

INTT software

Takashi Hachiya
Nara Women's University

Before Introduction

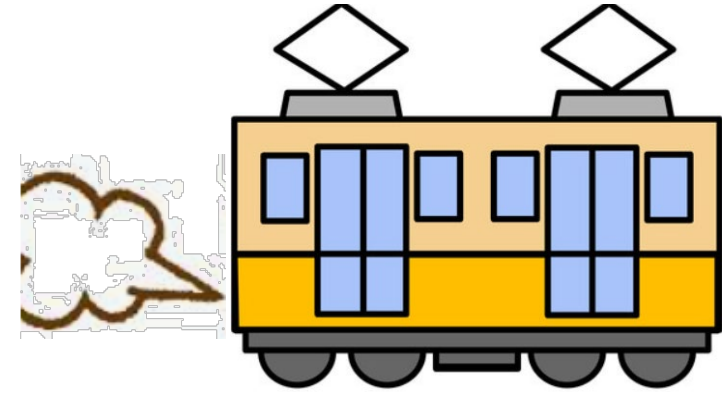


Hello, Takashi!
Can we discuss about INTT workshop session a little bit?

午後 4:04

既読

Sure! I



One thing we would like to get is, review session for **INTT related software**.

The point of this session is, we don't have to learn how to use during the session. We want to get some info, **what kind of software (analysis module) we have in sPHENIX github**, and what is purpose that module.

Oh, can we have our InttMapping review as well?

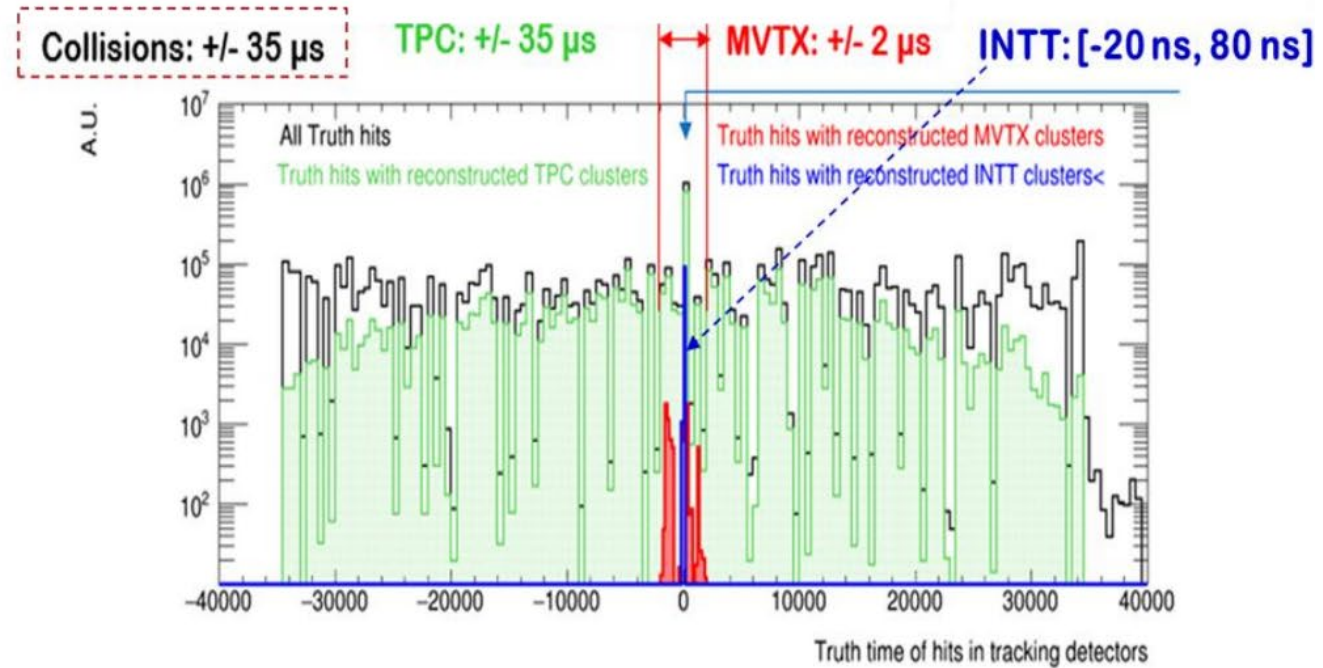
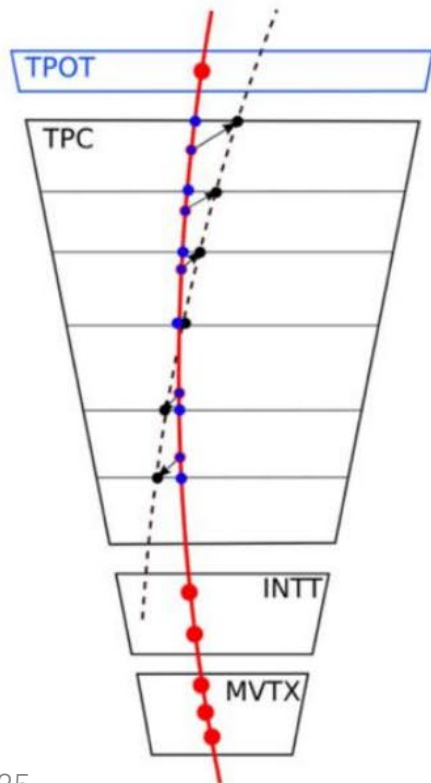
2024/11/25



Purposes of INTT

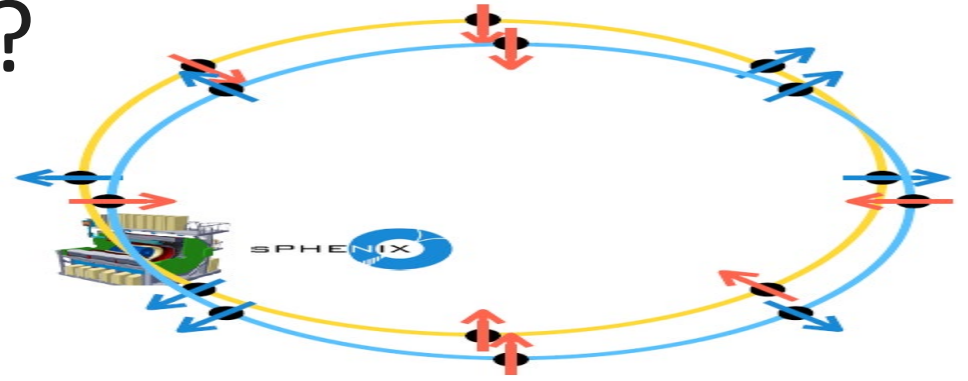
- Charged particle tracking
 - Together with MVTX, TPC, TPOT
 - More? Z-vertex, dN/deta, Reaction plane, Centrality for streaming data
- Beam crossing Identification
Treso ~ 100ns (INTT), ~5us(MVTX), ~80us (TPC)

Tracking Points and Timing

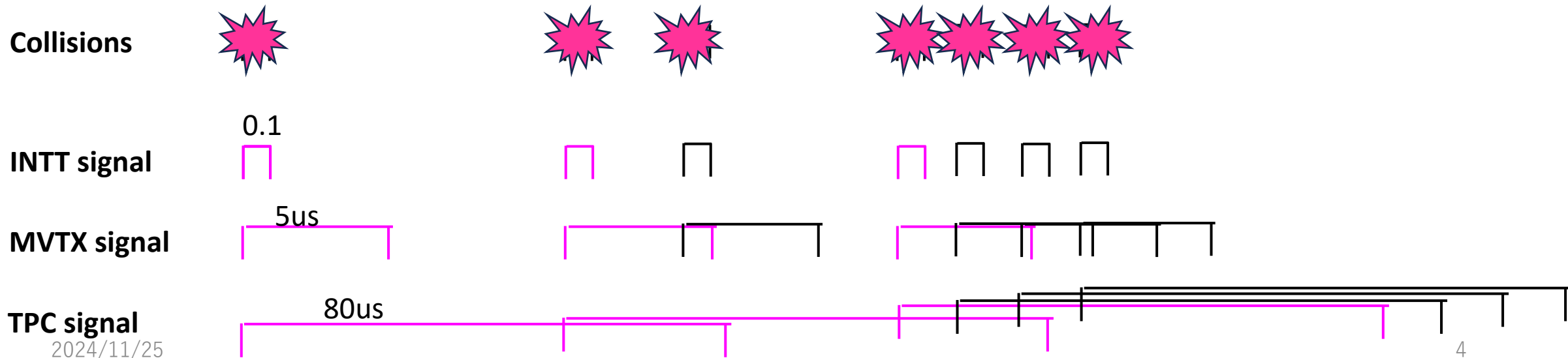


Why is beam crossing important?

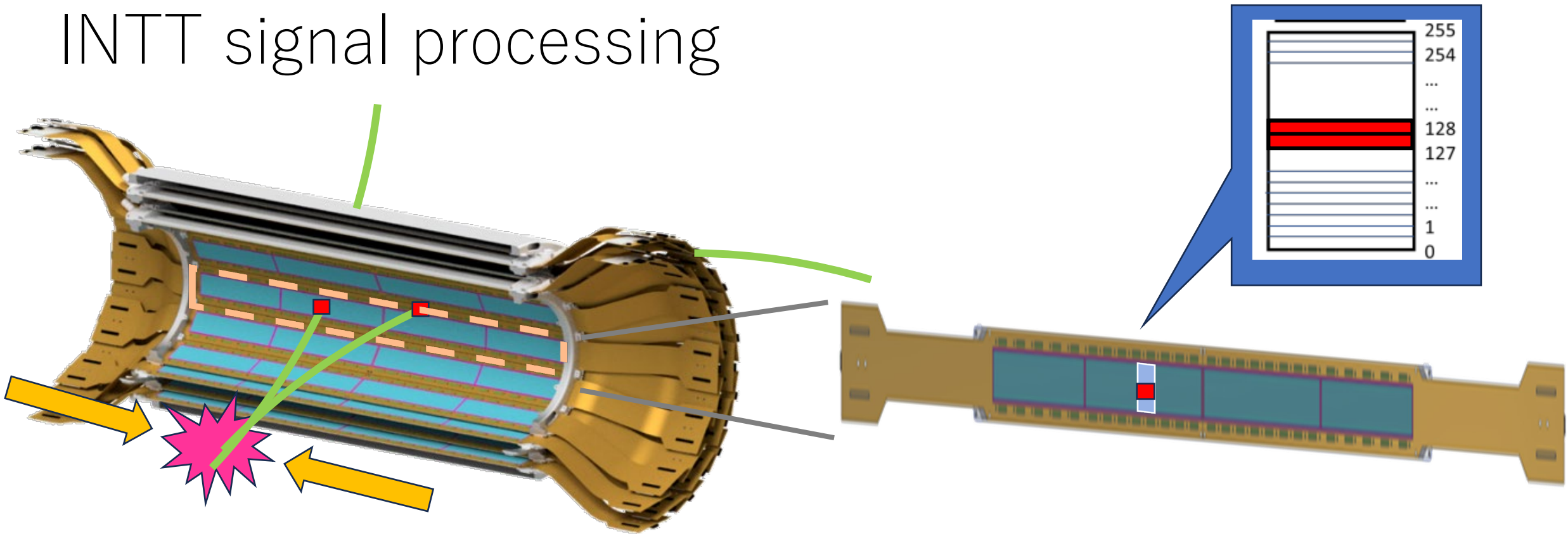
- The beam has a bunch structure
 - Bunch length is 106ns
 - Nbunchs is 120
- Treso $\sim 0.1\mu\text{s}$ (100ns) (INTT), $\sim 5\mu\text{s}$ (MVTX), $\sim 80\mu\text{s}$ (TPC)
 - Signals from different beam crossing mixed up
 - Physics we aim to measure is a quantity per **COLLISION**



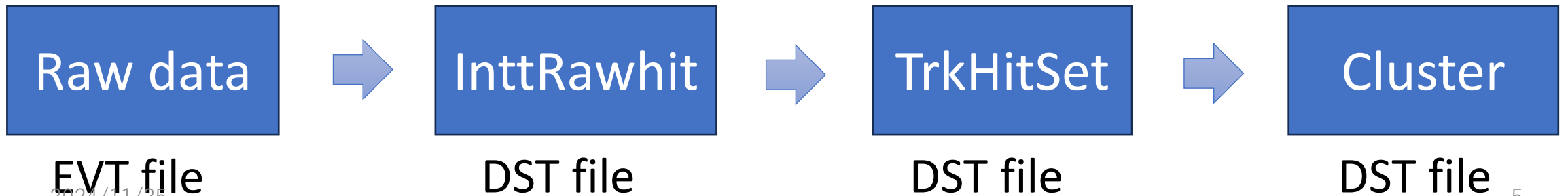
Timing Diagram



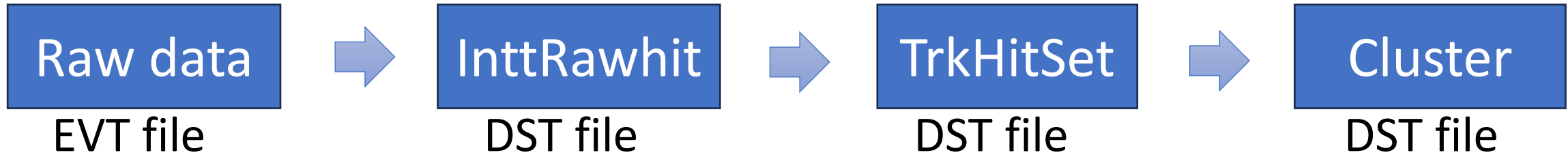
INTT signal processing



- Raw data is converted to 3D hit position step by step

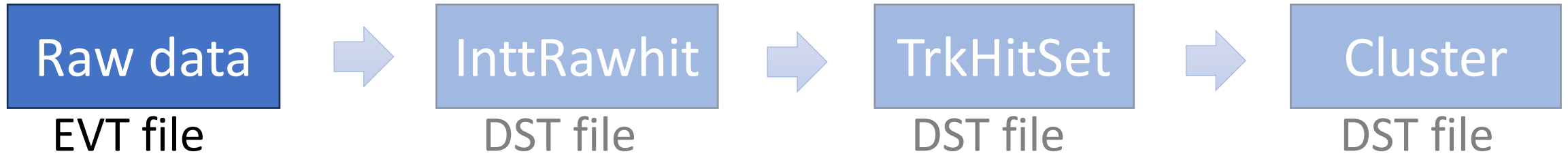


INTT data (format)



- Raw data: Binary data from INTT detector (FELIX)
 - a 32bit binary contains hit channel, chip, ladder, FELIX, ADC are stored in
 - Each FELIX server produce own EVT file -> INTT has 8 evt files
- InttRawhit: DST style data (C++ class)
 - Hit channel, chip, ladder, FELIX, adc are stored in the variables
 - Channel, chip, ladder, FELIX ID are defined by INTT team
- TrkrHitSet: DST style data but sPHENIX format for global tracking
 - Similar w/ InttRawhit
 - Channel, chip, ladder, FELIX ID are defined by sPHENIX tracking
- Cluster : DST style data
 - adjacent hits grouped to single cluster

INTT data (format)

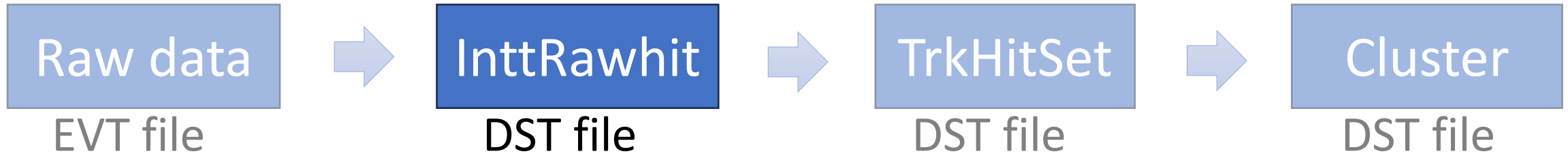


- Raw data: Binary data from INTT detector (FELIX)

- a 32bit binary contains hit channel, chip, ladder, FELIX, ADC are stored in
- Each FELIX server produce own EVT file -> INTT has 8 evt files

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1		Header				fee		len		Header																		header=0xF000CAFO						
2	1	Hit-Header				BCO				Hit-Header														Hit-header										
3	2	BCO																																
4	3	adc	chip_id		chan_id																				AMPL	full ROC	full FPX	FPHX_BCO		hit				
5	4	adc	chip_id		chan_id																				AMPL	full ROC	full FPX	FPHX_BCO		hit				
6	5	adc	chip_id		chan_id																				AMPL	full ROC	full FPX	FPHX_BCO		hit				
7	6	adc	chip_id		chan_id																				AMPL	full ROC	full FPX	FPHX_BCO		hit				
8	7	adc	chip_id		chan_id																				AMPL	full ROC	full FPX	FPHX_BCO		hit				

INTT raw data (format)



[InttRawhit in github](#)

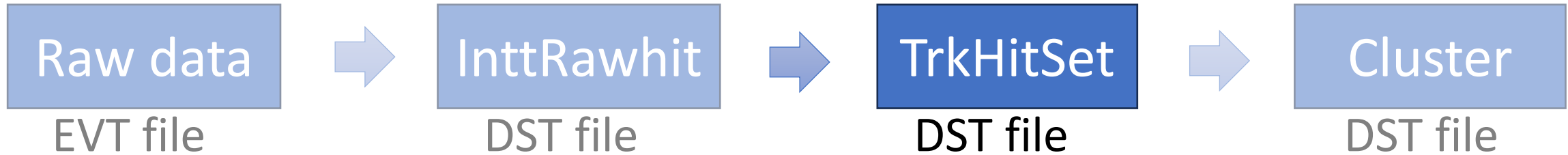
protected:

```
BCOFULL      uint64_t bco = std::numeric_limits<uint64_t>::max();
FELIX_ID     int32_t packetid = std::numeric_limits<int32_t>::max();
              uint32_t word = std::numeric_limits<uint32_t>::max();
Ladder       uint16_t fee = std::numeric_limits<uint16_t>::max();
Channel      uint16_t channel_id = std::numeric_limits<uint16_t>::max();
Chip         uint16_t chip_id = std::numeric_limits<uint16_t>::max();
Adc          uint16_t adc = std::numeric_limits<uint16_t>::max();
BCO          uint16_t FPHX_BCO = std::numeric_limits<uint16_t>::max();
              uint16_t full_FPHX = std::numeric_limits<uint16_t>::max();
              uint16_t full_ROC = std::numeric_limits<uint16_t>::max();
AMPL for calib uint16_t amplitude = std::numeric_limits<uint16_t>::max();
Added in v2   uint32_t event_counter = std::numeric_limits<uint32_t>::max();
```

- InttRawhit: DST style data (C++ class)

- Hit channel, chip, ladder, FELIX, adc are stored in the variables
- Channel, chip, ladder, FELIX ID are defined by INTT team

INTT TrkrHitSet(format)



[TrkrDef.h](#), [InttDef.h](#), [TrkrHitv2.h](#)

TrkrHitSet : double array [hitsetkey][hitkey] = ADC

hitsetkey :

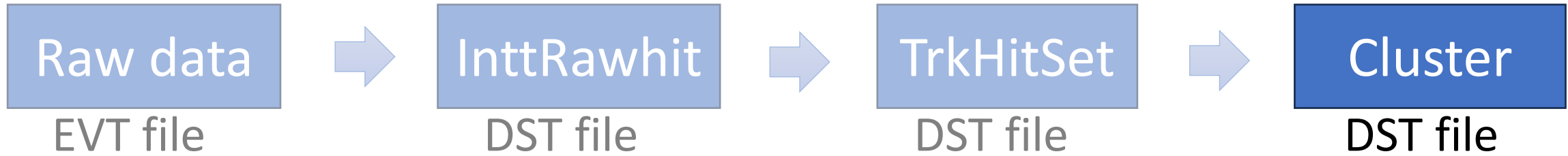
31-24	23-16	15-14	13-10	9-0
TrackerID	Layer	Z-ID	Ladder phi	BCO (time bucket)

hitkey : chip[] +channel[0-255 7:0]+BCO[]

Tracker ID and Layer are commonly used for
INTT, MVTX, TPC, TPOT

- TrkrHitSet: DST style data but sPHENIX format for global tracking
 - Similar w/ InttRawhit
 - Channel, chip, ladder, FELIX ID are defined by sPHENIX tracking

INTT Cluster(format)



[TrkrClusterv5 in github](#)

Cluster variables

TrkrCluster : array [cluskey] of cluster

cluskey : hitsetkey[63:32] + clusid[31:0]

hitsetkey :

31-24	23-16	15-14	13-10	9-0
Tracker ID	Layer	Z-ID	Ladder phi	BCO (time bucket)

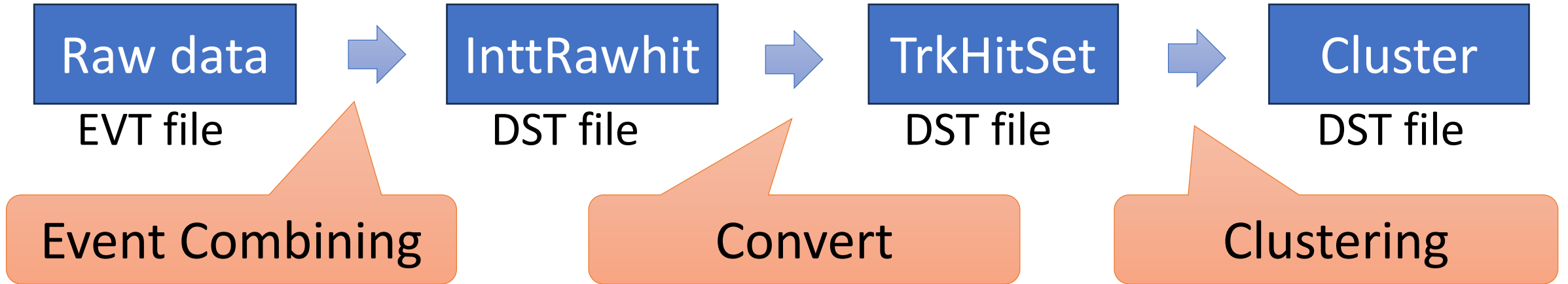
Clusid: i_cluster in the sensor

[TrkrDef.h in github](#)

2024/11/25

```
protected:  
float m_local[2]{}; //< 2D local  
TrkrDefs::subsurfkey m_subsurfkey; //< unique id  
float m_phierr;  
float m_zerr;  
unsigned short int m_adc; //< cluster sum ad  
unsigned short int m_maxadc; //< cluster sum ad  
char m_phisize; // 8bit  
char m_zsize; // 8bit  
char m_overlap; // 8bit  
char m_edge; // 8bit - cumul 2*
```

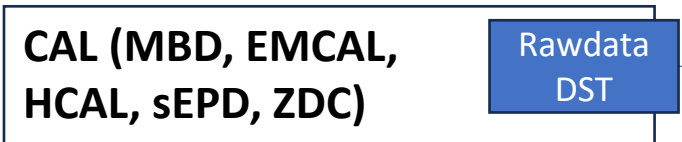
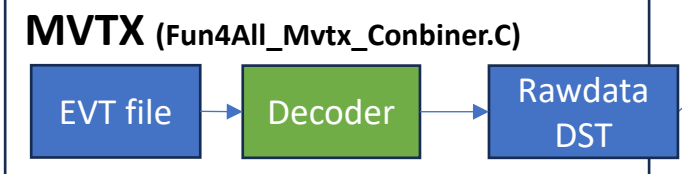
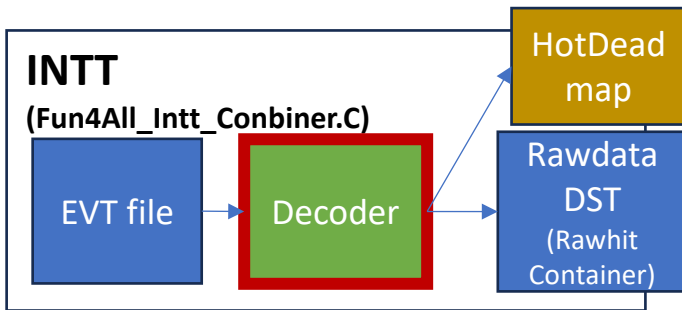
INTT data (format)



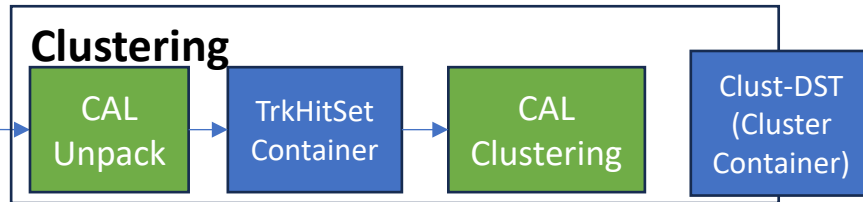
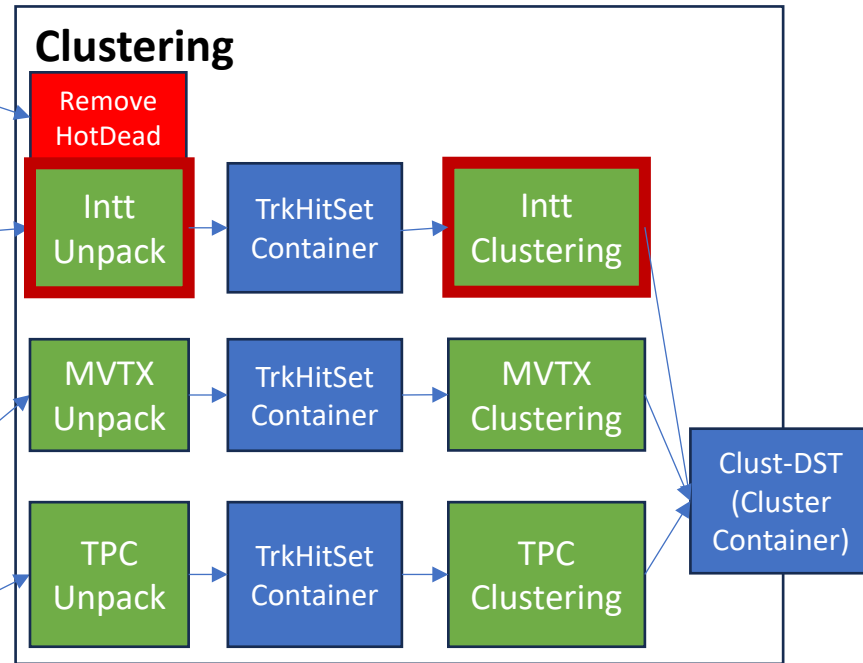
- Event Combining: SingleInttPoolInput.h
 - Convert rawdata to InttRawhit
 - Synchronize 8 FELIXs w/ BCOFULL
 - Synchronize INTT with GL1 and other detectors w/ BCOFULL
- Convert
 - Convert InttRawhit to TrkrHitSet
 - Hot/dead, BCO cut calibration applied to remove “BAD” hits
 - BCO -> Time (Timebucket)
- Clustering
 - Clustering hits sensor by sensor and time by time

Workflow of DST production: Multi-stage

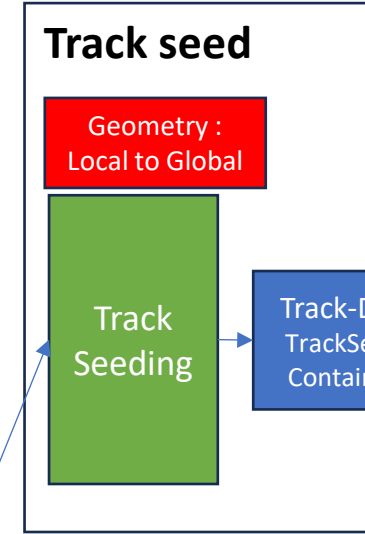
job -1: Offline Evt building



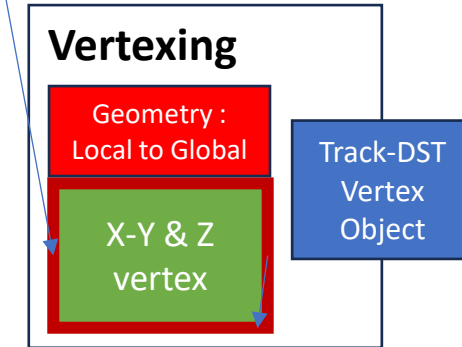
job0 Clustering



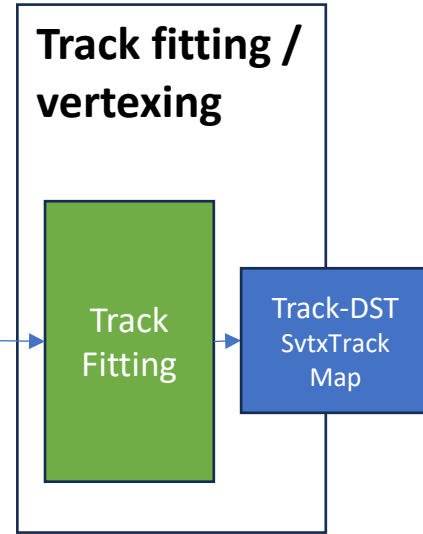
job-A Track seed



INTT-Vertexing

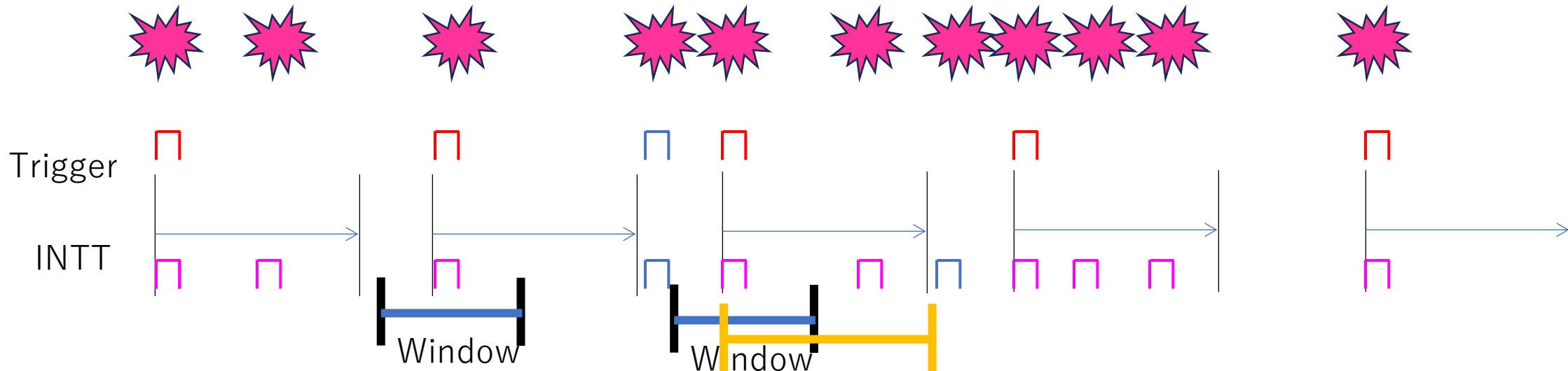


job-C Track fit/vtx



Event Combining

- Event combining is to combine data from other detectors using BCO(FULL)
 - The common method is applied for both (extended) trigger mode and streaming mode
- Method
 - Start with Reference BCOFULL (usually GL1)
 - Look for the hits having BCOFULL within “window” relative to Ref_BCOFULL



[Link to macro](#)

```
intt_sngl->SetNegativeBco(120-23);
```

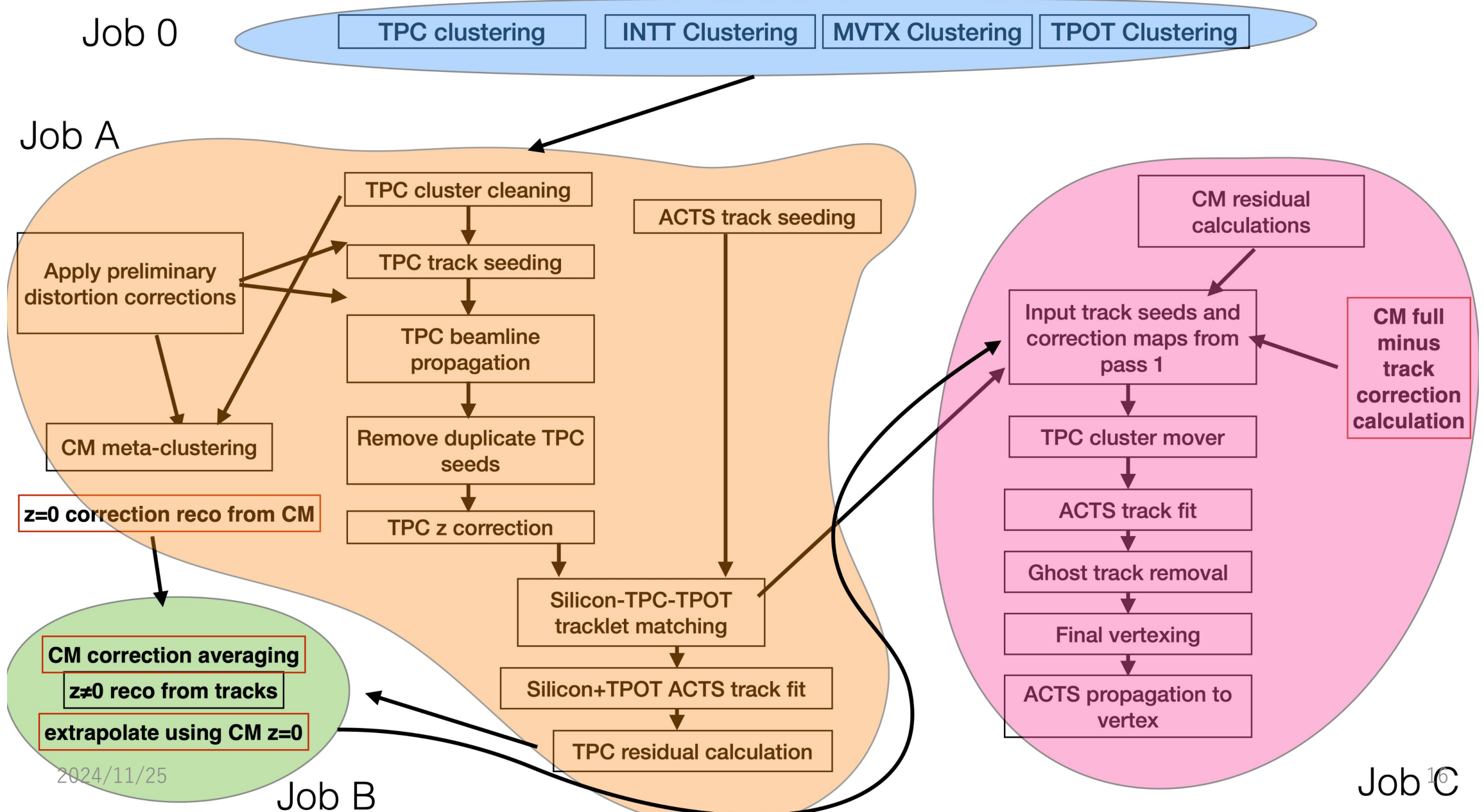
```
intt_sngl->SetBcoRange(500);
```

Convert

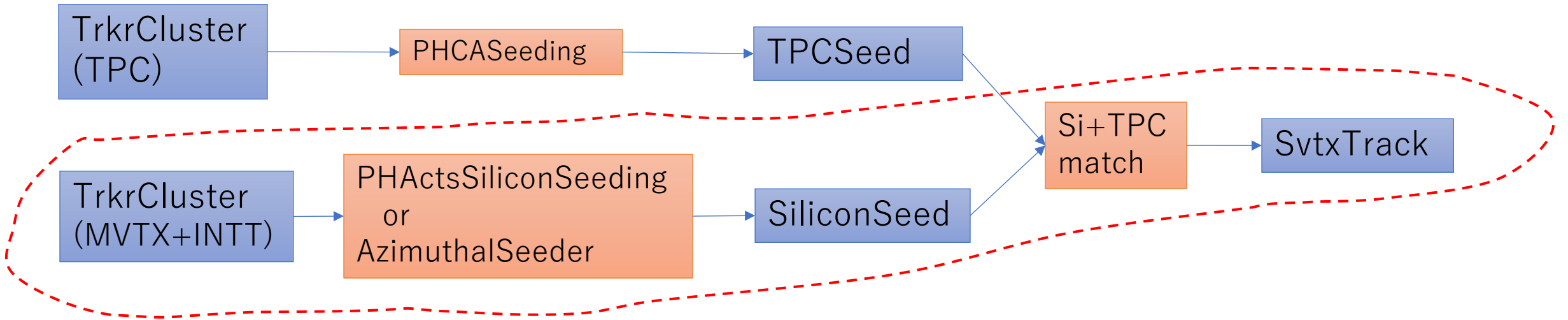
- Convert Ch in the detector(online) to Ch in sPHENIX software(offline)
- Convert BCO to Time (Timebucket)
 - Trigger: $\text{time} = \text{BCO} - \text{BCOFULL}$
 - Stream: $\text{time} = \text{BCOFULL}(\text{stream}) + \text{BCO} - \text{GL1BCO}$
- Remove “BAD” channels using Hot/DeadMAP
- BCO Cut (optional)
 - BCO-BCOFULL cut was made for Run2023 analysis but probably no longer used for 2024-25 data since Fun4All prepared similar function (but not same)

Clustering

Detailed flow of track reconstruction

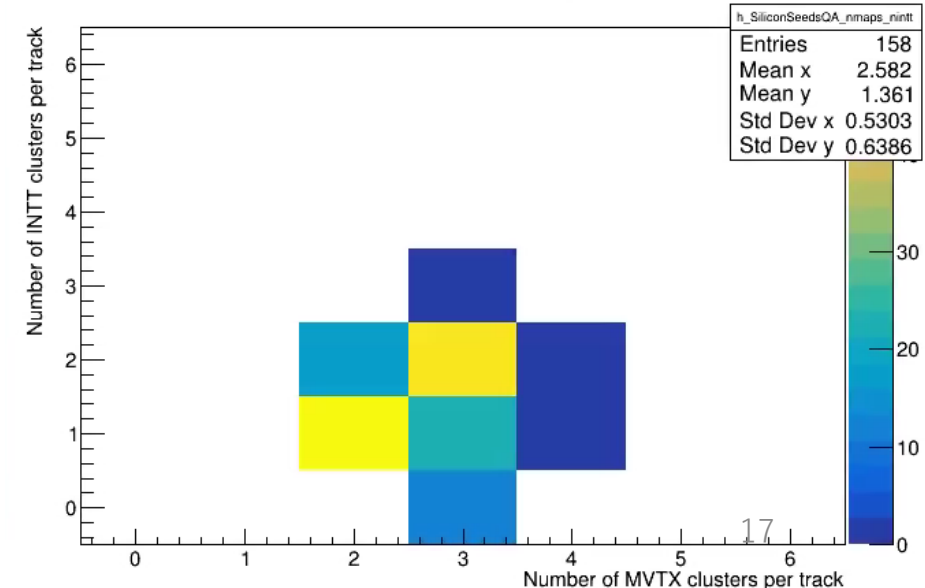


Silicon Seed



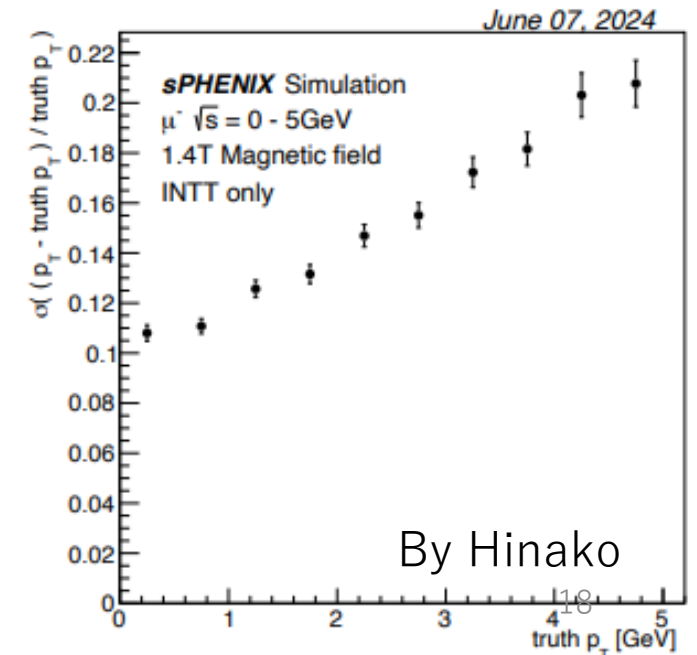
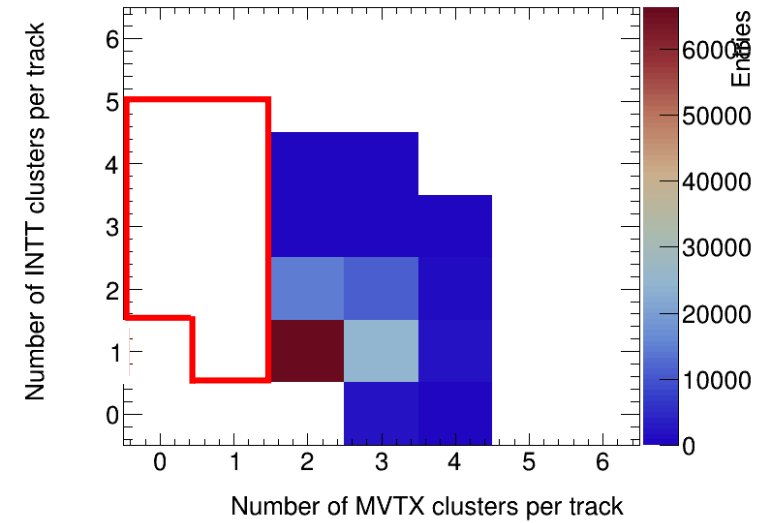
- https://github.com/sPHENIX-Collaboration/macros/blob/master/TrackingProduction/Fun4All_TrackAnalysis.C
- /gpfs/mnt/gpfs02/sphenix/user/hachiya/INTT/INTT/general_codes/hachiya/SiSeedAna/macro/Fun4All_TrackAnalysis.C

MVTX vs INTT clusters per track



Silicon seeding in tracking algorithm

- Tracking(Silicon Seeding) requires at least 2 MVTX + 1 INTT clusters
 - For low multiplicity such as p+p, this can be extended and relaxed.
 - S/N cannot be very good. Confirmation by outer detector (EMCAL) is needed.
- Furthermore, the momentum resolution can be improved by using EMCAL cluster
 - Help measuring Quarkonia by electron pairs
- Based on MC study by Hinako (NWU), INTT tracklet shows 10% pT resolution at pT=1GeV/c.



Seed variable checking w/ offline QA code

- <https://github.com/sPHENIX-Collaboration/coresoftware/blob/master/offline/QA/Tracking/SiliconSeedsQA.cc>
- **SiliconSeedsQA.cc**
 - `auto trackmap = findNode::getClass<SvtxTrackMap>(topNode, m_trackMapName);`
 - SvtxTrack contents
 - `virtual float get_p() const { return NAN; }`
 - `virtual float get_pt() const { return NAN; }`
 - `virtual float get_eta() const { return NAN; }`
 - `virtual float get_phi() const { return NAN; }`
 - https://github.com/sPHENIX-Collaboration/coresoftware/blob/fb24f0a8d57e65c68ba0b442ac3a913b894e5207/offline/packages/trackbase_historic/SvtxTrack.h

“Reference” is QA file from Jenkins (full tracking with TPC, TPOT and silicon)

“New” is single pion tracking with silicon-only

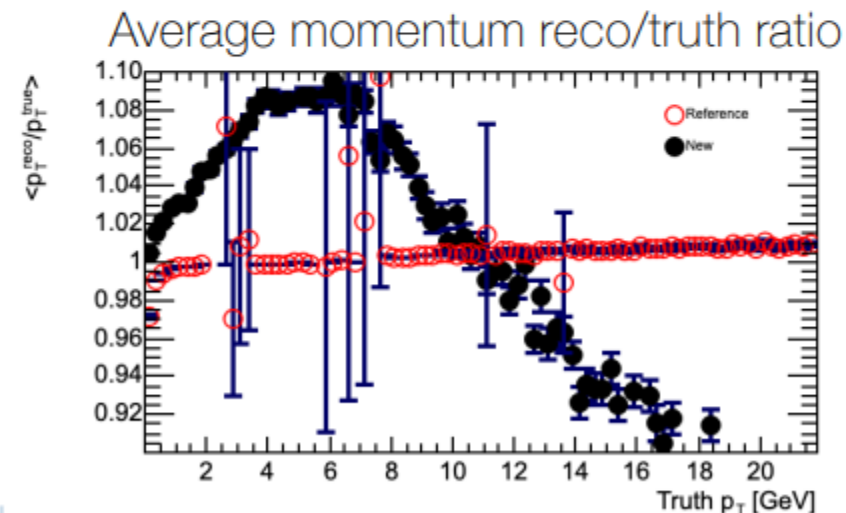
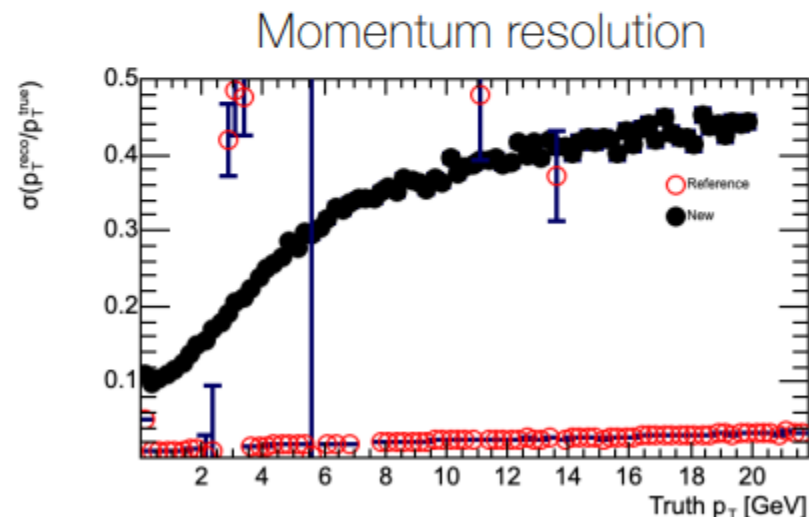
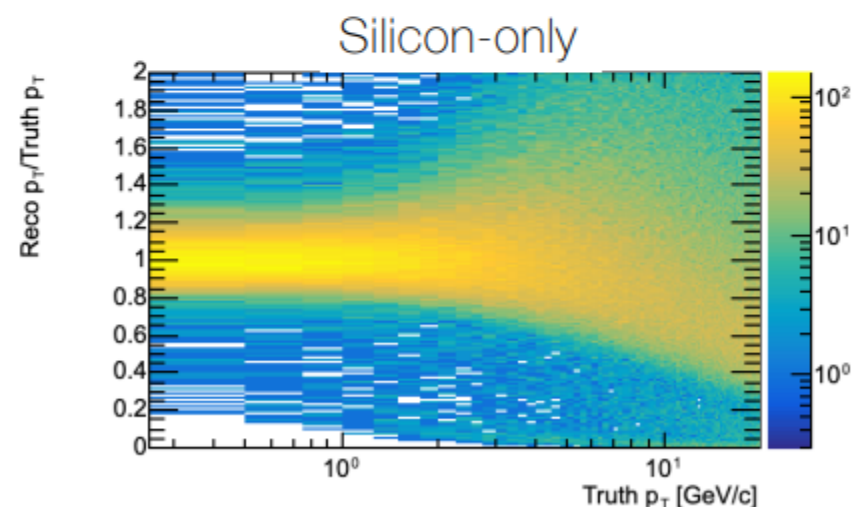
Thoughts

- Large divergence in reco to truth ratio (left) seems sensible at high p_T where tracks are straighter so difficult to measure curvature
- Increase in momentum resolution (middle) without TPC seems sensible and gets to around 45%

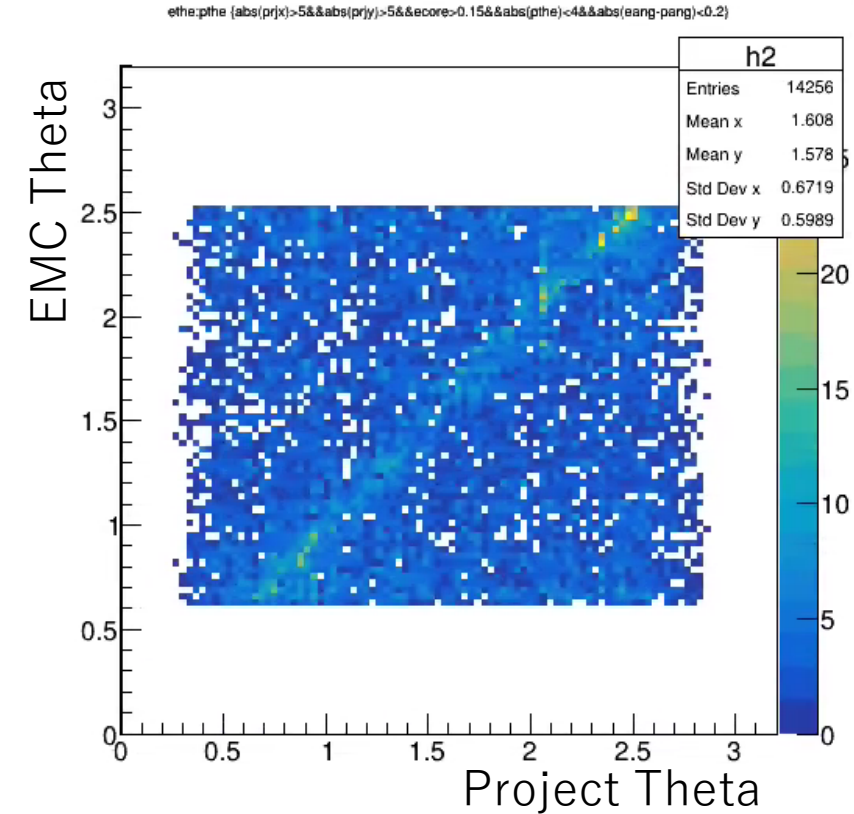
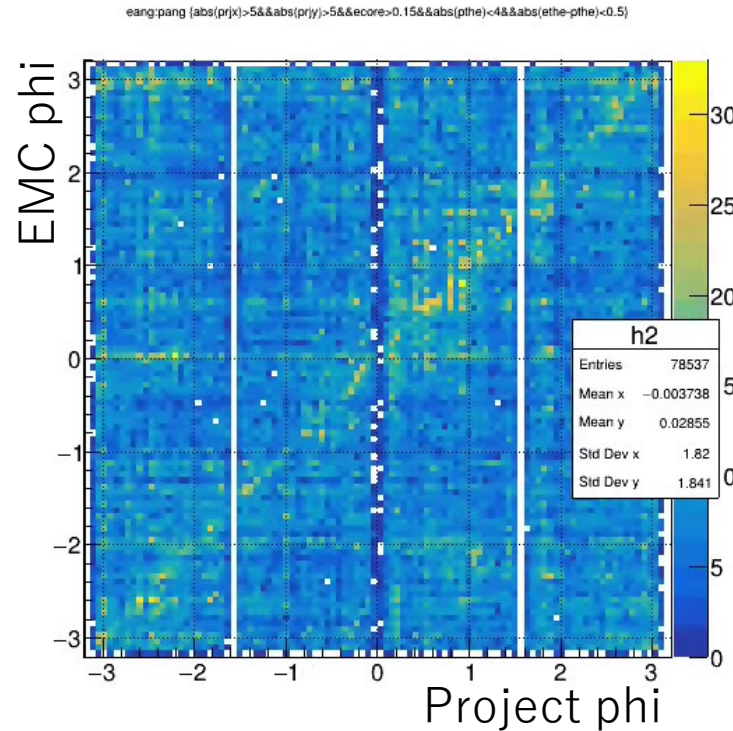
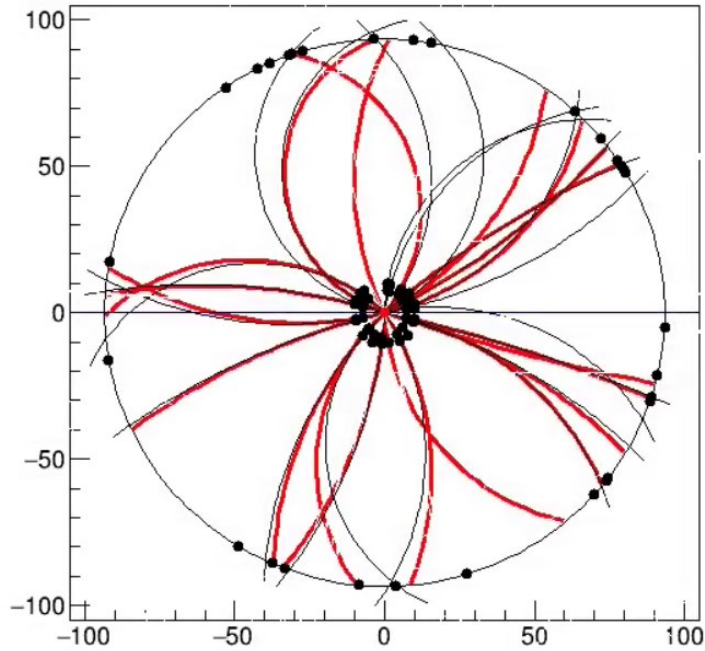
Middle and right plots

Red circles – pion gun, full tracking (TPC+TPOT+Silicon)

Black circles – pion gun, silicon-only



Projection of INTT tracklet to EMCAL



- MC(PYTHIA p+p MB)
 - Black line: INTT tracklet, Red: MC truth track
- INTT Tracklet is projected to EMCAL surface and search for the closest EMC cluster
 - INTT tracklet : 2 INTT clusters + XY vertex
 - Simple circle projection
- INTT + EMC matching works in p+p. Can be improved with MVTX

sPHENIX software on GitHub

<https://github.com/sPHENIX-Collaboration>

SPHENIX-Collaboration + 🔍 📄 📁 📦 🧑 11 🧑 265

Overview Repositories 74 Projects 1 Packages Teams 11 People 265



sPHENIX collaboration

sPHENIX is a new detector planned for the Relativistic Heavy Ion Collider facility at Brookhaven National Laboratory.

56 followers Upton, NY, USA <https://www.sphenix.bnl.gov/web/>

Follow

Pinned

coresoftware Public
Our big core software repository

C++ 27 199

macros Public
Official macros to run sPHENIX coressoftware. Welcome to fork and to edit for your own studies.

Jupyter Notebook 11 137

Singularity Public
Launcher and update macro for the sPHENIX Singularity container

Shell 5 9

tutorials Public
Our tutorials

Jupyter Notebook 10 35

analysis Public
Analysis repository hosting user modules that analyze the sPHENIX simulation and data

C 4 89

2024/11/25

calibrations Public
Calibration files and macros producing them. Automatically installed to RCF \$CALIBRATIONROOT/ during each nightly build

Jupyter Notebook 48

View as: Public

You are viewing the README and pinned repositories as a public user.

People



View all

Top languages

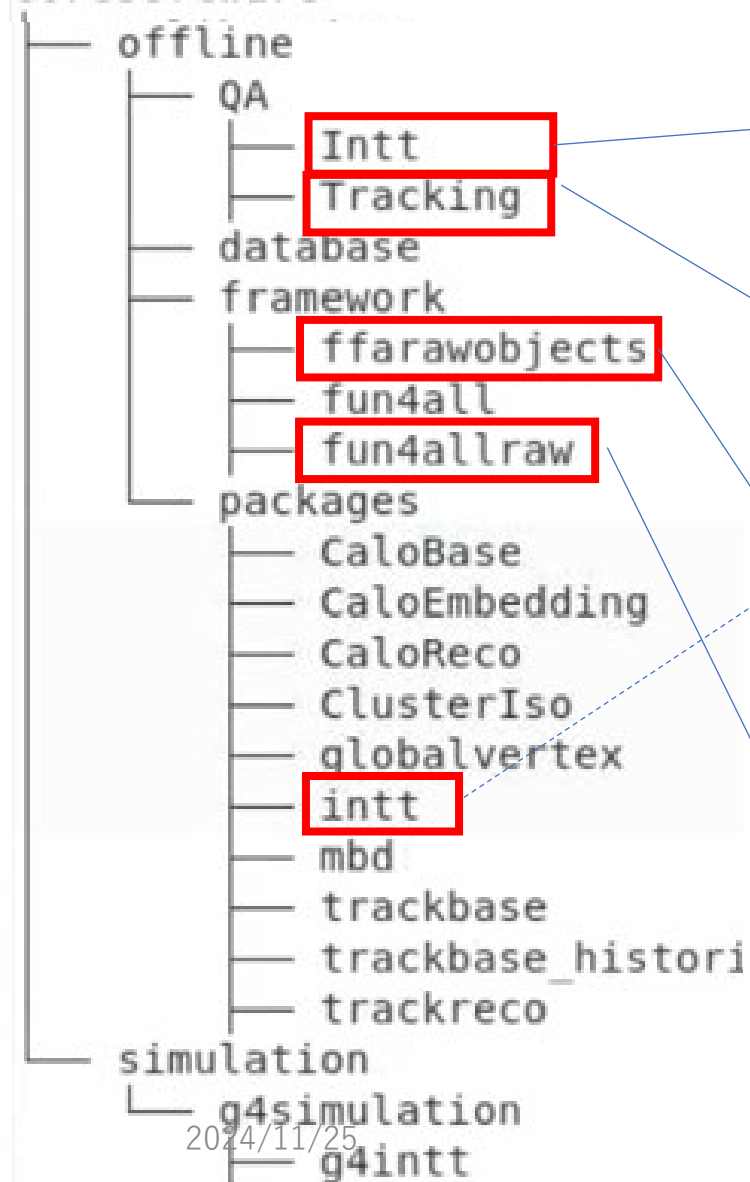
C++ C Python

Jupyter Notebook Rust

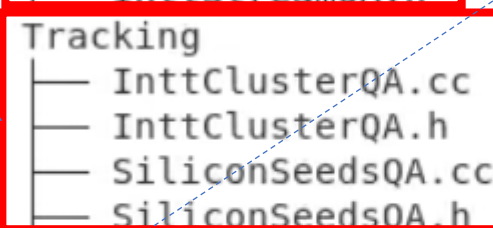
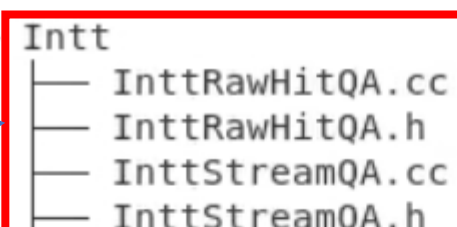
22

Coresoftware in github

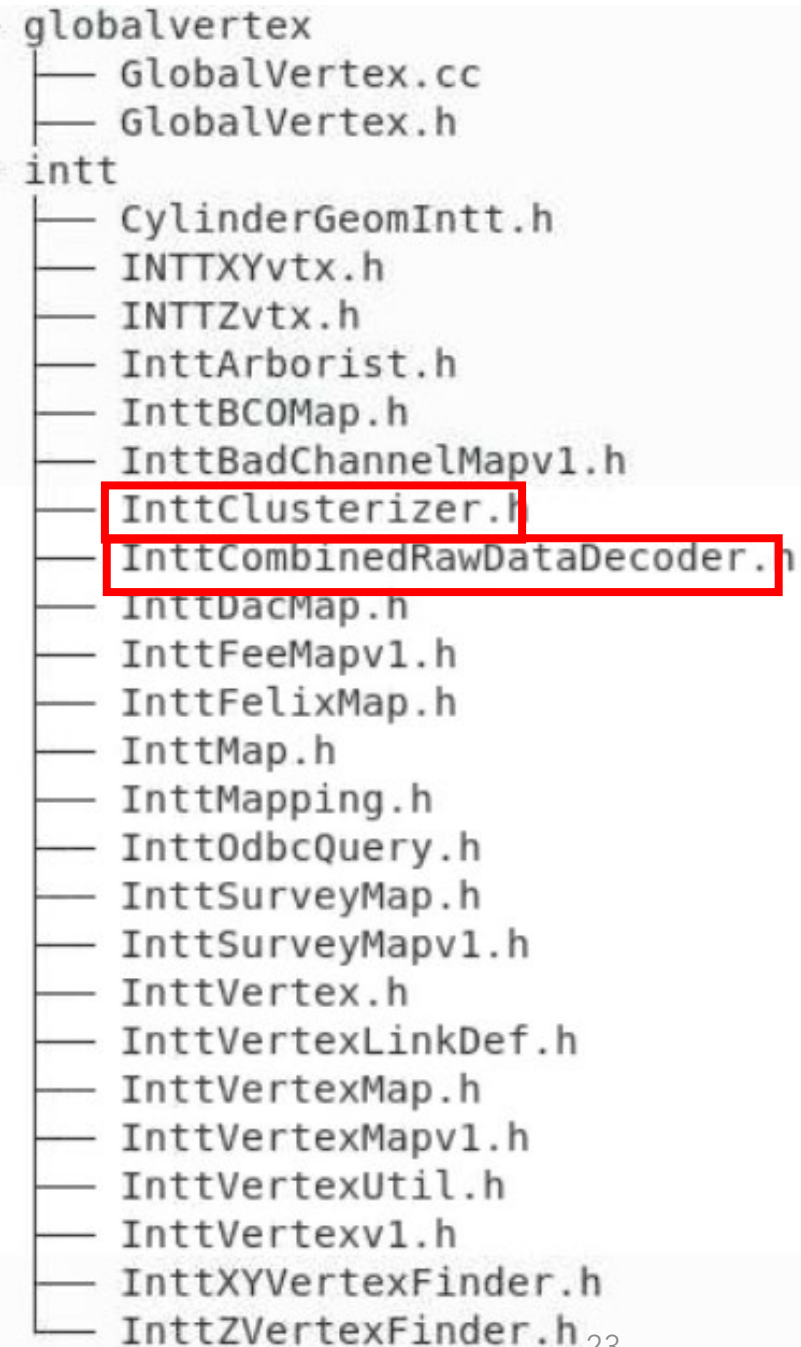
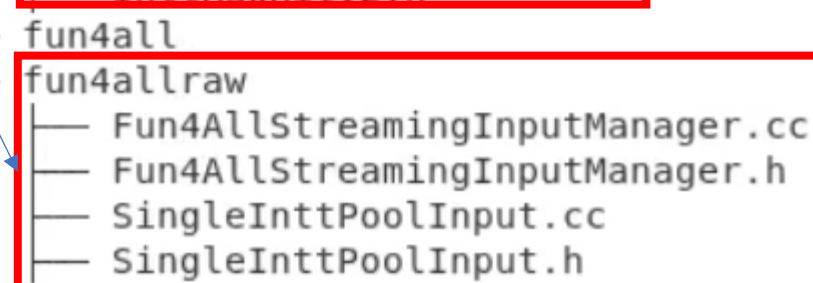
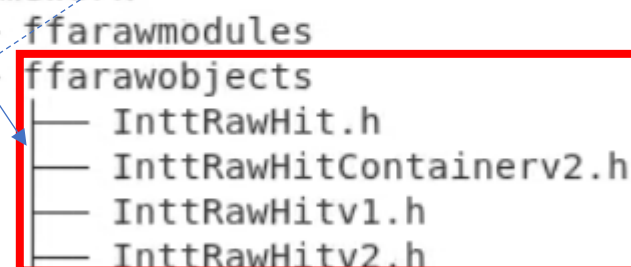
coresoftware



QA



framework



INTT vertex

- F4A module to calculate INTT XY and Z vertex available in sPHENIX-GITHUB

- These modules developed based on ChengWei's vertexing code

- It works but need to separate clusters by time (BCO)

```
globalvertex
├── GlobalVertex.cc
├── GlobalVertex.h
└── intt
    ├── CylinderGeomIntt.h
    ├── INTTXYvtx.h
    ├── INTTZvtx.h
    ├── InttArborist.h
    ├── InttBCOMap.h
    ├── InttBadChannelMapv1.h
    ├── InttClusterizer.h
    ├── InttCombinedRawDataDecoder.h
    ├── InttDacMap.h
    ├── InttFeeMapv1.h
    ├── InttFelixMap.h
    ├── InttMap.h
    ├── InttMapping.h
    ├── InttOdbcQuery.h
    ├── InttSurveyMap.h
    ├── InttSurveyMapv1.h
    ├── InttVertex.h
    ├── InttVertexLinkDef.h
    ├── InttVertexMap.h
    ├── InttVertexMapv1.h
    ├── InttVertexUtil.h
    ├── InttVertexv1.h
    ├── InttXYVertexFinder.h
    └── InttZVertexFinder.h
```

2024/11/25