Charge Light Matching Implementation to SPINE

Yuxuan Wang (Billy)

PI: Ryan Patterson

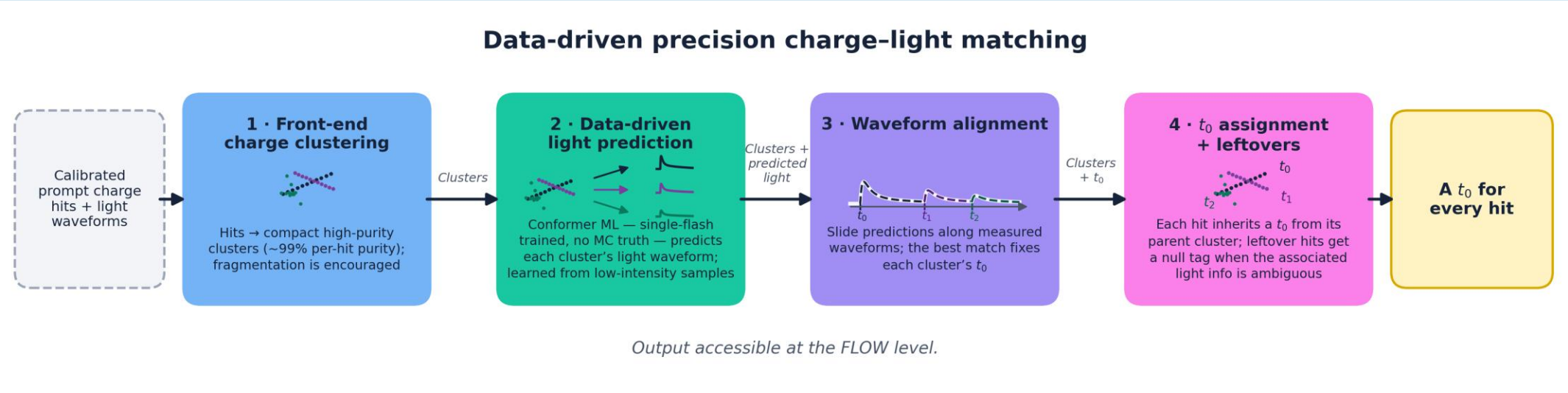
ML weekly meeting

Outline

- Background intro about the clustering stage of charge-light matching (CL matching)
- Comparing clustering methods in SPINE and CL matching
- Conclusion and how time info may come into SPINE

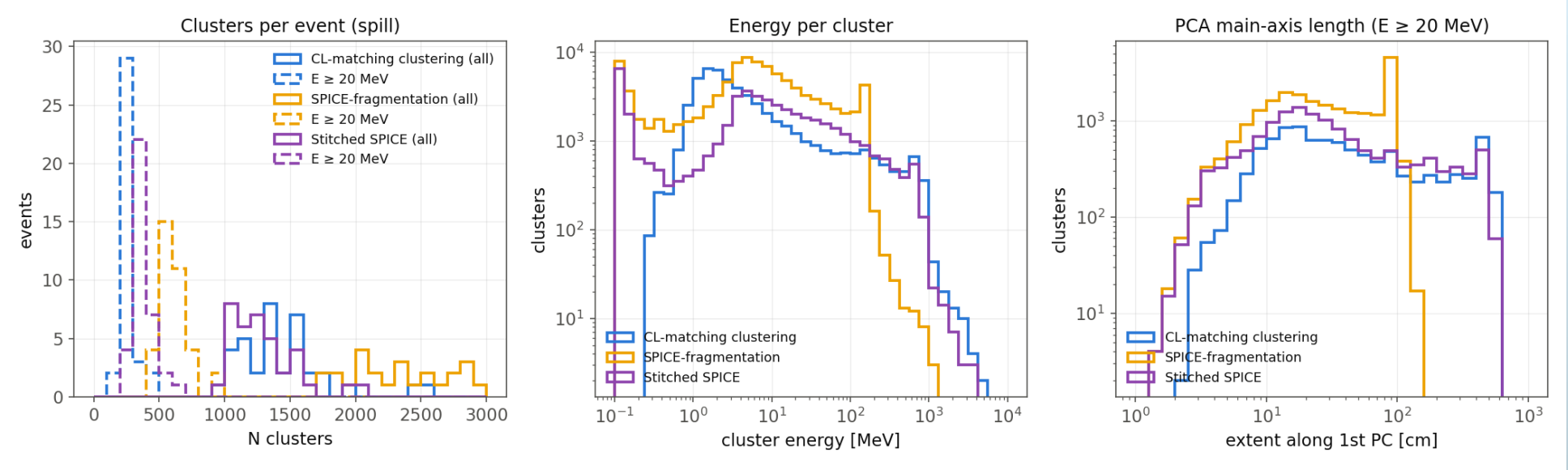
Background introduction

- Charge light matching relies on a front-end clustering to perform Ransac tracking + DBSCAN SCAN + cross-TPC stitching (no MC_truth involved)
  light readout has coarse spatial resolution → group charge hits until each cluster emits enough light to match reliably — cluster scale set by the *light* resolution, not the charge
- Spine have something similar, the SPICE fragmentation.
- Question: We can use SPICE directly, the workflow could be much more elegant to avoid "fragmenting two times"?



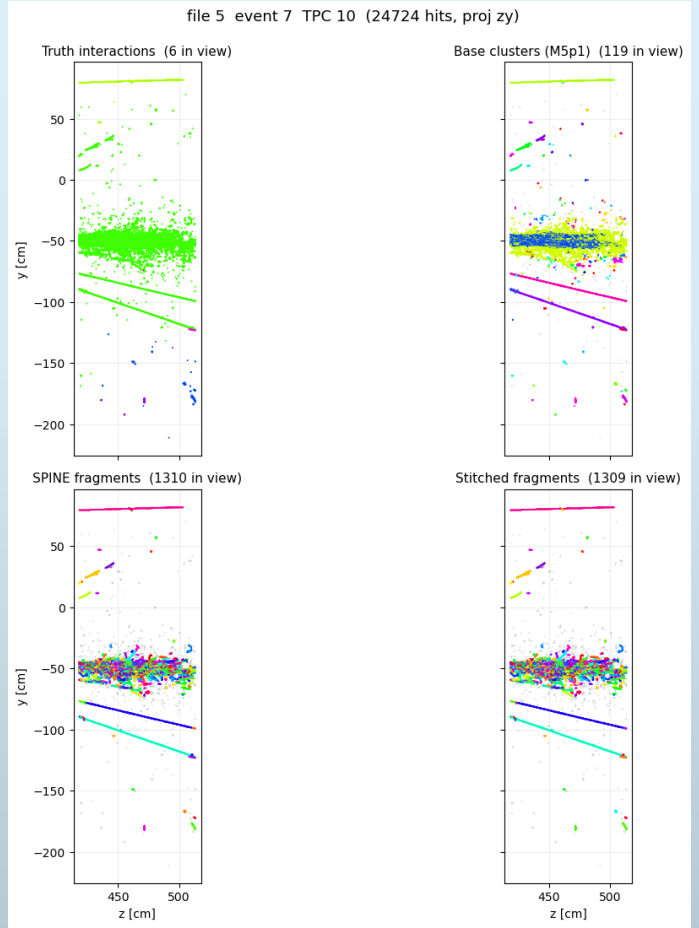
Comparing SPINE clusters vs. charge-light matching

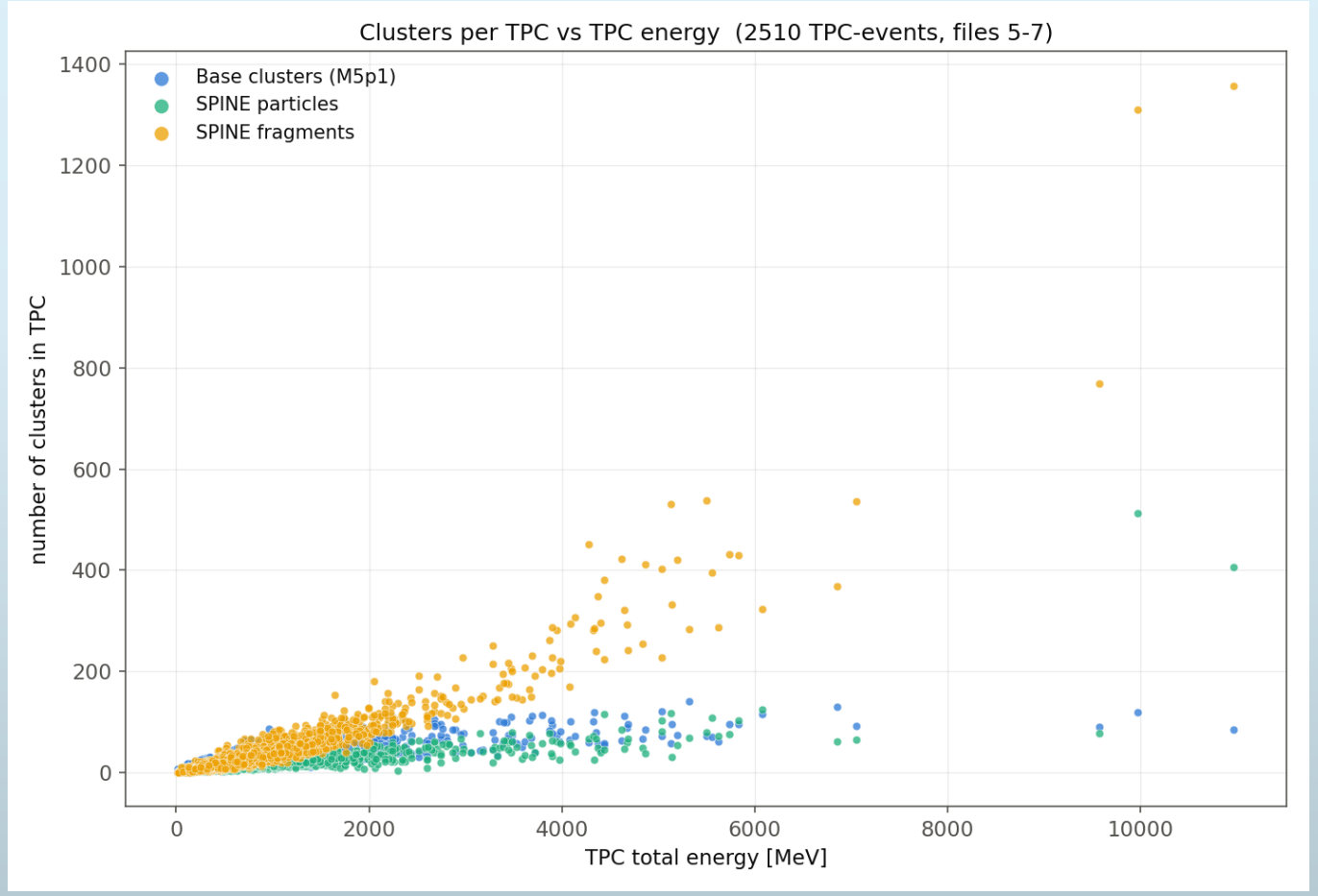
- Comparing the charge-light matching clusters vs. SPINE fragments/particles
- Way too many number of fragments in SPICE than in the current CL matching front-end
  → Need some level of stitching
- Note: 20.3% of the SPINE fragments have <=2 hits (0.3% of energy)



N fragments scales heavily with showering

- Example showering TPC: SPINE fragments produces an overwhelmingly lot of clusters

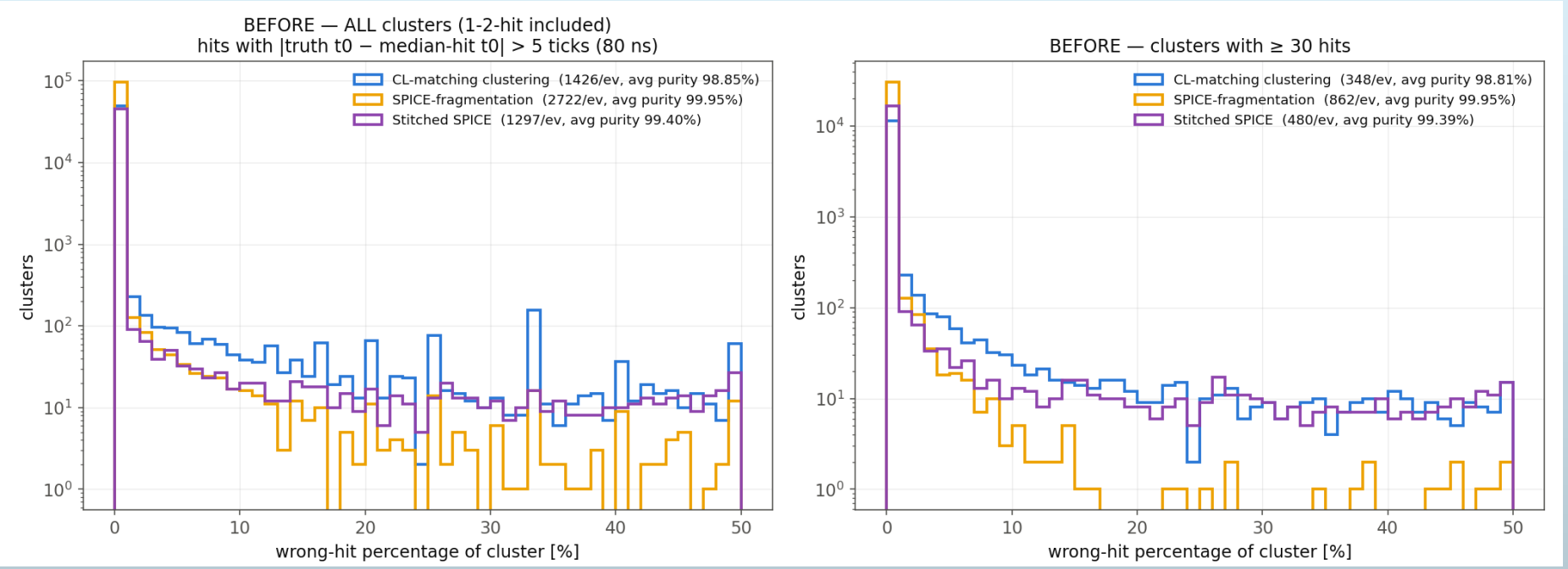




Purity

- Comparing the charge-light matching clusters vs. SPINE fragments/particles

	CLM front-end	SPICE-frag	Stitched SPICE-frag
Average Number of clusters / event	1427	2722	1297
avg energy-weighted purity	98.85%	99.95%	99.40%
Proportion of energy that are clustered	98.6%	93.9%	98.3%
total wrong hits	0.85%	0.05%	0.57%



Stitching needed

- SPICE is geared toward never merging unless it is ~100% confident merging is right → output extremely pure (99.95%) but tiny
  - median ~15 hits / 6 MeV, hard cutoff at the module boundary (~100 cm, ⩽200 MeV), only 0.9% of clusters span TPCs; ~2700 objects per spill
- CL-matching clustering follows the same purity-first ethic, but with
  1. Some level of lower bound (<3 hits → not clustered) to guarantee that enough light is produced by this cluster to make sure there is incident photon.
  2. Deliberately big shapes — cross-TPC tracks up to ~6 m, 33% multi-TPC — so the obvious objects match directly to light with info from multiple TPC's light detector

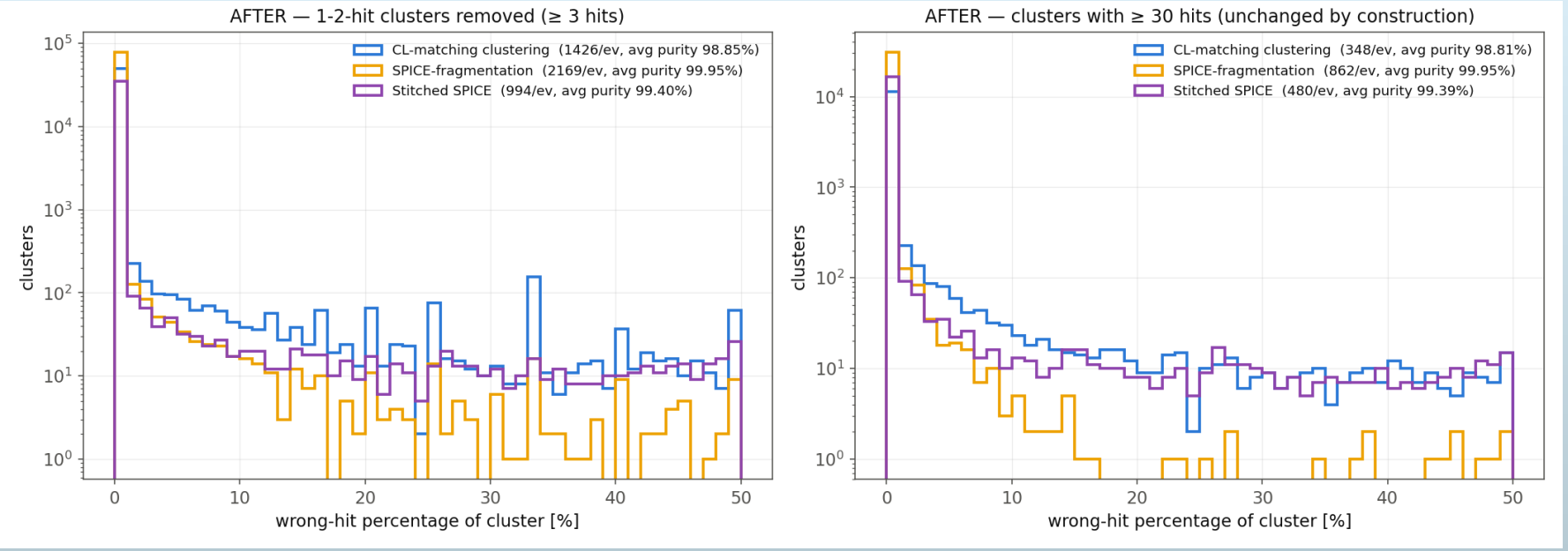
Stitching needed

- SPICE is geared toward never merging unless it is ~100% confident merging is right → output extremely pure (99.95%) but tiny
  - median ~15 hits / 6 MeV, hard cutoff at the module boundary (~100 cm, ⩽200 MeV), only 0.9% of clusters span TPCs; ~2700 objects per spill
- CL-matching clustering follows the same purity-first ethic, but with
  1. Some level of lower bound (<3 hits → not clustered) to guarantee that enough light is produced by this cluster to make sure there is incident photon.
  2. Deliberately big shapes — cross-TPC tracks up to ~6 m, 33% multi-TPC — so the obvious objects match directly to light with info from multiple TPC's light detector
- What if we tried to stitch SPICE a little?
  1. tracks — collinear chaining: join fragment ends across module boundaries
  2. showers — density-peak aggregation (per TPC): fragment centroids weighted by energy; local density peaks = shower cores, every other fragment attaches to its nearest-higher-density neighbor
  3. left-over low energy hits: perform a left-over DBSCan for the low energy hits and others

Purity

- After stitching, the Avg Purity are similar between Stitched SPICE-frag and CLM front-end

	CLM front-end	SPICE-frag	Stitched SPICE-frag
Average Number of clusters / event	1427	2722	1297
avg energy-weighted purity	98.85%	99.95%	99.40%
Avg purity	99.36%	99.93%	99.64%
Avg Purity excluding clusters with <=2 hits	99.36%	99.92%	99.53%
Proportion of energy that are clustered	98.6%	93.9%	98.3%



Implementing the stitched SPICE

- If we directly replace the current front-end clustering with SPINE fragments (basically SPICE output), would the performance of charge-light matching degrade too much?
- Note: in the future release version, we are thinking of allowing spatially ambiguous hits remain without any t_0 assignment since their t_0 can be assigned from whatever spatial decisions are made by later

metric	Front-end clustering	SPINE fragments + stitcher
energy efficiency	95.10%	94.15%
hit efficiency	95.37%	94.51%
backbone eff / energy share	98.26% / 70.3%	97.48% / 76.3%
non-shower TPC total eff	95.81%	95.13%
wall time/(2 events)	320 s	349 s

How to put into SPINE?

- The possibility of using Stitched- SPICE to replace front-end exists, but with weaker performance.
- Routes to implement into SPINE:
  1. Using Stitched – SPICE as a front-end, and run charge-light matching inside SPINE. For others who wants this info at flow level, just fill the value back to the flow files after the run.
  2. Keep the current charge-light matching, but integrate it into the SPINE after the SPICE stage by assigning each spice fragments a most probable reconstructed time (potentially just split the cluster if needed) and clusterid information, then proceed to further stages with this new information added

How to put into SPINE?

- The possibility of using Stitched- SPICE to replace front-end exists, but with weaker performance.
- **Using Stitched – SPICE**
  - More elegant, avoid two-times clustering
  - avoid high purity fragments being contaminated by charge-light clustering
- **Keep the current front-end clustering**
  - Better performance and easier to upgrade (designed for CL-matching)
  - Potentially still holds for purity if applied after the fragmentation stage
  - Keeps consistency as the entire algorithm is not supervised by MC_truth

Potential concerns in separating SPICE clusters

- One thing that could be done is that we ask charge-light matching to split a SPICE cluster into two whenever a SPICE fragment contains conflicting t0 assignment (or even conflicting charge-light matching clustering ids) from hits.
- Braking fragments to insist unique cluster id per fragment is likely the optimal way to carry out all the information from charge-light matching. This is especially important:
  - Two fragments with the same t0 but also the same clusterID, it knew that this might not be highly reliable as it could be just the charge-light matching incorrectly clustered things together at the front-end.
  - If t0 is the same but clusterID is different, than the association shall be considered strong.

splitting rule	extra pieces/event	fragments after	increase
Original fragmentation	0	2722	0
t0 conflict only (>5 ticks)	+32.4	2,746	**+1.19%**
any CLM cluster-id conflict	**+362.2**	**3,076**	**+13.35%**

Conclusion

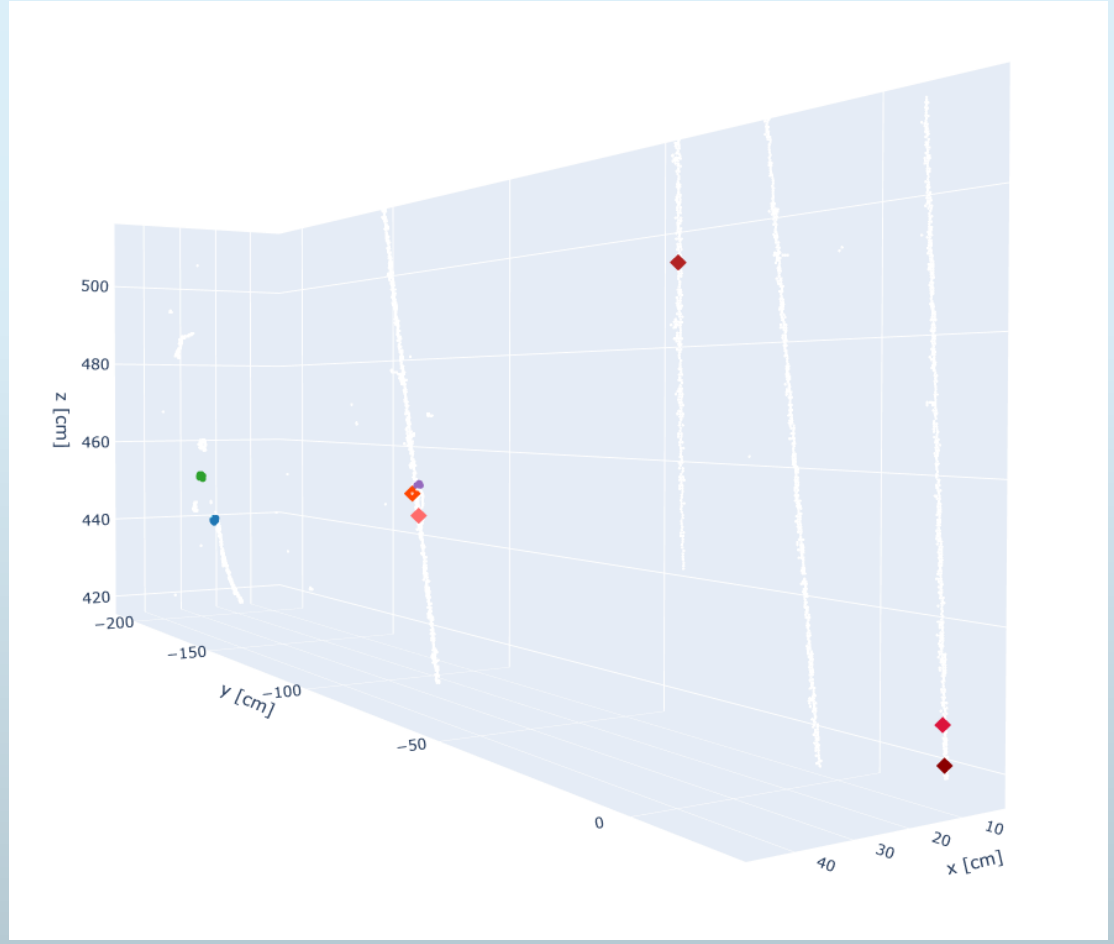
- The possibility of using Stitched- SPICE to replace front-end exists, but with weaker performance.
- **Using Stitched – SPICE**
  - More elegant, avoid two-times clustering
  - avoid high purity fragments being contaminated by charge-light clustering
- **Keep the current front-end clustering**
  - Better performance and easier to upgrade (designed for CL-matching)
  - Potentially still holds for purity if applied after the fragmentation stage
  - Keeps consistency as the entire algorithm is not supervised by MC_truth

  Figure out someway to implement charge-light matching without destroying the high purity nature of SPICE

Appendix: how the stitching happens

Looking closely at SPINE things

- A lot of these very small (ultra-low energy clusters = <1MeV of energy) are just next to a track
- Diamond represent clusters <1MeV
- Colored are clusters < 5MeV



display4_f0000005_ev2_tpc31.html

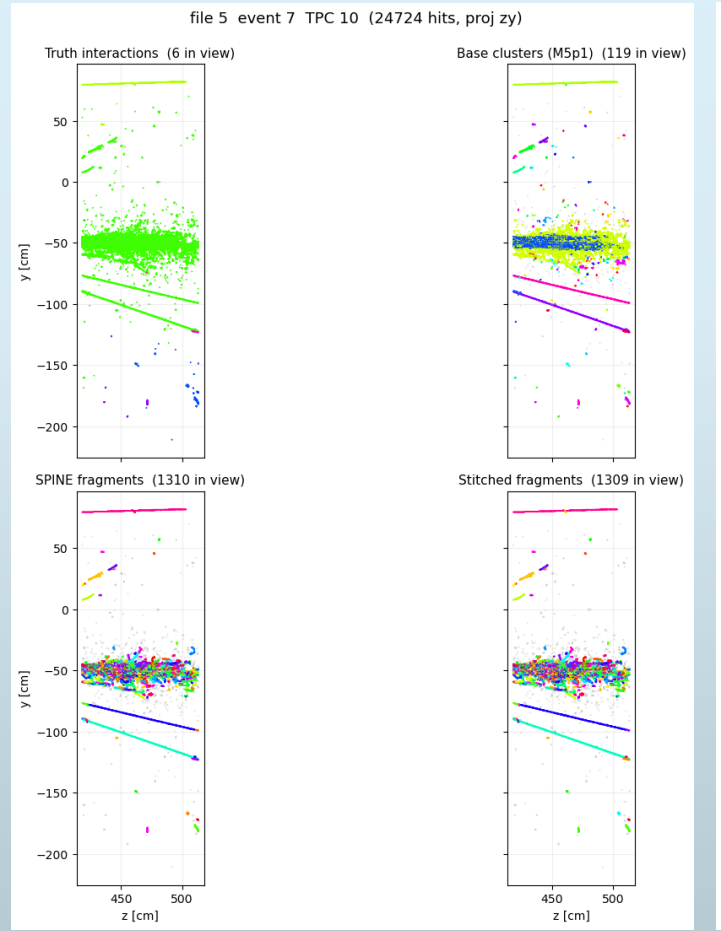
Absorbing clusters to track

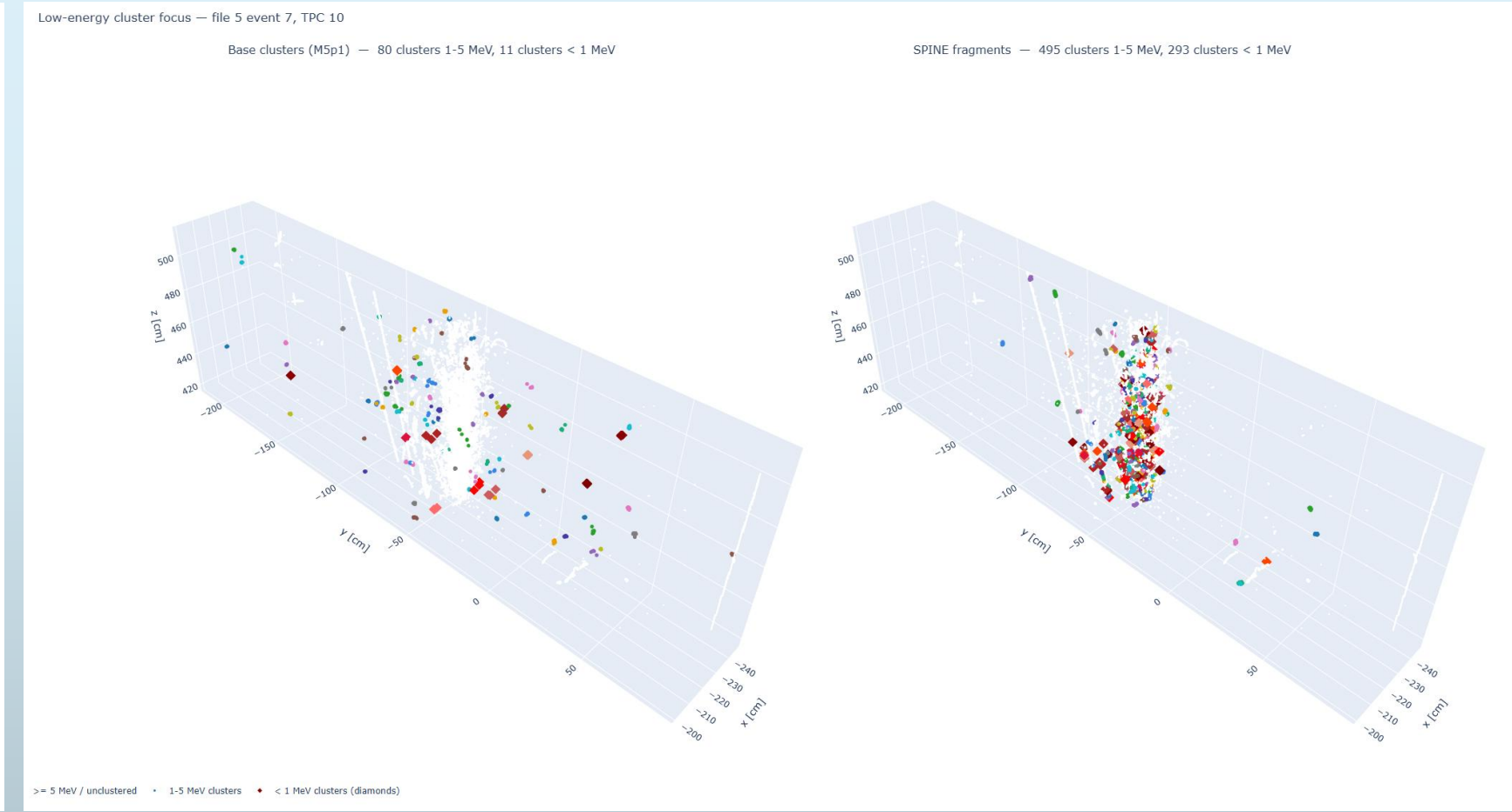
- A lot of these very small (ultra-low energy clusters = <1MeV of energy) are just next to a track
- If we allow track radius to be 3 times as much and absorb immediate ultra=low energy clusters to join:

	count	share
tiny (<1 MeV) clusters total	20,900	581/event
absorbed into 3R track cylinders	**13,322**	**63.7%** (54.3% by energy)
not absorbed	7,578	36.3%

N fragments scales heavily with showering

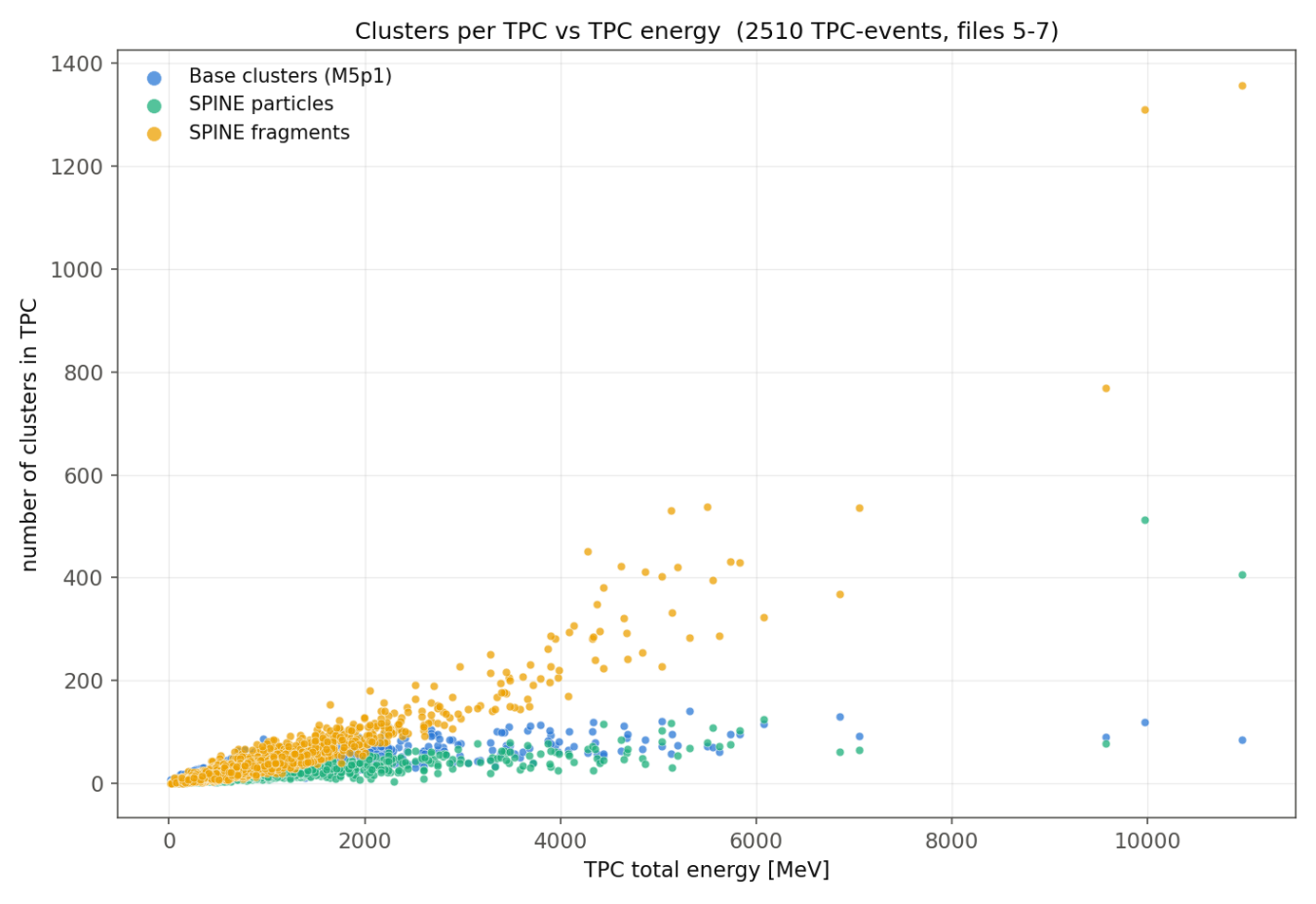
- In an example showering TPC, SPINE fragments produces an overwhelmingly lot of clusters





Too many small clusters in showering TPC

- We will need to merge some of them before performing the charge-light matching analysis



Summary slides

- In either ways, we will need to do some level of SPICE-stitching such that the computation efficiency and accuracy is preserved