Global Track Reconstruction and Data Compression Strategy in ALICE for LHC Run 3

David Rohr for the ALICE Collaboration

drohr@cern.ch, CERN

Connecting the Dots 2019

Valencia, Spain

4.4.2019

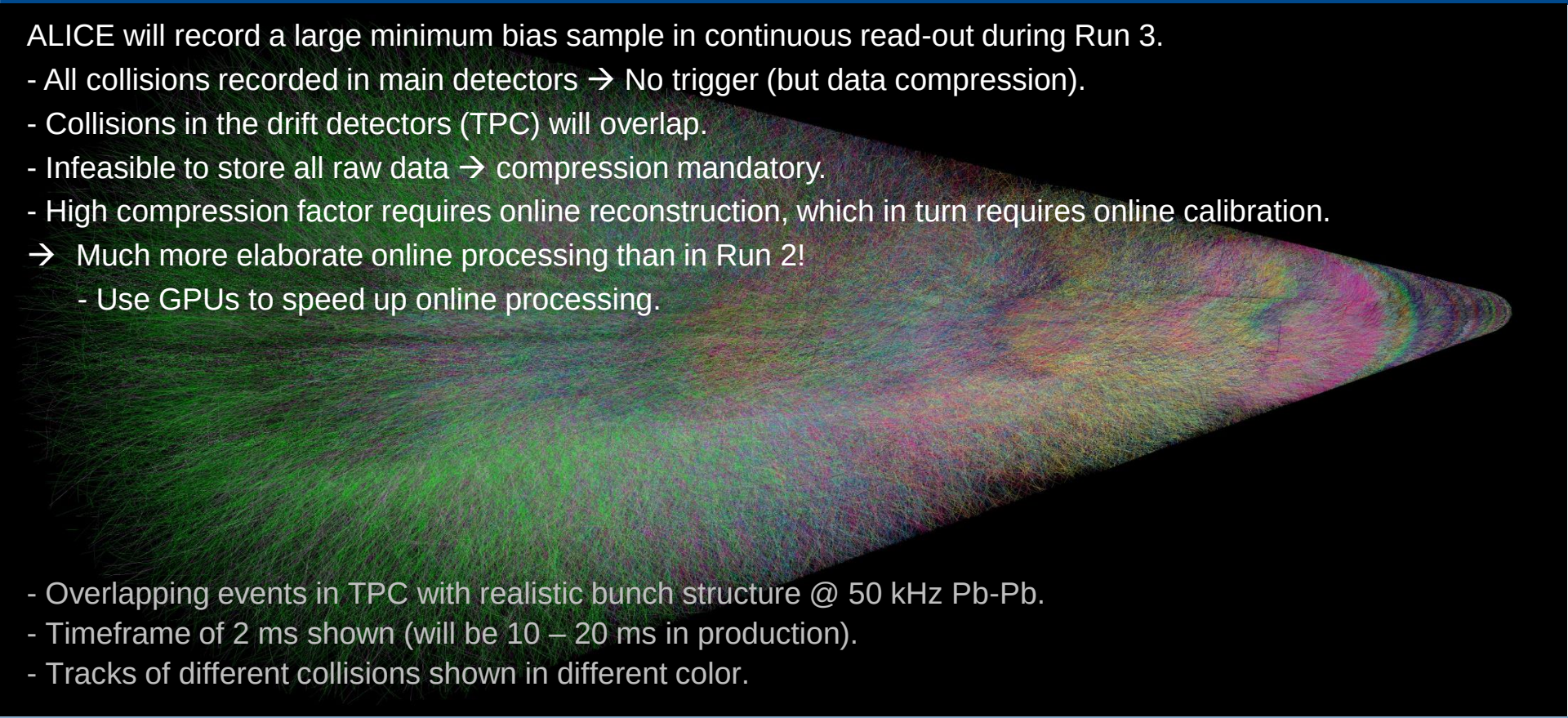
ALICE in Run 3

ALICE will record a large minimum bias sample in continuous read-out during Run 3.

- All collisions recorded in main detectors → No trigger (but data compression).
- Collisions in the drift detectors (TPC) will overlap.
- Infeasible to store all raw data → compression mandatory.
- High compression factor requires online reconstruction, which in turn requires online calibration.

→ Much more elaborate online processing than in Run 2!

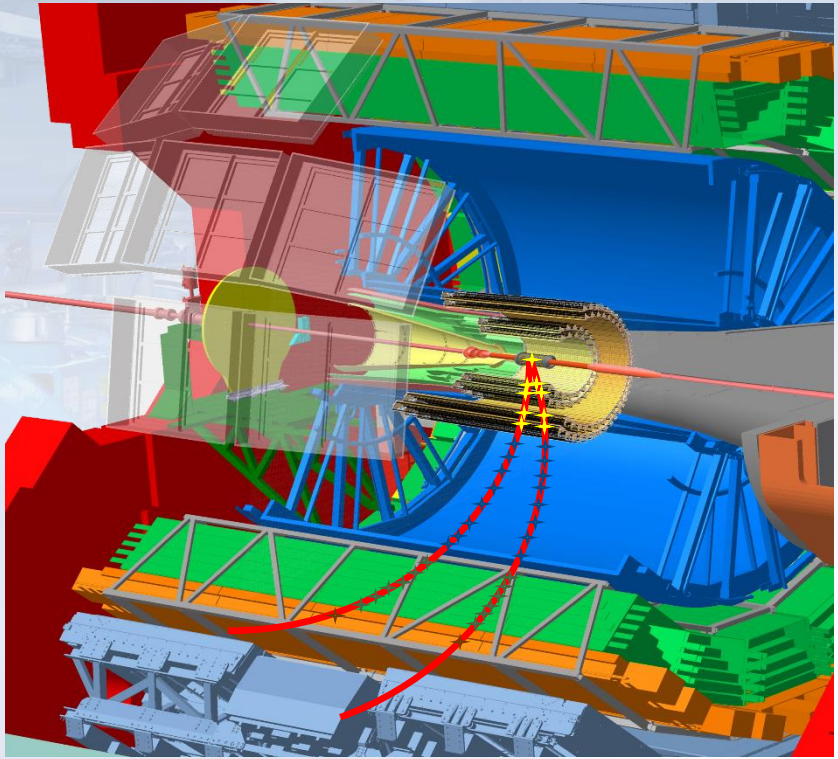
- Use GPUs to speed up online processing.



- Overlapping events in TPC with realistic bunch structure @ 50 kHz Pb-Pb.
- Timeframe of 2 ms shown (will be 10 – 20 ms in production).
- Tracks of different collisions shown in different color.

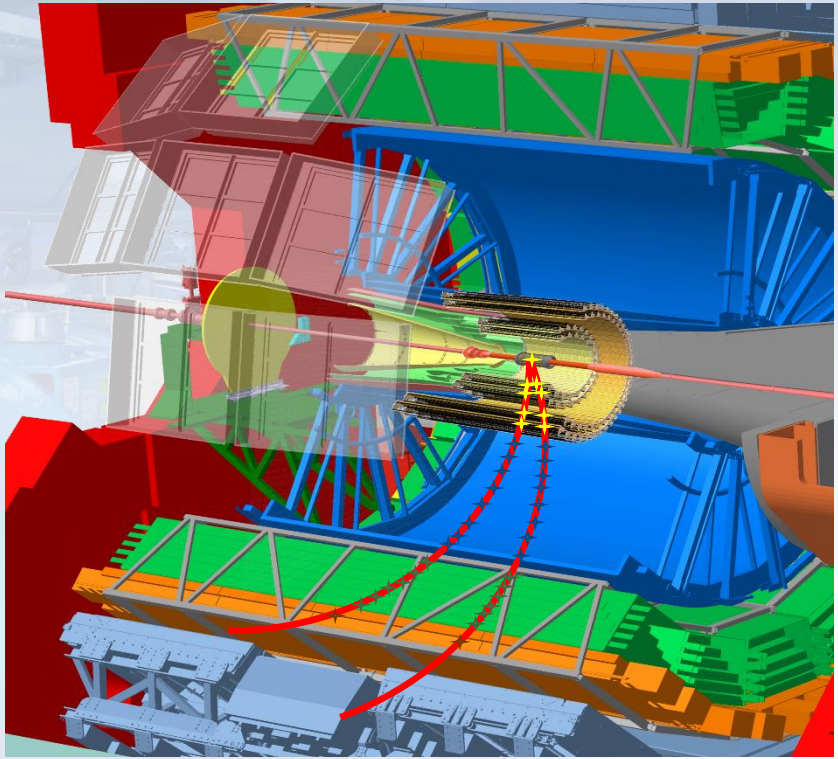
Tracking in ALICE in Run 3

- **ALICE uses mainly 3 detectors for tracking: ITS, TPC, TRD + (TOF)**
  - **7 layers ITS** (Inner Tracking System – silicon tracker)
  - **152 pad rows** TPC (Time Projection Chamber)
  - **6 layers TRD** (Transition Radiation Detector)
  - **1 layer TOF** (Time Of Flight Detector)



Tracking in ALICE in Run 3

- **ALICE uses mainly 3 detectors for tracking: ITS, TPC, TRD + (TOF)**
  - **7 layers ITS** (Inner Tracking System – silicon tracker)
  - **152 pad rows** TPC (Time Projection Chamber)
  - **6 layers TRD** (Transition Radiation Detector)
  - **1 layer TOF** (Time Of Flight Detector)



- **ALICE processing strategy**

Baseline solution
(almost available today):
TPC + part of ITS tracking on GPU

– Mandatory solution to keep up with the data rate online.

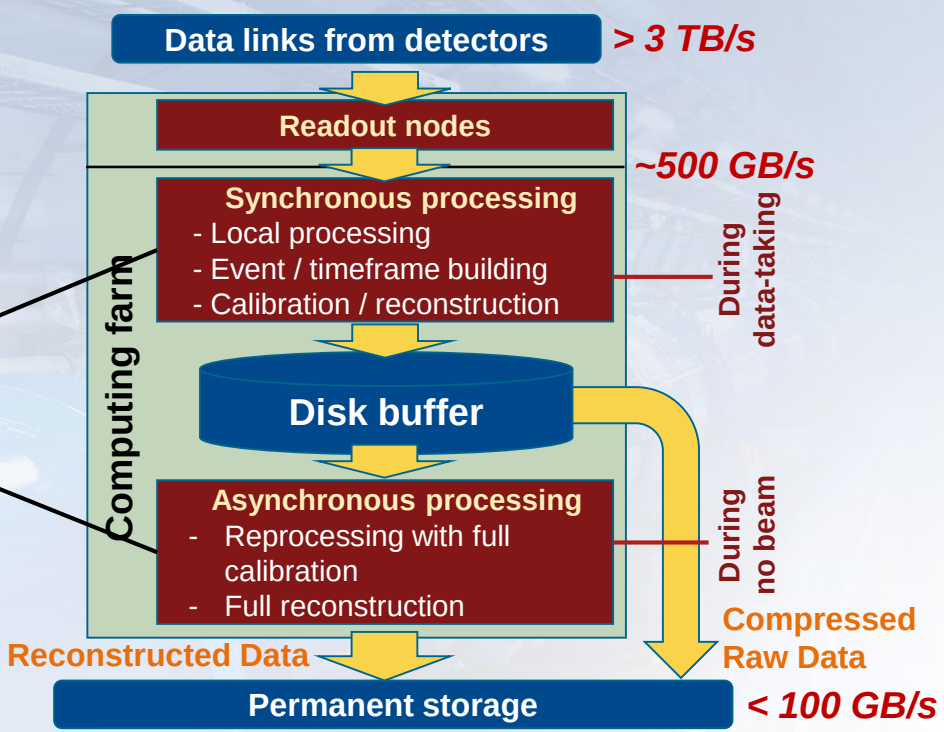
Optimistic solution
(what could we do in the ideal case):
Run most of tracking + X on GPU.

– Extension of baseline solution to make best use of GPUs.
  – Ideally, full barrel tracking without ever leaving the GPU.
  – In the end, we will probably be somewhere in between.

Online / Offline Computing in ALICE in Run 3

- **ALICE computing farm for Run 3**
  - On-site computing farm for online / offline.

- **Two reconstruction phases in Run 3:**
  - Synchronous reconstruction (during data taking):
    - **Calibration**
    - **Data compression**
  - Asynchronous reconstruction (when no beam):
    - **Full reconstruction with final calibration**



Online / Offline Computing in ALICE in Run 3

- **ALICE computing farm for Run 3**
  - On-site computing farm for online / offline.

- **Two reconstruction phases in Run 3:**
  - Synchronous reconstruction (during data taking):
    - **Calibration**
    - **Data compression**
  - Asynchronous reconstruction (when no beam):
    - **Full reconstruction with final calibration**

- **Partial ITS + TPC + TRD tracking for TPC calibration**
  - reduced statistics sufficient
  (TPC calibration based on matching of TPC / ITS / TRD tracks)
- *Other detectors without significant CPU load*

- **Full TPC tracking for TPC compression**
  - cluster to track residuals → better entropy coding
  - removal of tracks not used for physics
- *Entropy coding for other detectors*

Final reconstruction pass with final calibration

Online / Offline Computing in ALICE in Run 3

- **ALICE computing farm for Run 3**
  - On-site computing farm for online / offline.

- **Two reconstruction phases in Run 3:**
  - Synchronous reconstruction (during data taking):
    - **Calibration**
    - **Data compression**
  - Asynchronous reconstruction (when no beam):
    - **Full reconstruction with final calibration**

- **Partial ITS + TPC + TRD tracking for TPC calibration**
  - reduced statistics sufficient
  (TPC calibration based on matching of TPC / ITS / TRD tracks)
- *Other detectors without significant CPU load*

- **Full TPC tracking for TPC compression**
  - cluster to track residuals → better entropy coding
  - removal of tracks not used for physics
- *Entropy coding for other detectors*

Final reconstruction pass with final calibration

- **TPC calibration challenge: space charge distortions.**
  - **Run 3 GEM TPC without gating grid produces large number of ions at the end-plate (in contrast to Run 2).**
  - **Back-drifting ions will be dominant contribution to space charge (today only ions from primary ionization).**
  - **Space charge scales with collision rate: 50 kHz in Run 3 v.s. ~10 kHz in Run 2.**
  - **Space charge distorts the drifting electrons in the TPC by several cm.**

TPC Calibration

Distortions: Strategy A:

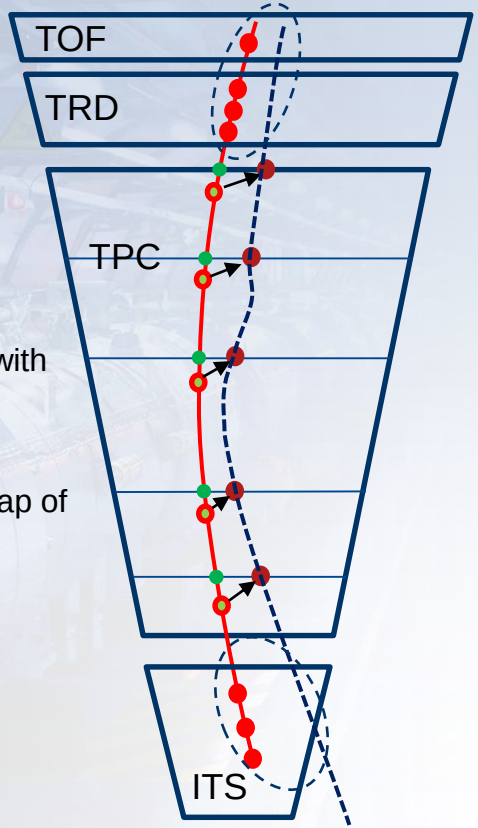
- Record all the current arriving at the TPC end plates (integrated digital current).
- Compute a time-dependent space-charge map inside the TPC.
- Compute the distortion-correction analytically from the map.

Use a combination: strategy B for absolute position, strategy A for short-time fluctuations.

Online processing needs at least correction for average distortions.

Distortions: Strategy B:

- Match TPC track to inner and outer detectors with precise position measurement.
- Refit track without TPC information.
- Residuals of clusters to refitted track yield a map of distortions in the TPC.



Drift Time: real-time calibration

- TPC drift velocity depends on certain factures such as pressure and temperature.
- Stable over a period of ~15 minutes.
- Can compute drift velocity by matching TPC to ITS tracks in a time-interval, and use this calibration for the following time interval in a feedback looped. (Deployed in Run 2 in the HLT).

ALICE Run 3 Data Taking & Online Computing Scheme

- Sub-event building
- Processing steps that need access to all data from a link (e.g. integrated digital current)

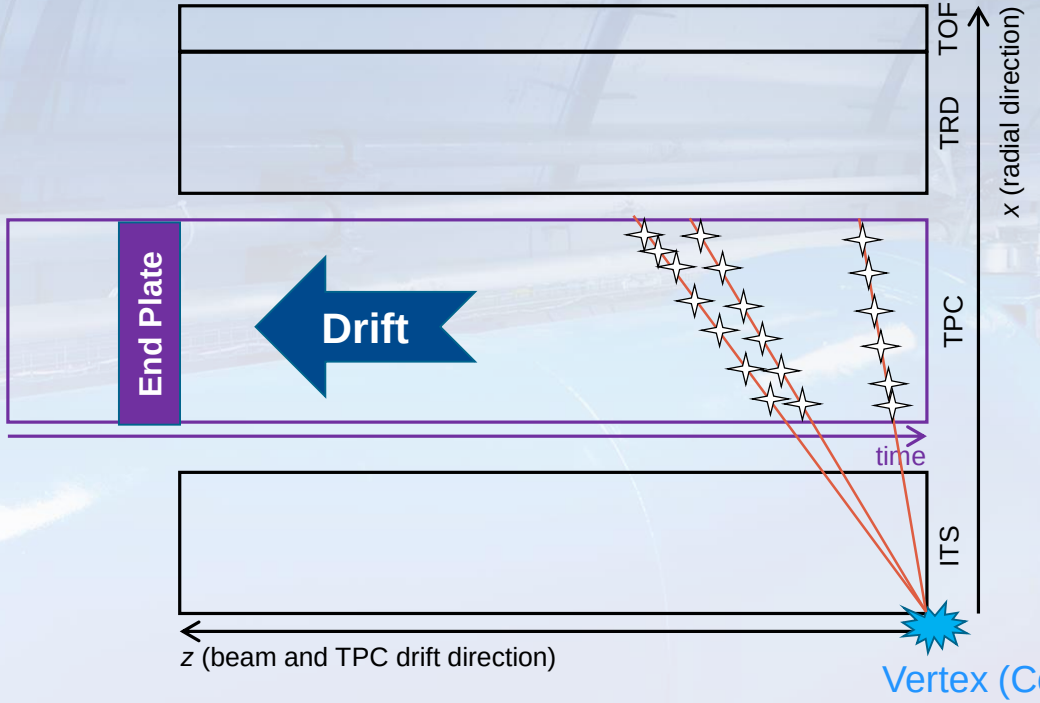
Round-robin distribution:
An EPN receives a full timeframe, and has 30 seconds for processing it

- **Synchronous:**
  - Full TPC tracking for compression (on GPUs).
  - ITS / TRD tracking of subset of events for calibration.
  - Integrated digital currents produced for calibration.
- **Postprocessing:**
  - Create space-charge and calibration map.
- **Asynchronous:**
  - Full reconstruction with final calibration for all detectors. (TPC not so dominant, split between Run 3 farm and GRID).

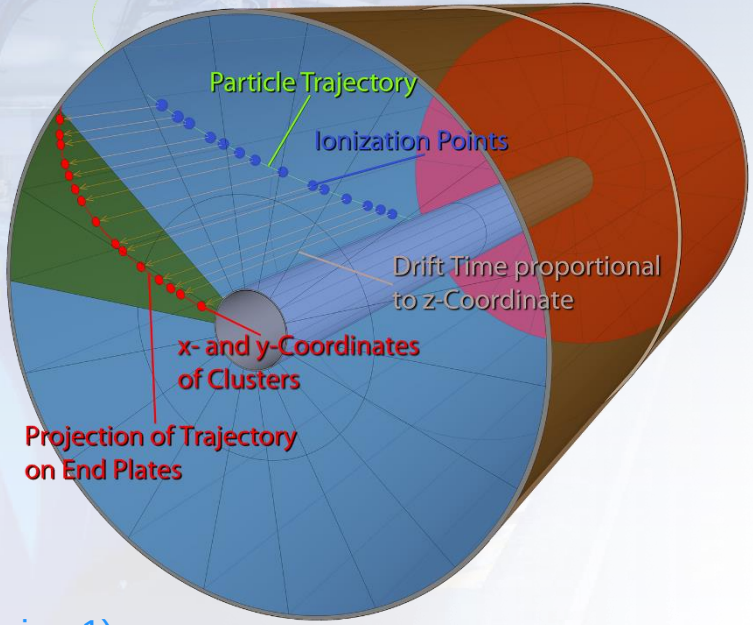
- **Synchronous TPC tracking @50kHz Pb-Pb defines peak-load for Run 3 farm.**
  - → GPU usage for TPC tracking mandatory!
  - Asynchronous reconstruction should leverage available GPU resources as good as possible.
  - Must run efficiently on CPUs on the GRID.
  - GPU reconstruction code written in a general way that runs on CPU & GPU with identical result.

The tracking challenge

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

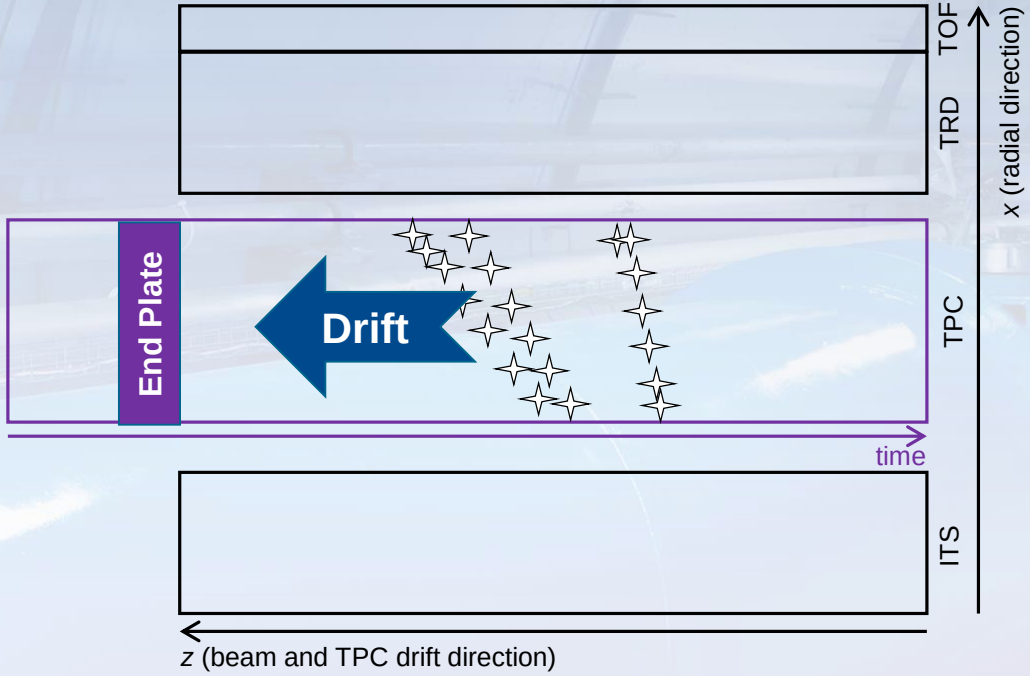


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**

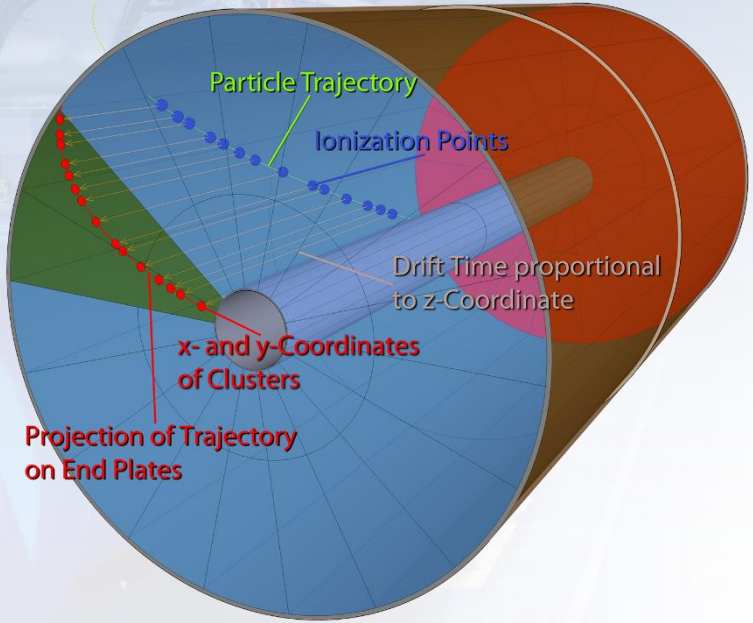


The tracking challenge

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

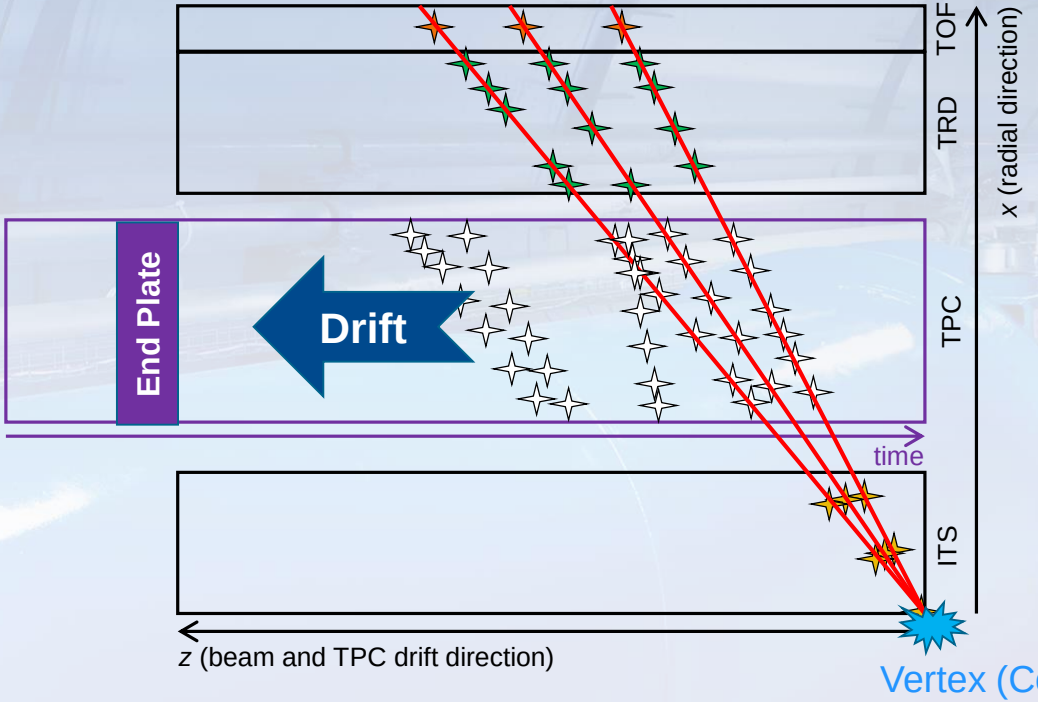


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**

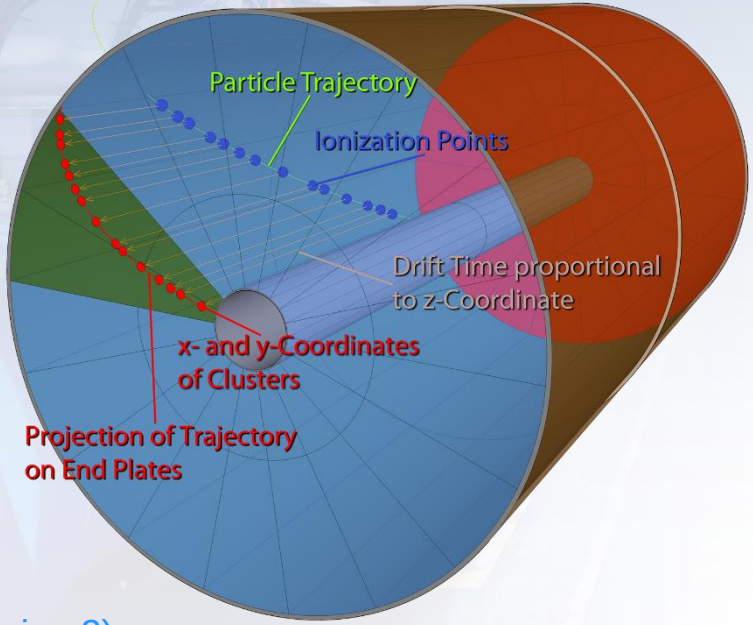


The tracking challenge

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

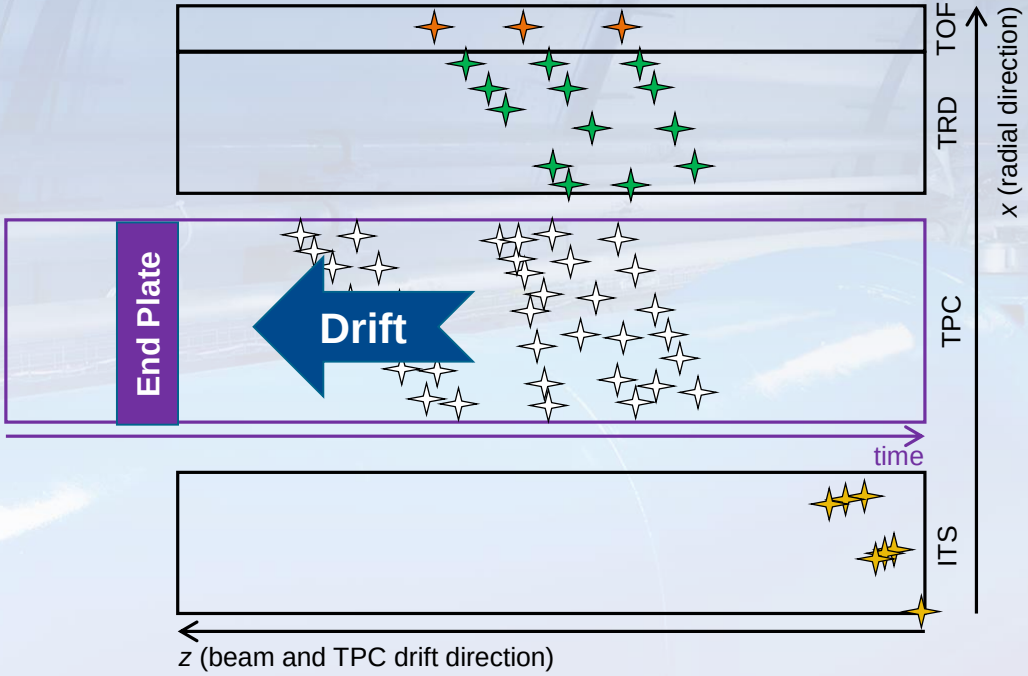


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**

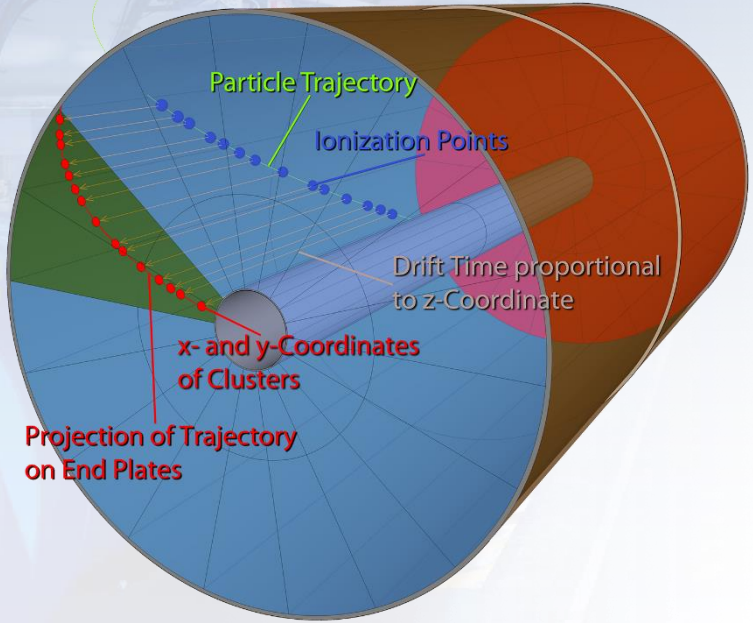


The tracking challenge

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

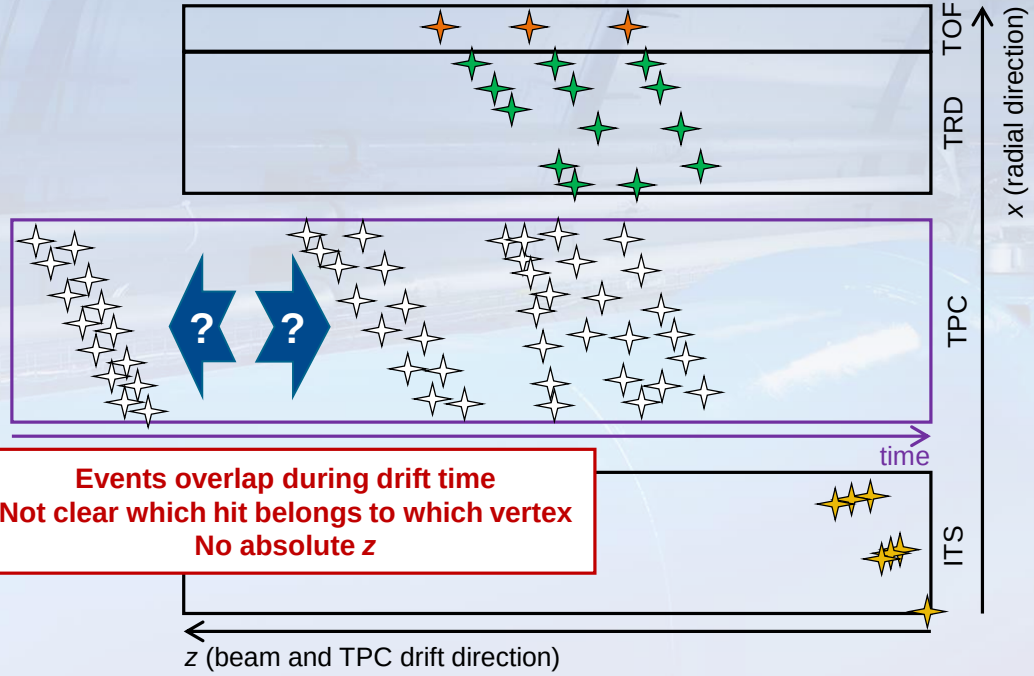


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**



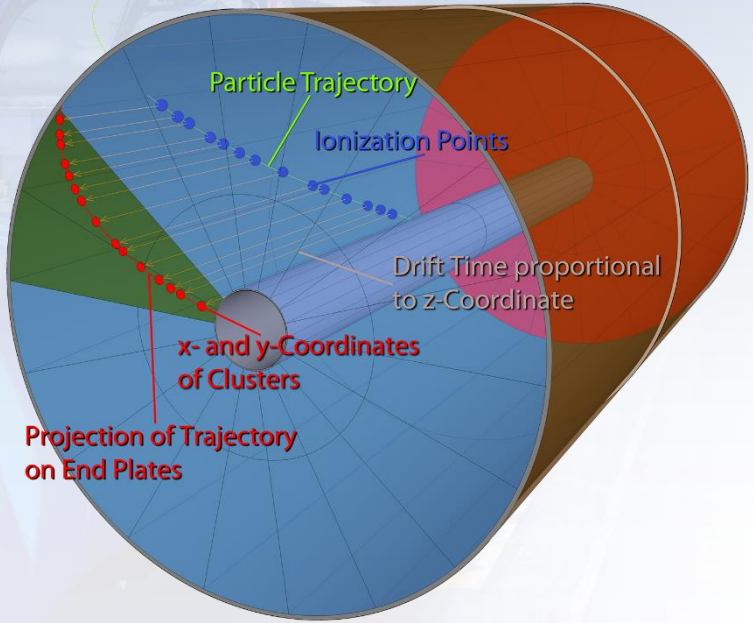
The tracking challenge

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



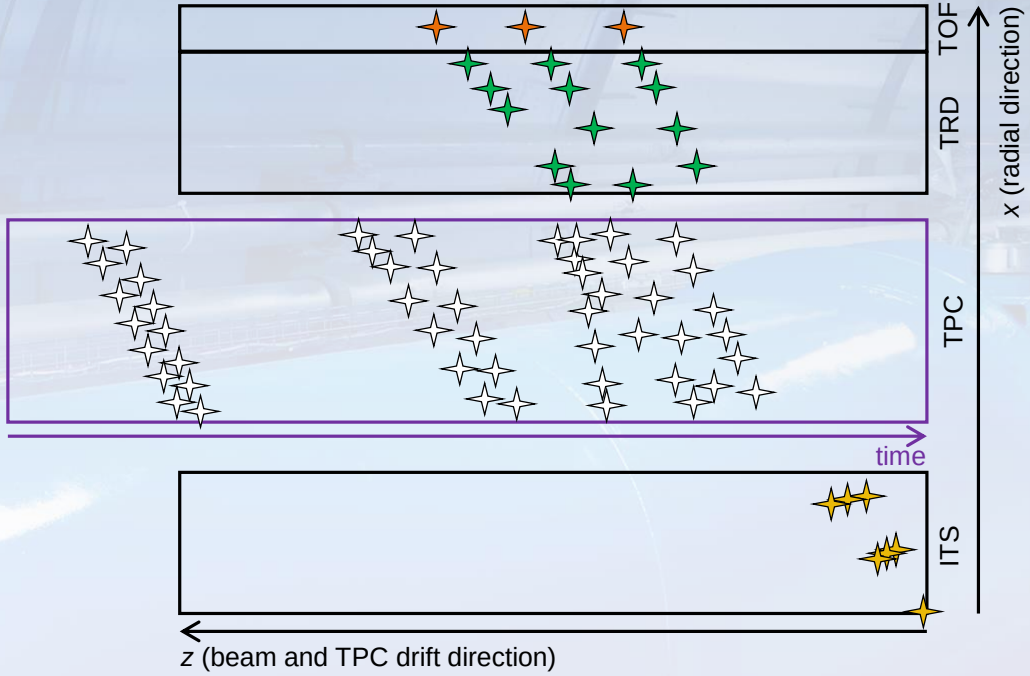
Events overlap during drift time
Not clear which hit belongs to which vertex
No absolute *z*

- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**



The tracking challenge

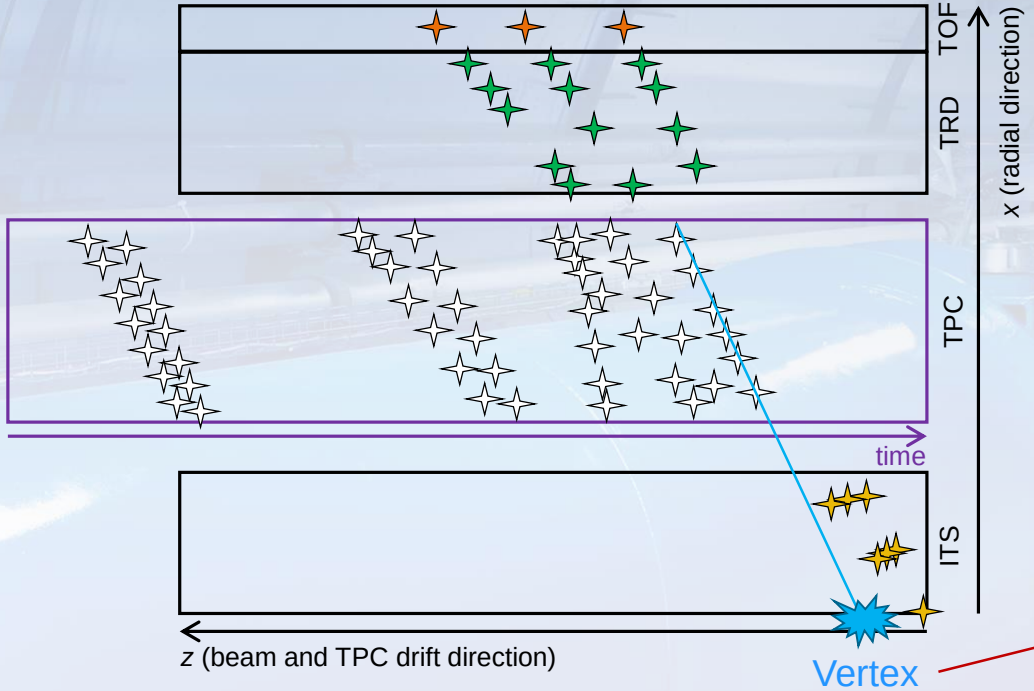
- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**

The tracking challenge

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



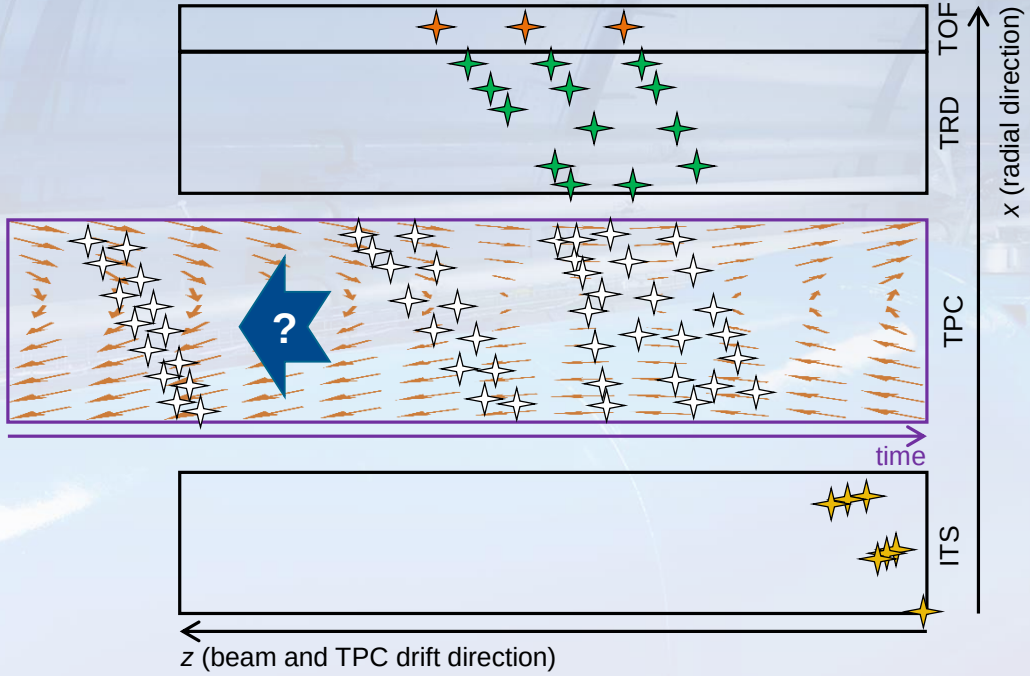
- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**

$z \sim t - t_{\mathrm{Vertex}}$

→ Need to identify the primary vertex, before assigning final *z* to cluster.

The tracking challenge

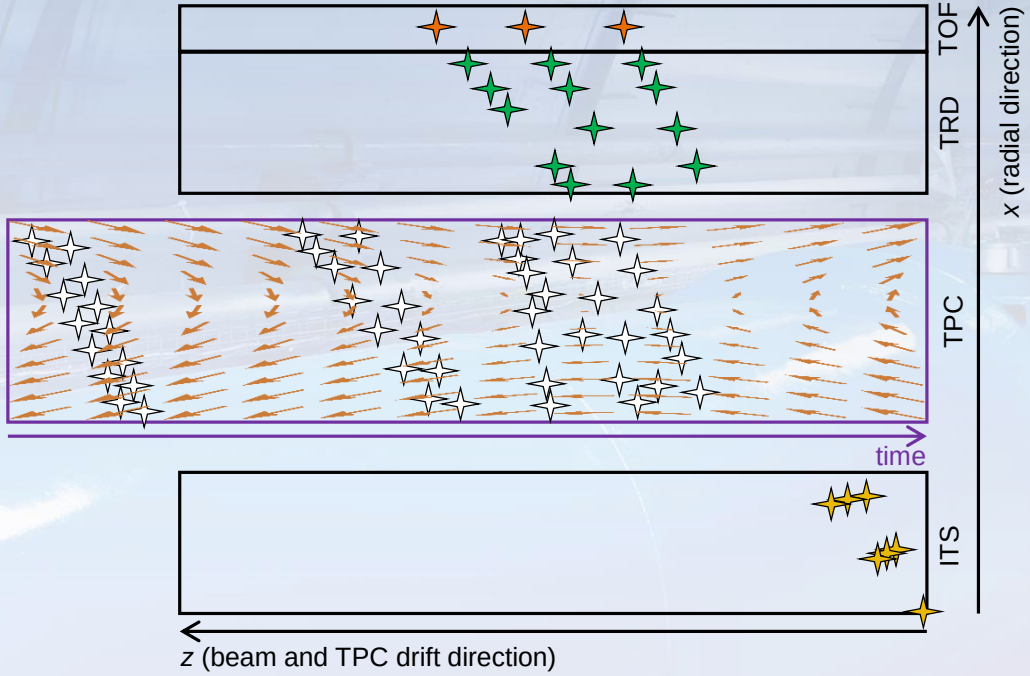
- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

The tracking challenge

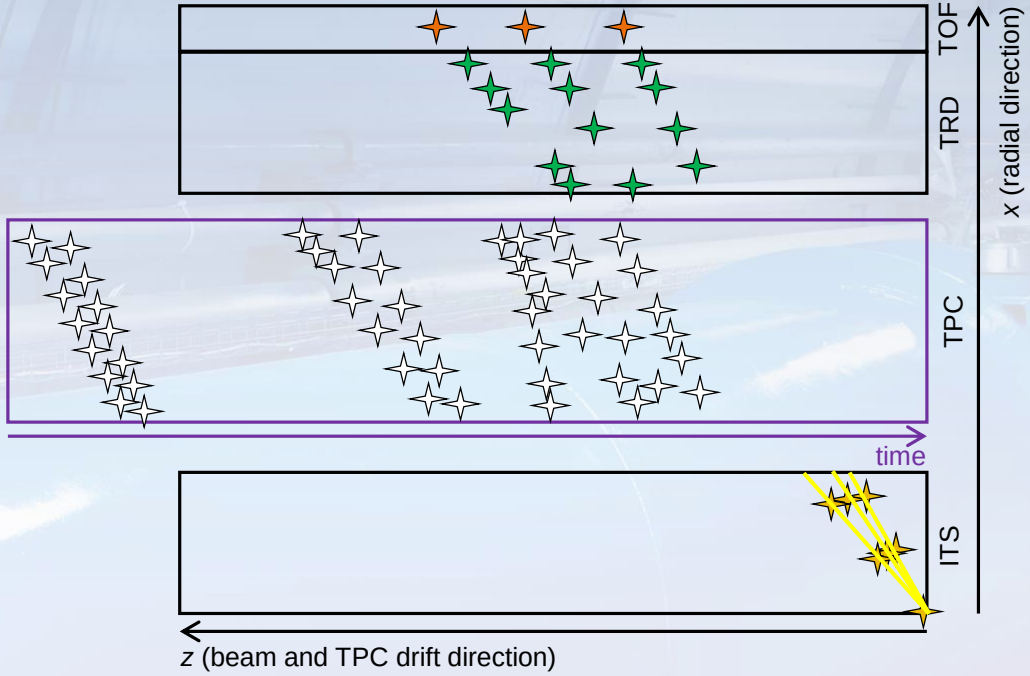
- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

The tracking challenge – How the tracking will work

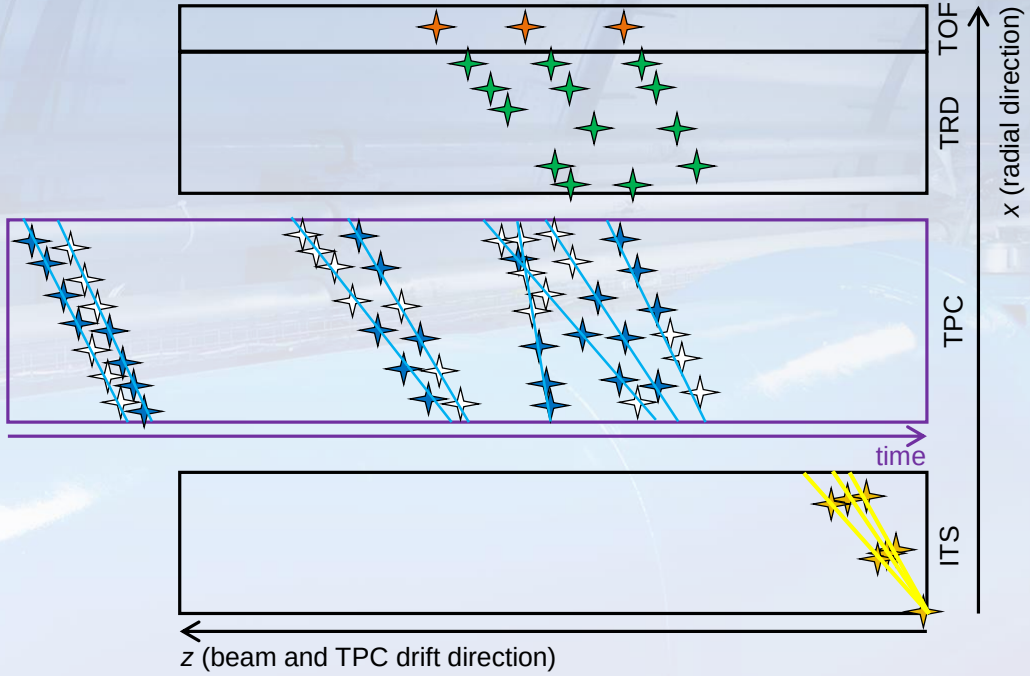
- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**
- Standalone ITS tracking.

The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.

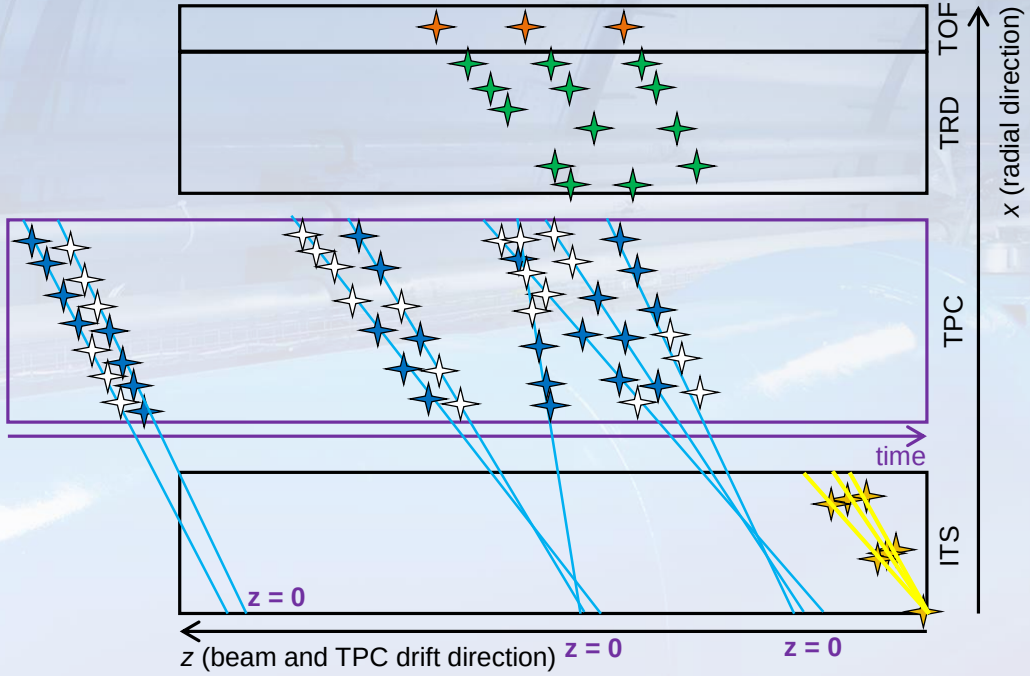
Precise tracking needs *z* for:

- Cluster error parameterization
- Inhomogeneous B-field
- Distortion correction

Effects smooth →
irrelevant for initial trackletting

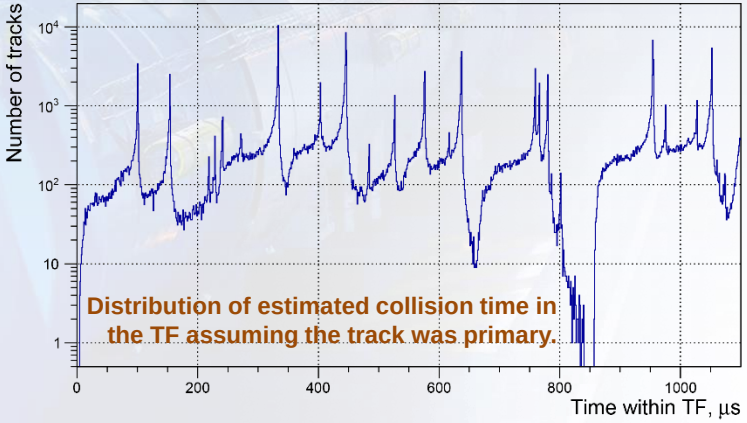
The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



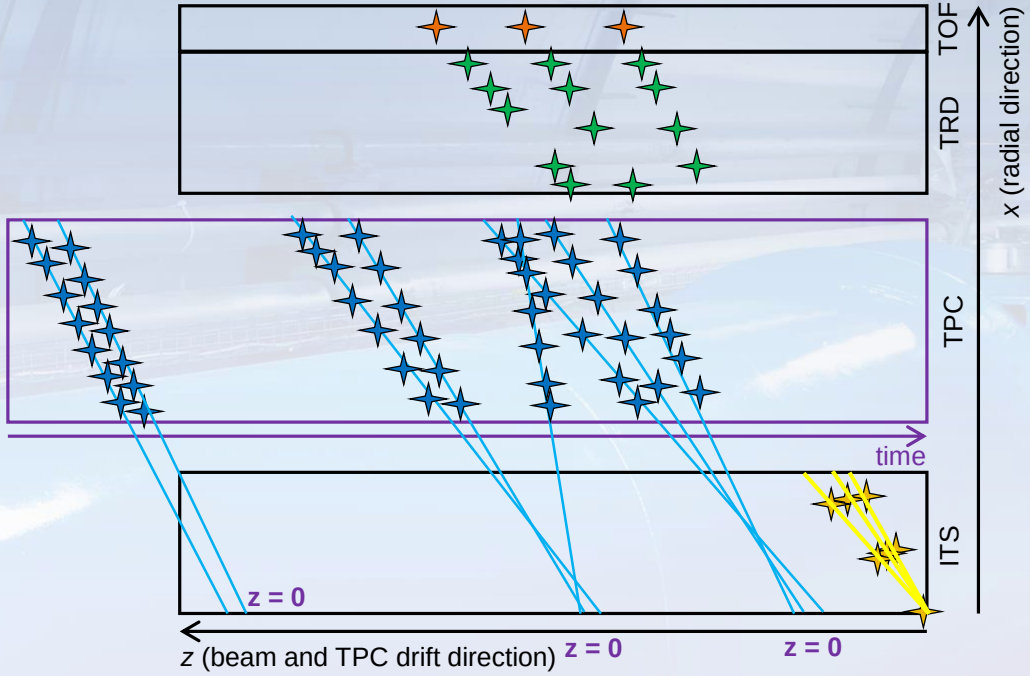
- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.



The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

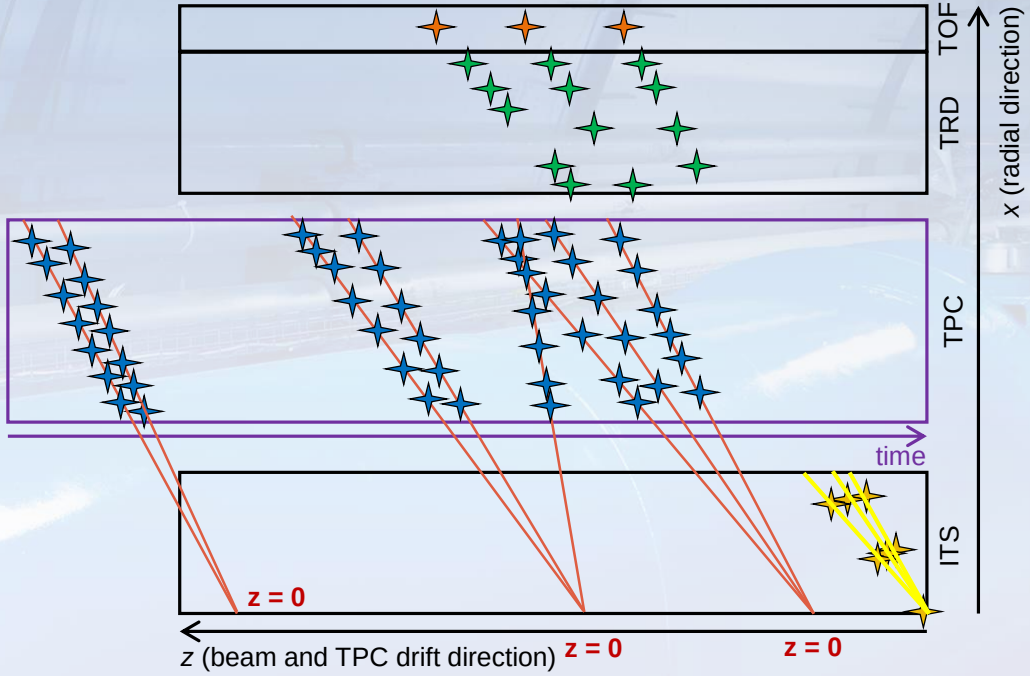


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.

The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

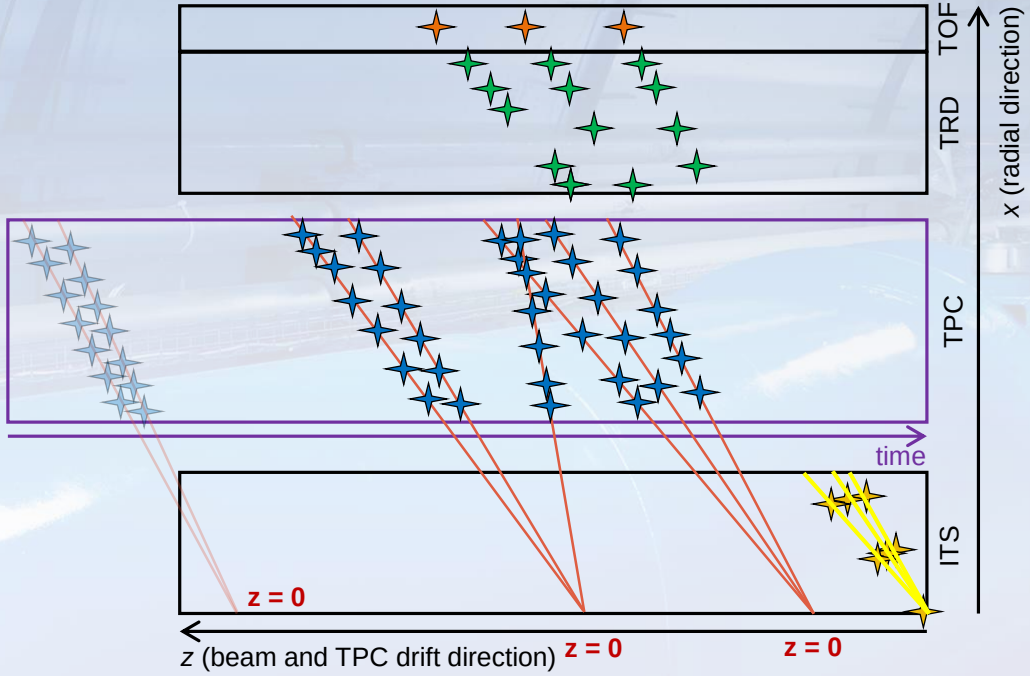


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision

The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

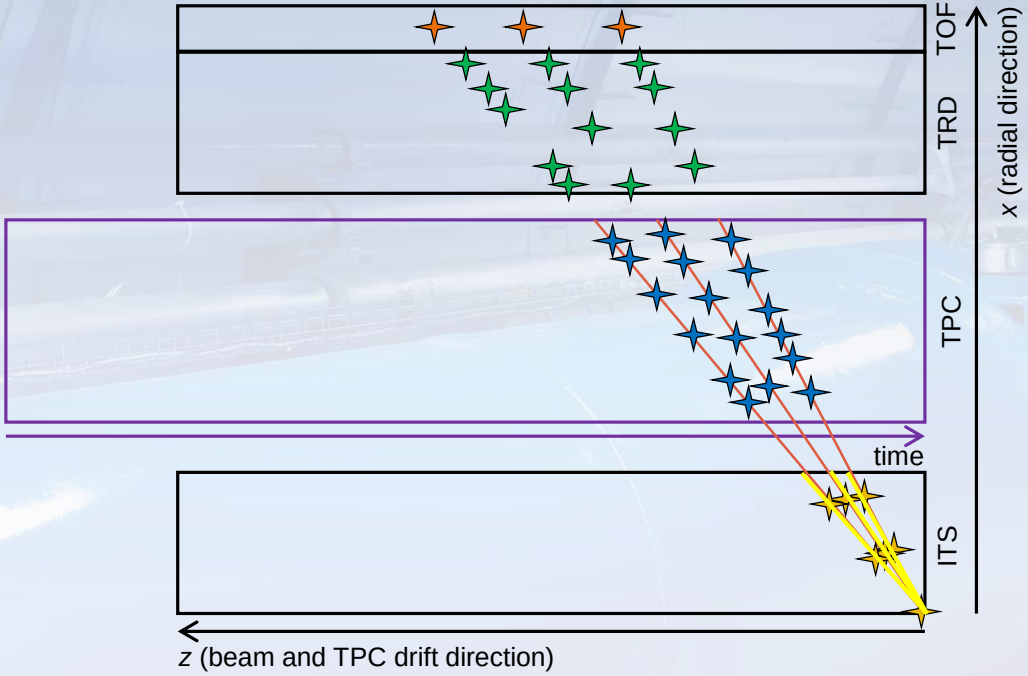


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision
- For the tracks in one ITS read out frame, select all TPC tracks with a compatible time (from $z = 0$ estimate).

The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

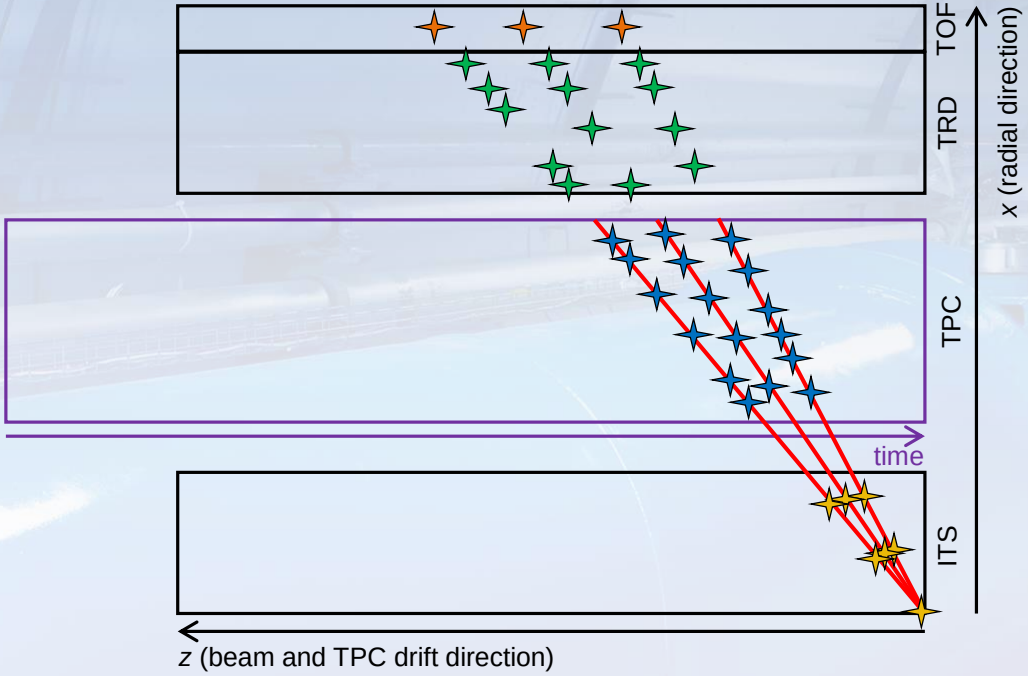


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision
- For the tracks in one ITS read out frame, select all TPC tracks with a compatible time (from $z = 0$ estimate).
- Match TPC track to ITS track, fixing z-position and time of the TPC track.

The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.

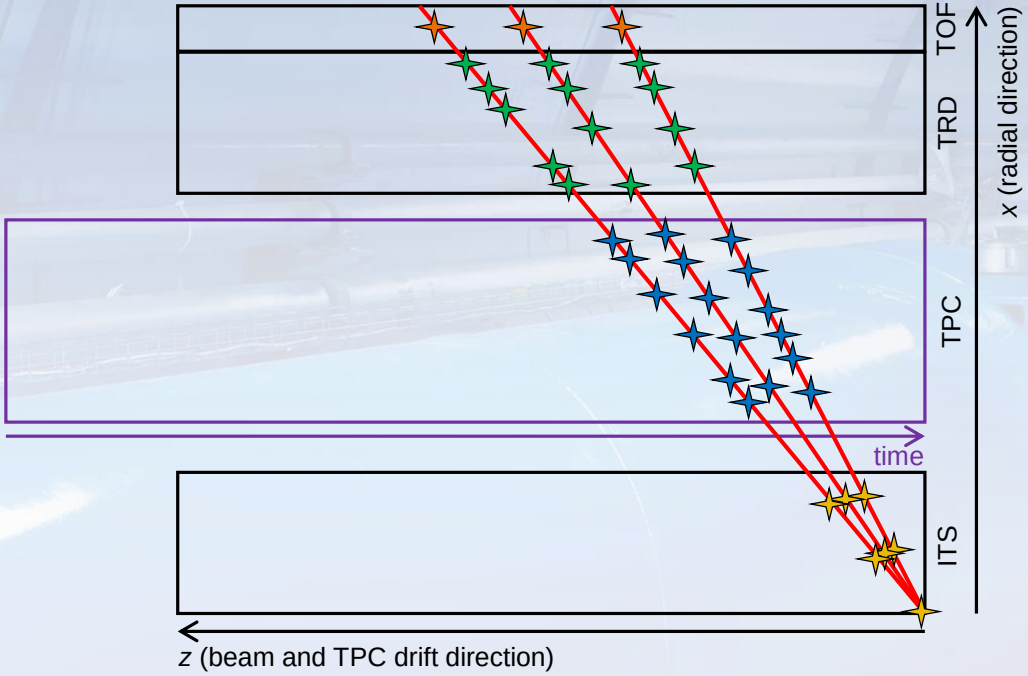


- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision
- For the tracks in one ITS read out frame, select all TPC tracks with a compatible time (from $z = 0$ estimate).
- Match TPC track to ITS track, fixing z-position and time of the TPC track.
- Refit ITS + TPC track outwards.

The tracking challenge – How the tracking will work

- **Tracking continuous data…**
  - The TPC sees **multiple overlapped collisions** (shifted in time).
  - Other detectors know the (rough) time of the collision.



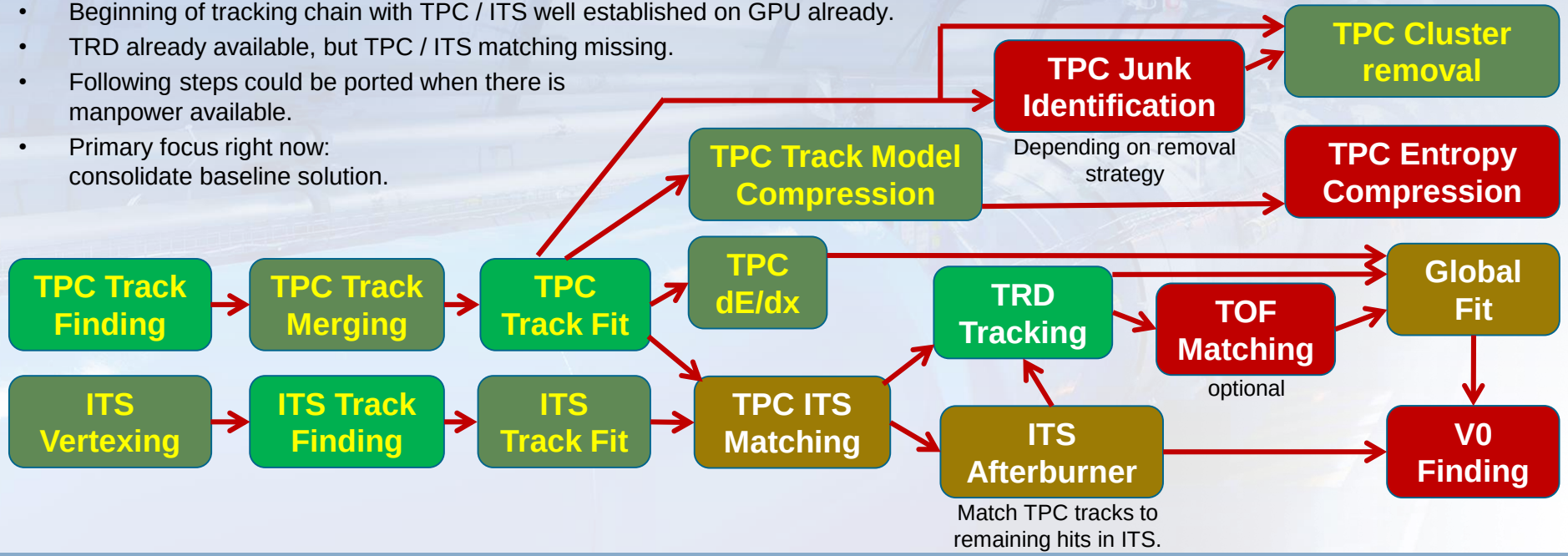
- **Problem: TPC clusters have no defined *z*-position but only a time. They can be shifted in *z* arbitrarily.**
- **GEM amplification produces ions that deflect the electrons during the drift. The correction of these space-charge distortions requires the absolute *z* position.**

- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision
- For the tracks in one ITS read out frame, select all TPC tracks with a compatible time (from $z = 0$ estimate).
- Match TPC track to ITS track, fixing z-position and time of the TPC track.
- Refit ITS + TPC track outwards.
- Prolong into TRD / TOF.

Barrel Tracking Chain

- Many steps of barrel tracking must run consecutively.
- Makes sense to port consecutive steps to GPU to avoid data transfer.
  - Although not strictly needed, depends also on data size. TPC clusters are most critical.

- Beginning of tracking chain with TPC / ITS well established on GPU already.
- TRD already available, but TPC / ITS matching missing.
- Following steps could be ported when there is manpower available.
- Primary focus right now: consolidate baseline solution.

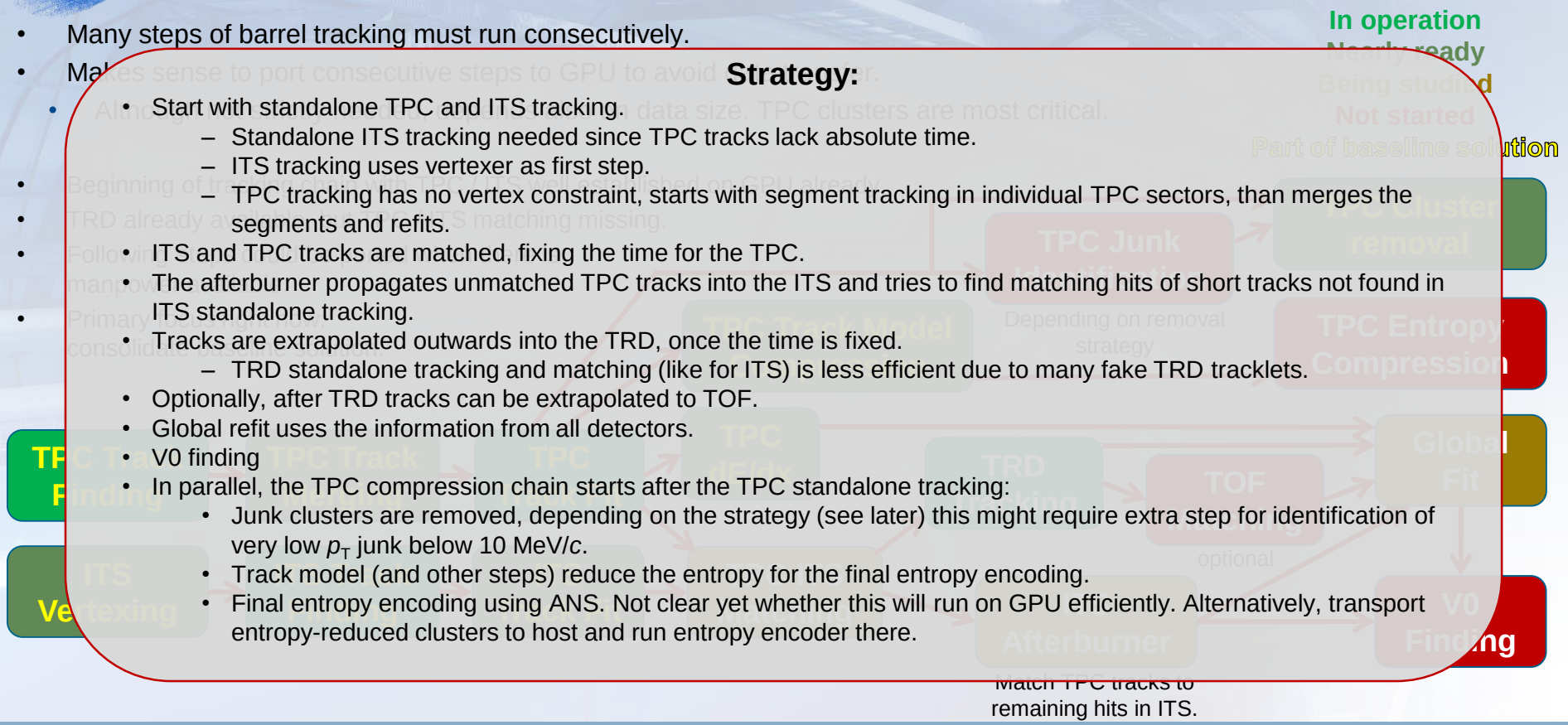


Barrel Tracking Chain

- Many steps of barrel tracking must run consecutively.
- Makes sense to port consecutive steps to GPU to avoid data transfer.
  - Aiming at reduction of data size. TPC clusters are most critical.
- Beginning of tracking chain [illegible] ported to GPU already.
- TRD already available, ITS matching missing.
- Following [illegible] manpower.
- Primary vertex [illegible] consolidate [illegible]

Strategy:

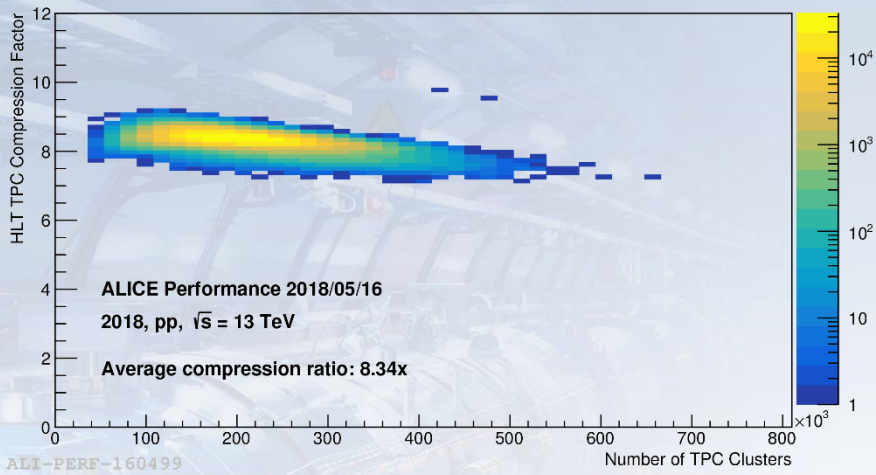
- Start with standalone TPC and ITS tracking.
  - Standalone ITS tracking needed since TPC tracks lack absolute time.
  - ITS tracking uses vertexer as first step.
  - TPC tracking has no vertex constraint, starts with segment tracking in individual TPC sectors, than merges the segments and refits.
- ITS and TPC tracks are matched, fixing the time for the TPC.
- The afterburner propagates unmatched TPC tracks into the ITS and tries to find matching hits of short tracks not found in ITS standalone tracking.
- Tracks are extrapolated outwards into the TRD, once the time is fixed.
  - TRD standalone tracking and matching (like for ITS) is less efficient due to many fake TRD tracklets.
- Optionally, after TRD tracks can be extrapolated to TOF.
- Global refit uses the information from all detectors.
- V0 finding
- In parallel, the TPC compression chain starts after the TPC standalone tracking:
  - Junk clusters are removed, depending on the strategy (see later) this might require extra step for identification of very low p_T junk below 10 MeV/c.
  - Track model (and other steps) reduce the entropy for the final entropy encoding.
  - Final entropy encoding using ANS. Not clear yet whether this will run on GPU efficiently. Alternatively, transport entropy-reduced clusters to host and run entropy encoder there.



TPC Data Compression

- TPC Data compression involves 3 steps:
  1. Entropy reduction (Track model, logarithmic precision, etc.)
  2. Entropy encoding (Huffman, Arithmetic, ANS)
  3. Removal of tracks not used for physics.

- Steps 1 + 2 implemented for Run 2.
  - Current compression factor **8.3x**.



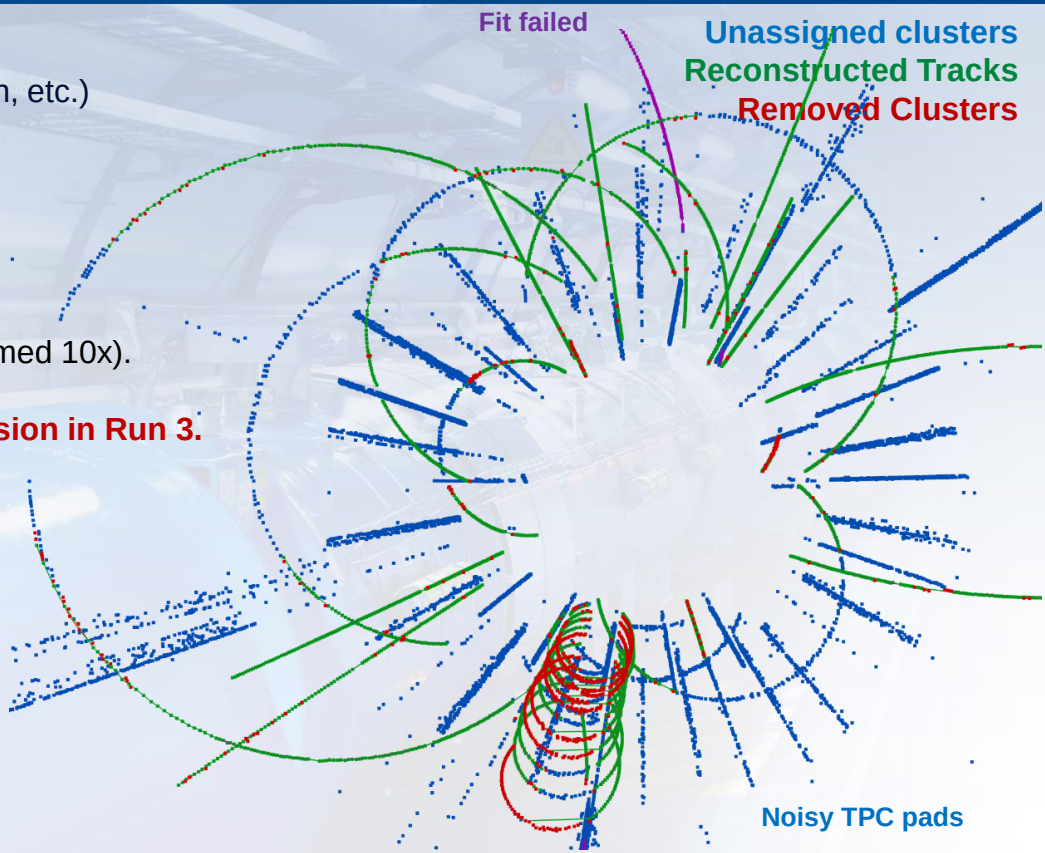
TPC Data Compression

- TPC Data compression involves 3 steps:
  1. Entropy reduction (Track model, logarithmic precision, etc.)
  2. Entropy encoding (Huffman, Arithmetic, ANS)
  3. Removal of tracks not used for physics.

- Steps 1 + 2 implemented for Run 2.
  - Current compression factor **8.3x**.
  - Prototype for Run 3 achieves factor **9.1x** (TDR assumed 10x).

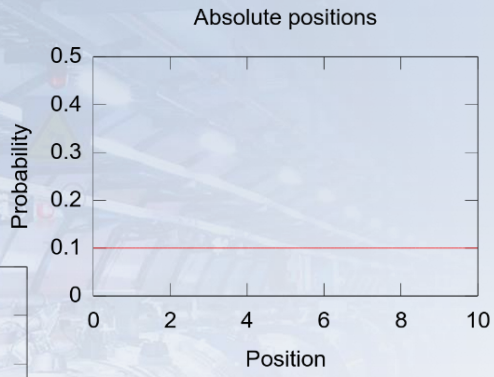
- **Step 3 must close the gap to the required compression in Run 3.**
  - Remove clusters from background / looping tracks.
    - Adjacent to low-p_T track < 50 MeV.
    - Adjacent to secondary leg of low-p_T track < 200 MeV.
    - Adjacent to any track with $\varphi > 70°$ in the fit.
  - Protect clusters of physics tracks.
    - Not Adjacent to any physics-track (except $\varphi > 70°$).

- In addition:
  - Use reconstructed track quantities to reduce entropy.

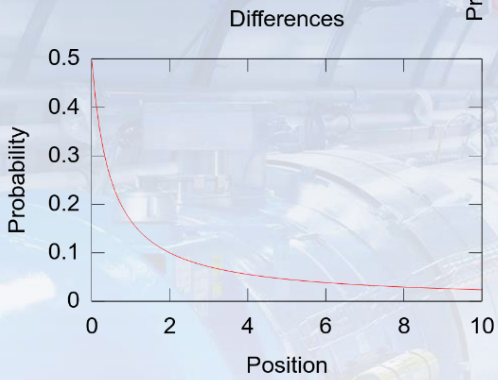


TPC Cluster Entropy Reduction

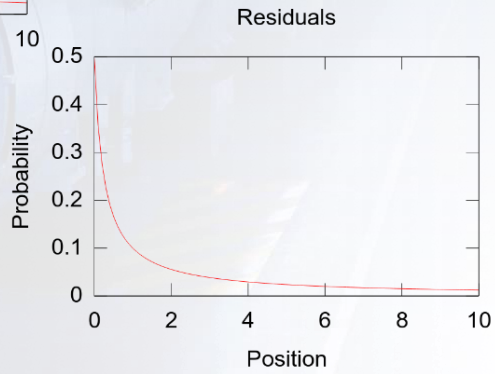
- Cluster Properties stored in integer format, such that 1 bit ~ required resolution.
- Exploit entropy encoding, i.e. some values are more probable than others.
  - Does not work well for absolute positions:
    - All positions have equal probability.



  - Can sort clusters and store only position differences.
    - Order is not important.
    - At high occupancy, all differences should be small.

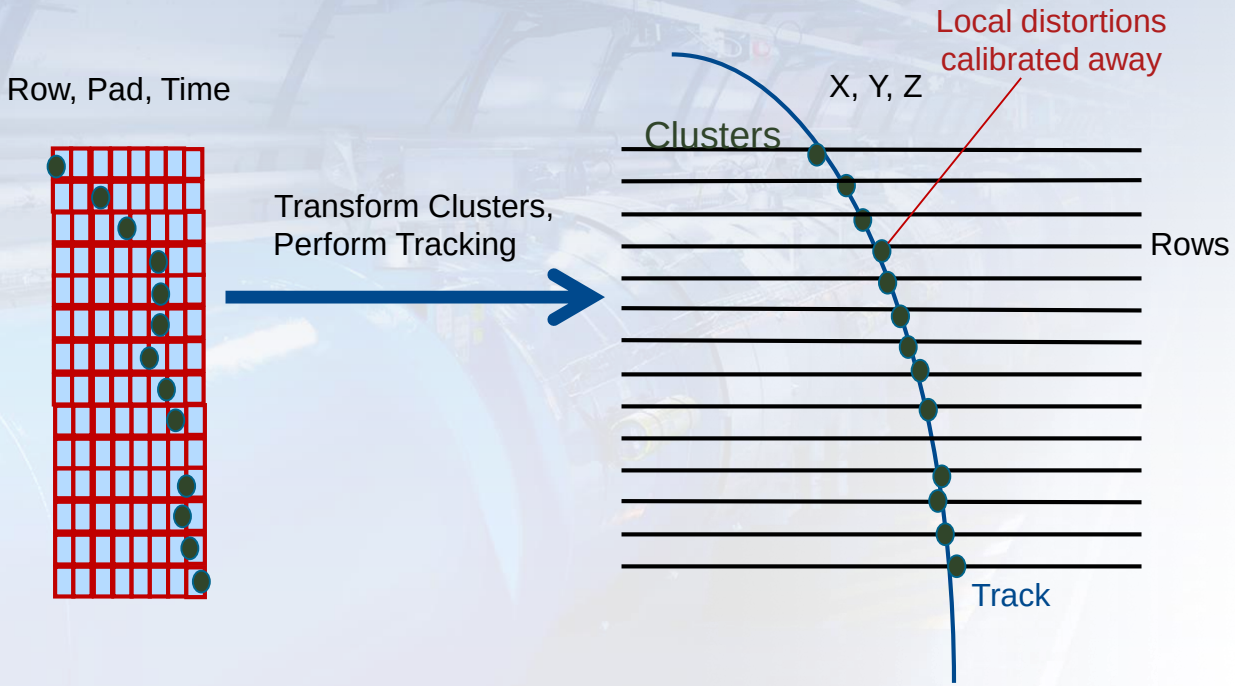


  - With tracking, store only cluster to track residuals
    - Even less entropy for attached clusters.
    - Stick to differences for unattached clusters.
      - Unfortunately, less clusters stored as differences (~50%).
      - → Larger differences (~ factor 2 → 1 bit).
      - → Need 0.5 more bits (1 * 50%) per unattached cluster.
      - Net compression still better.



TPC Cluster Entropy Reduction: Track Model

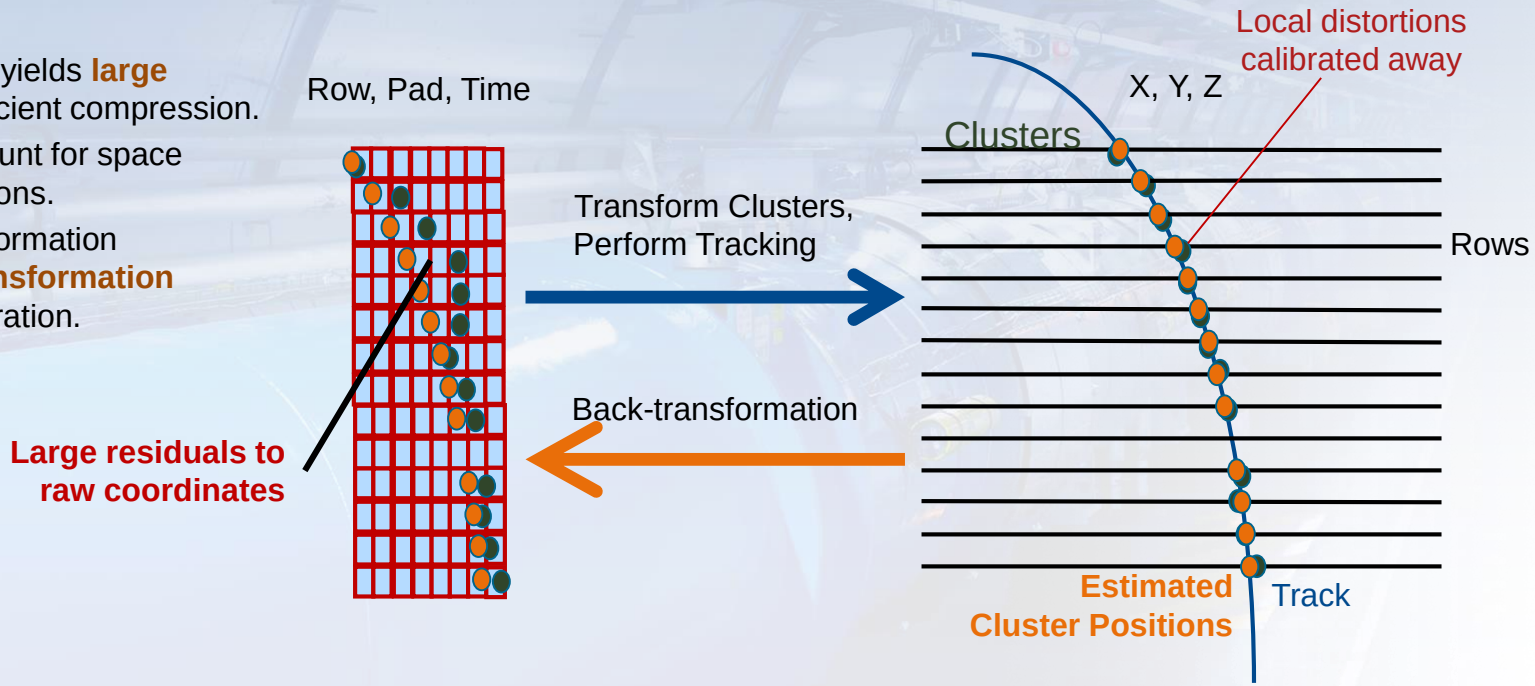
- **Minimize residuals**. (**Smaller entropy → Better Huffman compression.**)



TPC Cluster Entropy Reduction: Track Model

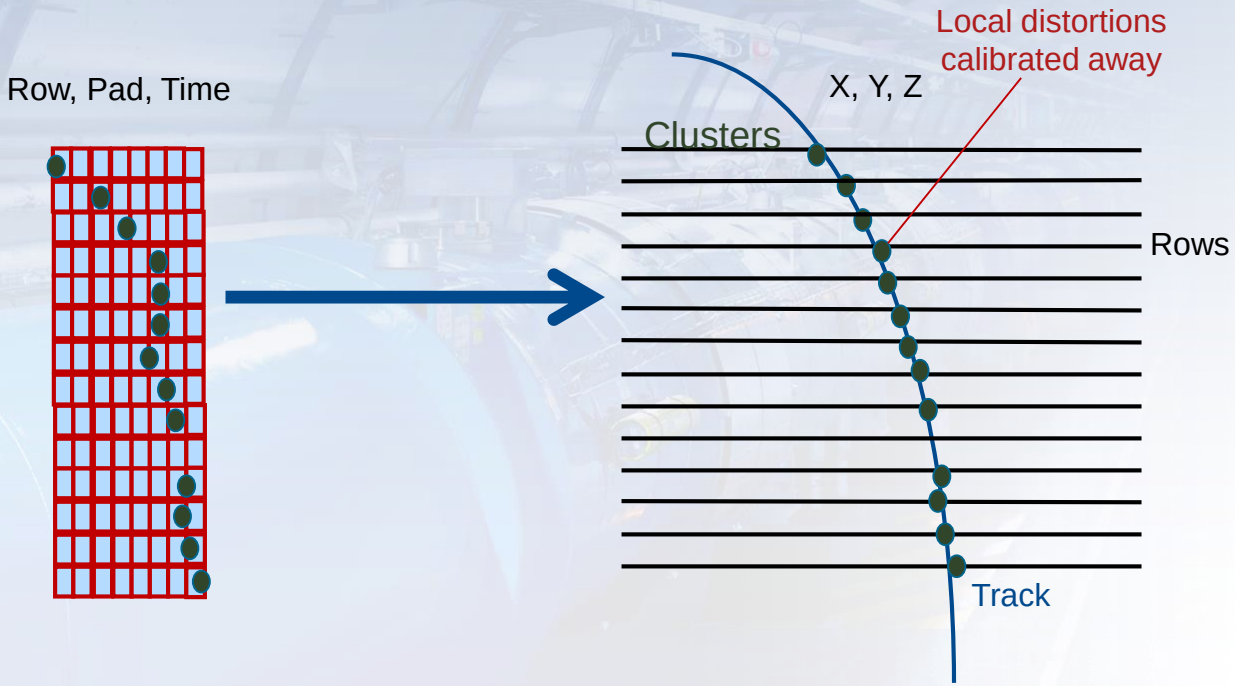
- **Minimize residuals**. (**Smaller entropy → Better Huffman compression.**)
  - **Constraint**: Clusters shall be stored in native TPC coordinates (Row, Pad, Time), independent from calibration.

- **Problems**:
  - Helix prolongation yields **large residuals** → inefficient compression.
    - Does not account for space charge distortions.
  - Linear back-transformation **cannot revert transformation** based on full calibration.



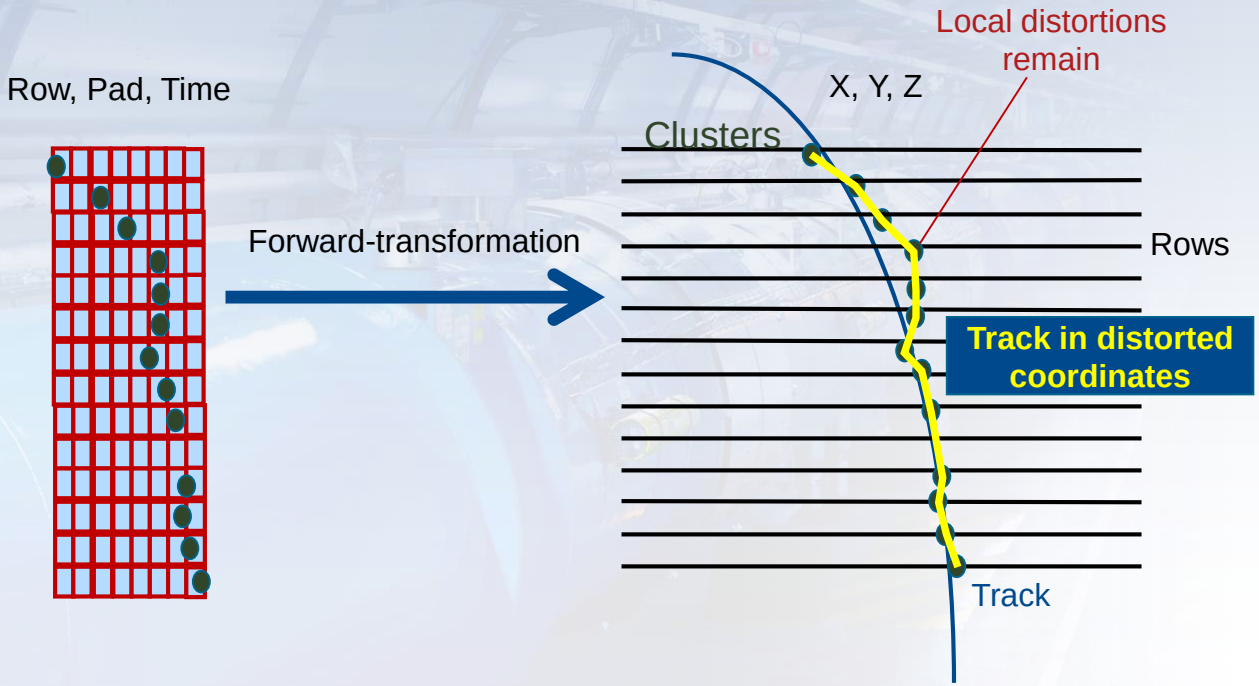
TPC Cluster Entropy Reduction: Track Model

- **Minimize residuals**. (**Smaller entropy → Better Huffman compression.**)
- **Back to start!**



TPC Cluster Entropy Reduction: Track Model

- **Minimize residuals**. (**Smaller entropy → Better Huffman compression.**)
  - Employ **fast, reversible linear approximation**. *(In principle, every transformation works.)*
  - **Refit track in distorted coordinate system**.

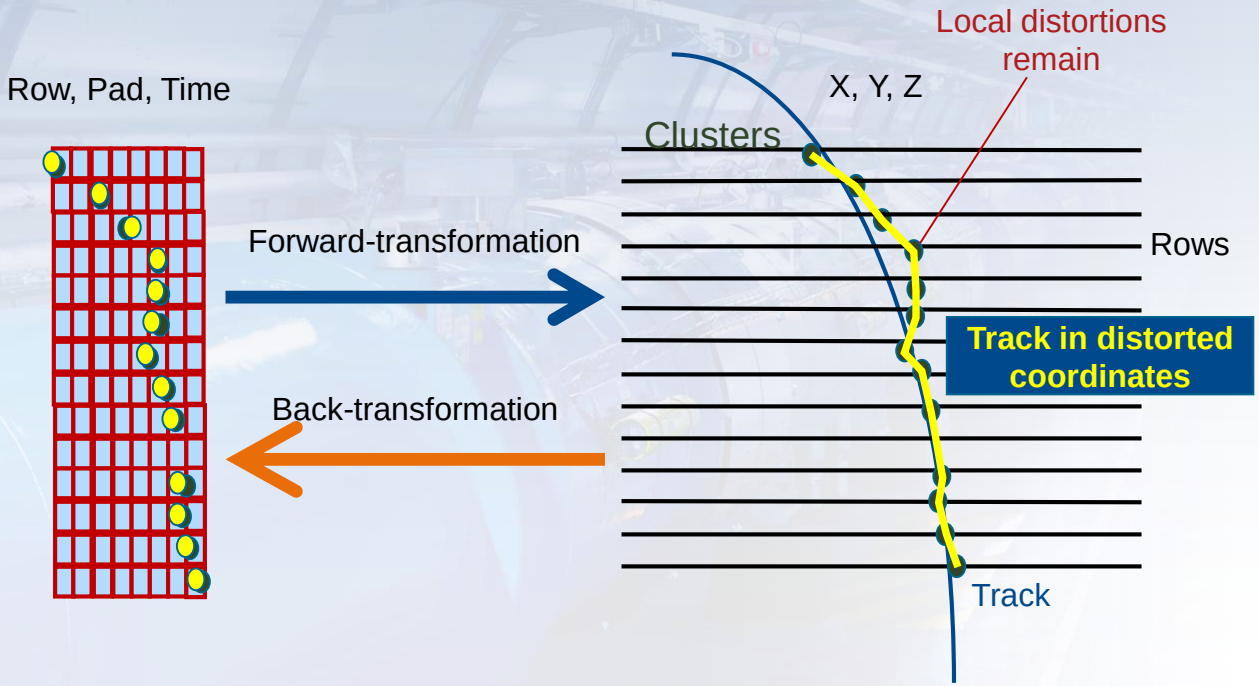


TPC Cluster Entropy Reduction: Track Model

- **Minimize residuals**. (**Smaller entropy → Better Huffman compression.**)
  - Employ **fast, reversible linear approximation**. *(In principle, every transformation works.)*
  - **Refit track in distorted coordinate system**.

- **Store residuals in pad, time.**
  - Currently, storing initial q/p_T, $\sin(\varphi)$, and $\tan(\lambda)$ with low precision.
  - During decompression, perform the same refit with linear transformation.

- **Additional benefit:**
  - Cluster to track association is stored intrinsically.



TPC Cluster Entropy Reduction

1. Can use track information for other properties: e.g. clusters of a track should have similar charge.
   - Can store charge wrt. average track charge.
   - Better use truncated mean / median to compensate fake clusters, could consider track angle (basically dE/dx).

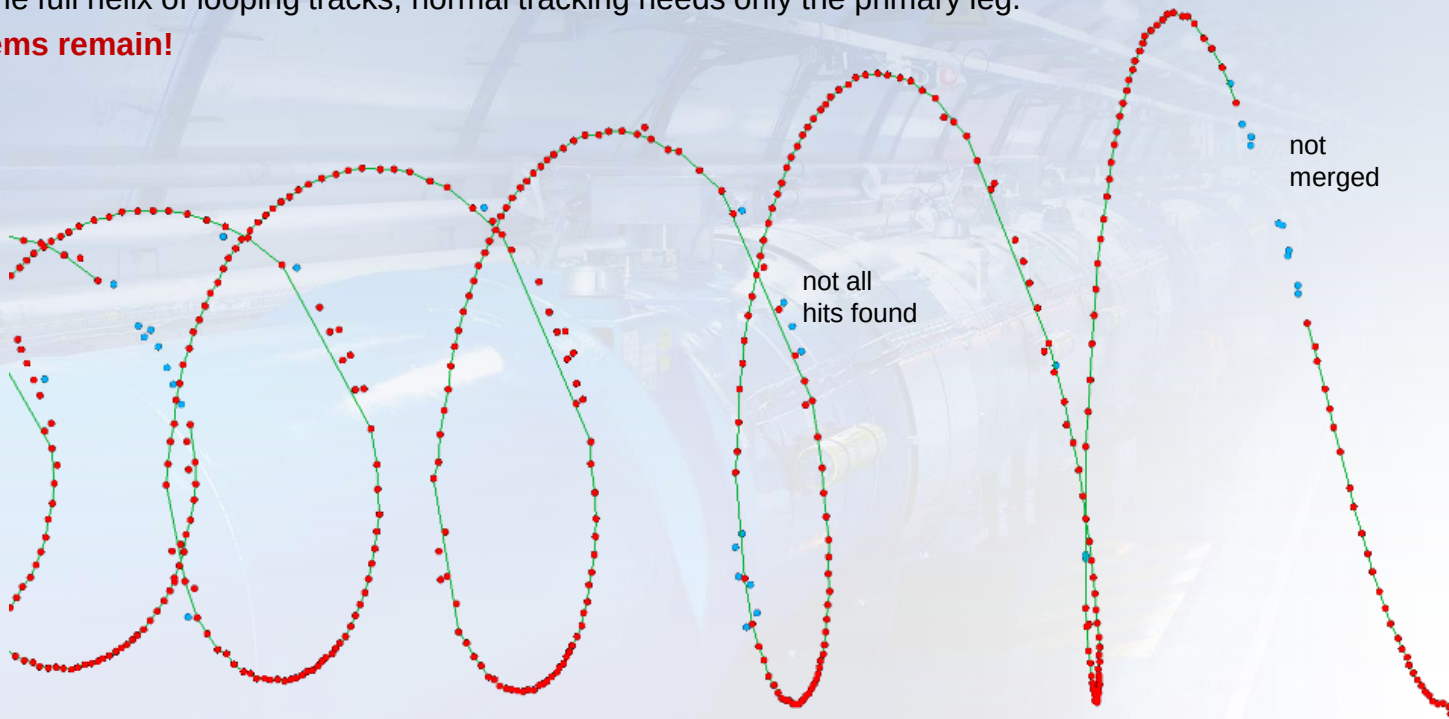
2. No need to store charge / width always with full precision.
   - Need only n significant bits (least significant bits irrelevant for large charge).
   - Basically we need a custom floating point format (no sign, custom size of exponent / mantissa).
   - Instead, we use our integer format and force all insignificant bits to 0 (with correct rounding).
     (insignificant = **n** bits after first non-zero bit: 00110111 → 00111000 for **n** = 3.)
     - Many values are prohibited, entropy coding assigns optimal short representations for allowed values.

3. Unfortunately, the gains of these two strategies do not accumulate directly:
   - **Strategy 1** reduces the numbers in general (and introduces negative numbers), while **strategy 2** yields same-size representation for all values.
   - Might only be able to reduce the **n** of **strategy 2** further in combination.

4. Can do the same for cluster shape / size.

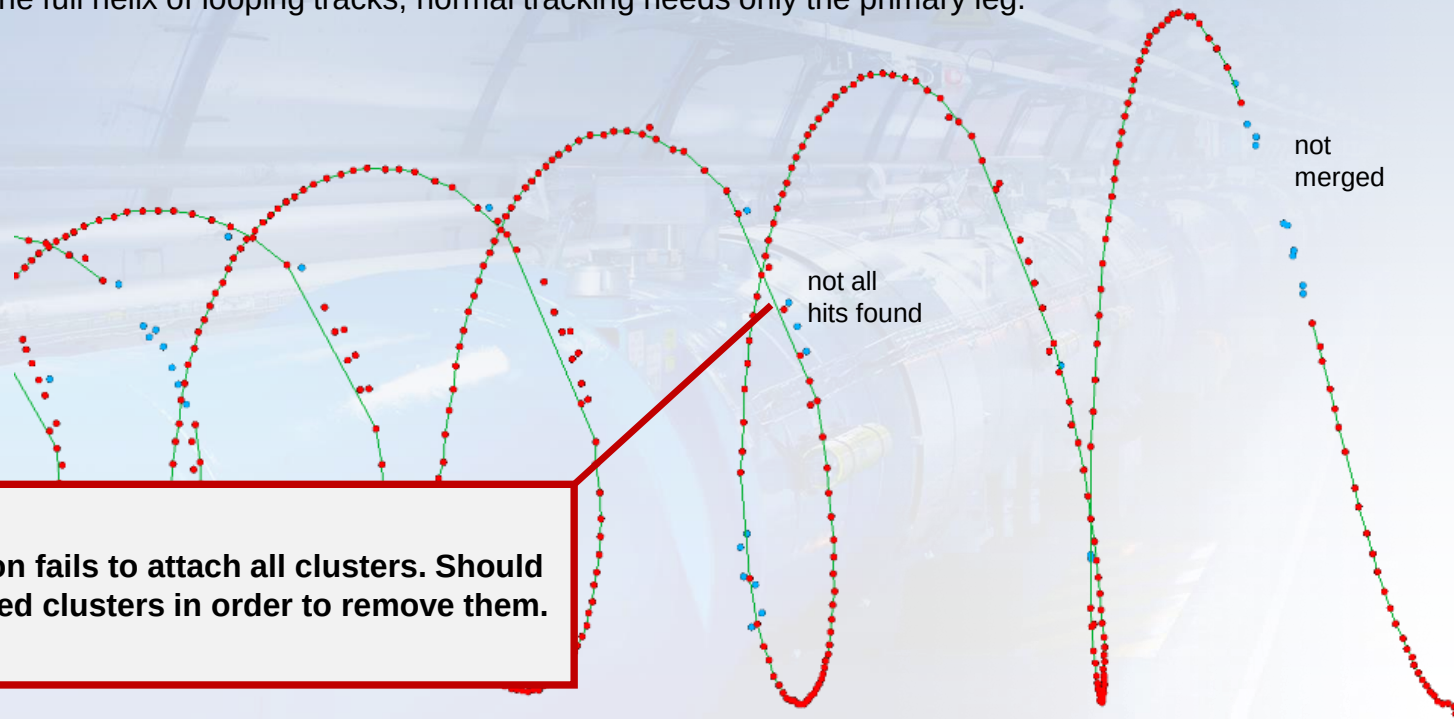
TPC Cluster rejection

- ALICE has implemented several improvements for low-p_T track finding.
- No impact on physics performance.
- Challenge: Needs to find the full helix of looping tracks, normal tracking needs only the primary leg.
- **Currently, several problems remain!**



TPC Cluster rejection

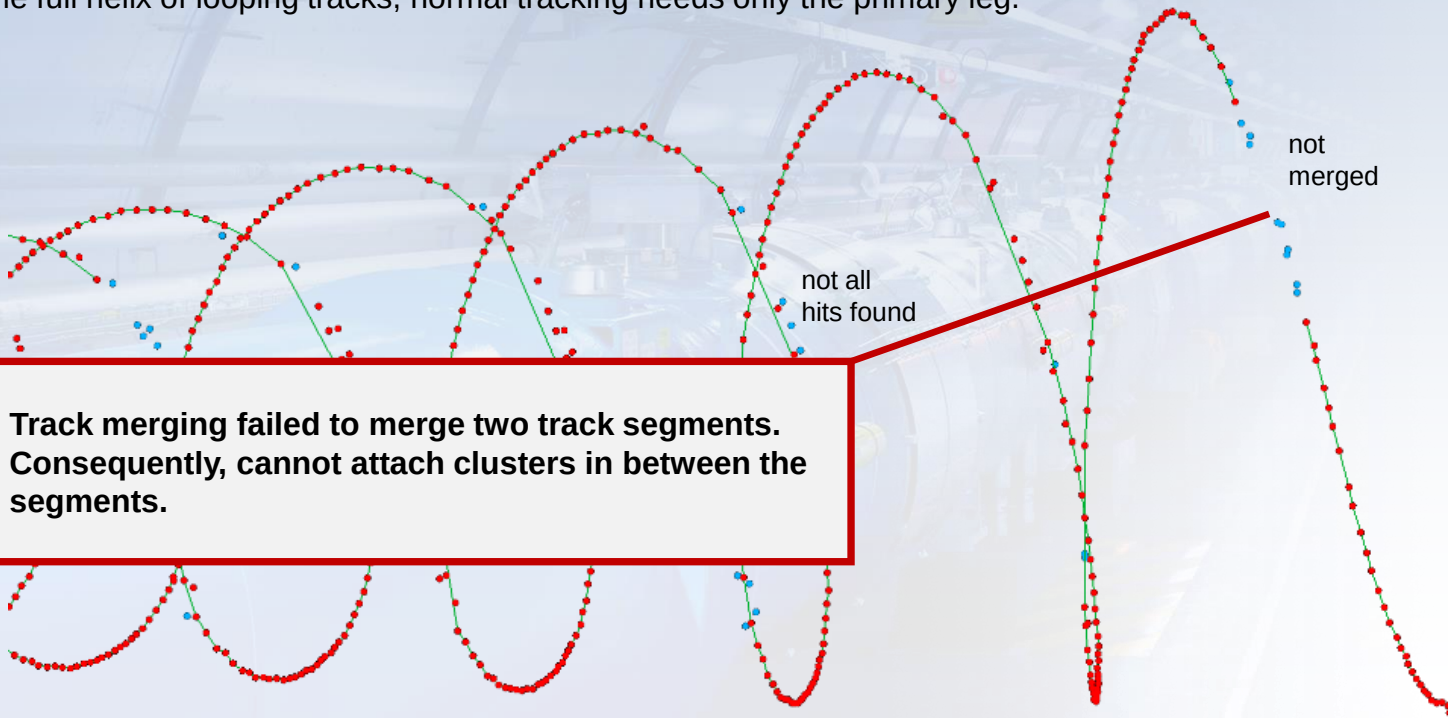
- ALICE has implemented several improvements for low-p_T track finding.
- No impact on physics performance.
- Challenge: Needs to find the full helix of looping tracks, normal tracking needs only the primary leg.



Extra- / interpolation fails to attach all clusters. Should identify all unneeded clusters in order to remove them.

TPC Cluster rejection

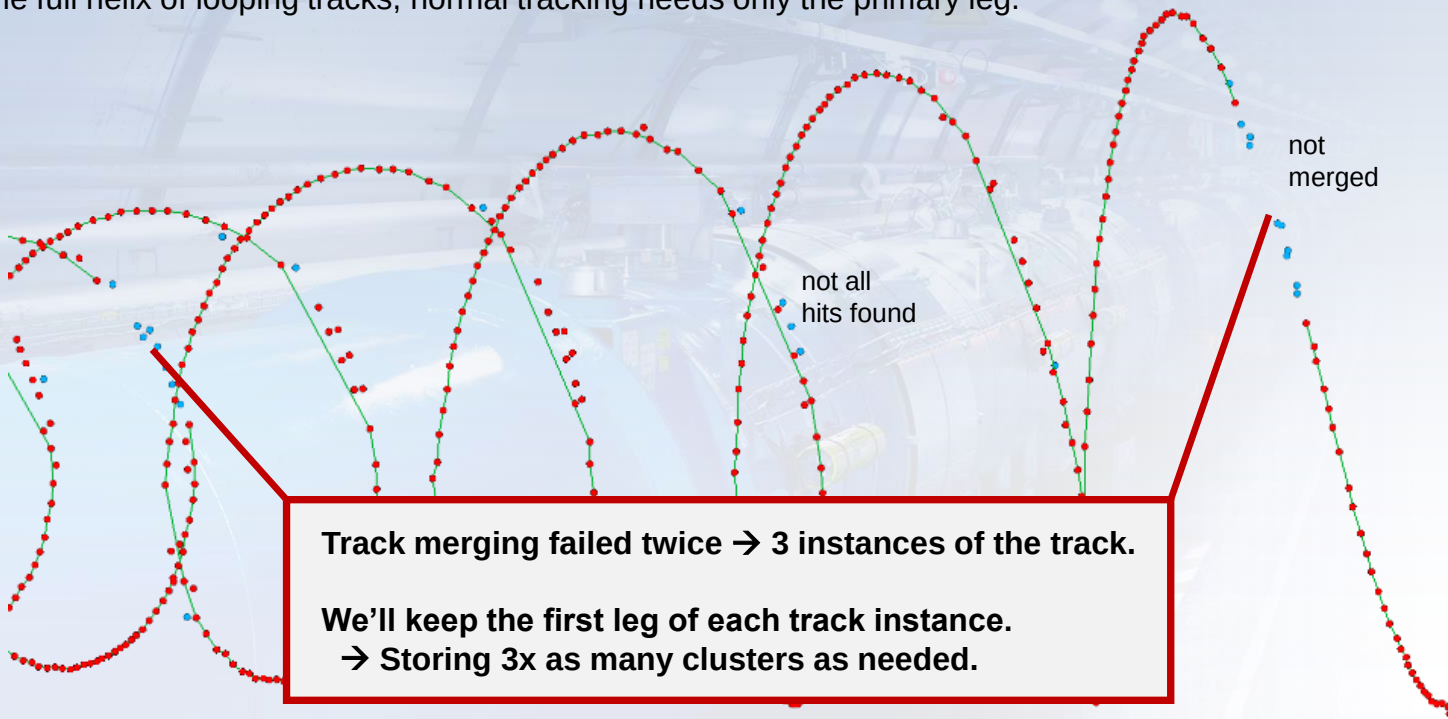
- ALICE has implemented several improvements for low-p_T track finding.
- No impact on physics performance.
- Challenge: Needs to find the full helix of looping tracks, normal tracking needs only the primary leg.



Track merging failed to merge two track segments. Consequently, cannot attach clusters in between the segments.

TPC Cluster rejection

- ALICE has implemented several improvements for low-p_T track finding.
- No impact on physics performance.
- Challenge: Needs to find the full helix of looping tracks, normal tracking needs only the primary leg.

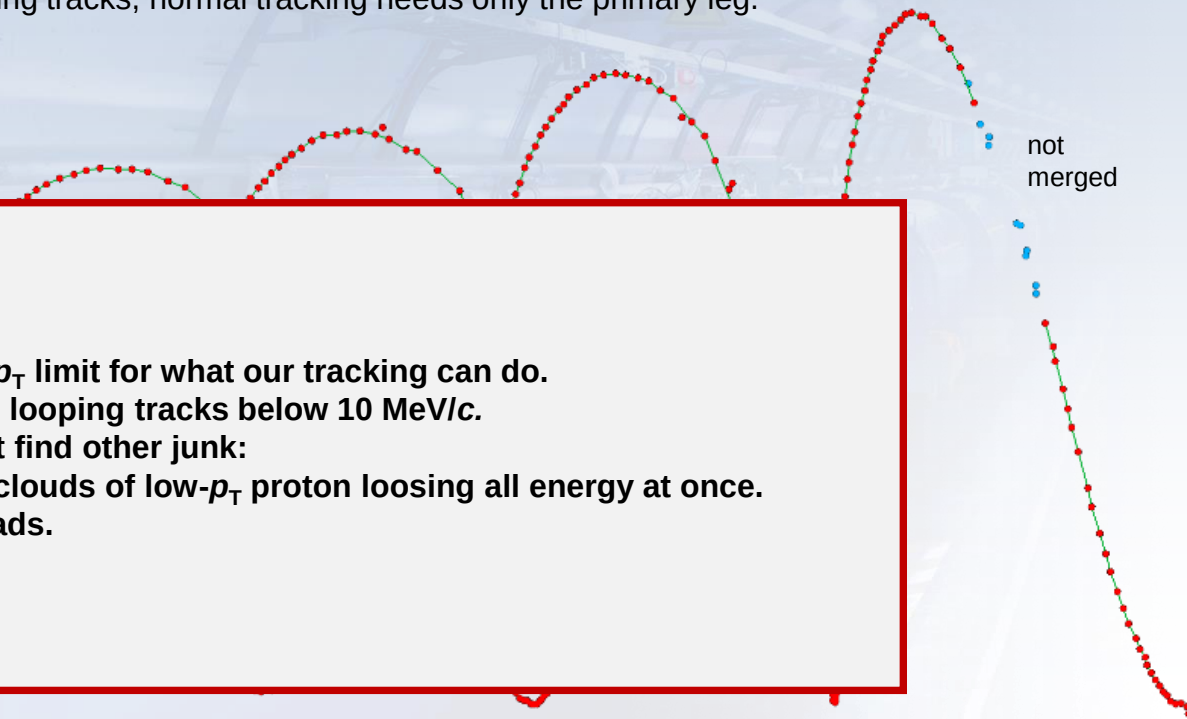


Track merging failed twice → 3 instances of the track.

We'll keep the first leg of each track instance.
→ Storing 3x as many clusters as needed.

TPC Cluster rejection

- ALICE has implemented several improvements for low-p_T track finding.
- No impact on physics performance.
- Challenge: Needs to find the full helix of looping tracks, normal tracking needs only the primary leg.



There is a lower p_T limit for what our tracking can do.

- **We do not find looping tracks below 10 MeV/*c*.**
- **We also do not find other junk:**
  - **Charge clouds of low-p_T proton loosing all energy at once.**
  - **Noisy pads.**

ALICE Run 3 Data Rates

- **Data rates of ALICE detectors with large data contribution.**
- **All rates in GB/s during 50 kHz Pb-Pb (peak rates).**
- **For reference: Data rates assumed in TDR: 88 (66.5 – 105.2).**

- **TPC Biggest contributor to data rate.**

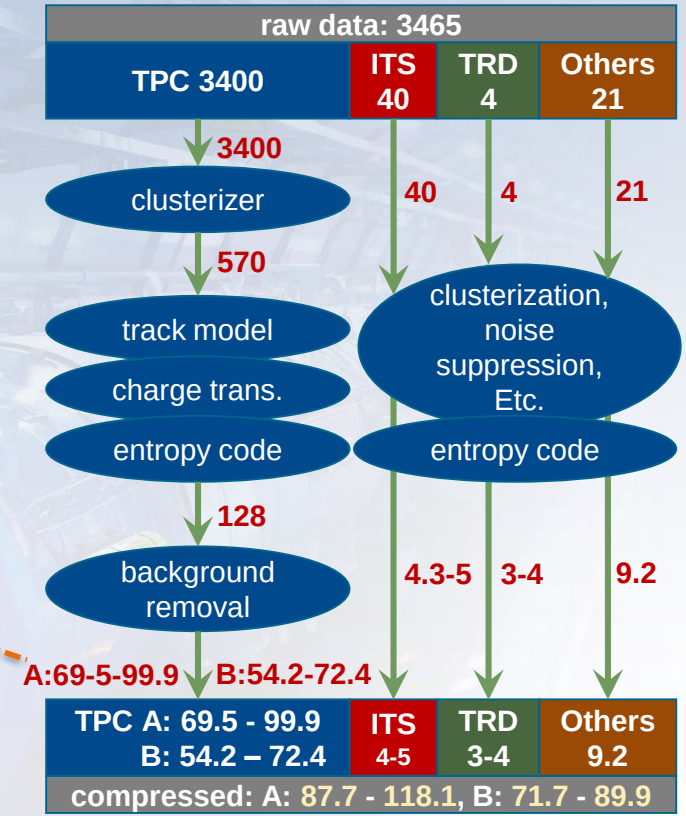
→ **TPC compression most critical.**

- Assumed factor 20x in TDR.
  (Factors badly comparable, as raw format changed → compare rates.)

TPC data rejection alternatives

A. Reject only clusters of identified background / tracks (loopers).
Rejects: 12.5% - 39.1%

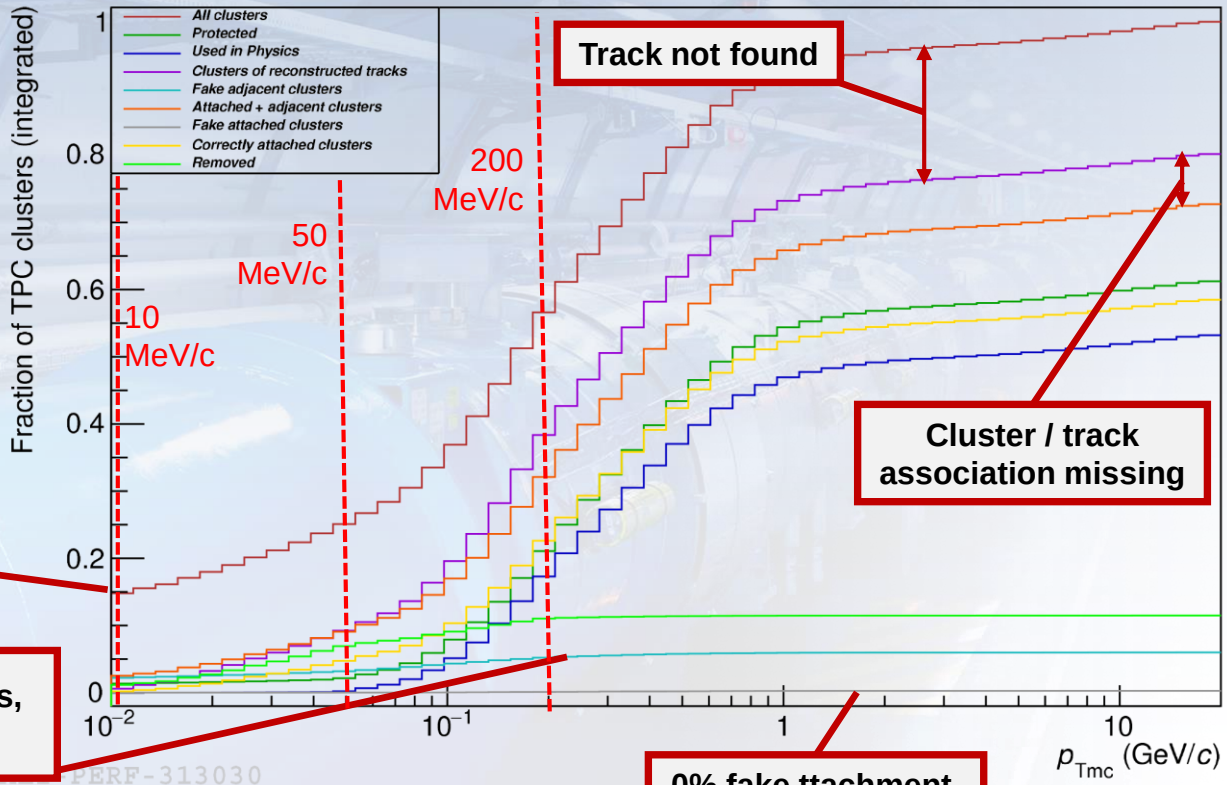
B. Keep only clusters attached or in proximity of identified signal tracks.
Rejects: 37.3% - 52.5%



TPC Cluster attachment ratios

- Blue: attached clusters (used in fit)
- Purple: possible adjacent clusters (all clusters of reconstructed tracks)
- Orange: Identified adjacent clusters

(**attached**: used in track fit / **adjacent**: in proximity but not attached)



13% of clusters inaccessible with tracking (< 10 MeV)

High fake adjacent rate for low-p_T tracks, adjacent to random higher-p_T track.

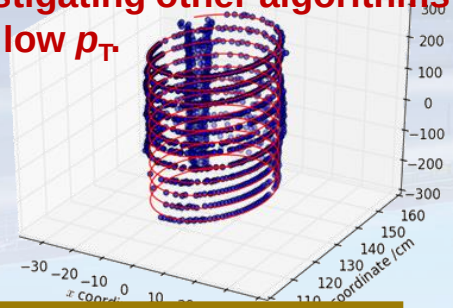
0% fake ttachment.

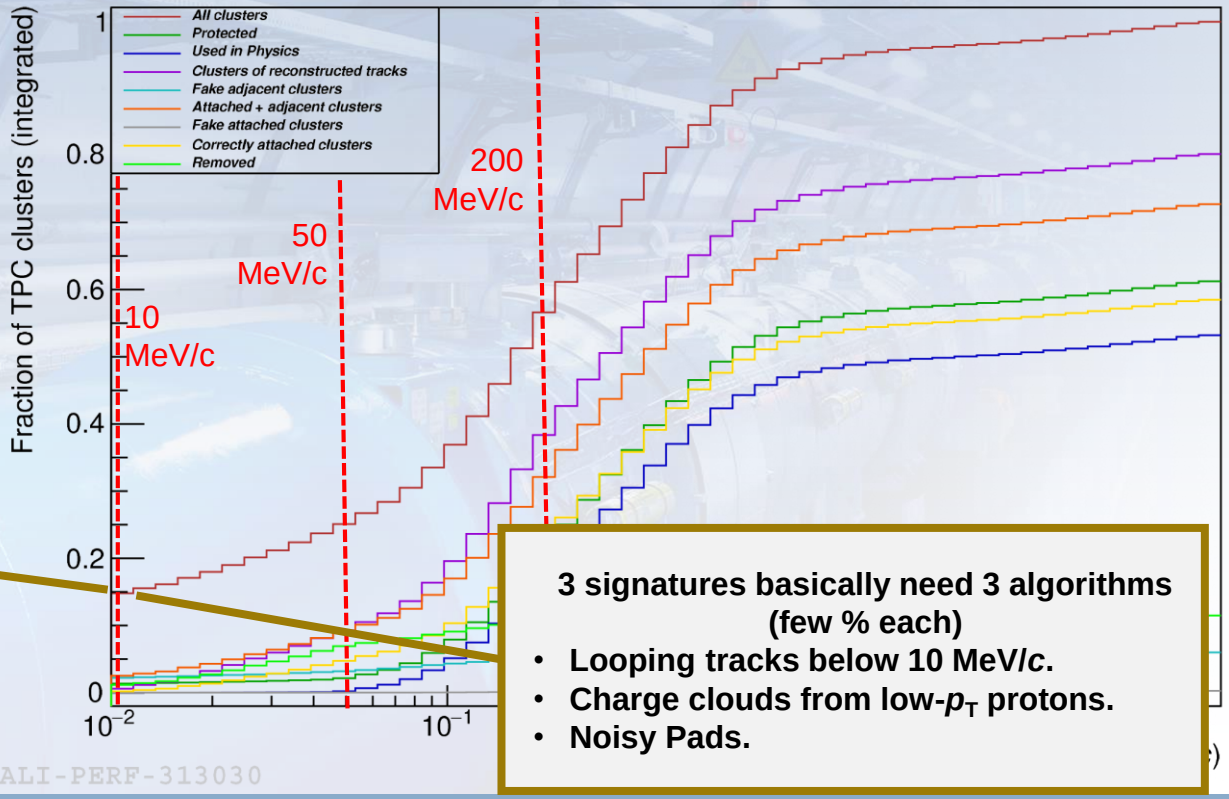
TPC Cluster attachment ratios

- Blue: attached clusters (used in fit)
- Purple: possible adjacent clusters (all clusters of reconstructed tracks)
- Orange: Identified adjacent clusters

(**attached**: used in track fit / **adjacent**: in proximity but not attached)

Investigating other algorithms for very low p_T.





13% of clusters inaccessible with tracking (< 10 MeV)

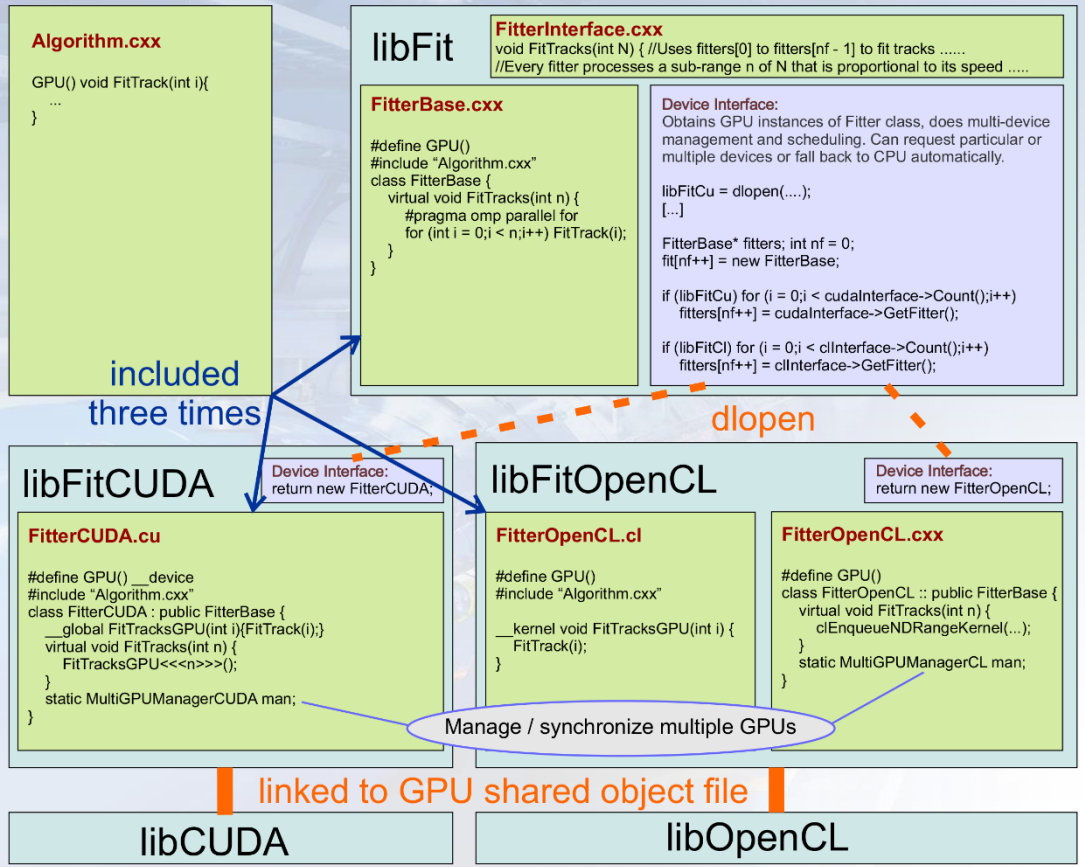
3 signatures basically need 3 algorithms (few % each)

- **Looping tracks below 10 MeV/*c*.**
- **Charge clouds from low-p_T protons.**
- **Noisy Pads.**

Detailed layout of GPU software (track fit example)

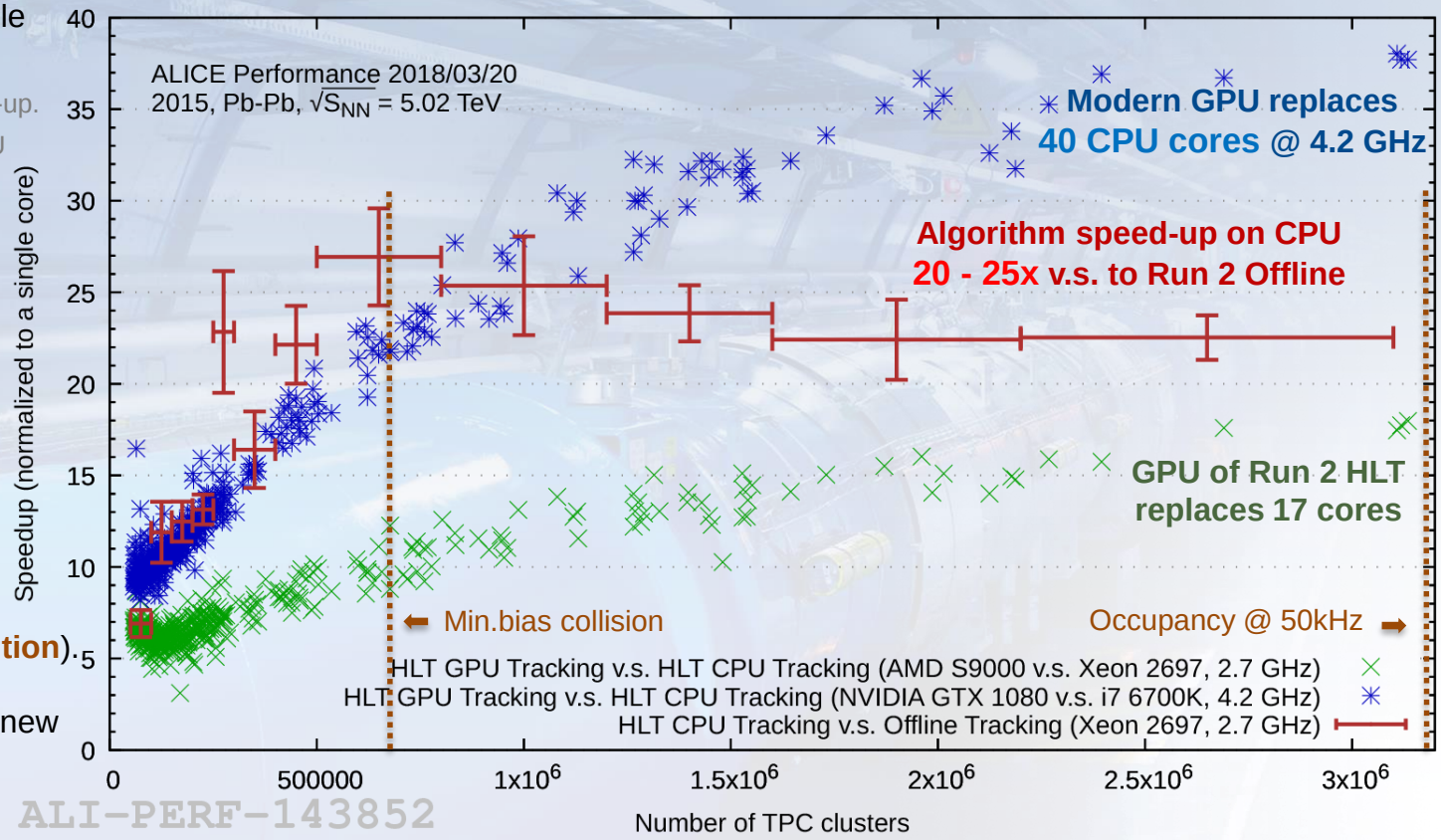
- **Detailed example of the track fit code.**
  - Majority of the code is in Algorithm.cxx, which is shared between CPU and all GPU versions.
  - libFit can be loaded on all compute nodes (no dependency on CUDA / OpenCL).
    - The cuda and OpenCL tracking libraries (libFitCUDA and libFitOpenCL) can be loaded when the respective runtime (libCUDA or libOpenCL) is present.

- **Common source code for CPU / GPU.**
  - Supporting CUDA
  - HIP (AMD)
  - OpenCL (2.2 or clang 9)



TPC Tracking performance

- Speed-up normalized to single CPU core.
  - Red curve: algorithm speed-up.
  - Other curves: GPU v.s. CPU speed-up corrected for CPU resources.
    - How many cores does the GPU replace.
- Significant gain with newer GPU (blue v.s. green).
- GPU with Run 3 algorithm replaces **> 800 CPU cores** Running Run 2 algorithm. (**blue * red**). (**at same efficiency / resolution**).
- We see ~**30%** speedup with new GPU generation (RTX 2080 v.s. GTX 1080)



Summary

- ALICE will take **50 kHz of minimum bias Pb-Pb** data in Run 3.
- There will be **no triggers** but data is compressed online in software.
- High data compression factors require online reconstruction, in particular TPC tracking.
- Full tracking for the TPC in the synchronous phase, tracking for ITS and TRD for few percent of the events.
- Full reconstruction with final calibration in the asynchronous phase.
- The majority of the synchronous phase will run on GPUs, asynchronous phase can run on GPU but also in the GRID.
- In an optimistic scenario, we can offload almost full barrel tracking to GPU.
- TPC Reconstruction more challenging than today due to space charge distortions.
- TPC Data compression still big issue:
  - Entropy compression factor of **9.1x** (10% short wrt. TDR).
  - **Cluster rejection turns out to be difficult.**
    - Random high-p_T tracks fake-protect junk clusters.
    - Incomplete track merging reduces the number of looping legs to be removed.
    - Still significant fraction of unattached clusters.
    - 13% of clusters not accessible by tracking (very low p_T, charge cloud from low p_T protons, noisy pads).
  - **Strategy B** could increase rejection ratio at the risk of loosing physics tracks in case of issue with calibration.
- Total data rate still in agreement with the TDR since we can save at other places.
- GPU code implemented in shared code also for CPU, algorithm speed-up ca **20x**, 1 GPU replaces ca **40 CPU cores**.