

Hot ChannelのAuto化

2023/8/23

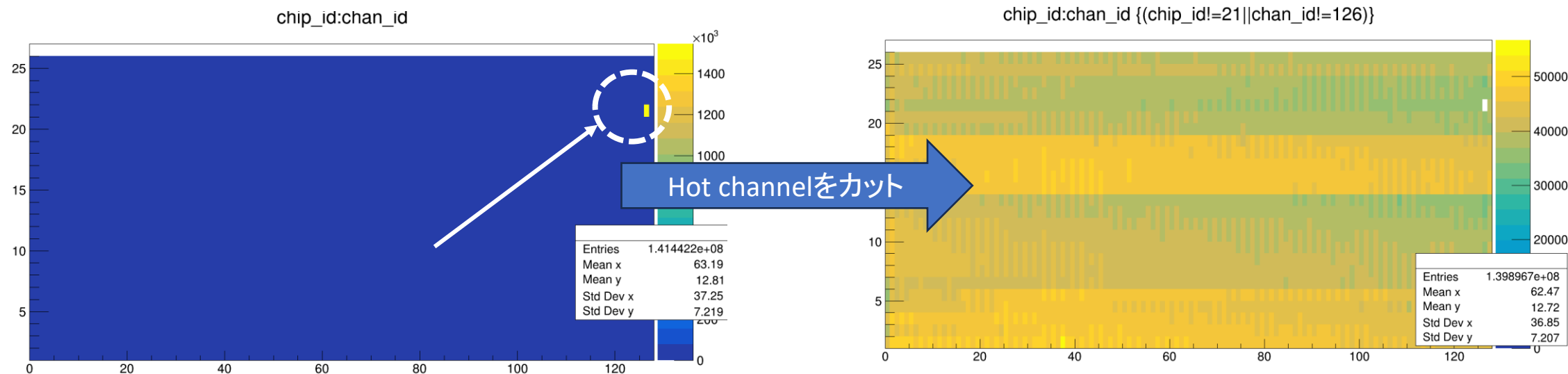
NWU M2 杉山由佳

Hot ChannelのAuto化

あるChannelで他の箇所よりもはるかにエントリー数が多いとき、そのChannelをHot Channelと呼んでいる。

今までtree->Draw()でChip:Chan分布を表示し、エントリー数が多いChannelを目視で判断し、そのHotChannelでのヒットをカットしていた。

しかし、ラダーごとにHotChannelを探すのは時間を要するなど手間がかかるため、HotChannelカットのAuto化を進めている。



Hot ChannelのAuto化

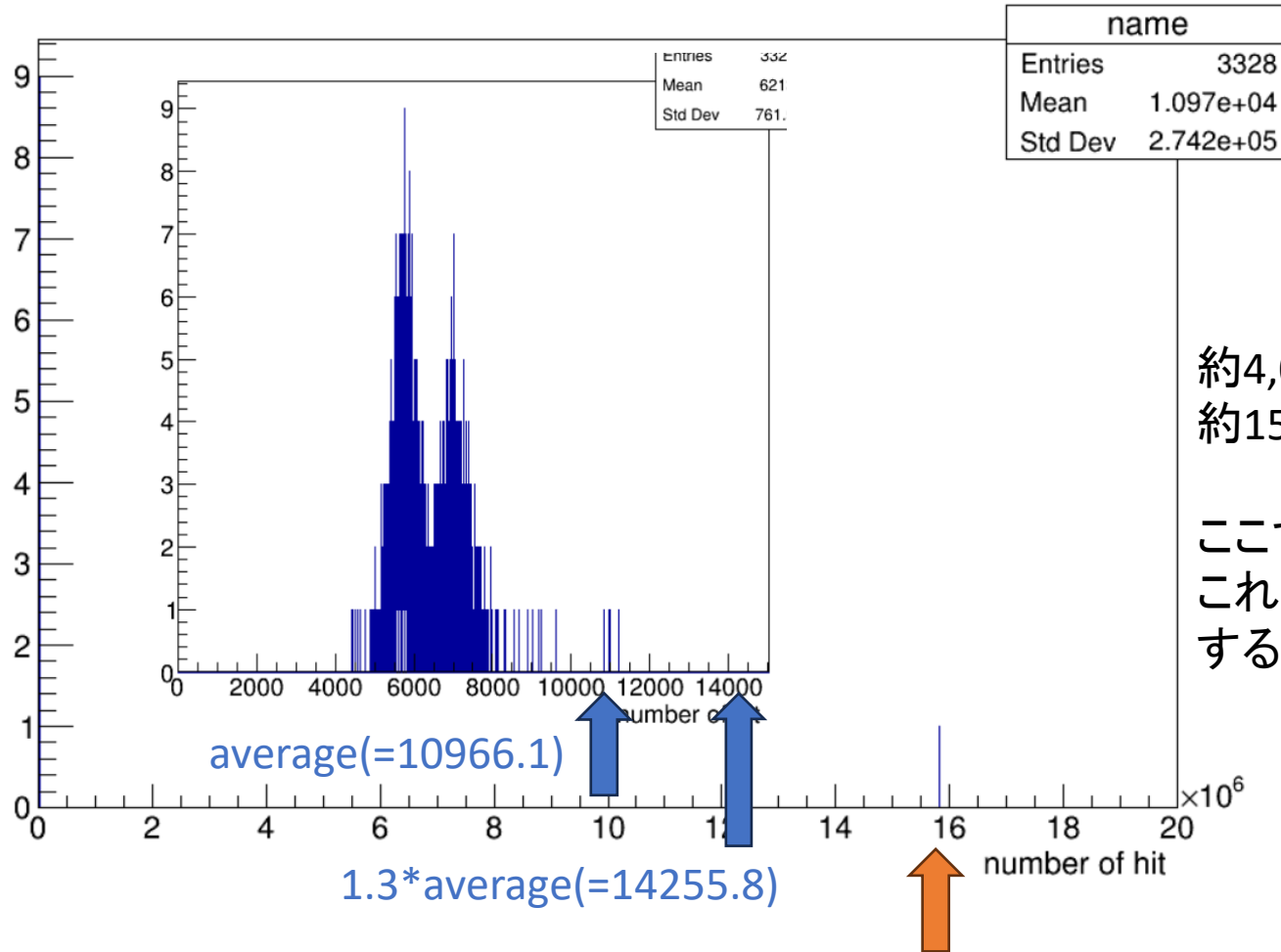
マクロの流れ

1. 2次元ヒストにChip番号、Chan番号の情報をつめる。`h1->Fill(chan_id, chip_id);`
2. 全Chanのエントリーの合計を求める。`count = count + h1->GetBinContent(a + 1, b + 1);`
3. 各Chanのエントリー数の平均値`average`を求める。`average = count / (26 * 128);`
4. (あるChanでのエントリー数) > $1.3 * average$ の場合、Hot Channelとみなす。
5. データを読み出すときにHotChanをカットし、データを読み出す。`if(HotChanのヒットを含まない)vector(chip_id);`

Hot Channelの基準の根拠は4ページに記す。

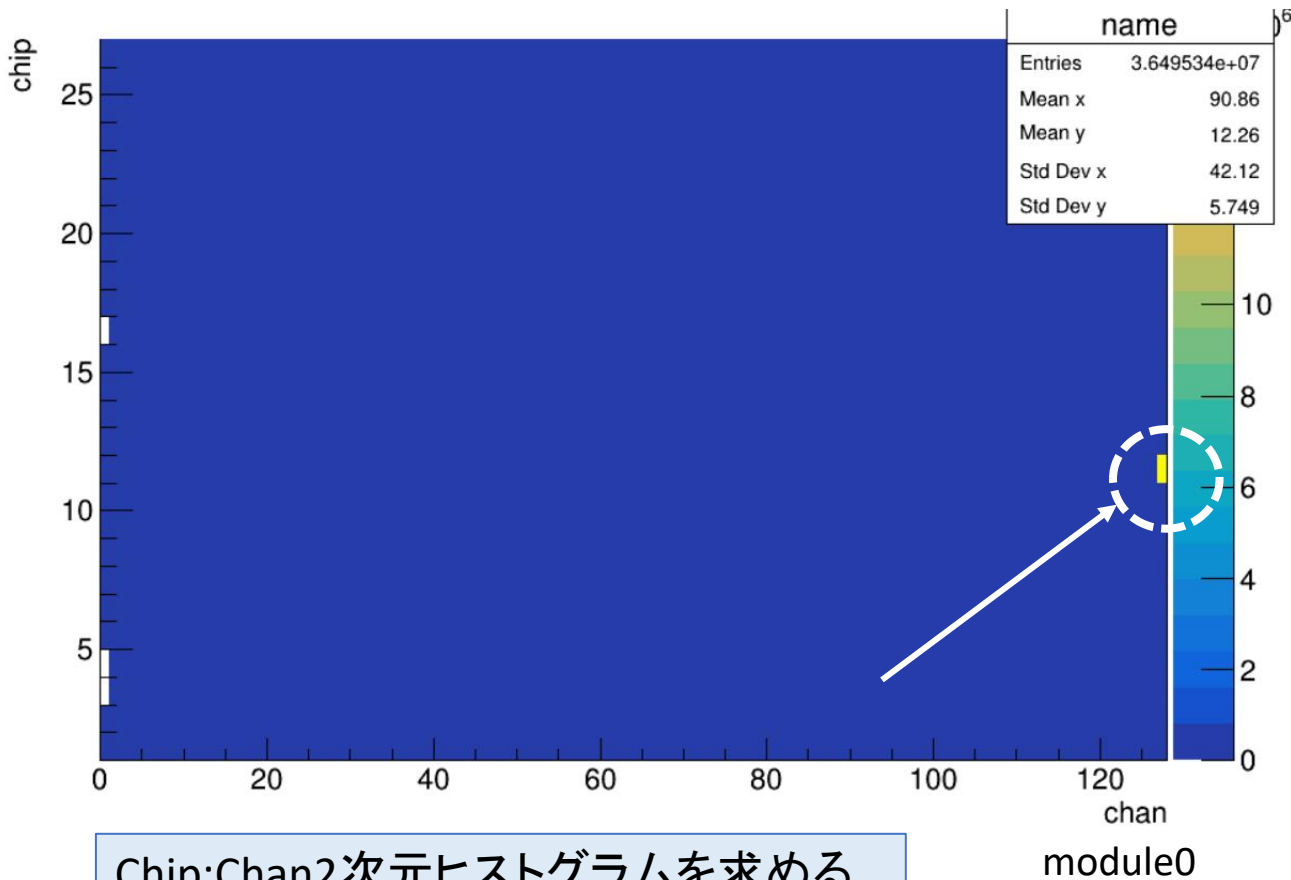
Run21537(ヒットベース)を解析し、今回はmodule0におけるHotChannelを求めた。

横軸Channelごとのエントリー数とした1次元ヒストグラム



約4,000-12,000にエントリー数が集まっている一方、約15,000,000とエントリーがはるかに多いChannelもある。

ここで、エントリーの平均値の1.3倍をHotChanの基準とした。これはmodule0の結果であるが、他ラダーについても今後確認する予定である。



これらの手順をマクロに書き記すことで、HotChannelのAuto化が可能になった。

Chip:Chan2次元ヒストグラムを求める

```
for (int a = 0; a < 26; a++) {
    for (int b = 0; b < 128; b++) {
        if (h1->GetBinContent(b+1, a + 1) > (1.3*average))cout << "Chip"<< a + 1 << " Chan" << b << endl;
    }
}
```

Chip11 Chan127

(あるChanでのエントリー数) > 1.3*average の場合、Hot Channelとする。

Hot Channel一覧

まとめ/今後の予定

Hot ChannelのAuto化マクロ

- “各Chanごとのエントリー数”と“エントリー数の平均値”の比較によりマクロを使ってHotChannelを求められた。
- 今後は他ラダーについてもChannelごとのエントリー数の1次元ヒストグラムを求めることで、HotChannelの基準値を決める予定である。

Back Up

```
int nEntry = tree->GetEntries(); //treeのEntry数を取得して代入する
for (int iEntry = 0; iEntry < nEntry; ++iEntry) { //すべてのEntryでループをまわす
    // for (int iEntry = 0; iEntry < 100000; ++iEntry) { //すべてのEntryでループをまわす
        tree->GetEntry(iEntry);
        if (module == l[dataset]) {
            int x = 0, y = 0;
            y = chip_id;
            x = chan_id;
            h1->Fill(x, y);
        }
    }
    c1->cd();
    h1->Draw("colz");
    c2->cd();
    h2->Draw("");

    double count = 0, average = 0;
    for (int a = 0; a < 26; a++) {
        for (int b = 0; b < 128; b++) {
            count = count + h1->GetBinContent(b+1, a + 1);
            h2->Fill(h1->GetBinContent(b+1, a + 1));
        }
    }
    cout<<h1->GetBinContent(128, 11)<<endl;
    average= count / (26 * 128);
    cout << average << endl;

    for (int a = 0; a < 26; a++) {
        for (int b = 0; b < 128; b++) {
            if (h1->GetBinContent(b+1, a + 1) > (1.3*average))cout << "Chip"<< a + 1 << " Chan" << b << endl;
        }
    }
}
```