In this sense, PidAlgoIdealCharged is doing its job coherent with all the other algorithms, and I have not understood the last comment. Could you please be more explicit?

Subject: Re: Mc Truth Match
Posted by Klaus Götzen on Tue, 20 Aug 2013 14:34:57 GMT
View Forum Message <> Reply to Message

Hi,

I think the basic problem for a perfect MC match is, that there is none! Sometimes reconstructed objects cannot by unambiguously assigned to MC truth objects, since a track candidate or cluster might consist of hits created by different particles.

This means, that a measured efficiency not only has a statistic error, but a systematic one in addition due to the imperfect assignment of the truth. We just don't care about it for the time being...

Best,
Klaus

Subject: Re: Mc Truth Match
Posted by Simone Esch on Tue, 20 Aug 2013 14:40:15 GMT
View Forum Message <> Reply to Message

Ciao Stefano,

Quote:
We have 5 particle hypothesis. If the pid algorithm fails, whatever it is, then all the 5 particles have the same probability -> 1/5 = 0.2 -> 20%. from the statistical point of view this is correct. Those tracks should not becut away, simply you don0t have pid information for them.
The point is that one should cut with probability more than such 20%, maybe the Loose condition should be moved to 25% just to be safe.

If we talk about real pid I agree with you. But if I am using the ideal algo, I expect all particles to have the right prob and not a prob in between which is not 1 or 0. If for the ideal algo something goes wrong I would like to know.

Quote:
I have not understood, the PidAlgoIdealCharged assigns the mc id to each candidate, as it should be, using the PndPidProbability object -> a mcpion will have probability of being a pion 1 and prog of e/mu/k/p = 0, so that this info can be used by FillList. This is not "ideal" parameters, this is ideal identification. The FillList picks the correct particles identified by MC.

In this sense, PidAlgoIdealCharged is doing its job coherent with all the other algorithms, and I

have not understood the last comment. Could you please be more explicit?

This is the point, the FillList picks not just the correct particles, but also wrong ones.
In my case (and for others the same) it collected also neutral particles and wrong particles in, due to there wrong charge. So I had to filter afterwards on the MCTruth characteristics to have a clean list.

And I think that this is missleading, that one has to filter afterwards for wrong particles, despite using a ideal algo.

Best Regards
Simone

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Tue, 20 Aug 2013 15:09:07 GMT
View Forum Message <> Reply to Message

The "Loose = 20%" selection is wrong, but the algorithm is correct.
Once you put a selection larger than 20% you should have the correct mc id from the FillList, and you don't need to filter the particles anymore.

If you want to know where the algorithm failed, you need to ask for particles with probability around 20%. This is our design for the particle identification.
The algorithm says which probability has the candidate, but the track check must be done from the candidate and not from the probability object.

Subject: Re: Mc Truth Match
Posted by Simone Esch on Tue, 20 Aug 2013 15:25:20 GMT
View Forum Message <> Reply to Message

Ciao Stefano,

but then I have still particles with a wrong charge which are grouped to the wrong list I have to filter on like Klaus reported earlier in this thread.
The particles with the wrong charge are not just in the wrong list and will be removed due to filtering, they are also missing in the other list, which drops your efficiency (I saw this alreaday, but I had no explanation for that).

I think in combination with the selection criteria this is a lot one needs to know in addition (which is not intuitive) to use the ideal algo correctly and I think the least people which use the Ideal algo know that.

Best Regards

Simone

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Tue, 20 Aug 2013 15:31:42 GMT
View Forum Message <> Reply to Message

These are reconstruction errors. You could have a real mc pion with a momentum which is completely screwed, and this is another reco error. The pid algorithm is simply associating to each track (bad or wrong, it does not care) one id from mc, when possible. When this is not possible, i.e. MVD noise tracklets, then it gives the same probability to all the ids.

If you want to have the perfect world, then you should use also ideal tracking w/o kalman, and not only ideal identification, to take out the reconstruction bias.

If you want to have a relly good ideal id on reco tracks, first one should define what is really a good track, and how to define really efficiency. But this is another issue, and the tracking discussion is also focused on this point.

Subject: Re: Mc Truth Match
Posted by Tobias Stockmanns on Tue, 20 Aug 2013 15:47:11 GMT
View Forum Message <> Reply to Message

Dear Klaus,

I think your argument that there is no perfect MC match is not right. For this purpose I have written the FairLinks which are working extremly well for comparison between reconstructed data and MC data. Maybe it is time to use them for the analysis as well.

For the rest of the discussion I see a discrepancy between what the designers of the code think their code is doing and the expectation of the users what the code should do according to the used names. I fear that the problem Simone and Martin had will happen to other users as well.

In addition I think there should be a method for an ideal list filling.

Cheers,

Tobias

Subject: Re: Mc Truth Match
Posted by Klaus Götzen on Tue, 20 Aug 2013 16:21:10 GMT
View Forum Message <> Reply to Message

Hi,

I hope you agree, that sometime there might be reconstructed objects consisting of hits from different particles. In this case there is no perfect MC match. There for sure exists a best one, but this is not necessarily the correct one, so we could have indeed MC mis-match, even when using the FairLink concept (as far as I see it).

We might also think about evaluation of the two or three best matches, or providing a match quality. You may remember the discussions about these issues, which we had quite some time ago.

What I understand is, that the documentation of the tools should very clearly define what is supposed to be delivered, and where are the limitations of each tool. I'll try to clarify these issues in the tutorial and docu page accordingly.

Best and thanks for all the feedback
Klaus

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Tue, 20 Aug 2013 17:30:55 GMT
View Forum Message <> Reply to Message

Tobias Stockmanns wrote on Tue, 20 August 2013 17:47Dear Klaus,

I think your argument that there is no perfect MC match is not right. For this purpose I have written the FairLinks which are working extremly well for comparison between reconstructed data and MC data. Maybe it is time to use them for the analysis as well.

The problem is not in the link, the problem is that one track can be made of different mc ids, i.e. a kaon decaying into a pion w/ a very small kink. In such sense, we take the most common mc index, which could be wrong. Moreover, the mc association depends on the hits, but the momentum reconstruction, the charge and so on depend on tracking. In this sense the MC matching is only our definition, no perfect matching.

It would be good if some FairLink expert could modify the code to get rid of the mctrackassociator and of the PndTrackID TCA, since for this part the code is somehow dupplicated. Do you know if somebody could so such job?

Quote:
For the rest of the discussion I see a discrepancy between what the designers of the code think their code is doing and the expectation of the users what the code should do according to the used names. I fear that the problem Simone and Martin had will happen to other users as well.

If the problem is in the semantics, then the PndPidIdealAssociatorTask count be renamed into PndPidMCAssociatorTask, if this could help.

Quote:
In addition I think there should be a method for an ideal list filling.

What do you mean exactly? The "ideal" information is already stored inside the MCTrack TCA and fully accessible, and this is the info that should be used for montecarlo plots. Even

because w/o covariance matrix the fitters will never work on mc tracks, and rho tools cannot be applied to mc info.
From the reconstructed candidate one can access to the mc truth, and this is available in the code.
Then, what is expected from an "ideal list filling"? If it is expected the copy from MCTrack, I remember some time ago (feb12) this was already present in the code, something like:

theAnalysis->FillList(mctrack, "McTruth");

I don't know if this code is still operational, anf if it provides what is expected.

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Tue, 20 Aug 2013 17:31:42 GMT
View Forum Message <> Reply to Message

I have seen only now that you already answered about the perfect mc matching

---

Subject: Re: Mc Truth Match
Posted by Tobias Stockmanns on Wed, 21 Aug 2013 06:25:06 GMT
View Forum Message <> Reply to Message

Dear Stefano and Klaus,

thank you for clarifying the ambiguities you can have for MC track matching for decaying particles. This is a point I have not thought about.

Nevertheless when using ideal track finding (as Simone did) you do not have these ambiguities. Together with the real kalman task you get tracks with covariance matrices which can be used for further fitters and the rho tools. In this case it would be good to have a (Ideal-)FillList which uses mc info to assign the correct tracks to the lists.

Cheers,

Tobias

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Wed, 21 Aug 2013 08:12:17 GMT
View Forum Message <> Reply to Message

But in such a case the PidIdeal algorithm should do the job, putting Tight instead of Loose in the FillList.
However, genfit introduces always some sort of bias, and sometime can produce a charge inversion or bad momentum. We need to wait for the genfit release 2 to understand if such problems are still present or not.

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Wed, 21 Aug 2013 11:16:39 GMT
View Forum Message <> Reply to Message

I would like to add that I have just cross-checked the method

```
theAnalysis->FillList(piall,"PionAll");
pimatchplus.Cleanup();
pimatchminus.Cleanup();
 // for pimatch
  for ( int ii=0 ; ii<piall.GetLength() ; ++ii ){
   if ( theAnalysis->McTruthMatch(piall[ii]) ){
    if ( 0 < piall[ii]->PdgCode() ){
     pimatchplus.Add( piall[ii] );
    }
    if ( 0 > piall[ii]->PdgCode() ){
     pimatchminus.Add( piall[ii] );
    }

   }
  }
```

with the following method

```
SelectWithMcTruthPdgSimulateMcMatch(piall, pdgpiplus, 211);

int SelectWithMcTruthPdgSimulateMcMatch(RhoCandList &lOld, RhoCandList &lNew, Int_t
SelectCandidatesWithPdgCode) {
 lNew.Cleanup();
 int notSelected = 0;
 for ( int ii=lOld.GetLength()-1; ii>=0; --ii ) {
  if ( ( lOld[ii]->GetMcTruth()->PdgCode() == SelectCandidatesWithPdgCode ) && (
lOld[ii]->PdgCode() == SelectCandidatesWithPdgCode ) ) {
   lNew.Add( lOld[ii] );
  } else {
   notSelected++;
  }
 }
 return notSelected;
}
```

for pions from 10k events simulated with the rho tutorial and 1k events of psi(4040) to D0 anti-D0 / D+ D- and all D decays allowed.

In other words, the McTruthMatch gives results consistent with checking the McTruth Pdg info directly (at least for charged particles, I have not checked resonances).

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Wed, 21 Aug 2013 15:26:05 GMT
View Forum Message <> Reply to Message

Well, one problem less

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Thu, 22 Aug 2013 15:53:35 GMT
View Forum Message <> Reply to Message

To avoid problems, I changed the "Loose" definition from 0.2 to 0.25.
Now, with PioneLoose, the ideal charged matching should work.
Could you please try and let me know?

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Thu, 22 Aug 2013 16:57:16 GMT
View Forum Message <> Reply to Message

If I just rerun the analysis macro I see differences between

theAnalysis->FillList(mcpiplus,"PionLoosePlus","PidAlgoIdealCharged");

and

theAnalysis->FillList(mcpiplus,"PionTightPlus","PidAlgoIdealCharged");

For a simulations of 10k events simulated with the rho tutorial macros I get (in total) 3 piplus and 1 piminus more with "Loose" than with "Tight". 3 misidentified particles have PdgCode 1000010020 and 1 has 1000010030. PID probability arrays are 0.2 0.2 0.2 0.2 0.2 in all cases.

If I only simulate 1k such events, I get no differences.

For a simulations of 1k events of psi(4040) to D0 anti-D0 / D+ D- and all D decays allowed I get (in total) 3 piplus and 1 piminus more with "Loose" than with "Tight". All misidentified particles have PdgCode 1000010020 and PID probability arrays are 0.2 0.2 0.2 0.2 0.2

I will rerun all simulations starting from the simulation macro and check again, but that will take until tomorrow.

[EDIT: I have just noticed that I incremented the event counter twice per event, so the number of inspected events is only half of what I wrote above]

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Thu, 22 Aug 2013 17:11:46 GMT
View Forum Message <> Reply to Message

Quote:PID probability arrays are 0.2 0.2 0.2 0.2 0.2 in all cases.

Then there is something strange, since the Loose cut now is > 0.25. Are you using the standard macro/params/pidana.par all.par ? Did you update such folder?

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Thu, 22 Aug 2013 17:25:02 GMT
View Forum Message <> Reply to Message

Yes, I was very surprised, too. I believe my trunk should be standard (apart from the addition of 2 externals). Here is my svn status and info:

svn info
Path: .
Working Copy Root Path: /home/pandaroot_jun13/trunk
URL: https://subversion.gsi.de/fairroot/pandaroot/trunk
Repository Root: https://subversion.gsi.de/fairroot
Repository UUID: 0381ead4-6506-0410-b988-94b70fbc4730
Revision: 21362
Node Kind: directory
Schedule: normal
Last Changed Author: spataro
Last Changed Rev: 21361
Last Changed Date: 2013-08-22 17:17:46 +0200 (Thu, 22 Aug 2013)

svn status
 M   .
X   GenfitTools/trackrep/GeaneTrackRep2
X   GenfitTools/trackrep/LSLtrackRep
X   GenfitTools/trackrep/RKTrackRep
X   MbsAPI
X   base
X   cmake
X   dbase
X   eventdisplay
X   fairtools
X   geane
X   generators
X   genfit
X   geobase
X   ora
X   parbase
X   scripts

X templates
X trackbase

Performing status on external item at 'eventdisplay':

Performing status on external item at 'generators':

Performing status on external item at 'base':

Performing status on external item at 'MbsAPI':

Performing status on external item at 'genfit':

Performing status on external item at 'parbase':

Performing status on external item at 'dbase':

Performing status on external item at 'geane':

Performing status on external item at 'cmake':

Performing status on external item at 'GenfitTools/trackrep/RKTrackRep':

Performing status on external item at 'ora':

Performing status on external item at 'scripts':

Performing status on external item at 'templates':

Performing status on external item at 'GenfitTools/trackrep/LSLtrackRep':

Performing status on external item at 'trackbase':

Performing status on external item at 'fairtools':

Performing status on external item at 'GenfitTools/trackrep/GeaneTrackRep2':

Performing status on external item at 'geobase':

I am currently rerunning all simulations from simulation to analysis and will post tomorrow about the results.

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Thu, 22 Aug 2013 20:29:07 GMT
View Forum Message <> Reply to Message

... strange...

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 08:49:19 GMT
View Forum Message <> Reply to Message

Good morning everybody.

Unfortunately, I still see the difference between

theAnalysis->FillList(mcpiplus,"PionLoosePlus","PidAlgoIdealCharged");

and

theAnalysis->FillList(mcpiplus,"PionTightPlus","PidAlgoIdealCharged");

with trunk revision 21361. I reran all my test simulations with increased statistics over night. Here are the details:

For a simulations of 20k events simulated with the rho tutorial macros I get (in total) 23 piplus and 1 piminus more with "Loose" than with "Tight". The misidentified particles have PdgCode 1000010030 and 1000010020. PID probability arrays are 0.2 0.2 0.2 0.2 0.2 in all cases.

For a simulations of 10k events of psi(4040) to D0 anti-D0 / D+ D- and all D decays allowed I get (in total) 153 piplus and 44 piminus more with "Loose" than with "Tight". All misidentified particles have PdgCode 1000010020 or 1000010030 (except for very few with 1000020040) and PID probability arrays are 0.2 0.2 0.2 0.2 0.2

Can somebody confirm these observations?

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 09:20:36 GMT
View Forum Message <> Reply to Message

Hello all,

when investigating the problem with Fill List in combination with PidAlgoIdealCharged I ran into another problem:

It seems that "Loose" even fills candidates (in very rare cases) into the list which return a 0 pointer when I try to access GetMcTruth. However, the five charged particle probabilities were set to 0.2.

I noticed that error in a simulation of 20k events with the rho tutorial macros. It happened once there.

In a simulation of 10k events for the (physically impossible) decay corresponding to the dec file quoted below I saw the same error once, too.

Decay psi(2S)
  1.0 J/psi gamma PHSP;
Enddecay

End

[PS: I only wanted photons in the events to test how PandaRoot handles them and did not care whether the decay makes any sense or not.]

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Fri, 23 Aug 2013 10:08:52 GMT
View Forum Message <> Reply to Message

In your ana macro, do you see such message:

[ERROR] init() ANAPidSelections not initialized
?

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 10:14:18 GMT
View Forum Message <> Reply to Message

Yes, here is the complete output that comes before I see my user generated content (for 10k events of psi(2S) to jpsi gamma and all jpsi decays).

root [0]
Processing mcmatchtestRemove.C...
(int)0
Warning in <TStreamerInfo::BuildCheck>:
   The StreamerInfo of class PndDrc read from file simparams.root
   has the same version (=11) as the active class but a different checksum.
   You should update the version to ClassDef(PndDrc,12).
   Do not try to write objects with the current class definition,
   the files will not be readable.

Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Double_t fPackingFraction; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Bool_t fStopTime; //

Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Double_t fPhoMaxTime; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Bool_t fTakeDirect; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Bool_t fTakeReflected; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Int_t fFocusing; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Bool_t fTakeRealReflectivity; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Bool_t fStopSecondaries; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Bool_t fStopChargedTrackAfterDIRC; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Bool_t fSetBlackLens; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  TArrayI fProc; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Int_t fPdgCode; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Double_t fThetaC; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Int_t fEventID; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
  Float_t fTime_out; //

```
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Float_t fTime_in; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Float_t fLength_in; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Float_t fLength_out; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Float_t fPEnergy; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Float_t fLambda1; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Float_t fDeltaT; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Double32_t fTrackTime; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Int_t fSenId1; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Int_t fSenId2; //
Warning in <TStreamerInfo::CompareContent>: The following data member of
the in-memory layout version 11 of class 'PndDrc' is missing from
the on-file layout version 11:
   Int_t fSenIdBar; //
FairRootManager::OpenOutFile("output.root")
[INFO   ] The input consists out of the following trees and files:
[INFO   ]  - cbmsim
[INFO   ]     - pid_complete.root
[INFO   ] Parameter and input file are available, Assure that basic info is there for the run!
[INFO   ] The number of entries in chain is 10000
[INFO   ]  Branch: MCEventHeader.  not found in Tree
[INFO   ]  Branch: MCEventHeader.  not found in Tree

************************************************************
    initialisation for run id 1377190965
************************************************************
```

```
-I- FairRunTimeDB::InitContainer() FairBaseParSet
Info in (PndGeoHandling::Instance): Making a new instance using the framework.
Error in <TBufferFile::CheckByteCount>: object of class PndDrc read too many bytes: 27185 instead of 27177
Warning in <TBufferFile::CheckByteCount>: PndDrc::Streamer() not in sync with data on file simparams.root, fix Streamer()
Info in <TGeoManager::CloseGeometry>: Geometry loaded from file...
Info in <TGeoManager::SetTopVolume>: Top volume is cave. Master volume is cave
Info in <TGeoNavigator::BuildCache>: --- Maximum geometry depth set to 100
Info in <TGeoManager::Voxelize>: Voxelizing...
Info in <TGeoManager::CountLevels>: max level = 12, max placements = 4896
Info in <TGeoManager::CloseGeometry>: 2008273 nodes/ 5804 volume UID's in FAIR geometry
Info in <TGeoManager::CloseGeometry>: ----------modeler ready----------
Container FairBaseParSet initialized from ROOT file.

************************************************************
    initialisation for run id 1377190965
************************************************************
-I- FairRunTimeDB::InitContainer() FairBaseParSet
-I- FairRunTimeDB::InitContainer() PndSensorNamePar
Container PndSensorNamePar initialized from ROOT file.
PndFieldCreator::SetParm() 0x8890830

************************************************************
    initialisation for run id 1377190965
************************************************************
-I- FairRunTimeDB::InitContainer() FairBaseParSet
-I- FairRunTimeDB::InitContainer() PndSensorNamePar
-I- FairRunTimeDB::InitContainer() PndMultiFieldPar
Container PndMultiFieldPar initialized from ROOT file.
OBJ: PndTransPar        PndTransPar     Trans. Field parameter container
OBJ: PndDipole1Par      PndDipole1Par   Dipole Field parameter container
OBJ: PndDipole2Par      PndDipole2Par   Dipole Field parameter container
OBJ: PndSolenoid1Par    PndSolenoid1Par Solenoid 1st region parameter container
OBJ: PndSolenoid2Par    PndSolenoid2Par Solenoid 2nd region parameter container
OBJ: PndSolenoid3Par    PndSolenoid3Par Solenoid 3rd region parameter container
OBJ: PndSolenoid4Par    PndSolenoid4Par Solenoid 4th region parameter container
[INFO   ] PndFieldMap: Reading field map from ROOT file
/home/pandaroot_jun13/trunk/input/TransMap.0890.root
[INFO   ] PndFieldMap: Reading field map from ROOT file
/home/pandaroot_jun13/trunk/input/DipoleMap1.0890.root
[INFO   ] PndFieldMap: Reading field map from ROOT file
/home/pandaroot_jun13/trunk/input/DipoleMap2.0890.root
[INFO   ] PndFieldMap: Reading field map from ROOT file
/home/pandaroot_jun13/trunk/input/SolenoidMap1.root
[INFO   ] PndFieldMap: Reading field map from ROOT file
/home/pandaroot_jun13/trunk/input/SolenoidMap2.root
[INFO   ] PndFieldMap: Reading field map from ROOT file
/home/pandaroot_jun13/trunk/input/SolenoidMap3.root
[INFO   ] PndFieldMap: Reading field map from ROOT file
/home/pandaroot_jun13/trunk/input/SolenoidMap4.root
```

-I- PndAnalysis::Init(): Trying now.
[INFO   ] Branch: not found in Tree
[INFO   ] Branch: not found in Tree
-I- PndAnalysis::ReadTCA(): No array found.
-I- PndAnalysis::Init(): Trying SttMvdGenTrack now.
[INFO   ] Branch: LheGenTrack not found in Tree
[INFO   ] Branch: LheGenTrack not found in Tree
-I- PndAnalysis::ReadTCA(): No LheGenTrack array found.
-I- PndAnalysis::Init(): Trying SttMvdGenTrack now.
[INFO   ] Branch: SttMvdGenTrack not found in Tree
[INFO   ] Branch: SttMvdGenTrack not found in Tree
-I- PndAnalysis::ReadTCA(): No SttMvdGenTrack array found.
-I- PndAnalysis::Init(): Trying SttMvdGemGenTrack now.
[INFO   ] Branch: SttMvdGemGenTrack not found in Tree
[INFO   ] Branch: SttMvdGemGenTrack not found in Tree
-I- PndAnalysis::ReadTCA(): No SttMvdGemGenTrack array found.
-I- PndAnalysis::Init(): Trying BarrelGenTrack now.
[INFO   ] Branch: BarrelGenTrack not found in Tree
[INFO   ] Branch: BarrelGenTrack not found in Tree
-I- PndAnalysis::ReadTCA(): No BarrelGenTrack array found.
-E- PndAnalysis::Init(): No track inpt array. Make a Zero TclonesArray.
-I- PndAnalysis::Init(): Second: Trying now.
[INFO   ] Branch: not found in Tree
[INFO   ] Branch: not found in Tree
-I- PndAnalysis::ReadTCA(): No array found.
-I- PndAnalysis::Init(): Second: Trying FTSGenTrack now.
[INFO   ] Branch: FTSGenTrack not found in Tree
[INFO   ] Branch: FTSGenTrack not found in Tree
-I- PndAnalysis::ReadTCA(): No FTSGenTrack array found.
-I- PndAnalysis::Init(): Second: Trying FTSTrkIdeal now.
[INFO   ] Branch: FTSTrkIdeal not found in Tree
[INFO   ] Branch: FTSTrkIdeal not found in Tree
-I- PndAnalysis::ReadTCA(): No FTSTrkIdeal array found.
[INFO   ] Branch: PndMcTracks not found in Tree
[INFO   ] Branch: PndMcTracks not found in Tree
-I- PndAnalysis::ReadTCA(): No PndMcTracks array found.
-I- PndAnalysis::Init(): Trying mc stack now.

****
    initialisation for run id 1377190965
****
-I- FairRunTimeDB::InitContainer() FairBaseParSet
-I- FairRunTimeDB::InitContainer() PndSensorNamePar
-I- FairRunTimeDB::InitContainer() PndMultiFieldPar
-I- FairRunTimeDB::InitContainer() ANAPidSelections
[ERROR] init() ANAPidSelections not initialized
Error in <FairRuntimeDb::initContainers()>: Error occured during initialization
[INFO   ] The number of entries in chain is 10000
[INFO   ] The number of entries in chain is 10000

[Edit: The output above is for the gamma simulation, I do not see the PndDrc errors in the

simulation of psi(4040) to D0 anti-D0 / D+ D- and all D decays or the standard rho tutorial decay. The rest of the output is identical apart from run numbers.]

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Fri, 23 Aug 2013 10:38:51 GMT
View Forum Message <> Reply to Message

Ok, I found that the tutorial analysis macro is not loading the ascii param file.
Can you try the following:

```
      // *** initialization
      FairRunAna* fRun = new FairRunAna();
      FairRuntimeDb* rtdb = fRun->GetRuntimeDb();
      fRun->SetInputFile(inPidFile);

+      TString anaParFile = gSystem->Getenv("VMCWORKDIR");
+      anaParFile +=  "/macro/params/all.par";
+      FairParAsciiFileIo* parIo1 = new FairParAsciiFileIo();
+      parIo1->open(anaParFile.Data(),"in");

      FairParRootFileIo* parIO = new FairParRootFileIo();
      parIO->open(inParFile);
      rtdb->setFirstInput(parIO);
+      rtdb->setSecondInput(parIo1);

      rtdb->setOutput(parIO);
```

Please add the "+" lines in your analysis macro, and let me know.

(I realized that by default the "clear" value for Loose is 0.2, and that you were not loading the updated pidana.par).

After you should see:

```
-I- FairRunTimeDB::InitContainer() ANAPidSelections
ANAPidSelections initialized from Ascii file
```

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 10:47:47 GMT
View Forum Message <> Reply to Message

Thank you very much Stefano!

I have just tried that fix and now I do not see any difference between Loose and Tight any more in all simulations that I investigated previously.

Subject: Re: Mc Truth Match

Posted by Klaus Götzen on Fri, 23 Aug 2013 11:10:08 GMT
View Forum Message <> Reply to Message

Hi,

just a comment to the missing McTruth. The values of 0.2 for all particle species are the default values, i.e. you don't have any knowledge about the true particle type.

If not McTruth object can be assigned (what sometimes happens for whatever reason), the defaults are not altered anymore, even in PidAlgoIdealCharged.

One could think about excluding those particles from the list. Any opinions?

Best,
Klaus

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 11:15:27 GMT
View Forum Message <> Reply to Message

Maybe PidAlgoIdealCharged could assign 0 for all of the 5 particle hypotheses we currently have if the track is not associated to any mc truth track or to a track which is neither electron, nor muon, pion, proton or kaon? Then it should also not matter which numerical value Loose corresponds to.

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Fri, 23 Aug 2013 11:19:57 GMT
View Forum Message <> Reply to Message

Once you merge more than 1 algorithm, if one algorithm is not able to provide a selection (prob = 0.2), the 2nd algorithm will provide a number and in the pid moltiplication the 2nd will win. In this sense the track has not to be rejected (i.e. an emc algorith w/o emc signal).

If you multiply the ideal algo with a real algo, you can do purity/eficiecny calculation in an easy way. For this reason we decided to set the prob at 0.2 and not to exclude them.

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Fri, 23 Aug 2013 11:22:01 GMT
View Forum Message <> Reply to Message

Setting the pdfs at zero bring to a division by zero (pdf/sum(pdf)=0), and it crashes once you calcualte the probability.

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 11:43:22 GMT
View Forum Message <> Reply to Message

Ah, ok, I did not know that. It makes sense to me now.

Subject: Re: Mc Truth Match
Posted by Simon Reiter on Fri, 23 Aug 2013 12:04:03 GMT
View Forum Message <> Reply to Message

I suggest to let the probabilities at 0.2 to get a sum of 1. But what about implementing a check if all probabilities are the same, and set the pdg information to 0? I guess this can implemented in PndAnaPidSelector::Accept, or?

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 12:10:11 GMT
View Forum Message <> Reply to Message

The pdg information is a (mass) hypothesis which is needed for calculating invariant masses of resonances and so on. Therefore, it should be assigned to what you specify when you use FillList if the probability of the track being the particle species you asked for is >= your desired probability.

Of course, one can filter the list lateron using McTruthMatch for instance to filter out wrong assignments for efficiency calculations and so on.

Subject: Re: Mc Truth Match
Posted by MartinJGaluska on Fri, 23 Aug 2013 12:38:16 GMT
View Forum Message <> Reply to Message

How about introducing a sixth category: "Charged, other than electron, muon, pion, kaon, proton"? PidAlgoIdealCharged could set all probabilities to 0 and the sixth probability to 1. Or would that introduce other problems?

Subject: Re: Mc Truth Match
Posted by StefanoSpataro on Fri, 23 Aug 2013 15:15:53 GMT
View Forum Message <> Reply to Message

Apart from the "fake" definition,
can I flag this topic as "FIXED"?

Subject: Re: Mc Truth Match
Posted by Klaus Götzen on Fri, 23 Aug 2013 15:31:10 GMT
View Forum Message <> Reply to Message

This depends on whether we want to change structure or add another PID type different to e, mu, pi, K, p.