Progress Report

M2 Takumi Kumayama

困っていること

▶ **ITkの同期が取れていない。**

- SWのバージョンアップによるデータフレームの変更は無し。
- 高頻度で大量にデータを落としていそう。
- スキップイベントを頑張るしかなさそう。
- ITk telだけでなくv2の同期も取れていない。

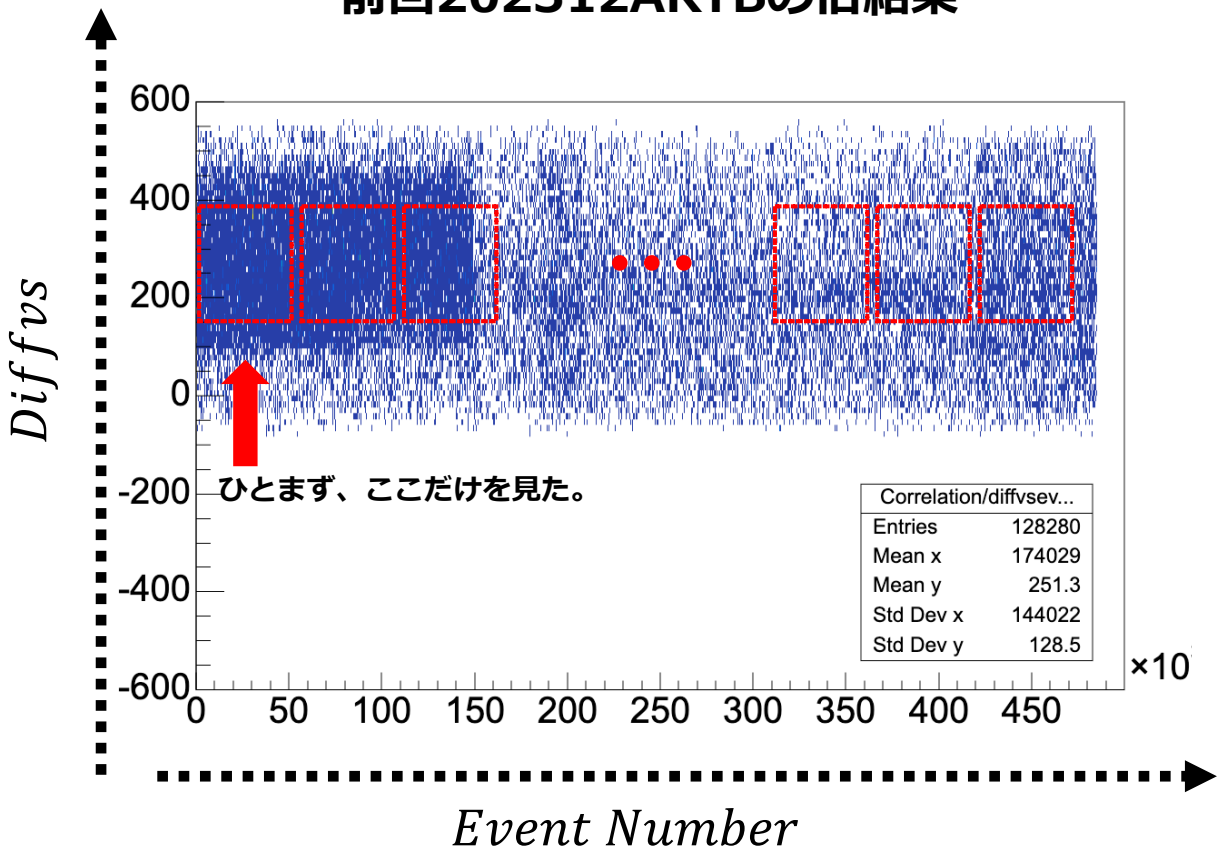
▶ **今後の方針。**

- ひとまず、イベントスキップを頑張る(?)。
- ただ、高頻度で落とし続けているようなら修復は無理な気がしてきている。
- アドバイス求むです。

同期復活のアルゴリズム考案①

▶ ノーヒントでHSIO2のskip Eventを見つけ出すのは無理すぎる。

前回202312ARTBの旧結果



① 1event目を1skipしたときの赤四角領域のエントリーをカウント。

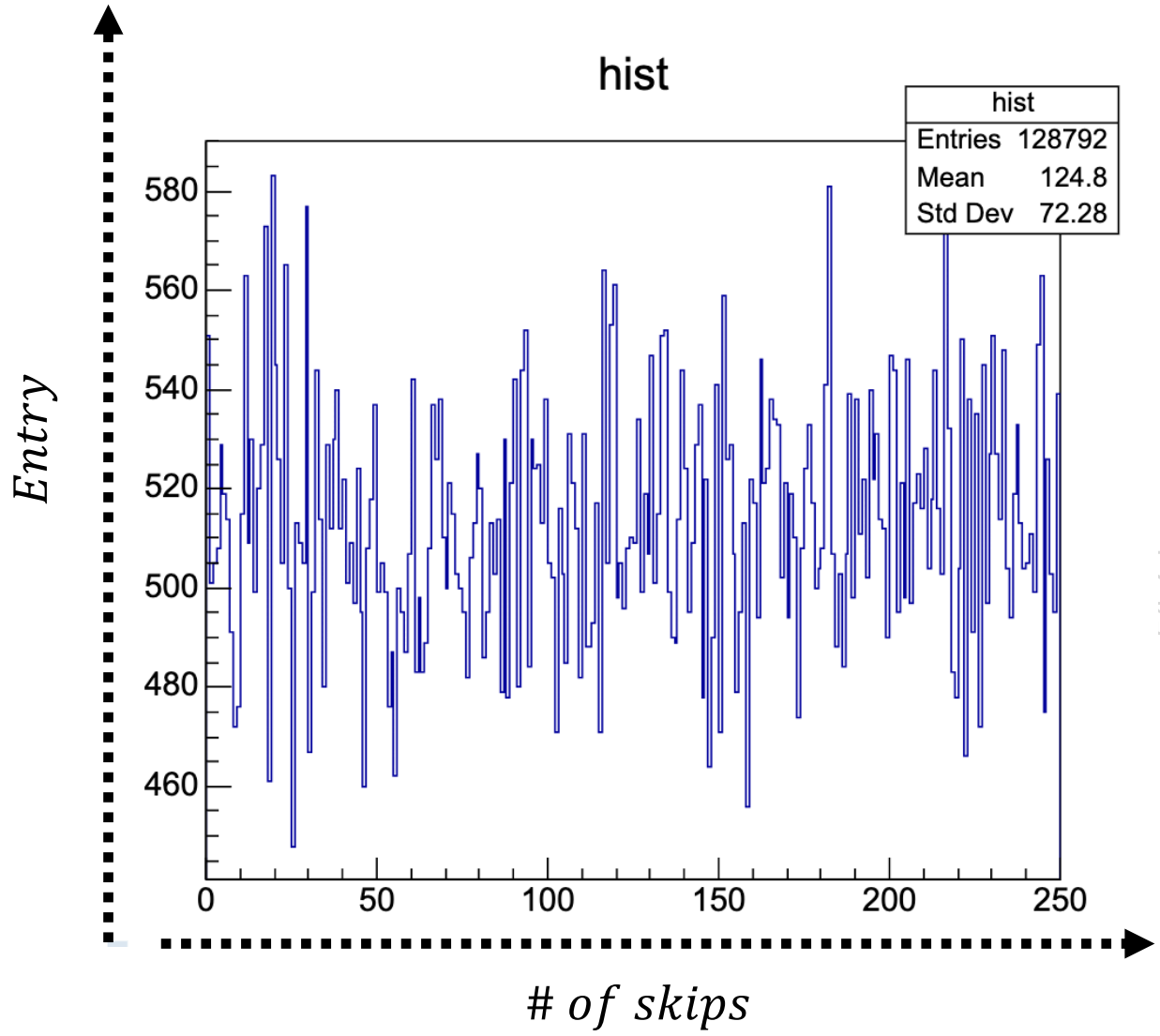
② 同期が途中で途切れる可能性があるため領域は細かく見る(500events)。

③ 同期が取れているタイミングではエントリーが増え、同期が切れるとエントリーが減ると予想。

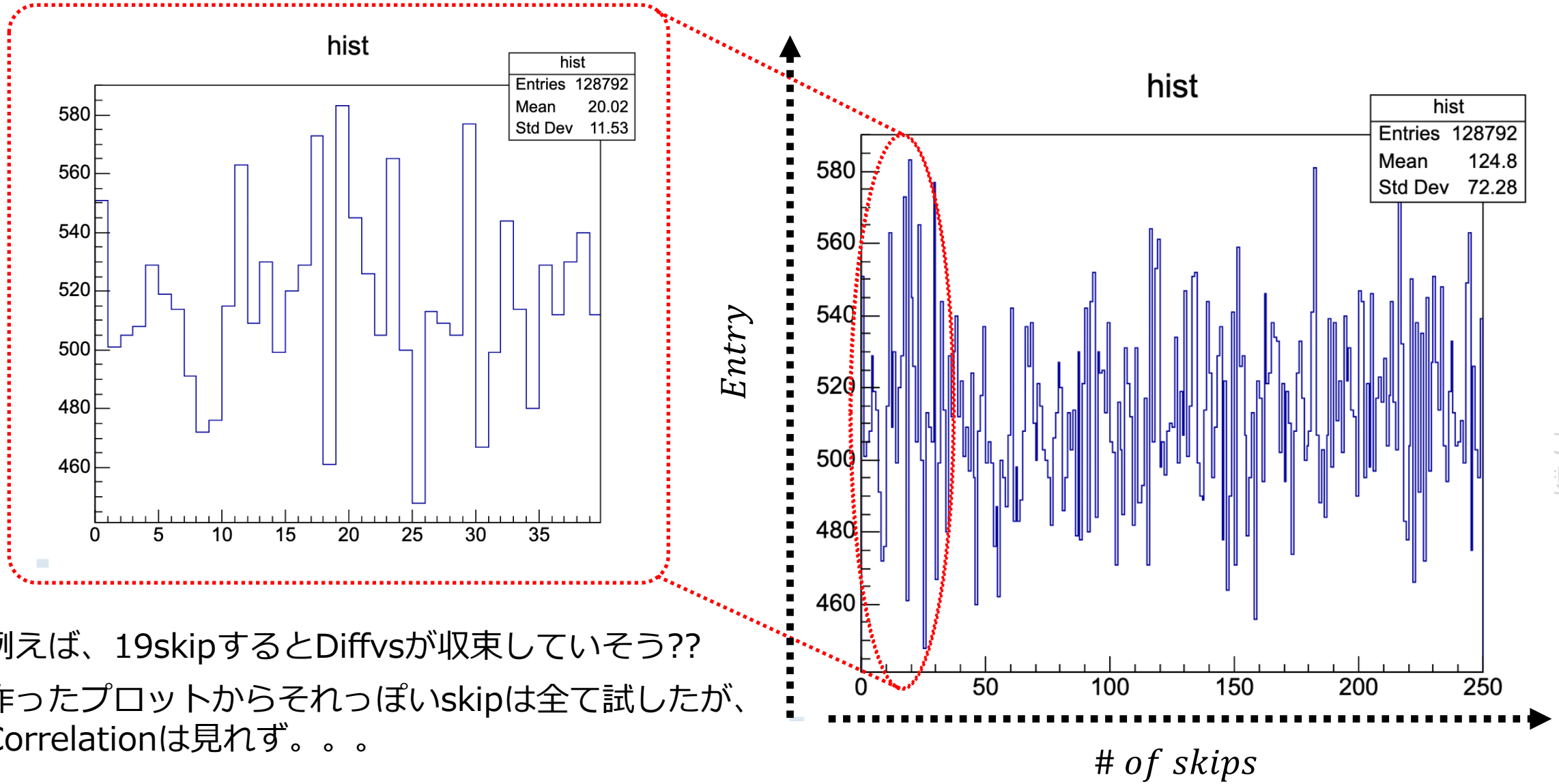
④ 頭からskipするイベント数を増やしていく。先頭から250skipまで繰り返す。

skip数 vs エントリー数のプロットを作成

同期復活のアルゴリズム考案①



同期復活のアルゴリズム考案①



- 例えば、19skipするとDiffvsが収束していそう??
- 作ったプロットからそれっぽいskipは全て試したが、Correlationは見れず。。。

Back Up

困っていること

▶ **ITkの同期が取れていない。**

・ITkがイベントをランダムに落としてしまうせいで同期が取れない。

・12月のARTBではrunの途中で同期が切れていたが、今回は最初からコリが見れない。

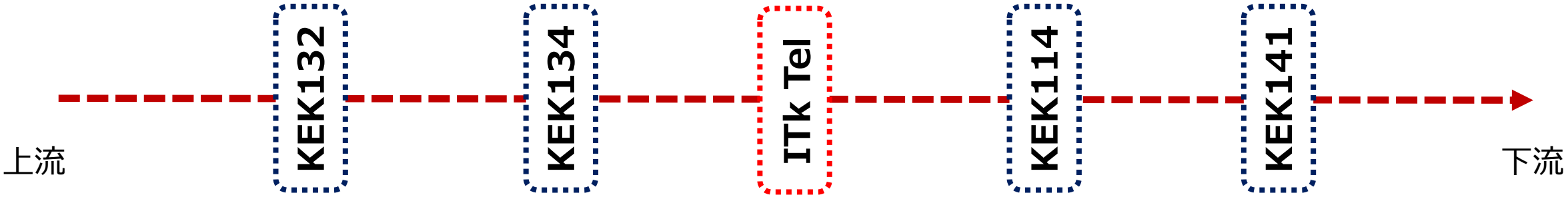
・ITk telだけでなく、v2モジュールでも同期が取れていない。ITk全滅。。

▶ **熊山なりに頑張って見たこと。**

・データフレームが旧YARRで取得したものと変わっているか確認。→ 何も変わってない。

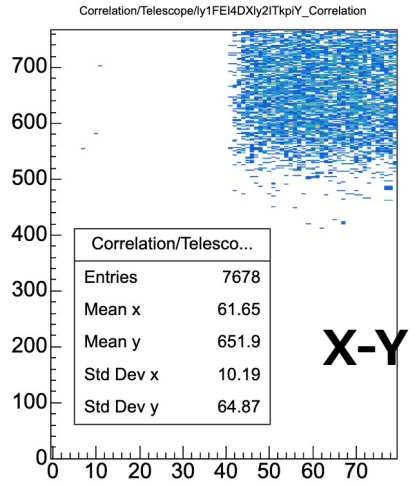
・アルゴリズムを組んで適切なスキップイベント数を探してみた。 → 未だ同期取れず。

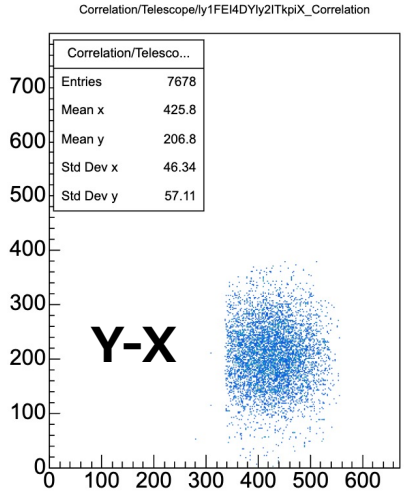
set up



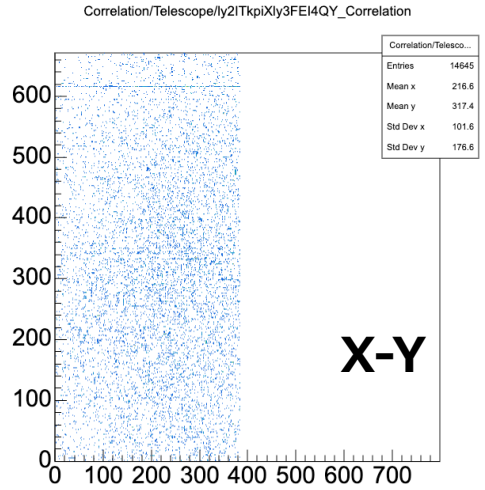
ROTATE_Z	180	270	0	90	180

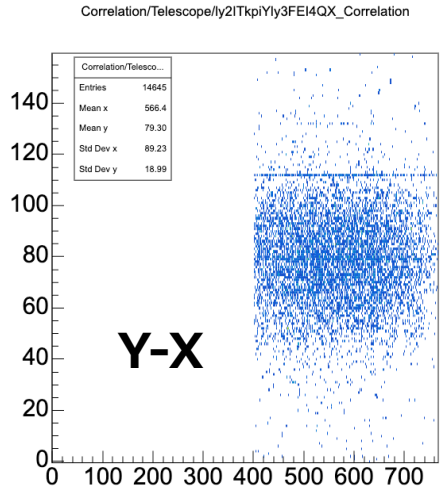
134とITkのコリレーション





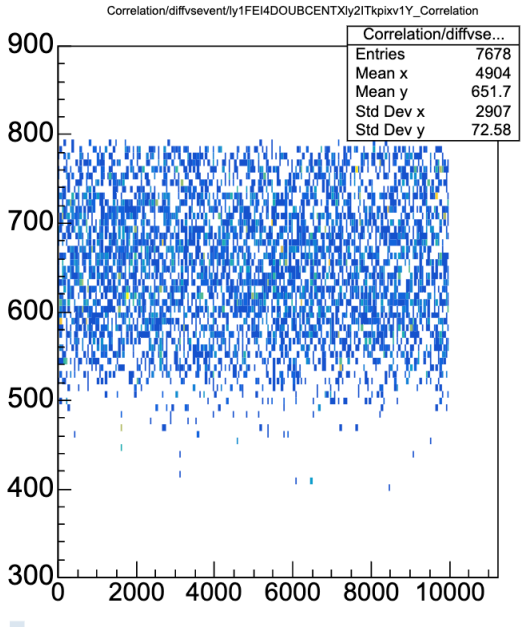
ITkと114のコリレーション



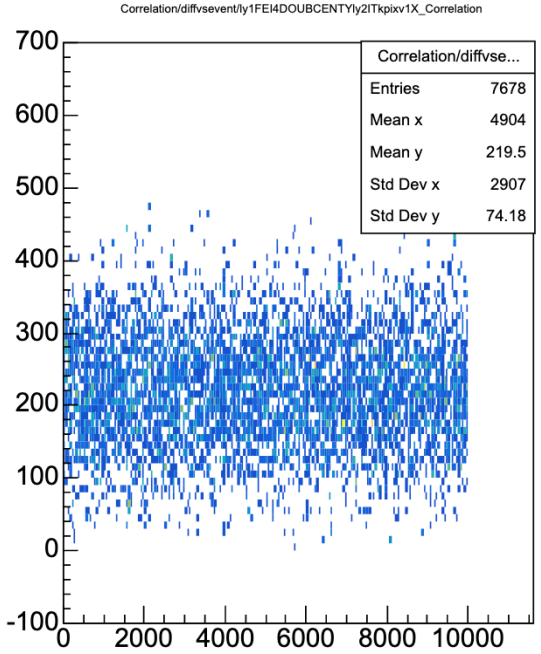


Diffvs KEK134-ITktel

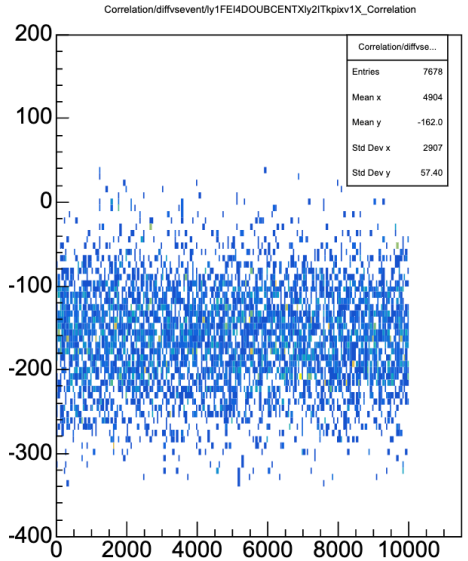
同期が最初から取れていない。どこからskipすれば回復するのか当たりが付けられない。



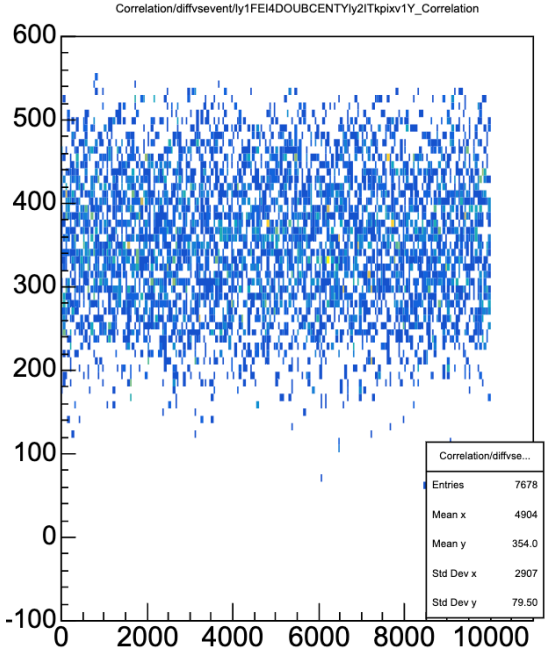
X-Y



Y-X



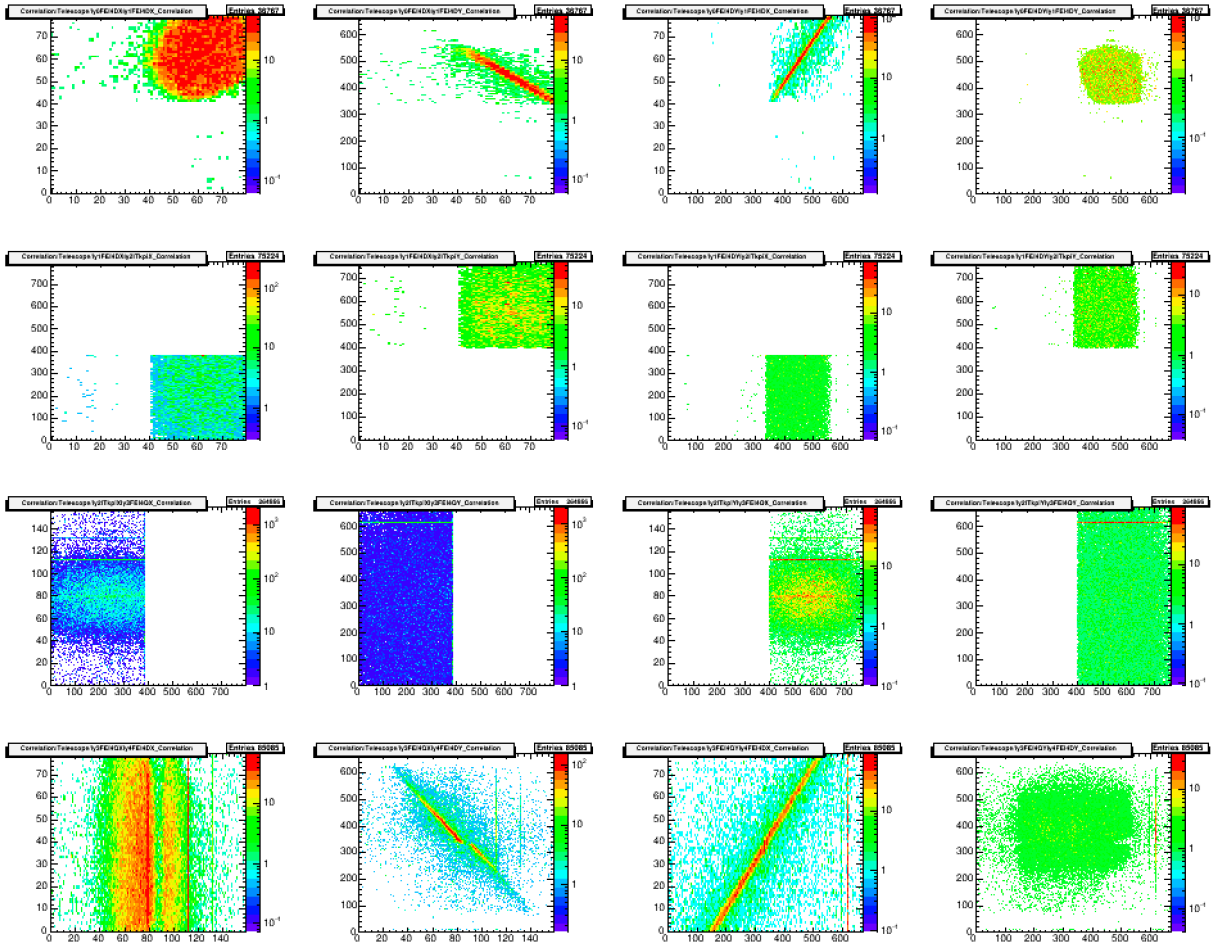
X-X



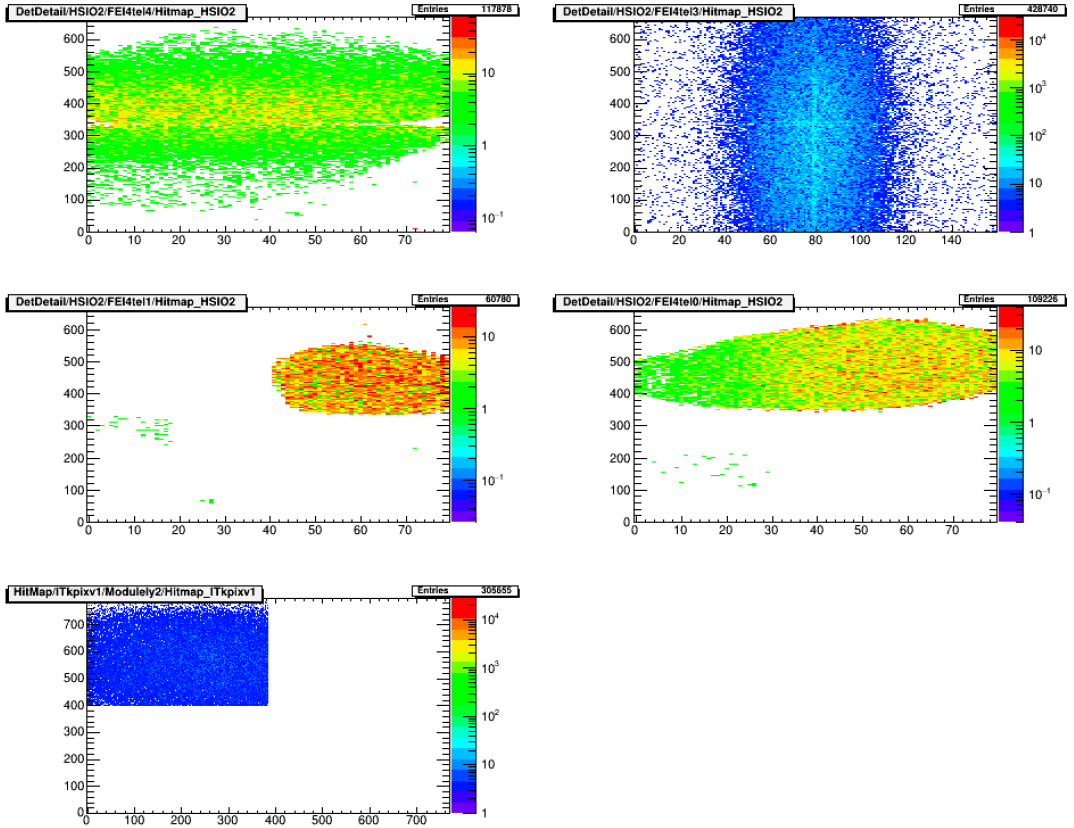
Y-Y

Onlinemon TLU=340

Correlation

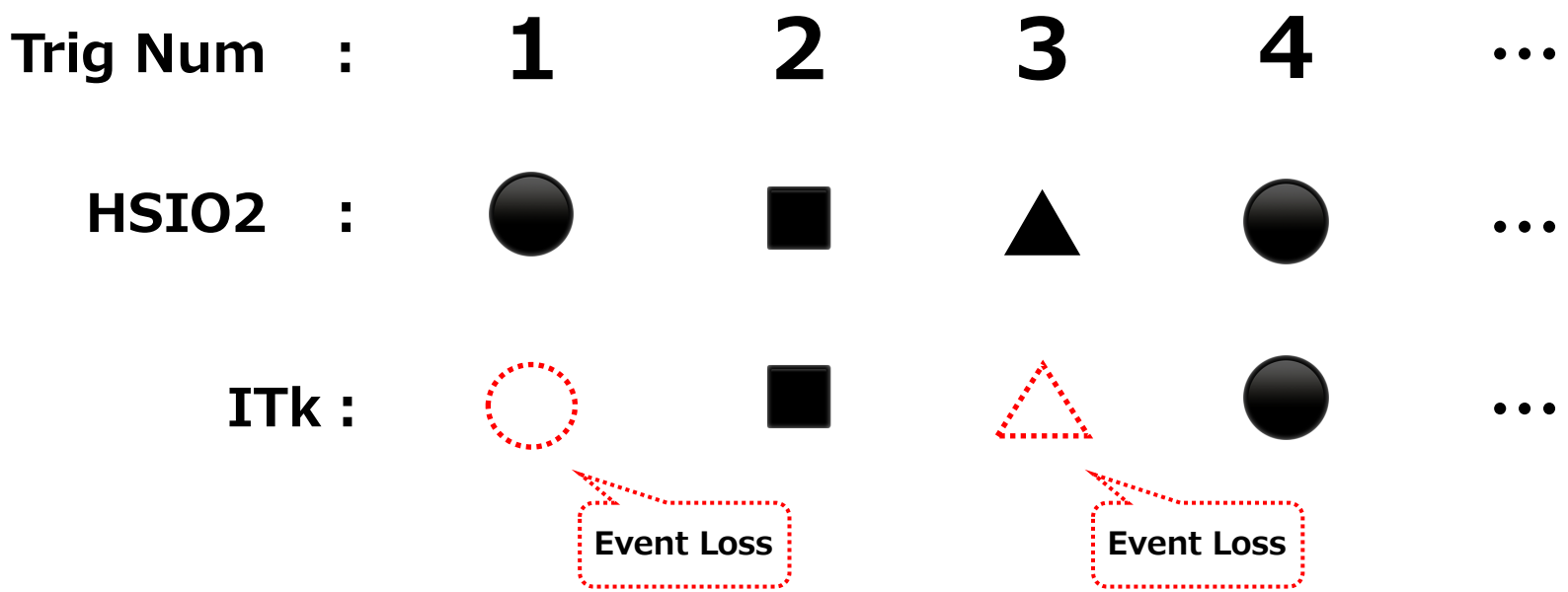


Hit Map



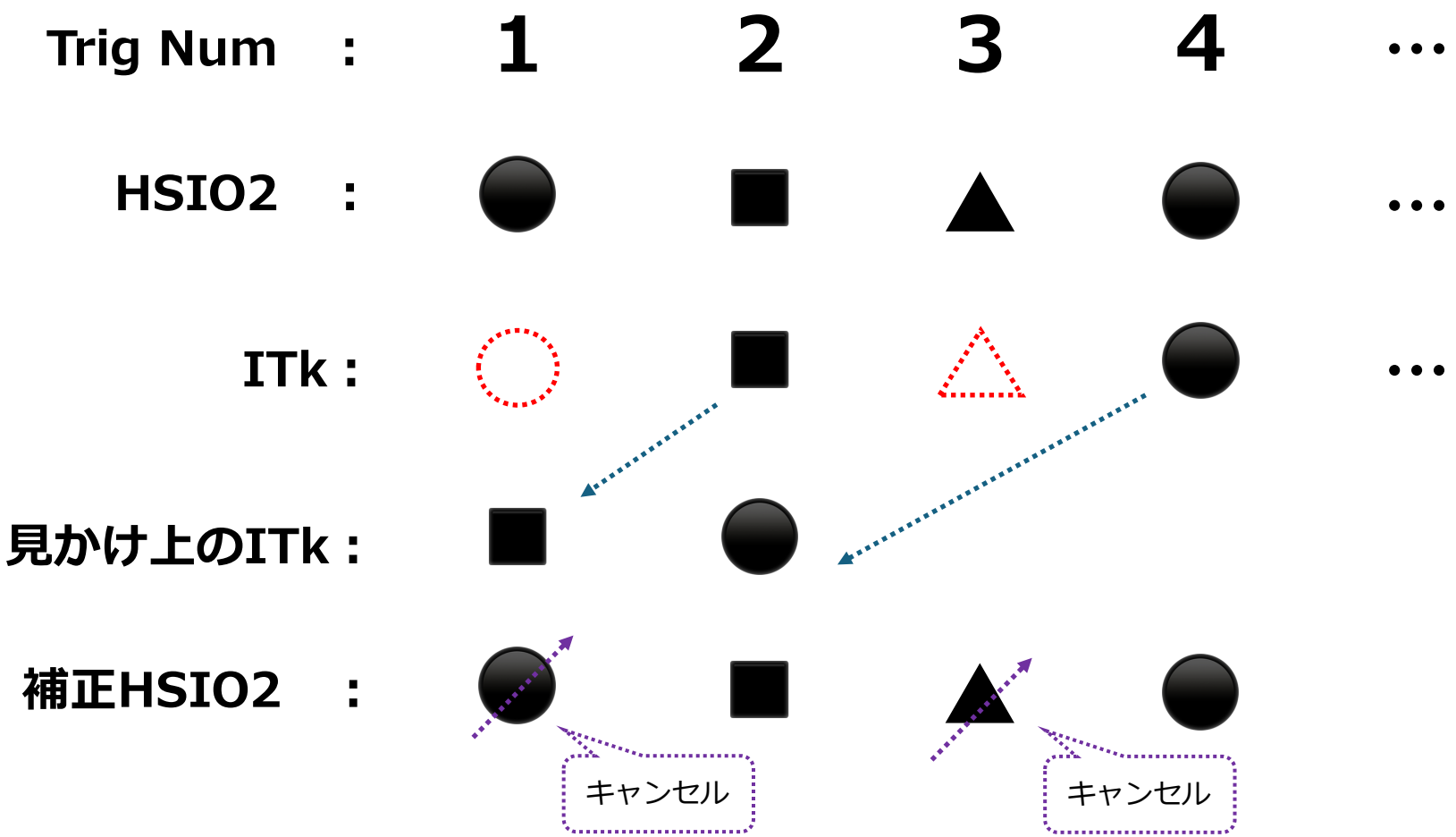
ITkがイベントを落とす問題

▶ ITkがランダムにイベントを落としてしまうため、HSIO2のイベントをskipしないといけない。



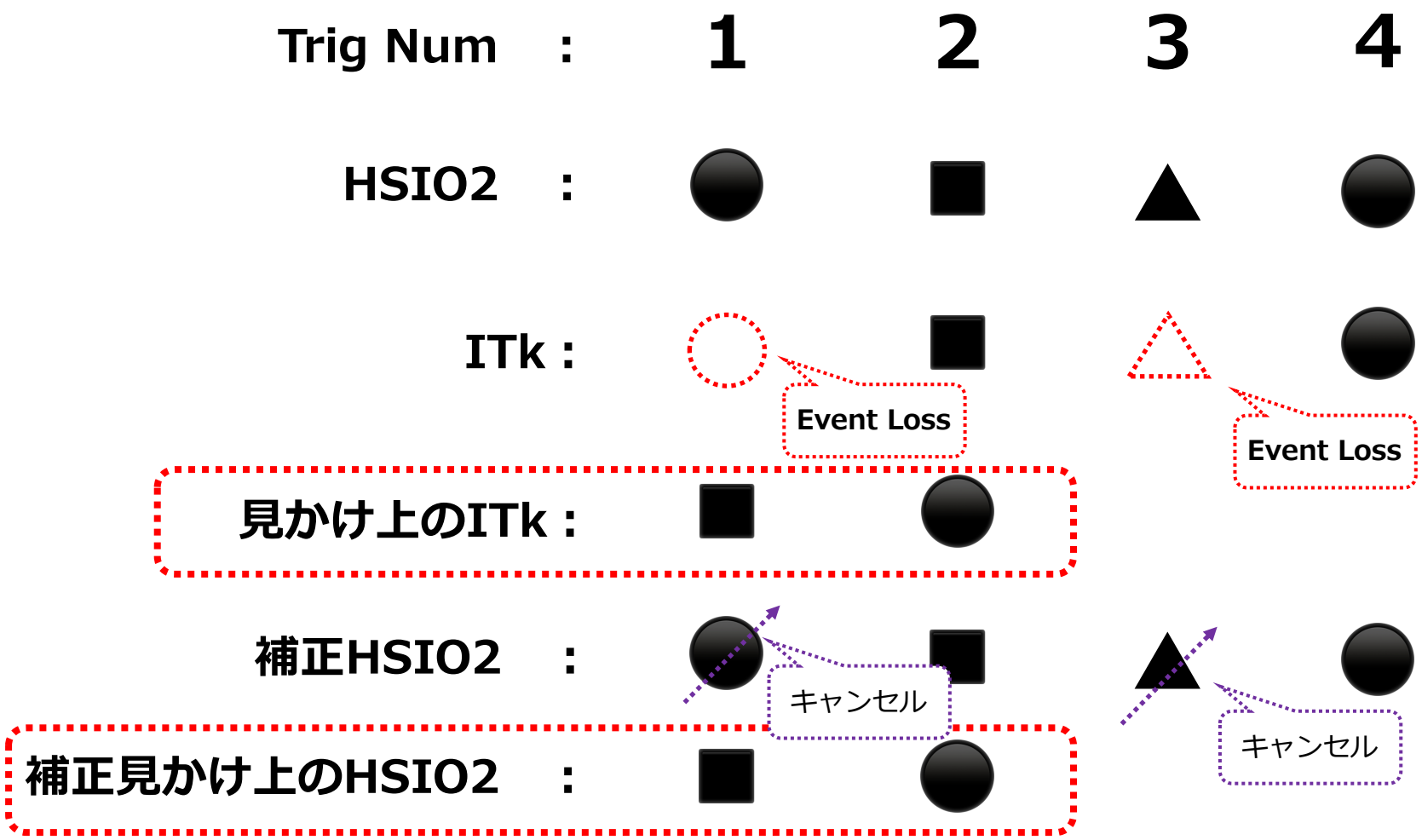
ITkがイベントを落とす問題

▶ ITkがランダムにイベントを落としてしまうため、HSIO2のイベントをskipしないといけない。



ITkがイベントを落とす問題

▶ ITkがランダムにイベントを落としてしまうため、HSIO2のイベントをskipしないといけない。

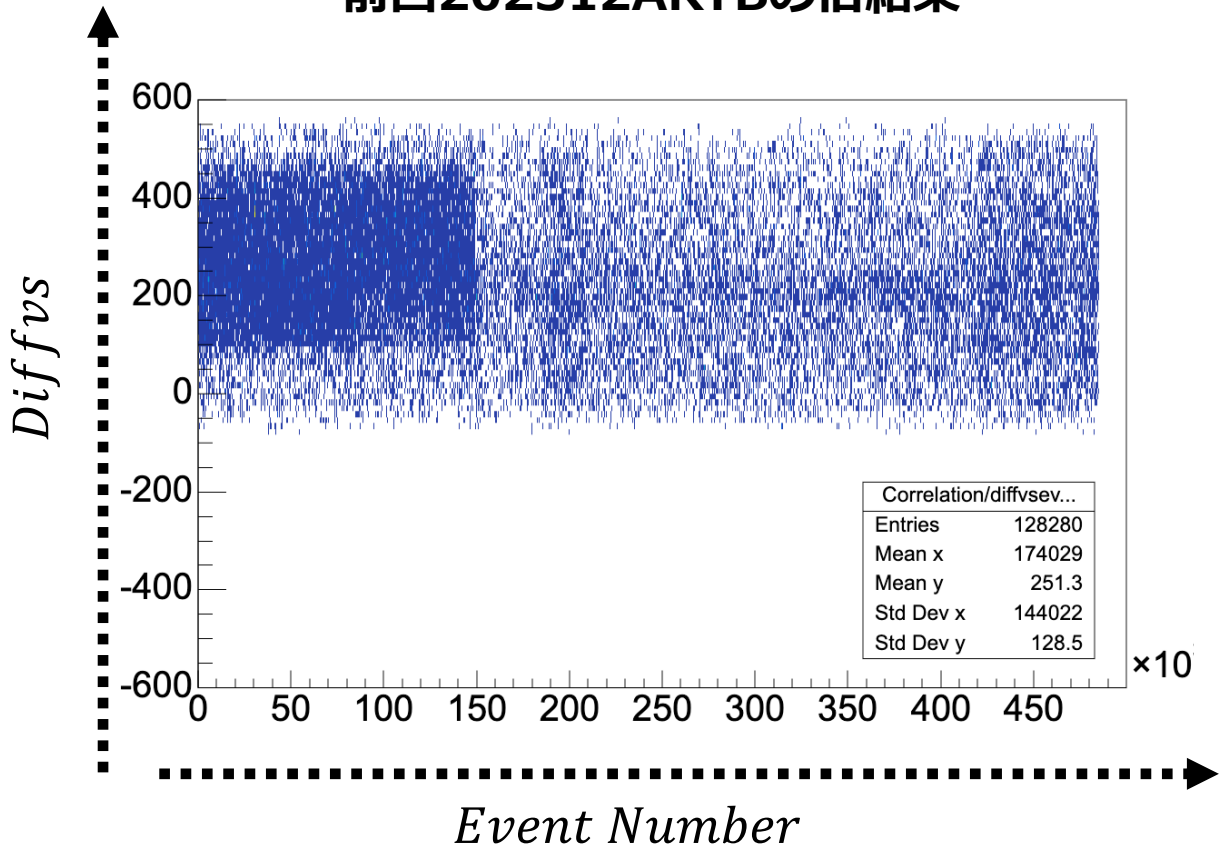


前回より同期取るのがむずい

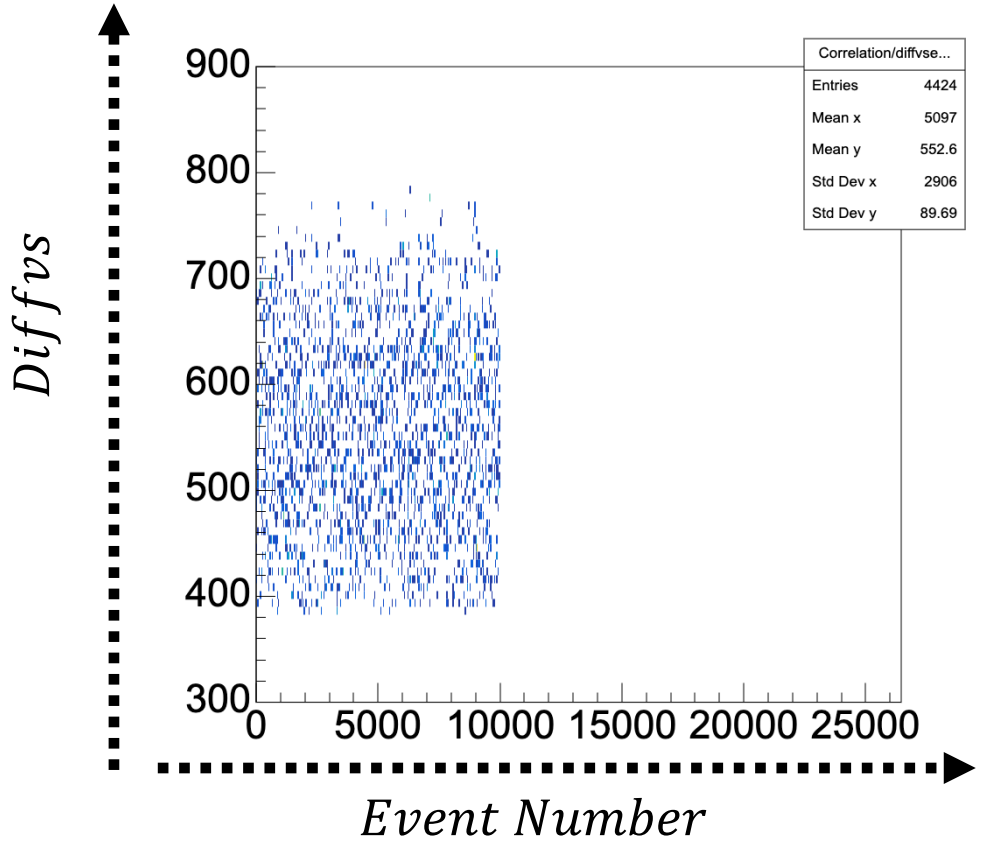
▶ Diffvsを見て同期の復活を試みる。

Diffvs … LocalHit位置の差分。同期が取れている間は一定の値を取り続けるはず。同期が切れると発散する。

前回202312ARTBの旧結果

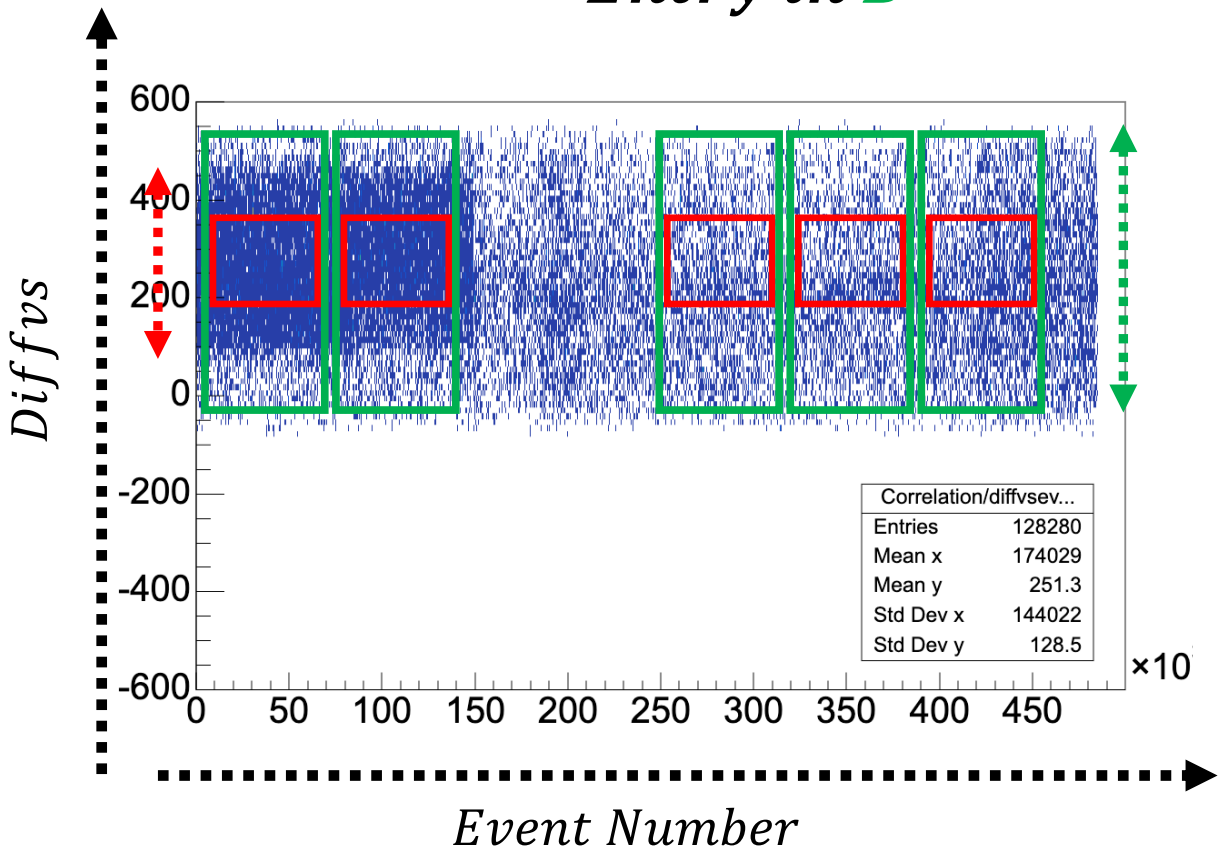


今回KEK132X(ly2)とITktelY(ly3)のDiffvs



同期復活のアルゴリズム考案②

$$DiffRate = \frac{Entry\ in\ A}{Entry\ in\ B}$$



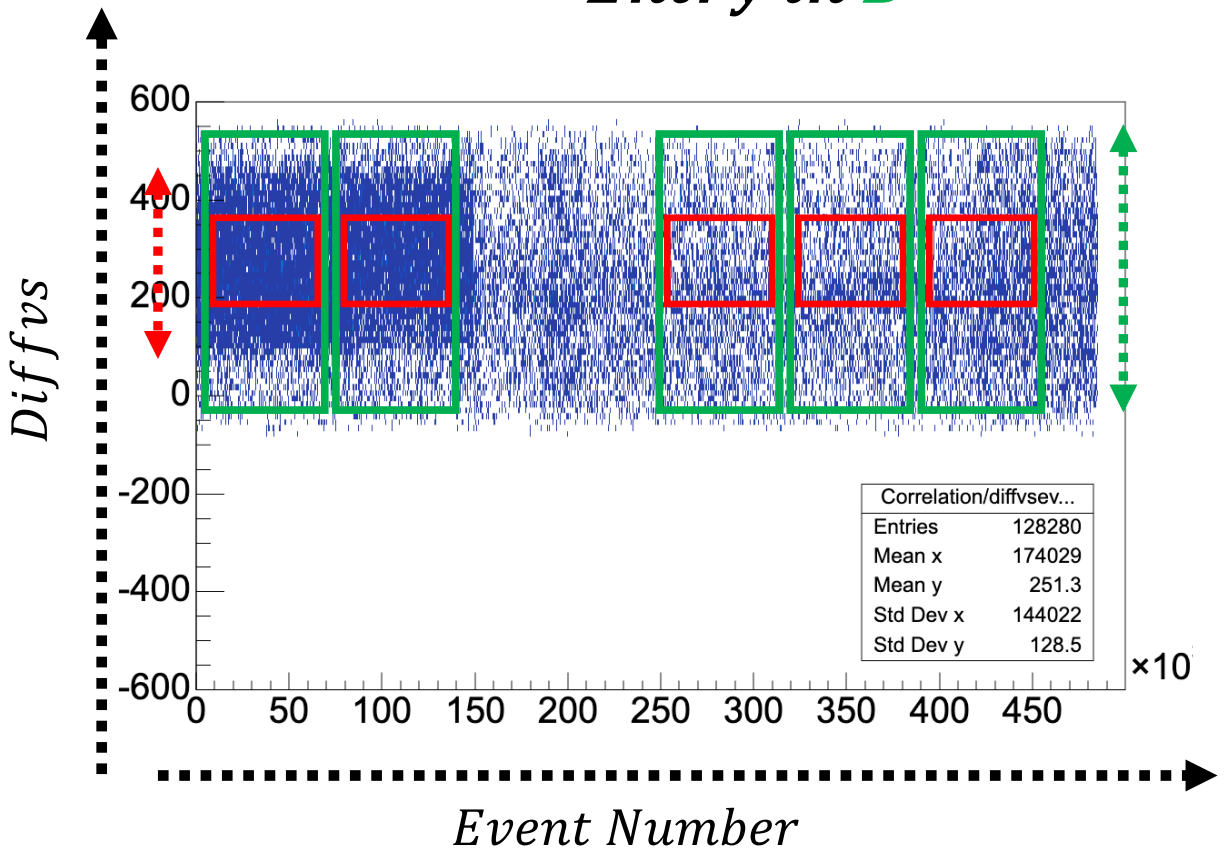
左図の例だと…

- 同期が取れている時は… 高めの値をとるはず。
- 同期が切れている時は… 低めの値に収束するはず。

X：skip数　Y：DiffRate　Z：entryを作成。

同期復活のアルゴリズム考案②

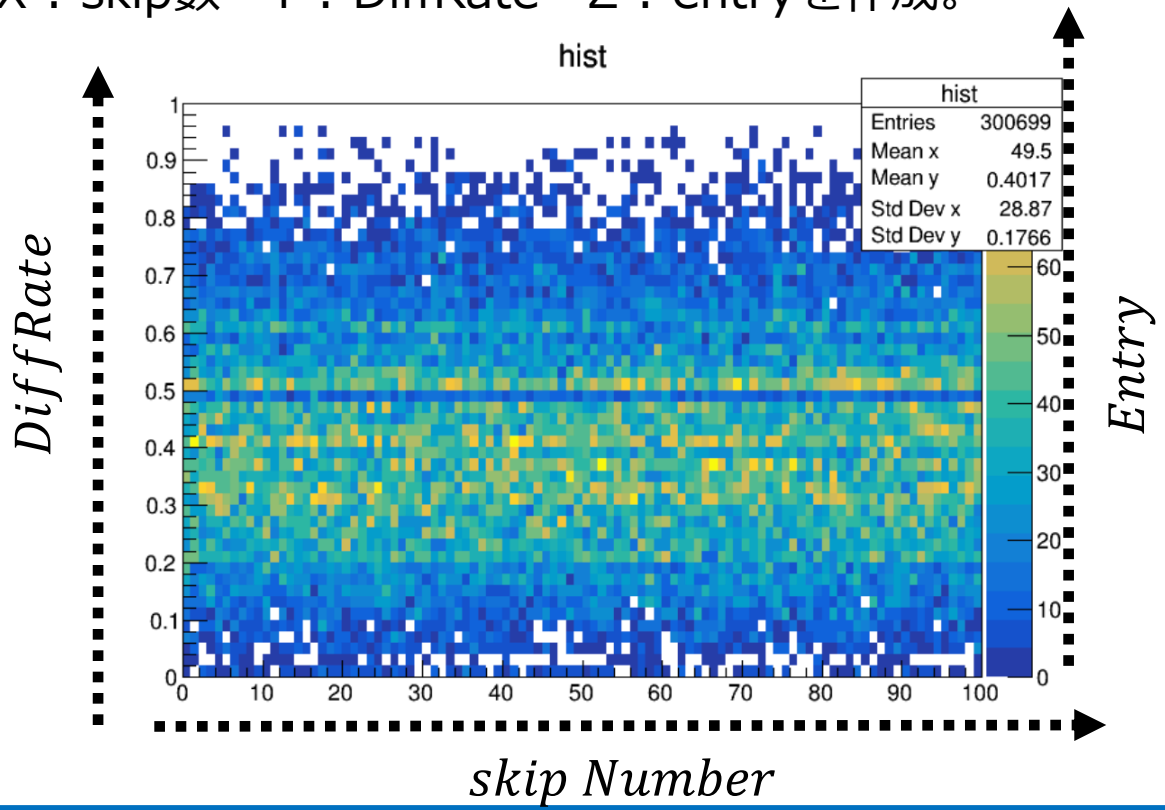
$$DiffRate = \frac{Entry\ in\ A}{Entry\ in\ B}$$



左図の例だと…

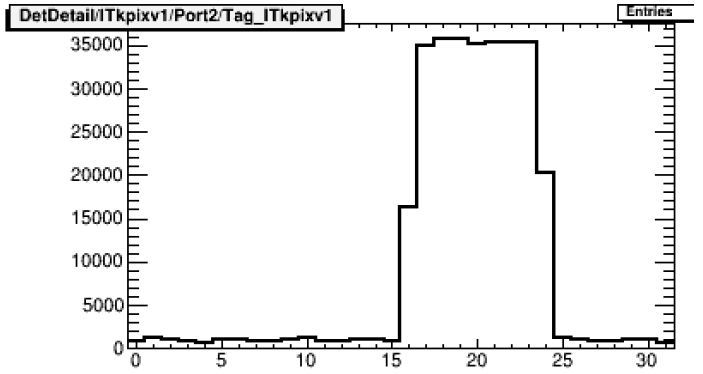
- 同期が取れている時は… 高めの値をとるはず。
- 同期が切れている時は… 低めの値に収束するはず。

X：skip数　Y：DiffRate　Z：entryを作成。



まとめ & 考え得る要因

▶ ITkがランダムにイベントを落とすため、同期が取れなくて困っている。



▶ 考え得る要因を羅列。

・yarrをreadyにしてからトリガー出したか。→ 間違いない。大丈夫。

・latencyは適切で合ったか。→ online Monitorで確認していた。

・トップアップの影響はあり得るか..? → veto刺さってた??

・ARTB202312と比べてビームレートが高かった → 1.7kHzから3kHzそんな柔じゃない?

・頻繁にイベントロスしている可能性が高いと考えている。

▶ Trig Tag FWを実装すべきだった。

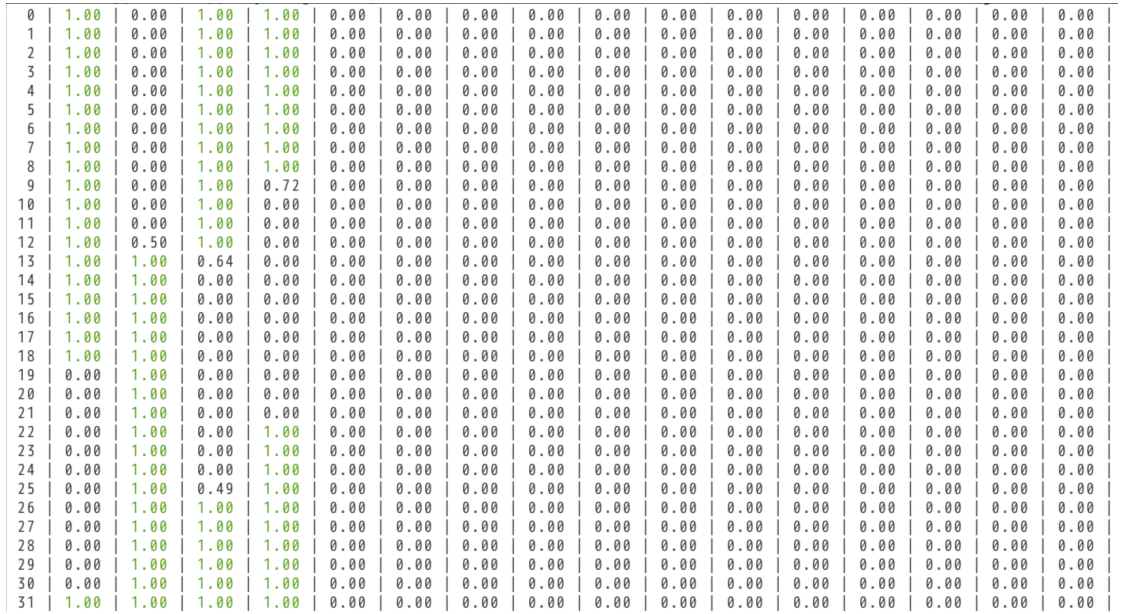
・v1.1だけでも別のFWを試すべきだった(時間が足りなかった。言い訳。)

▶ とりあえず、同期取れるように試行錯誤してみる(To Do)。

ITk telescope Debug

chip	rx	chipID	serEnLane	DataMergeOut	config
1	1	14→12	15	0/0/0/0	○
2	0	13	15	0/0/0/0	(○)
3	3	12→14	15	0/0/0/0	(○)
4	2	15	15	0/0/0/0	×

▶ Eye Diagram

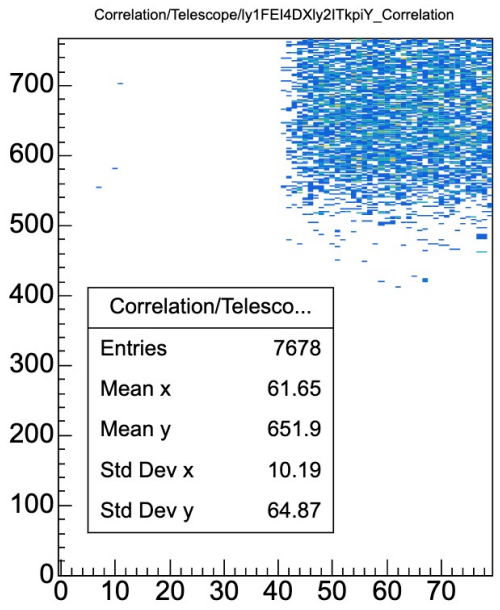


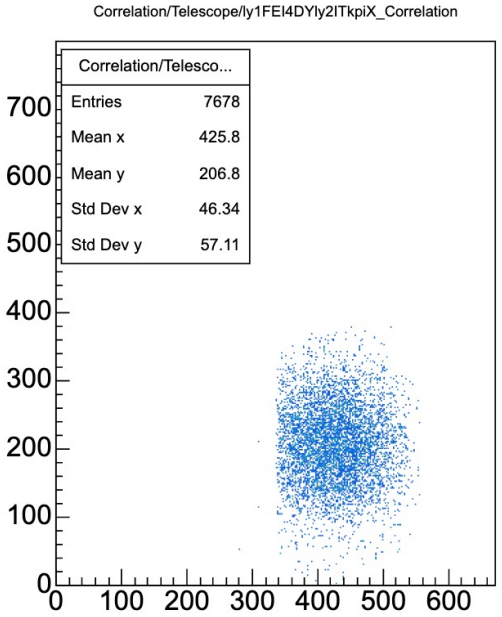
▶ Summary

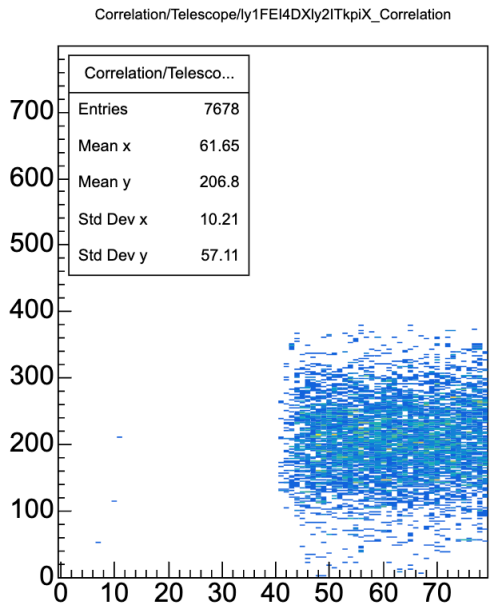
- eyeが初めて4レーン開通した。
- wireのchipIDがswapしていたのが問題だった。
- chip2, 3：digital scan回るが、time outで最後まで走り切れない。(これはchipの問題??)
- chip4はconfigが通らない。
  (LBNLと同じ。だが、eyeは通っている。)

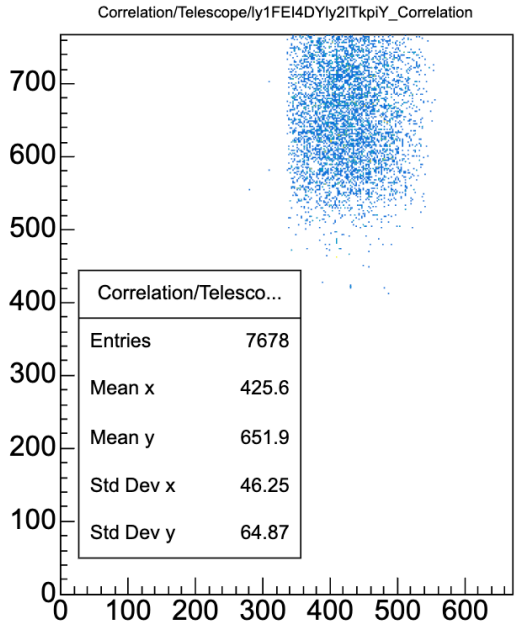
→ ヘルシーなモジュールを用意したい。

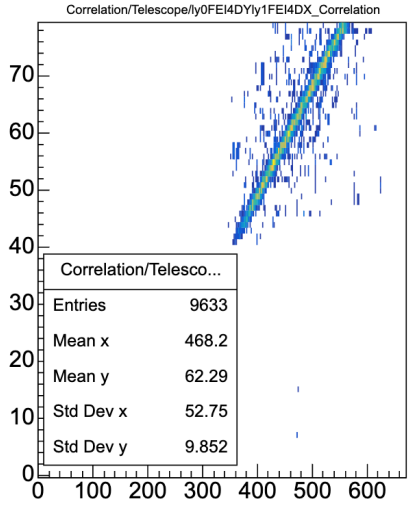
Correlation

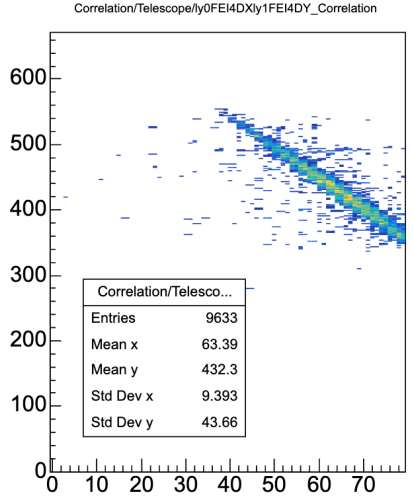


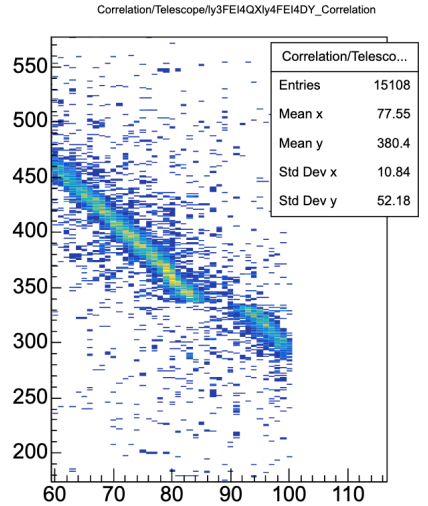


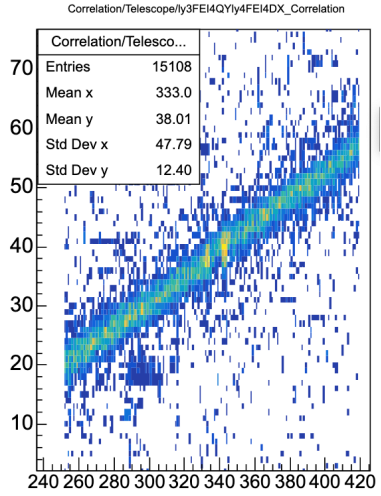




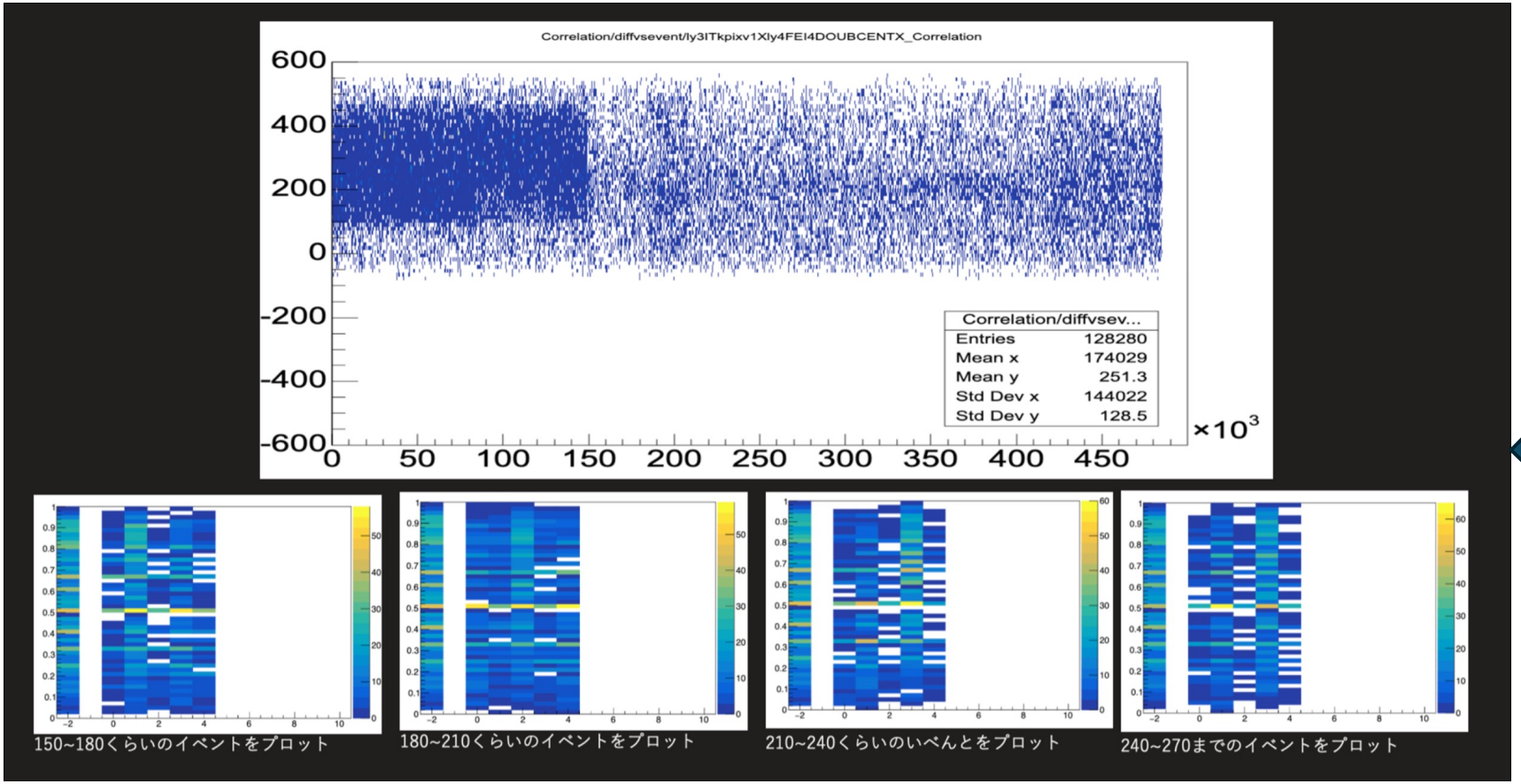




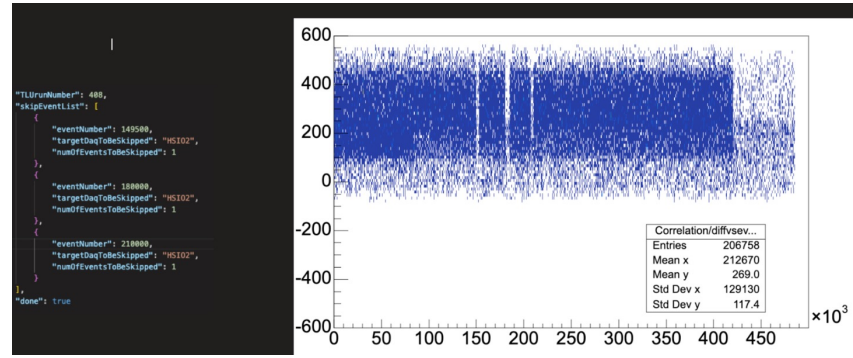




ARTB202312の同期は復活できる



同期復活



アルゴリズム②を使って去年のITk同期復活できることはわかった。

Raw Data Converter debug

↓BinaryからRaw Dataを読んでいる箇所

RawDataConverter/YarrPCI/src/ITkpixv1converter.cxx

```
FrontEndEvent *event = new FrontEndEvent();
event->fromFileBinary(file);
eventlist.push_back(event);
```

Binaryから読んできたデータには

- tag
- l1id
- bcid
- t_hits

の４つが含まれている。

▶こいつらをcoutした。

fromFileBinaryの中身

RawDataConverter/YarrPCI/src/EventData.cxx

```
void FrontEndEvent::fromFileBinary(std::fstream &handle) {
    uint16_t t_hits = 0;
    handle.read((char*)&tag, sizeof(uint32_t));
    handle.read((char*)&l1id, sizeof(uint16_t));
    handle.read((char*)&bcid, sizeof(uint16_t));
    handle.read((char*)&t_hits, sizeof(uint16_t));
    for (unsigned ii = 0; ii < t_hits; ii++) {
        FrontEndHit hit;
        handle.read((char*)&hit, sizeof(FrontEndHit));
        this->addHit(hit);
    } // ii
}
```

Raw Data Converter debug

今回のデータ：ITk telescope TLU=370

```
 #### Event 500 ####
nHits in this event = 1
 (link, ly ,name ,RJ,col,row,posX,posY,posZ,ntot,ntot2,lv1,type)=(0, 2, ITkTelescope, 1, 267, 671, 13.325, 33.725, 0, 7, 0, 0, ITkpixv1)
L1 count: 1 at event 500 L1ID(666) BCID(666) TAG(0) HITS(2)
L1 count: 2 at event 500 L1ID(666) BCID(666) TAG(1) HITS(0)
L1 count: 3 at event 500 L1ID(666) BCID(666) TAG(2) HITS(0)
L1 count: 4 at event 500 L1ID(666) BCID(666) TAG(3) HITS(0)
L1 count: 5 at event 500 L1ID(666) BCID(666) TAG(4) HITS(0)
L1 count: 6 at event 500 L1ID(666) BCID(666) TAG(5) HITS(0)
L1 count: 7 at event 500 L1ID(666) BCID(666) TAG(6) HITS(0)
L1 count: 8 at event 500 L1ID(666) BCID(666) TAG(7) HITS(0)
L1 count: 9 at event 500 L1ID(666) BCID(666) TAG(8) HITS(0)
L1 count: 10 at event 500 L1ID(666) BCID(666) TAG(9) HITS(0)
L1 count: 11 at event 500 L1ID(666) BCID(666) TAG(10) HITS(0)
L1 count: 12 at event 500 L1ID(666) BCID(666) TAG(11) HITS(0)
L1 count: 13 at event 500 L1ID(666) BCID(666) TAG(12) HITS(0)
L1 count: 14 at event 500 L1ID(666) BCID(666) TAG(13) HITS(0)
L1 count: 15 at event 500 L1ID(666) BCID(666) TAG(14) HITS(0)
L1 count: 16 at event 500 L1ID(666) BCID(666) TAG(15) HITS(0)
L1 count: 17 at event 500 L1ID(666) BCID(666) TAG(16) HITS(2)
L1 count: 18 at event 500 L1ID(666) BCID(666) TAG(17) HITS(0)
L1 count: 19 at event 500 L1ID(666) BCID(666) TAG(18) HITS(0)
L1 count: 20 at event 500 L1ID(666) BCID(666) TAG(19) HITS(0)
L1 count: 21 at event 500 L1ID(666) BCID(666) TAG(20) HITS(0)
L1 count: 22 at event 500 L1ID(666) BCID(666) TAG(21) HITS(0)
L1 count: 23 at event 500 L1ID(666) BCID(666) TAG(22) HITS(0)
L1 count: 24 at event 500 L1ID(666) BCID(666) TAG(23) HITS(0)
L1 count: 25 at event 500 L1ID(666) BCID(666) TAG(24) HITS(0)
L1 count: 26 at event 500 L1ID(666) BCID(666) TAG(25) HITS(0)
L1 count: 27 at event 500 L1ID(666) BCID(666) TAG(26) HITS(0)
L1 count: 28 at event 500 L1ID(666) BCID(666) TAG(27) HITS(0)
L1 count: 29 at event 500 L1ID(666) BCID(666) TAG(28) HITS(0)
L1 count: 30 at event 500 L1ID(666) BCID(666) TAG(29) HITS(0)
L1 count: 31 at event 500 L1ID(666) BCID(666) TAG(30) HITS(0)
L1 count: 32 at event 500 L1ID(666) BCID(666) TAG(31) HITS(0)
```

前回のデータ：KEKPreprodQ03 TLU=408

```
 #### Event 332 ####
nHits in this event = 2
 (link, ly ,name ,RJ,col,row,posX,posY,posZ,ntot,ntot2,lv1,type)=(2, 3, KEKPreprodQ03, 3, 640, 256, 32.175, 12.775, 0, 7, 0, 0, ITkpixv1)
 (link, ly ,name ,RJ,col,row,posX,posY,posZ,ntot,ntot2,lv1,type)=(2, 3, KEKPreprodQ03, 3, 640, 257, 32.175, 12.825, 0, 7, 0, 0, ITkpixv1)
L1 count: 1 at event 332 L1ID(666) BCID(666) TAG(0) HITS(4)
L1 count: 2 at event 332 L1ID(666) BCID(666) TAG(1) HITS(0)
L1 count: 3 at event 332 L1ID(666) BCID(666) TAG(2) HITS(0)
L1 count: 4 at event 332 L1ID(666) BCID(666) TAG(3) HITS(0)
L1 count: 5 at event 332 L1ID(666) BCID(666) TAG(4) HITS(0)
L1 count: 6 at event 332 L1ID(666) BCID(666) TAG(5) HITS(0)
L1 count: 7 at event 332 L1ID(666) BCID(666) TAG(6) HITS(0)
L1 count: 8 at event 332 L1ID(666) BCID(666) TAG(7) HITS(0)
L1 count: 9 at event 332 L1ID(666) BCID(666) TAG(8) HITS(0)
L1 count: 10 at event 332 L1ID(666) BCID(666) TAG(9) HITS(0)
L1 count: 11 at event 332 L1ID(666) BCID(666) TAG(10) HITS(0)
L1 count: 12 at event 332 L1ID(666) BCID(666) TAG(11) HITS(0)
L1 count: 13 at event 332 L1ID(666) BCID(666) TAG(12) HITS(0)
L1 count: 14 at event 332 L1ID(666) BCID(666) TAG(13) HITS(2)
L1 count: 15 at event 332 L1ID(666) BCID(666) TAG(14) HITS(2)
L1 count: 16 at event 332 L1ID(666) BCID(666) TAG(15) HITS(0)
L1 count: 17 at event 332 L1ID(666) BCID(666) TAG(16) HITS(0)
L1 count: 18 at event 332 L1ID(666) BCID(666) TAG(17) HITS(0)
L1 count: 19 at event 332 L1ID(666) BCID(666) TAG(18) HITS(0)
L1 count: 20 at event 332 L1ID(666) BCID(666) TAG(19) HITS(0)
L1 count: 21 at event 332 L1ID(666) BCID(666) TAG(20) HITS(0)
L1 count: 22 at event 332 L1ID(666) BCID(666) TAG(21) HITS(0)
L1 count: 23 at event 332 L1ID(666) BCID(666) TAG(22) HITS(0)
L1 count: 24 at event 332 L1ID(666) BCID(666) TAG(23) HITS(0)
L1 count: 25 at event 332 L1ID(666) BCID(666) TAG(24) HITS(0)
L1 count: 26 at event 332 L1ID(666) BCID(666) TAG(25) HITS(0)
L1 count: 27 at event 332 L1ID(666) BCID(666) TAG(26) HITS(0)
L1 count: 28 at event 332 L1ID(666) BCID(666) TAG(27) HITS(0)
L1 count: 29 at event 332 L1ID(666) BCID(666) TAG(28) HITS(0)
L1 count: 30 at event 332 L1ID(666) BCID(666) TAG(29) HITS(0)
L1 count: 31 at event 332 L1ID(666) BCID(666) TAG(30) HITS(0)
L1 count: 32 at event 332 L1ID(666) BCID(666) TAG(31) HITS(0)
```

・どちらも同じ形式のデータが入っている様に見える。

▶ データフレームはv2になっても同じってこと？。

Raw Data Converter debug

これらのヘッダーがこれで正しいのか？

10. Data Output

The RD53C data output consists of tagged events, which enables the readout to automatically recover from transmission errors without any action from the DAQ. While tagged data would permit event building to be performed off chip if desired, RD53C builds events on-chip, such that a full event is output before sending any data for the next event. The characteristics of the physical data output ports are described in Sec. 10.1. The transmission protocol used is a subset of the Aurora 64b66b protocol [3], as detailed in Sec. 10.2. This provides industry standard frame alignment, DC balance and multi-lane serial transmission suitable for high speed data, but does not define the data content. The Aurora protocol can be thought of as a "wrapper" placed around the RD53C data. Before the Aurora wrapper, the hit data are packaged in *streams*, not fixed frames. A stream is a self-contained, variable length data container beginning with a tag (8 bits) and followed by a mix of hit data and possibly other tags (called internal tags, which are 11 bits). Streams and their contents are described in Sec. 10.3 to 10.9. This variable length format is approximately 25% more efficient (fewer bits per hit) than the fixed frame format previously used in RD53A.

RD53C manual

これらのヘッダーのbit指定を適切にすれば解消する？

fromFileBinaryの中身

RawDataConverter/YarrPCI/src/EventData.cxx

```
void FrontEndEvent::fromFileBinary(std::fstream &handle) {
    uint16_t t_hits = 0;
    handle.read((char*)&tag, sizeof(uint32_t));
    handle.read((char*)&l1id, sizeof(uint16_t));
    handle.read((char*)&bcid, sizeof(uint16_t));
    handle.read((char*)&t_hits, sizeof(uint16_t));
    for (unsigned ii = 0; ii < t_hits; ii++) {
        FrontEndHit hit;
        handle.read((char*)&hit, sizeof(FrontEndHit));
        this->addHit(hit);
    } // ii
}
```

- tag … 32bitの整数をbinaryから取得
- l1id … 16bitの整数をbinaryから取得
- bcid … 16bitの整数をbinaryから取得
- t_hits … 16bitの整数をbinaryから取得