Studies on combined GNN + CKF tracking

12.10.2023

Benjamin Huth[1]

Lukas Heinrich[2], Andreas Salzburger[3], Tilo Wettig[1]

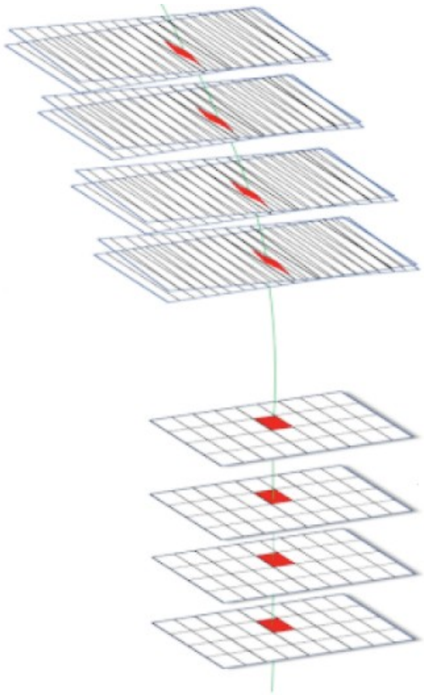
(1) Universität Regensburg, (2) TU München, (3) CERN

Outline

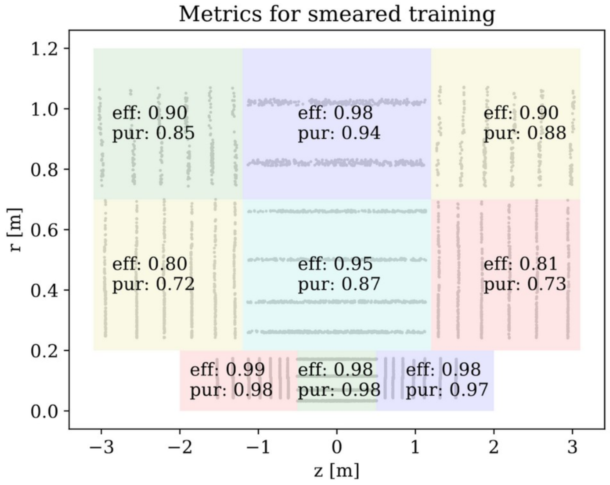
1) Motivation and idea
2) ACTS, the ODD & the Exa.TrkX pipeline
3) Experiment setup
4) Performance analysis

Motivation

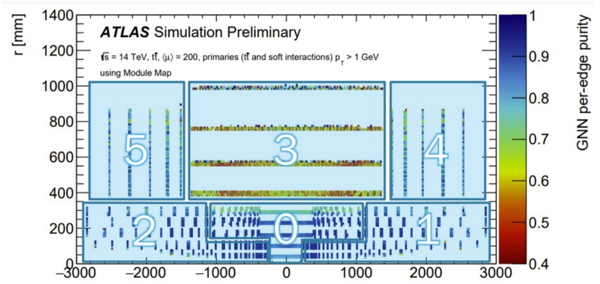
- Graph Neural Network (GNN) based tracking
  - Very promising results achieved recently
  - But: issues observed with low resolution spacepoints [1], [2]
- How to solve this?
  - Idea: Use GNN only in pixels, follow up with Combinatorial Kalman Filter (CKF) → **presented here**
  - Heterogenous networks [2]



From [3]



From [1], ODD, fast simulation only

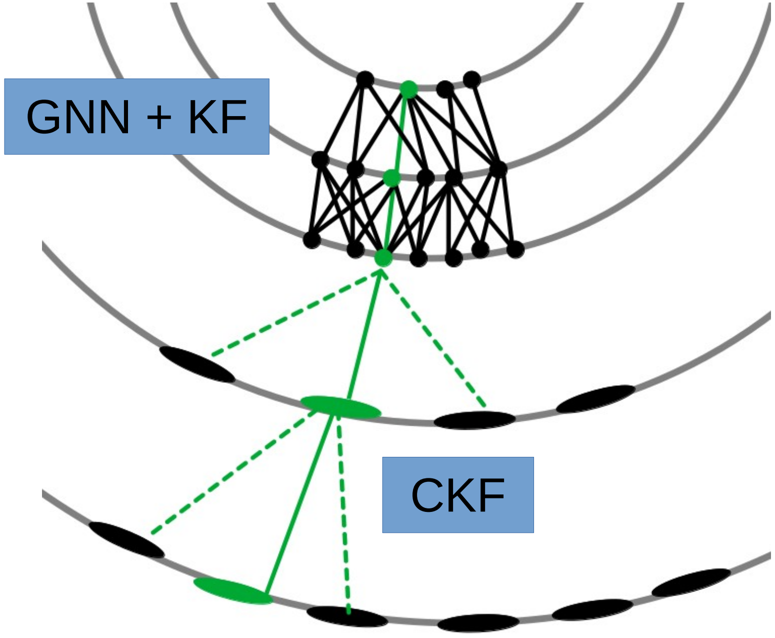


From [3]

[1] 2022, B. Huth, Applying and optimizing the Exa.TrkX Pipeline on the OpenDataDetector with ACTS
[2] 2022, D. Murnane, Heterogeneous GNN for tracking
[3] 2023, P. Calafiura, The Exa.TrkX project

Idea

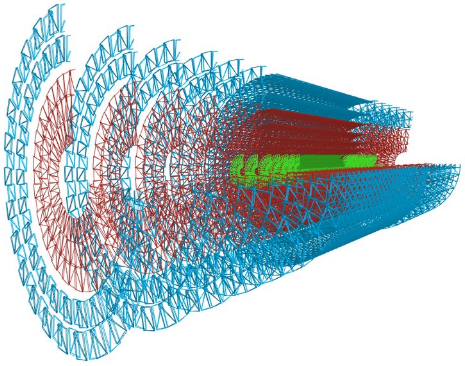
- **GNN:**
  - Resolve combinatorics with high resolution spacepoints in pixels
  - Use ordinary KF here
- **CKF**
  - Completes tracks in strips
- **Benefits of combination:**
  - High quality seeds without duplicates for CKF
  - Use CKF in region with lower density (→ less branching)
  - CKF can e.g. use single strip measurements
  - Smaller graph (pixel only)

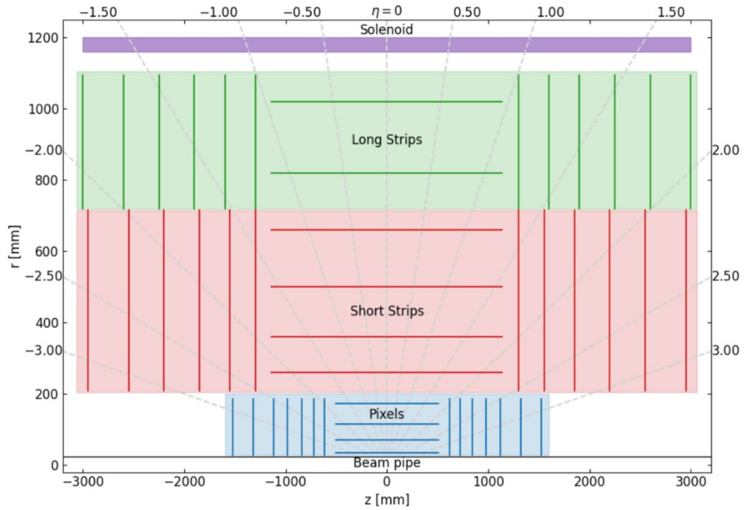


„Full GNN pixel seeding + CKF“

ACTS & OpenDataDetector

- ACTS [1]:
  - Charged particle tracking software package
  - Usage: both production & R&D
  - This work based on the **ACTS examples framework**
- OpenDataDetector (ODD) [2]
  - Virtual silicon detector based on DD4hep
  - Structure:
    - Pixels: 2D, 15µm resolution
    - Short strips: 2D, 43µm/1.2mm
    - Long strips: 1D (stereo angle), 72µm





[1] github.com/acts-project/acts
[2] Paul Gessinger et al, 2021, The Open Data Detector - Tracking and Vertexing

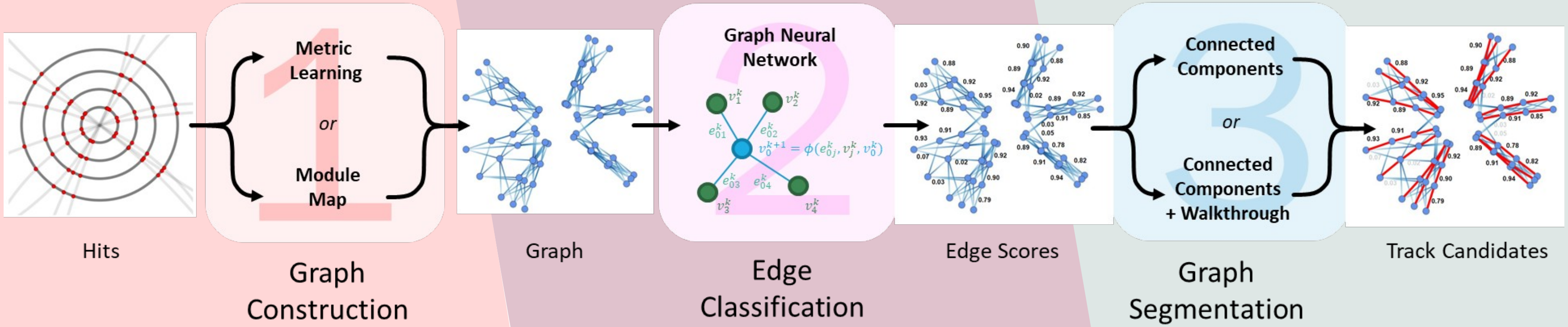
GNN pipeline

- Training with custom branch of GNN4Itk common framework [2]
- Acts Exa.TrkX-plugin based on torchscript and Boost.Graph [3]

Graph construction
- metric learning approach

Edge classification
- simple MLP filter
- GNN

Track building
- connected components



From [1]

[1] 2023, P. Calafiura, The Exa.TrkX project
[2] gitlab.cern.ch/bhuth/commonframework
[3] acts.readthedocs.io/en/latest/plugins/exatrkx.html

KF-CKF-Hybird implementation

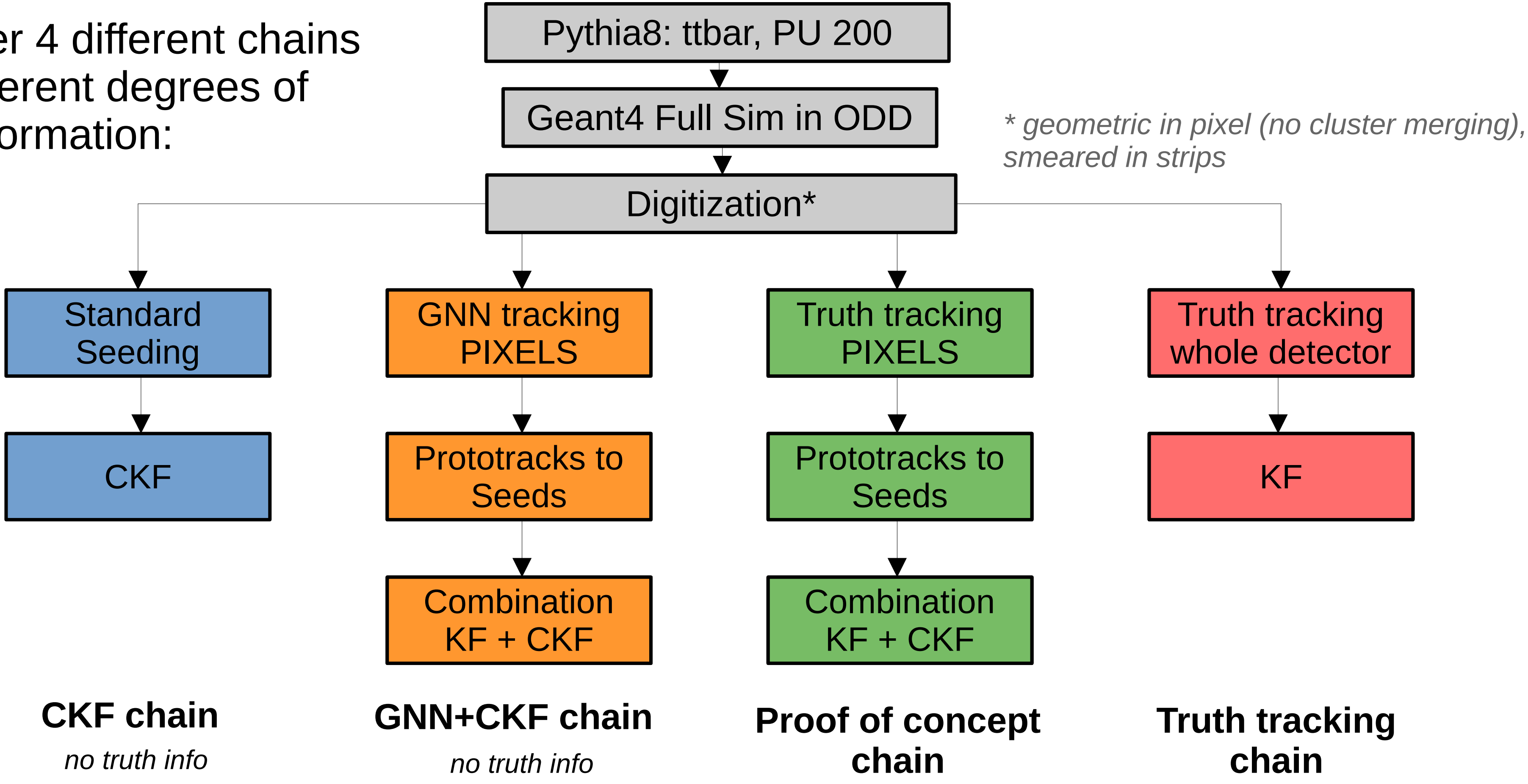
- **Very easy** implementation (~200 lines C++)
  - custom *measurement-accessor* for CKF*
    - Use measurements from GNN prototracks in pixel volumes
    - Use all available measurements in strip volumes
  - Shows flexibility of ACTS algorithms

*actually: `SourceLinkAccessorDelegate`

→ *More details in backup*

Setup

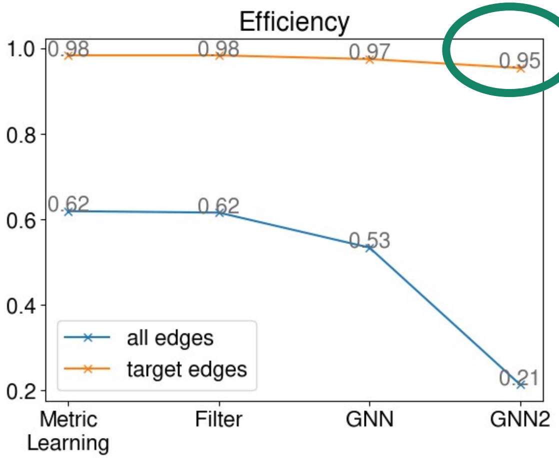
- Consider 4 different chains with different degrees of truth information:

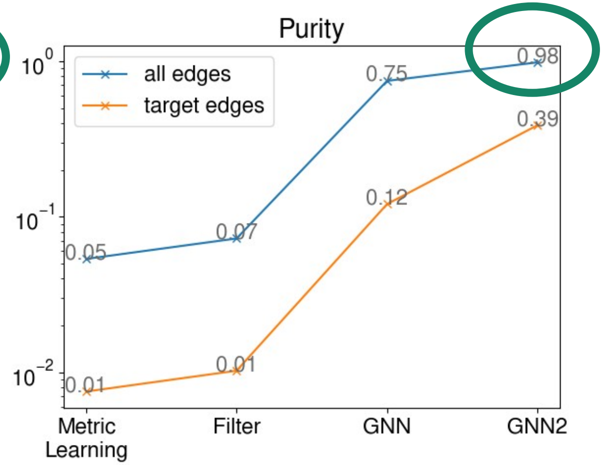


→ *More details in backup*

Pipeline training

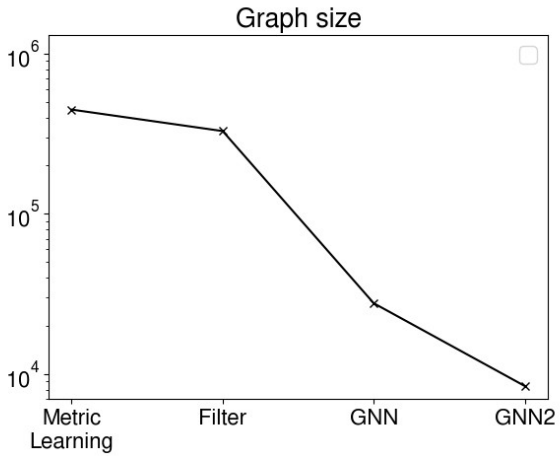
- Target particles (up-weighted in training):
  - min pT: 1GeV
  - min pixel hits: 3
- Training:
  - 2K events (1500, 250, 250)
  - Features:
    - r, phi, z
    - cluster features (metric learning only)
  - Goal:
    - high target efficiency
    - high total purity
- 2nd GNN stage improved performance





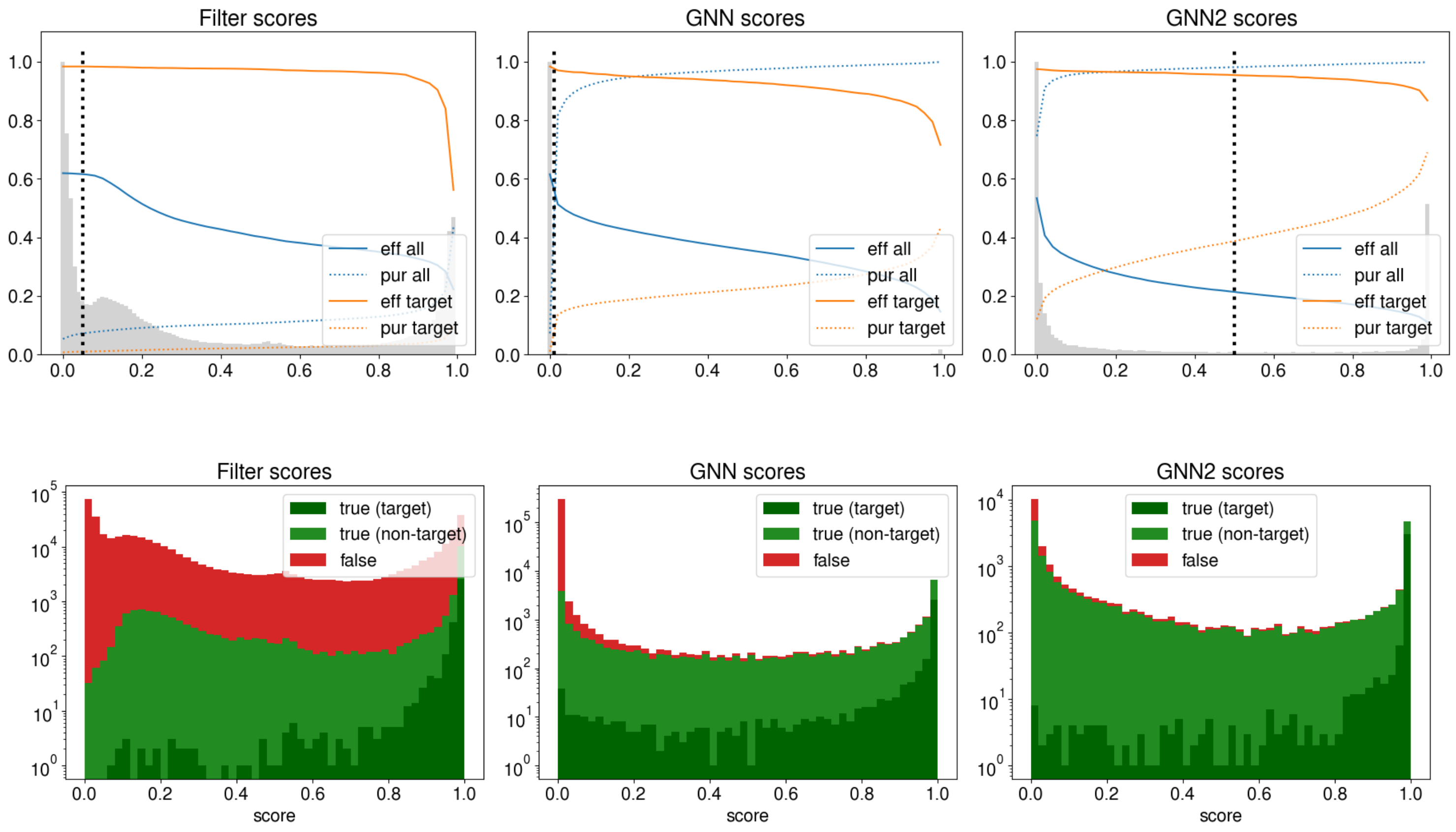
$$\text{eff} = \frac{\text{true positive edges}}{\text{true edges}}$$

$$\text{pur} = \frac{\text{true positive edges}}{\text{positive edges}}$$



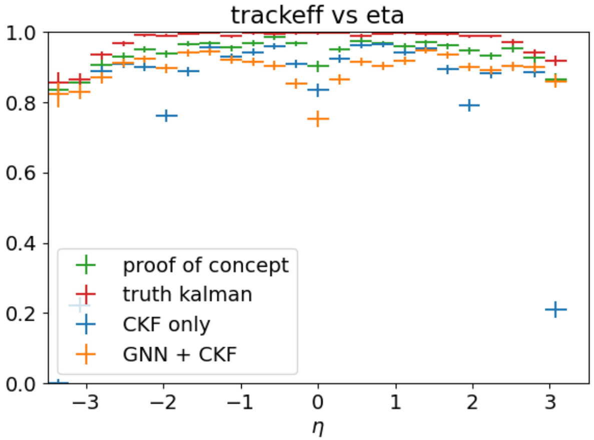
Pipeline training

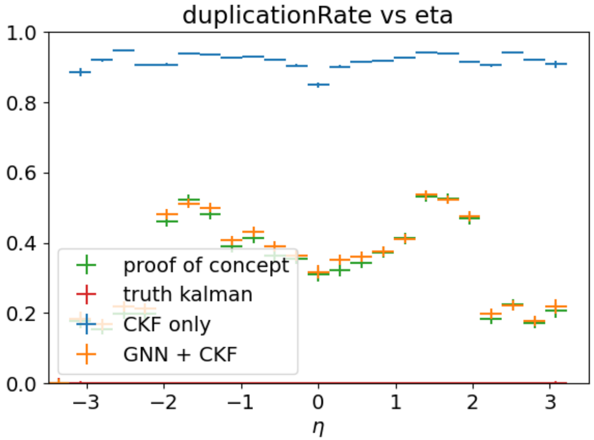
- Use low edge cut in early filter stages to preserve efficiency
- Hypothesis:
  - GNN2 gives improvement because graph is better balanced?



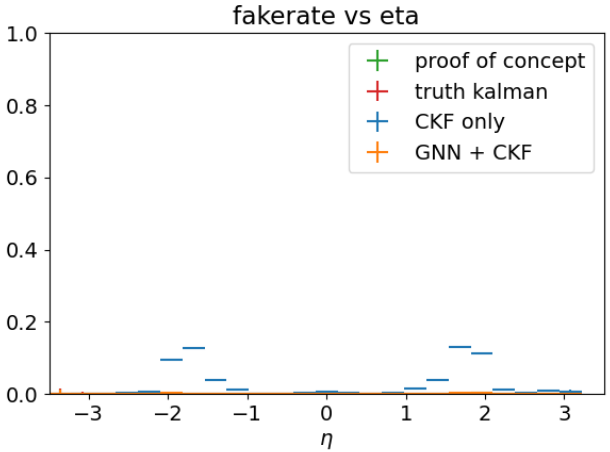
Track finding performance

- Plots based on the ACTS CKF Performance Writer
  - 20 events
  - No ambiguity solution
- Truth matching:
  - Particles with pT > 1GeV, #hits >= 7
    - Also apply track selection based on fitted momentum and found hits
  - Track is matched if > 50% of its hits belong to one particle
  - Otherwise: fake
- **Efficiency not optimal overall**
  - Loose particles both in GNN and CKF stage
  - CKF performance not yet understood fully (see also Andi's Talk)
  - Drop at η=0 needs to be investigated





- GNN+CKF chain:
  - Almost zero fakerate
  - **Still duplication from CKF stage**
    - in proof-of-concept as well



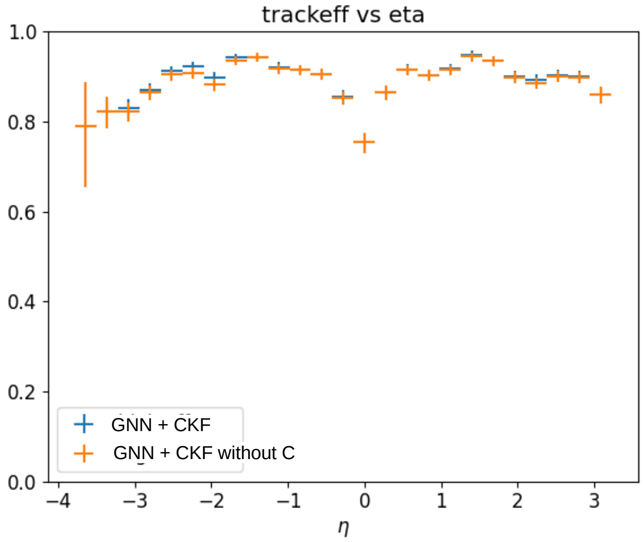
Remove the C

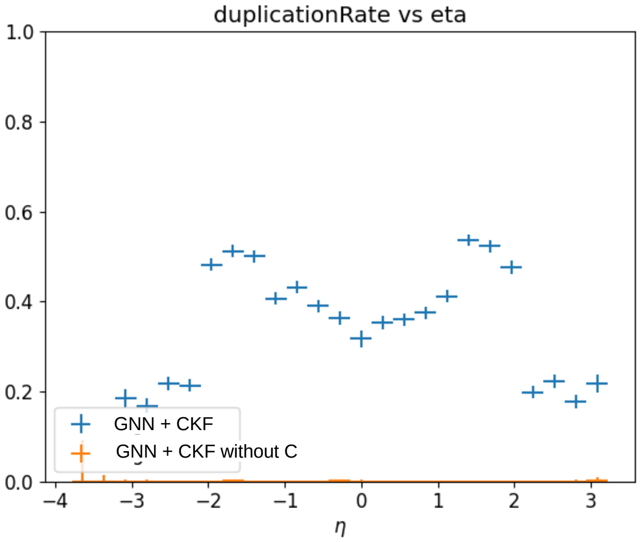
~~C~~KF

10 (default) → 1

```
acts.MeasurementSelector.Config(
    [(acts.GeometryIdentifier(), ([], [chi2Cut], [nMeasurementsCut]))]
)
```

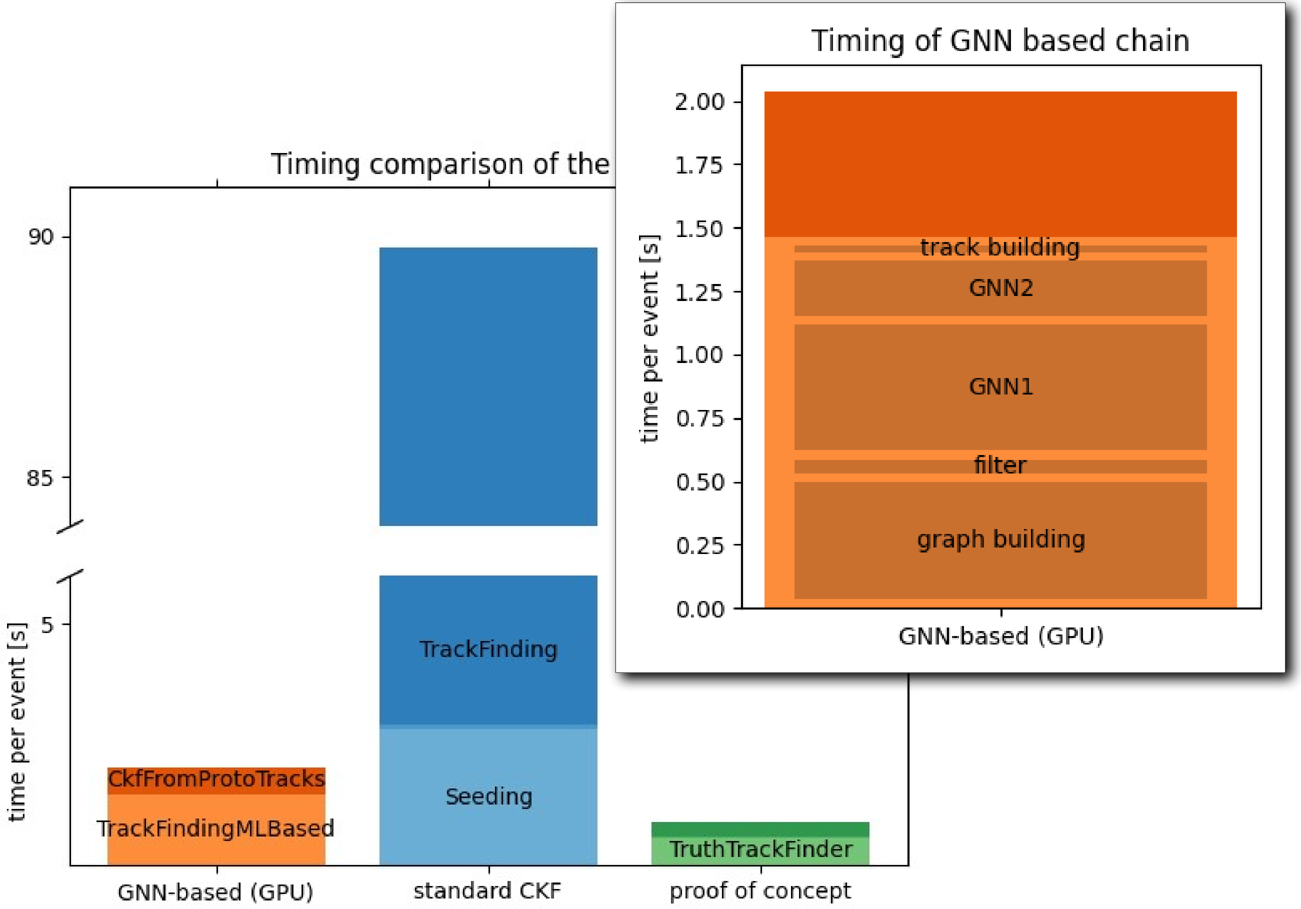
- Remove combinatorial aspect
  - Minimal efficiency cost
  - Zero duplication rate
- **Again:** I did not try to optimize this for standard CKF



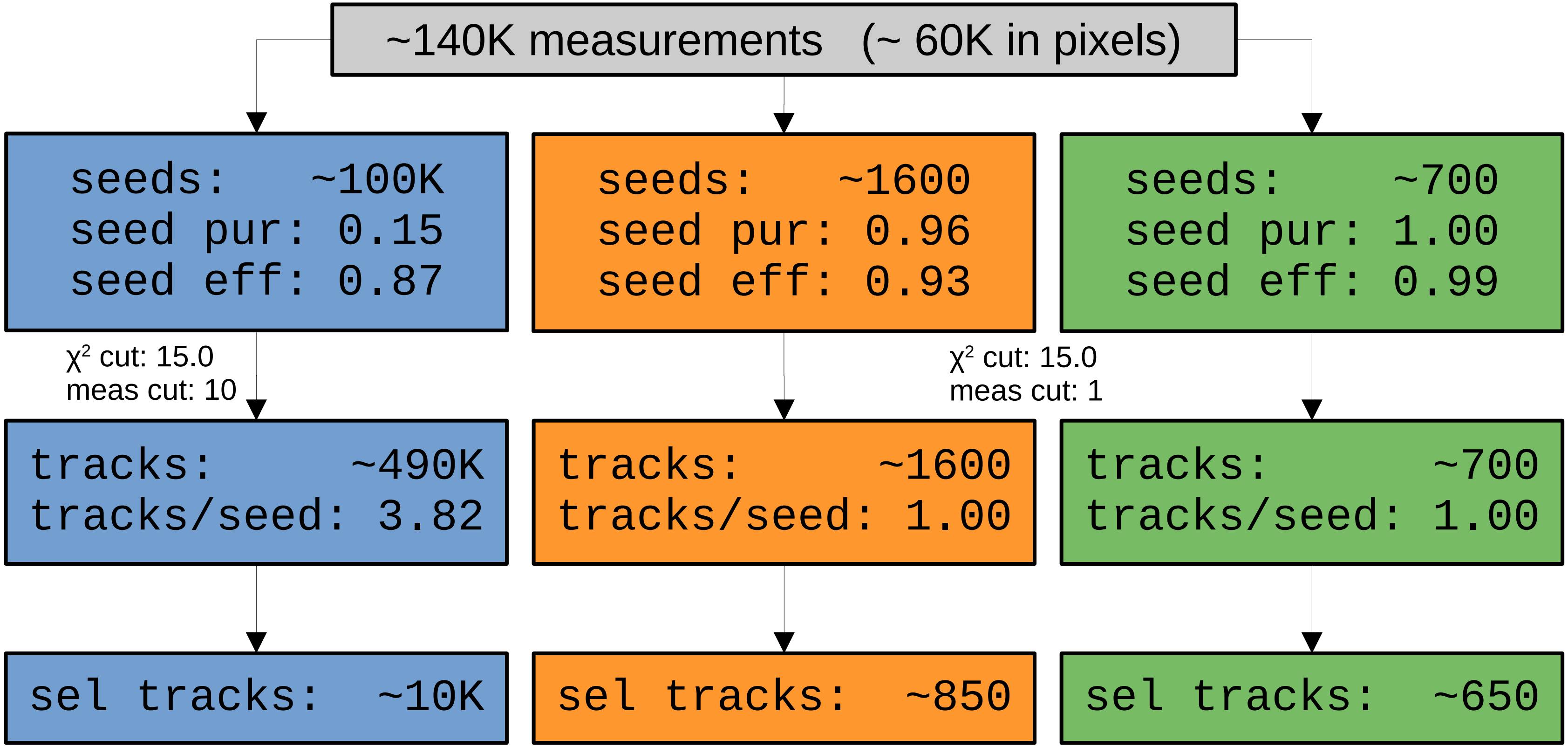


Preliminary timing

- **Caveat:**
  - CKF not optimized for this study (seed filtering, ...)
  - CPU timing seems to be very machine dependent
- Large combinatoric overhead of vanilla CKF configuration
- GPU: Nvidia A100 40GB



Combinatorics



C(KF) stage

Track selection:
- min pT: 1 GeV
- min hits: 7

CKF **GNN+CKF** **Proof of concept**

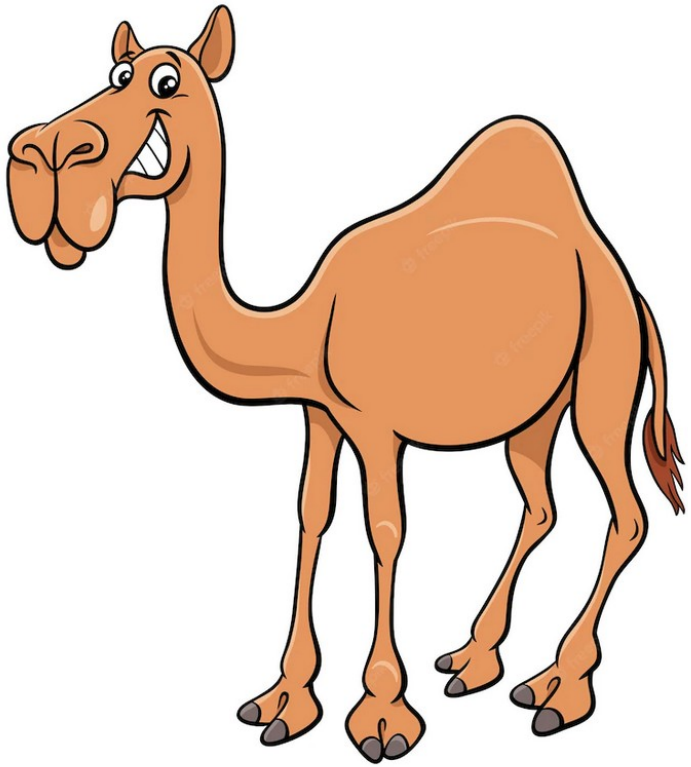
Conclusion

- Combination of GNN + CKF easy to implement in ACTS
  - Example for flexibility of ACTS tools (+ examples framework)
- Tracking performance:
  - Very low duplication and fakerates
  - suboptimal efficiency (both due to GNN and CKF)
- Promising compute performance
  - Even more interesting, if CKF on GPU is available

Special thanks to Lukas Heinrich and MPG for providing access to GPU clusters

Reproducible workflow via **snakemake**

Backup



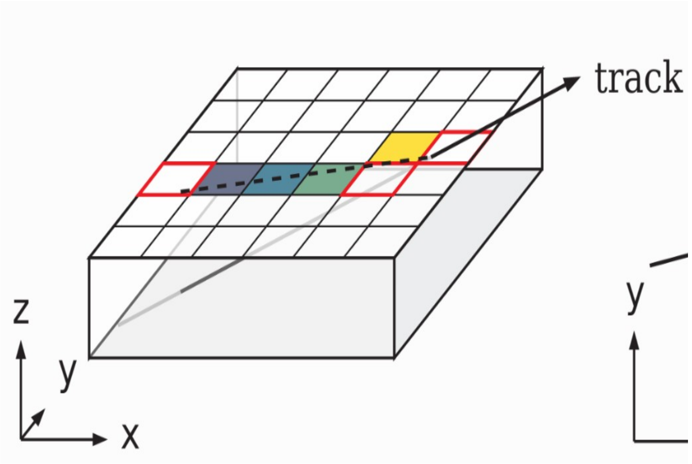
ACTS Exa.TrkX plugin

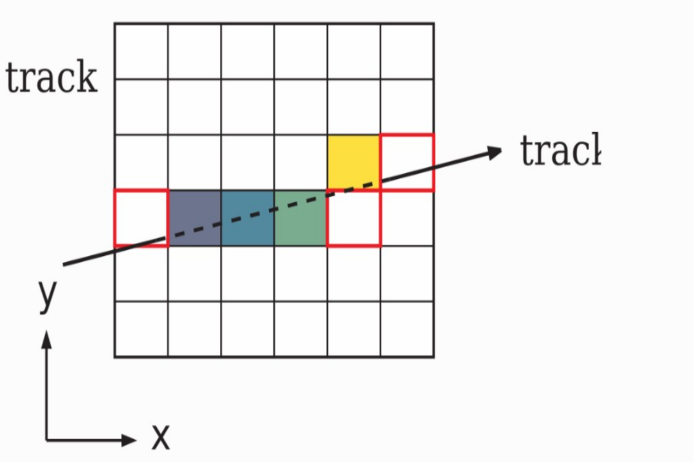
- Supports CPU & GPU
  - torchscript / FRNN / Boost.Graph
- Examples-framework integration
  - Supports cluster features
  - Edge-metrics hook based on truth sim data

```
20:25:22    MetricsHook    INFO    Metrics for total graph:
20:25:22    TrackFinding   INFO    Efficiency=0.195985, purity=0.983137
20:25:22    MetricsHook    INFO    Metrics for target graph (pT > 1 GeV, nHits >= 3):
20:25:22    TrackFinding   INFO    Efficiency=0.972825, purity=0.405324
```

Geometric digitization

- Geometric digitization:
  - Compute paths through pixels
  - Path corresponds to charge deposit
  - Configurable parameters:
    - Path threshold:
    - path smearing: N(0,
- 4 cluster features for Metric learning:
  - Cell-count
  - accumulated cell-activation
  - Cluster size in l_0 and l_1

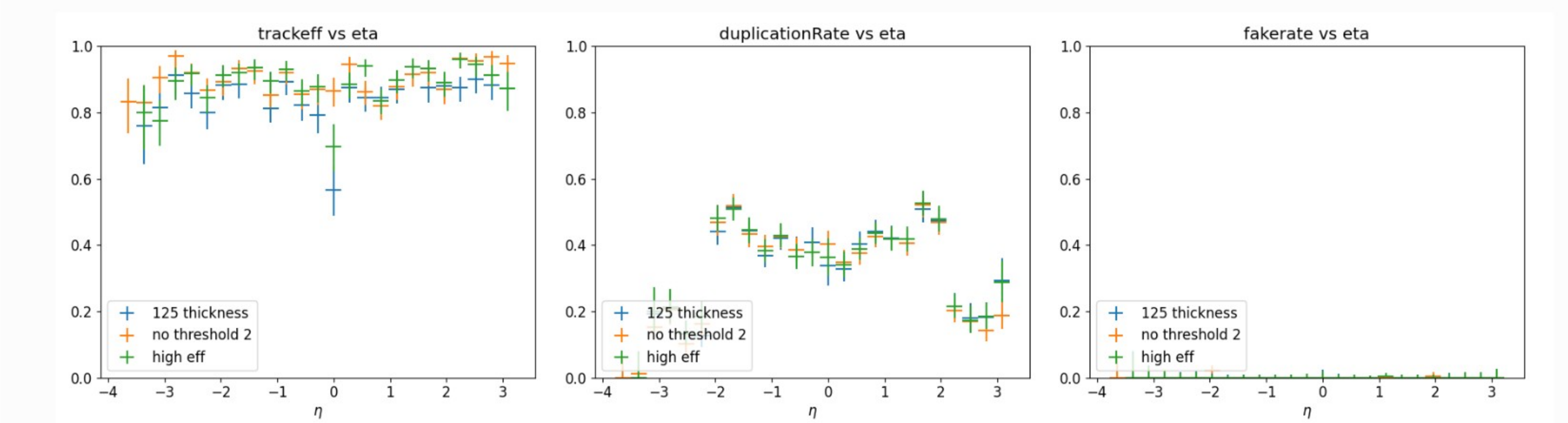




Improvements with cluster features

- With cluster features
  - Comparable efficiency
  - Improved purity

Different models / configurations



Label in plot	Description
„high eff“	setup presented before
„125 thickness“	Setup with 1 GNN
„no threshold“	Setup with 1 GNN + idealized geometric digitization (no threshold, no charge smearing)

Details on implementation

- CKF can be configured by `SourceLinkAccessor` template
  - Returns source-links / measurements for surface

```
struct ProtoTrackSourceLinkAccessor
    : GeometryIdMultisetAccessor<IndexSourceLink> {
  Container protoTrackSourceLinks;

  auto range(const Acts::Surface& surface) const {
    const auto& logger = *loggerPtr;

    if (protoTrackSourceLinks.contains(surface.geometryId())) {
      auto [begin, end] =
          protoTrackSourceLinks.equal_range(surface.geometryId());
      return {Iterator{begin}, Iterator{end}};
    }

    auto [begin, end] = container->equal_range(surface.geometryId());
    return {Iterator{begin}, Iterator{end}};
  }
};
```

Where do we lose tracks?

- Disentangle unphysical track candidates
    - Some graph algorithms tested in python but not yet implemented in ACTS
- CKF performance not yet 100% understood