Update Hot Channel Finder

Jaein Hwang

Thursday 14 Sep 2023

Motivation

- Previous Definition of Hot Channel

$\frac{\#\text{ of hits of a channel}}{\#\text{ of hits of a chip}} > C \rightarrow$ channel is classified as a hot channel.

1) In the case all channels in a chip are relatively hot compared to others, this algorithm cannot find this.

2) In cosmics run, the statistics is not enough for all channels to have some hits.

- Hot Channel masking depends on the purpose of analysis.

Aggressive masking cut? or loose masking cut?

- Users can load masking files quickly and use them in their analysis.

Hot Channel Finder(1)

Location of code (SDCC machine)

SDCC /sphenix/tg/tg01/commissioning/INTT/work/jaein/HotChannelFinder/orginalcode

INTT GitHub

sPHENIX/INTT/general_codes/Jaein/HotChannelFinder

- HotChannel.C : root macro to make the hot channel list.

- run_hot.sh: bash shell script to run HotChannel.C for every 8 Felix servers simultaneously.

- HotChannelApply.cc/h : C++ based function to load(use) the hot channel list in your analysis code.

Hot Channel Finder(2)

Definition of Hot Channel

OLD : $\frac{(\#\text{ of hits of a channel})}{(\#\text{ of hits of a chip})} > C$

NOW : $(\#\text{ of hits of a channel}) > \frac{(\text{Total \# of hits in a server})}{(\#\text{ of hit Channels in a server})} \times C$

User can decide C constant value in your analysis.

List of value : 1.3(most aggressive cut) 1.4 1.5 2.0 2.5 3.0(loosest cut) for now.

Ex) If you need an aggressive hot channel list, pick up 1.3 as constant value.

HotChannel.C

- Making the hot channel list to use hit-based file
  Requirement : hit-based file at /sphenix/tg/tg01/commissioning/INTT/root_files
- How to use
  root -l -q 'HotChannel.C({your_run_number},{Felix server})’

  ex) Runnumber 25992, felix server = 1

  root -l -q 'HotChannel.C(25992,1)’

  To make the hot channel list for all felix server, run_hot.sh is strongly recommended.

  bash run_hot.sh {your_run_number}

  ex) bash run_hot.sh 25992

Hot lists are saved at :
/sphenix/tg/tg01/commissioning/INTT/work/jaein/HotChannelFinder/hotchannelmap
Do not change the directory!

HotChannelApply.cc/h

- C++ function to load the HotChannel map in your analysis code.
- How to use

0. Make the hot Channel list through HotChannel.C

1. Copy HotChannelApply.cc/h to your directory you wans to use.

2. Add #include “HotChannelApply.h/cc” in your code.

3. Make bool MyMap[8][14][27][128]
(Felix server = 0~7, ladder = 0~13 ,chip 1~26 ,channel 0~127)

4. Load Hot Channel list

HotChannelApply(myMap,{$1},{$2},{$3});

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <sstream>
#include <cstring>
#include "HotChannelApply.h"
#include "HotChannelApply.cc"

void Test_HotChannelApply()
{
  bool myMap[8][14][27][128];
  for (int i = 0; i < 8; i++)
  {
    HotChannelApply(myMap, 25478, i, 1.5);
  }
```

$1 : run number $2 : Felix server, $3 : Constant value for Hot channel selection. **1.3, 1.4, 1.5, 2.0 2.5 3.0**

Ex) Felix 6, ladder 3, chip 1, channel 127 is(is not) hot, MyMap[6][3][1][127] returns TRUE(FLASE)

Example to use the code

Ex) Felix 6, ladder 3, chip 1, channel 127 is(is not) hot, MyMap[6][3][1][127] returns TRUE(FLASE)

```
26  TFile *ofile = new TFile(of.c_str());
27  TTree *tree = (TTree *)ofile->Get("tree");
28  tree->Print();
29  int module, chan_id, chip_id, adc;
30  tree->SetBranchAddress("module", &module);
31  tree->SetBranchAddress("chan_id", &chan_id);
32  tree->SetBranchAddress("chip_id", &chip_id);
33  tree->SetBranchAddress("adc", &adc);
34  tree->SetBranchStatus("*", 0);
35  tree->SetBranchStatus("module", 1);
36  tree->SetBranchStatus("chip_id", 1);
37  tree->SetBranchStatus("adc", 1);
38  tree->SetBranchStatus("chan_id", 1);
39  for (int i = 0; i < tree->GetEntries(); i++)
40  {
41    tree->GetEntry(i);
42    histbefore->Fill(chan_id, chip_id);
43    if (myMap[felix][module][chip_id][chan_id])
44      continue;
45    histafter->Fill(chan_id, chip_id);
46  }
```

Example code is at
/sphenix/tg/tg01/commissioning/INTT/work/jaein/HotChannelFinder/Test_HotChannelApply.C

The idea is :

1. User just add the bool types to ignore hot channel in your analysis.

2. It's also possible to add this logic before merging the 8 files from individual Felix servers to 1 file.

If channel is(not) hot, return is true(false)

Future plan

- Maintenance of code, some debugging

Welcome any comments, any problems about the hot channels

- Code is needed hit-based files. Change to event-based files? (If needed)

Another cut option for analysis

- BCOFULL-BCO cut is needed to filter out some noise.

- Automation algorithm to find the BCO peak (ongoing)

- Code to apply the BCO cut