Robustness Studies of Graph-Based Track Seeding (GBTS) Algorithm

Jason Ni

Supervisors: Jyoti Biswal, Tim Adye, Tim Martin

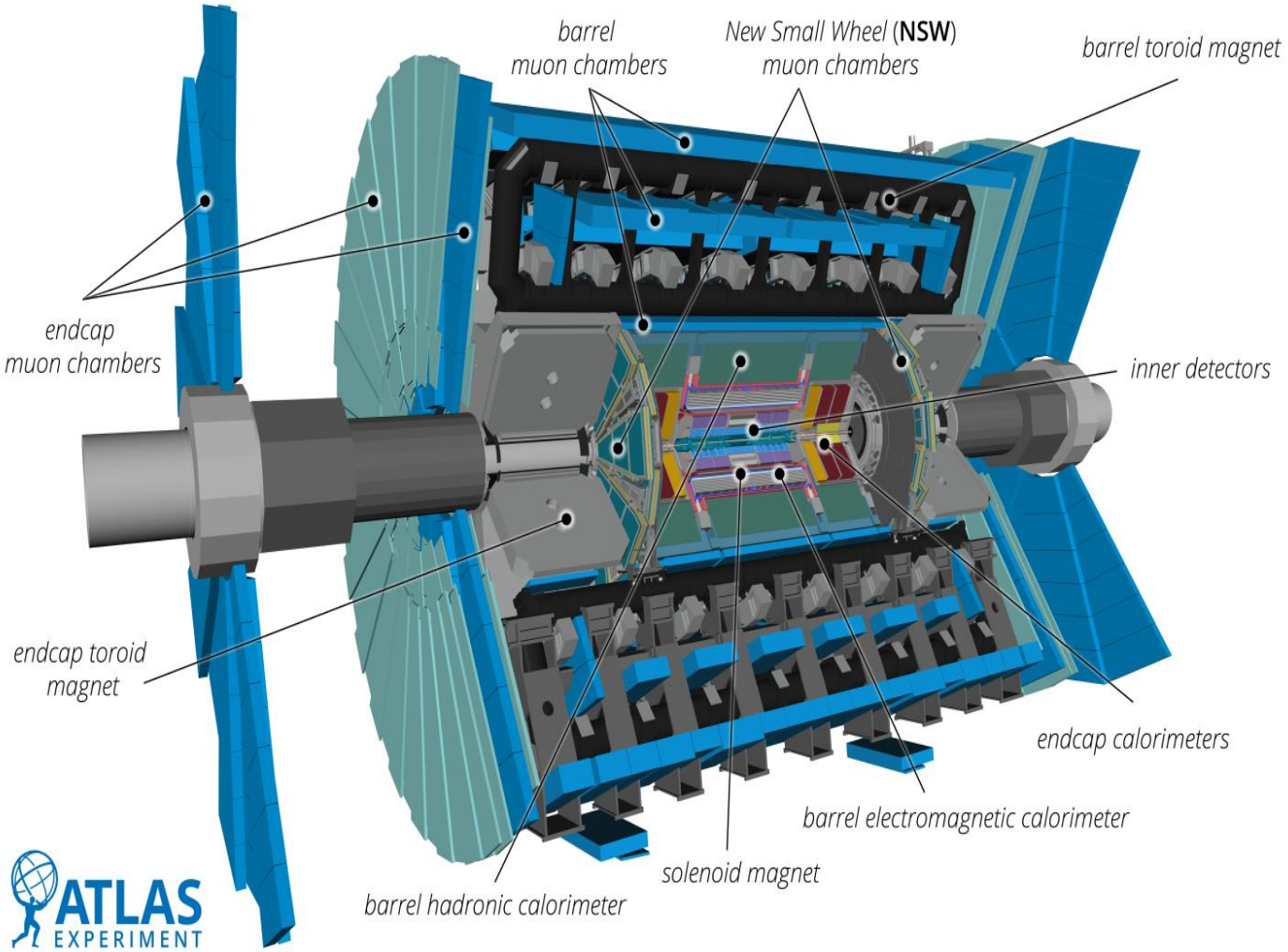
PPD Seminar

20 August 2025

Rutherford Appleton Laboratory

ATLAS Trigger

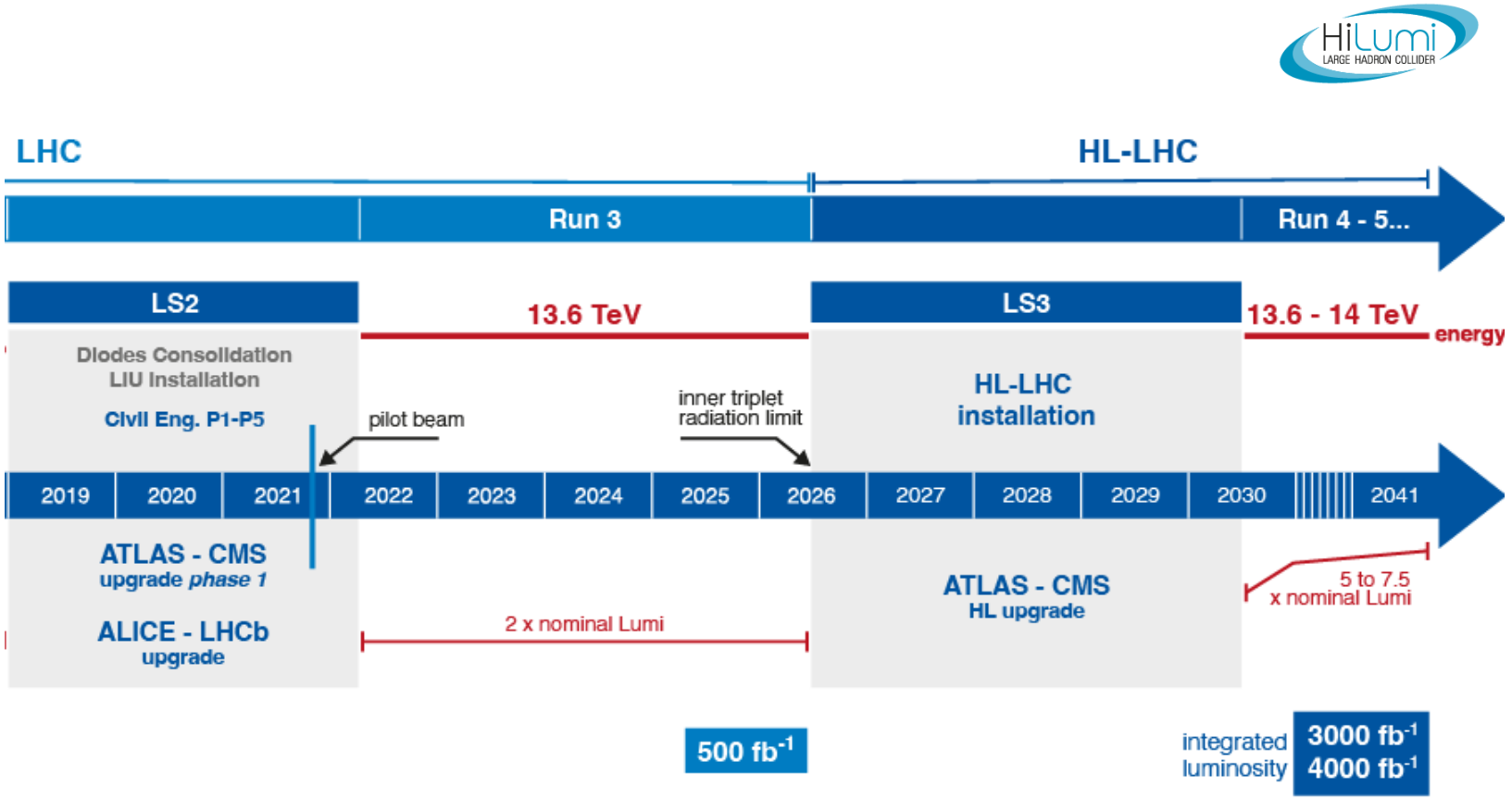
- ATLAS is a particle detector studying particle interactions at the LHC
- Don't have the space or computing power to store and analyse every event
- Use trigger to select "interesting" events to keep
- Current ATLAS trigger rate: 3 kHz [1]



Run 3 ATLAS configuration; Source: CERN

[1] The ATLAS Collaboration, *The ATLAS trigger system for LHC Run 3 and trigger performance in 2022,* https://iopscience.iop.org/article/10.1088/1748-0221/19/06/P06029

High-Luminosity Large Hadron Collider (HL-LHC)



- HL-LHC upgrades will increase number of interactions even further
- Much higher luminosity and pile up -> more things for trigger to process
- Final event rate 10 kHz [2]

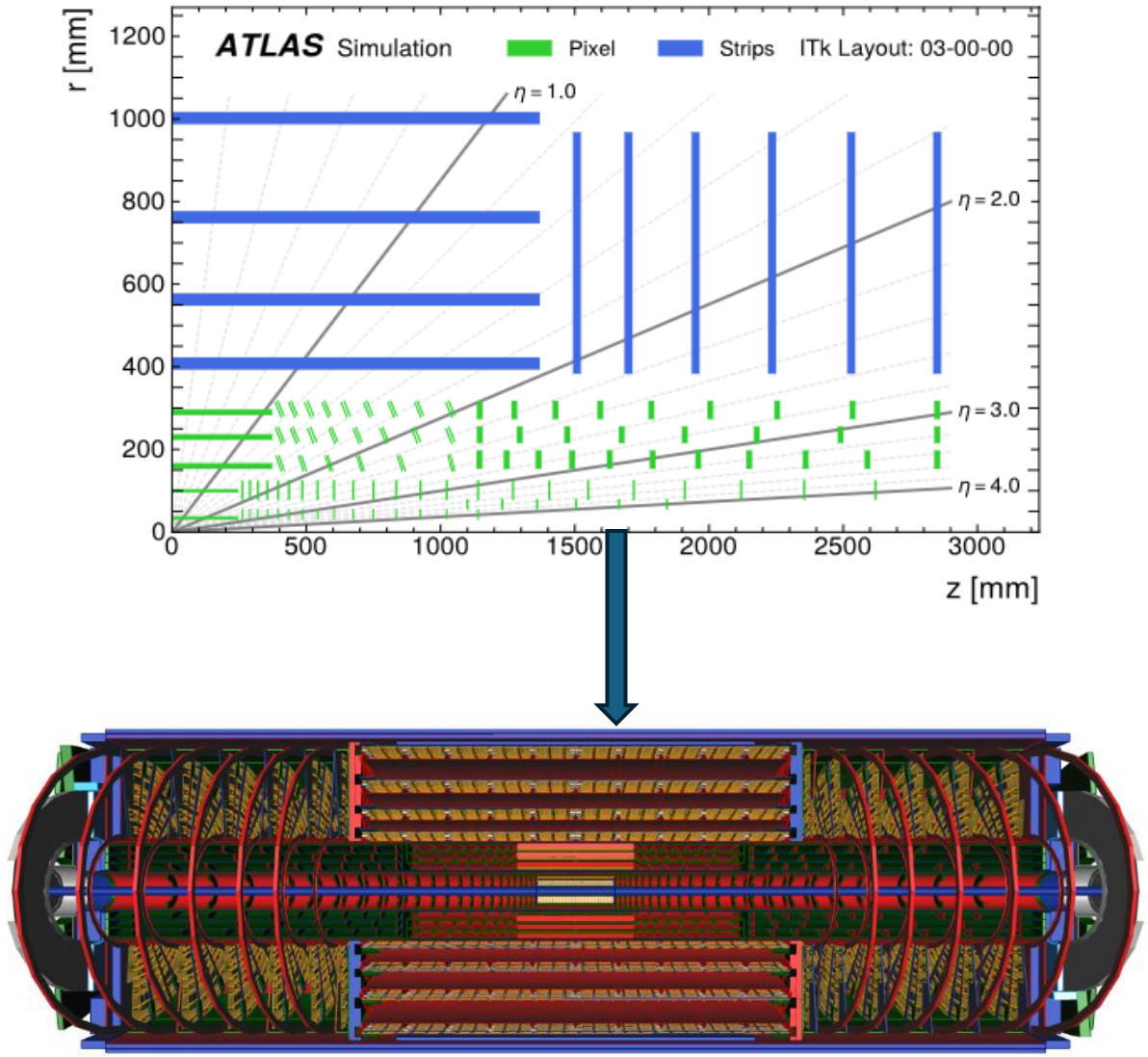
Source: Hi Lumi Project

[2] The ATLAS Collaboration, *Technical Design Report for the Phase-II Upgrade of the ATLAS Trigger and Data Acquisition System - Event Filter Tracking Amendment*, https://cds.cern.ch/record/2802799/

Inner Tracker (ITk)

- Inner Detector to be replaced with Inner Tracker for HL-LHC
- 2 parts: inner Pixel Detector and outer Strip Detector
- Will provide higher granularity
- Can now access pseudorapidity $2.5 < |\eta| < 4$
- Need new software to take advantage of this

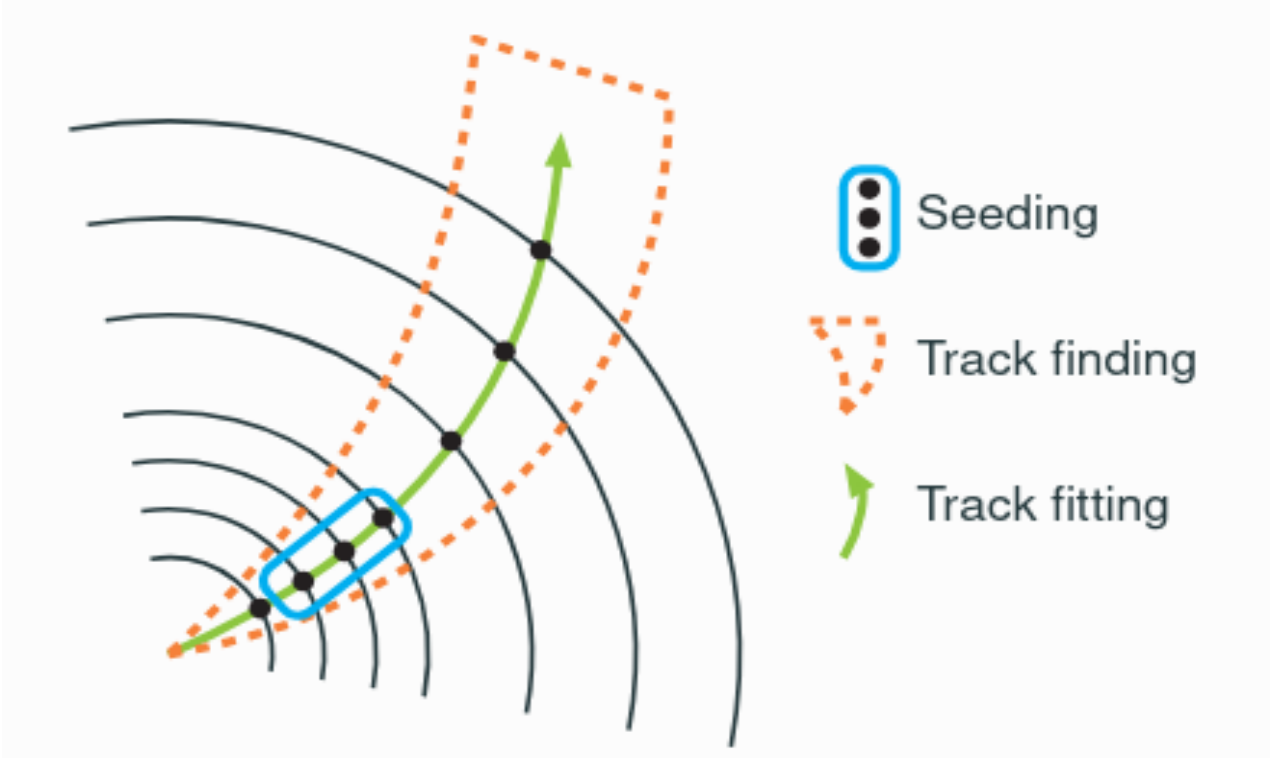
Source: The ATLAS Collaboration



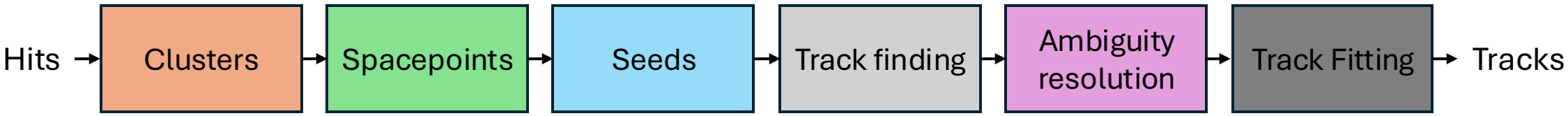
ITk diagram; Source: The ATLAS Collaboration

Tracking

- Tracking is a key part of the software trigger
- Important for other processes like vertex reconstruction used by trigger to make final decision
- Reconstruct particle tracks from detector hits
- The **ACTS (A Common Tracking Software)** project is an implementation of track reconstruction software of the LHC era towards HL-LHC and beyond.

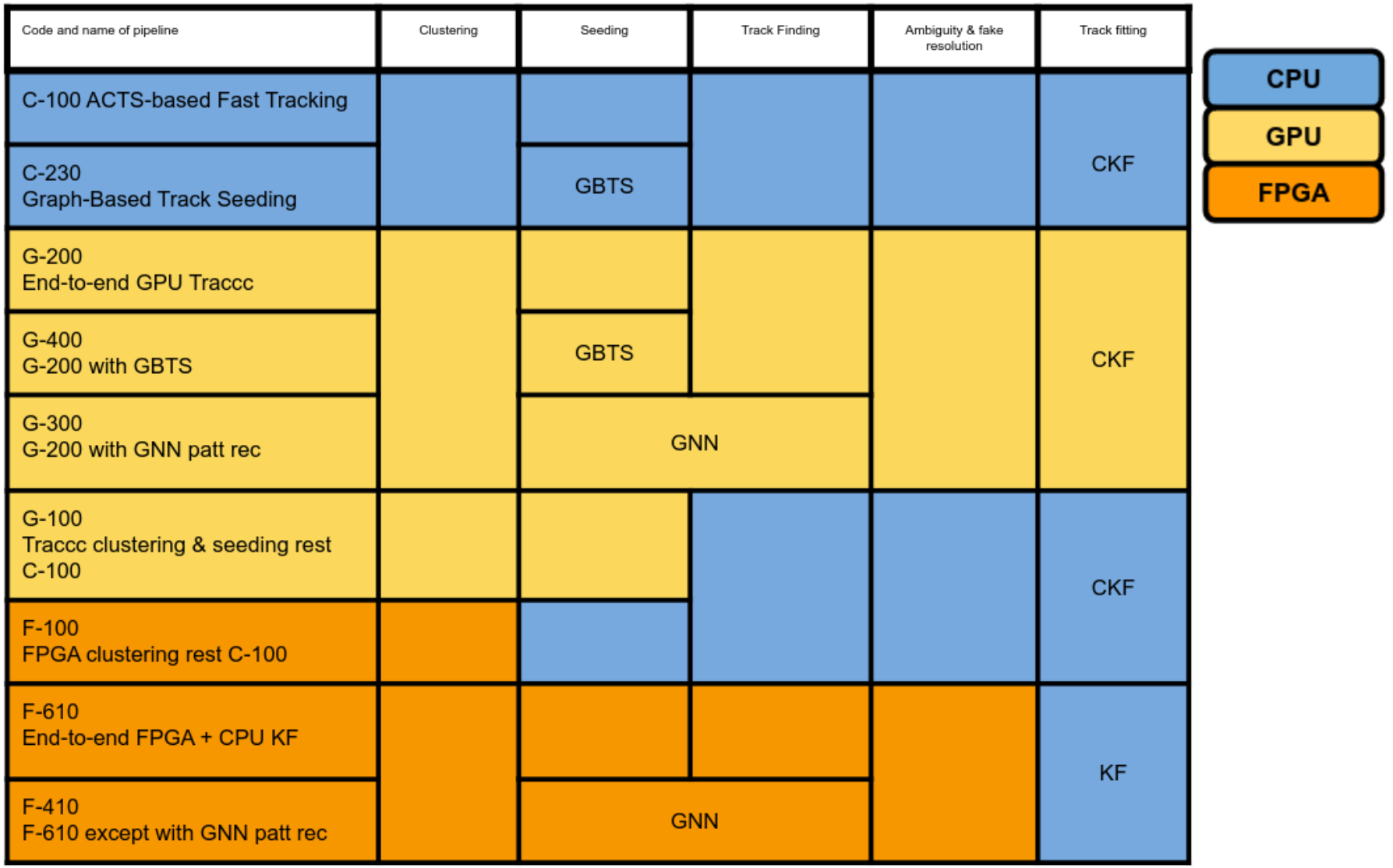


Source: ACTS Project



Tracking Pipelines

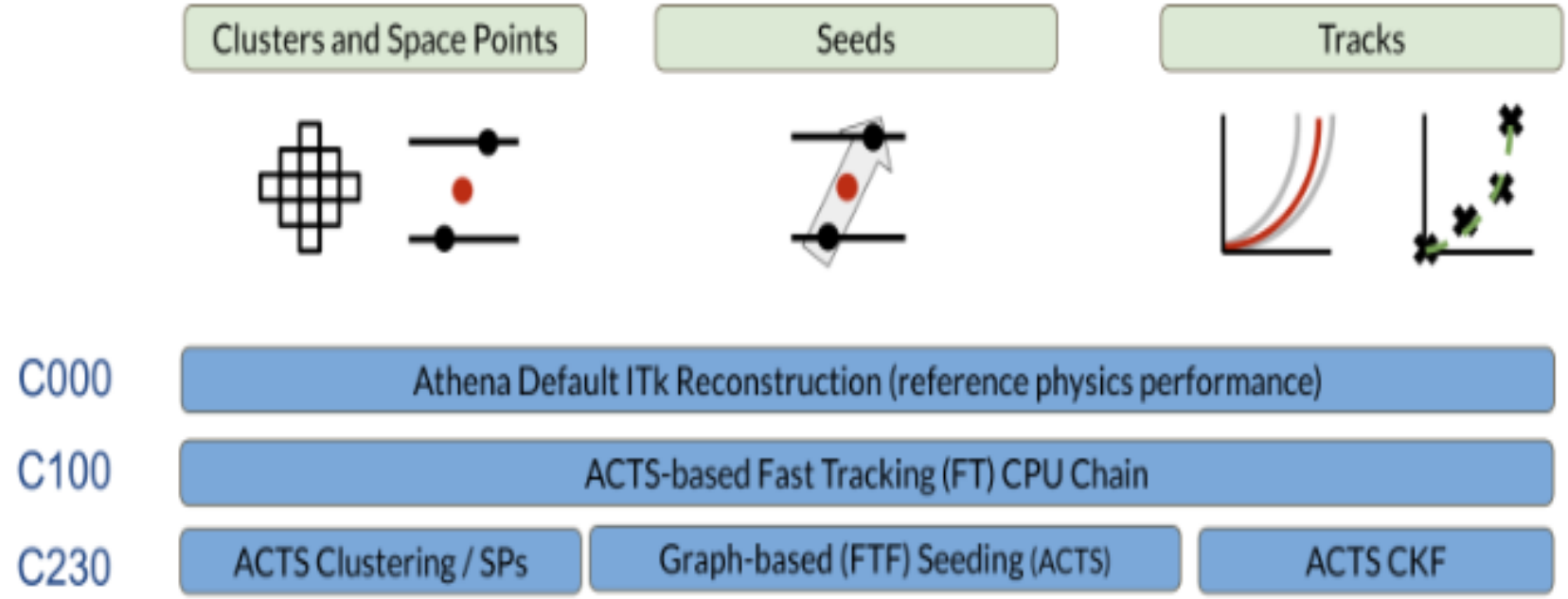
- Many different tracking pipelines in development.
- Some run on different architectures, some with different software
- By end of the year, Technology Choice Committee will make recommendation of which technology to use
- **C for CPU, G for GPU, F for FPGA**



Source: [Veronique Boisvert & Stephanie Majewski]

C100 vs C230

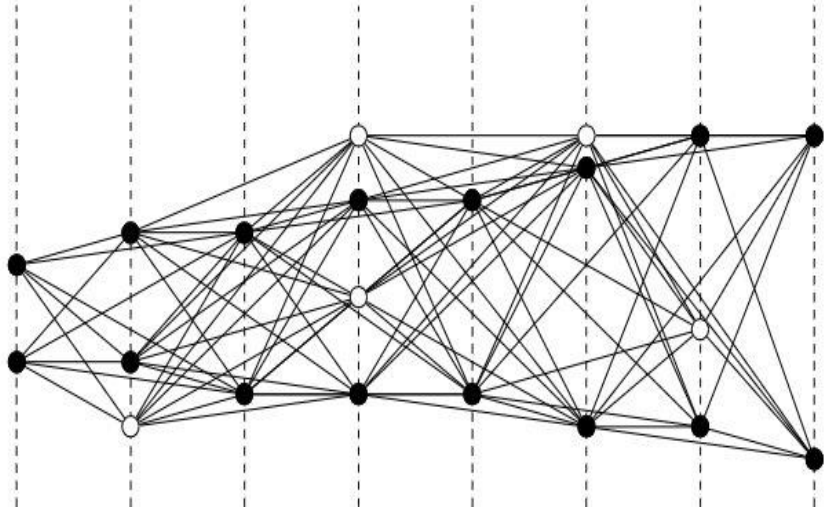
- Both run on CPUs
- C230 uses GBTS seeding instead of default ACTS seeding used in C100
- Else they are the same
- We want to compare the performance of the two pipelines with defects

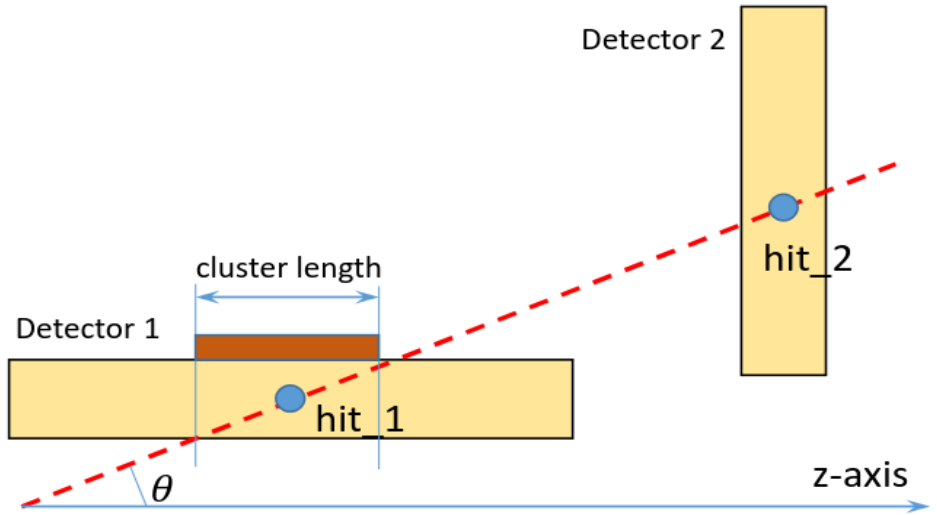


Source: The ATLAS Collaboration

Graph-Based Track Seeding (GBTS)

- Split detector into η bins of roughly equal numbers of spacepoints
- Form graph of 2-spacepoint track segments
- Use cluster dimensions to predict range of acceptable θ for segments
- Track segments that share spacepoints with matching track parameters connected
- Find longest sequences of connected track segments
- Extract tracklets (longer than ACTS 3-spacepoint seeds)





Source: Dmitry Emeliyanov

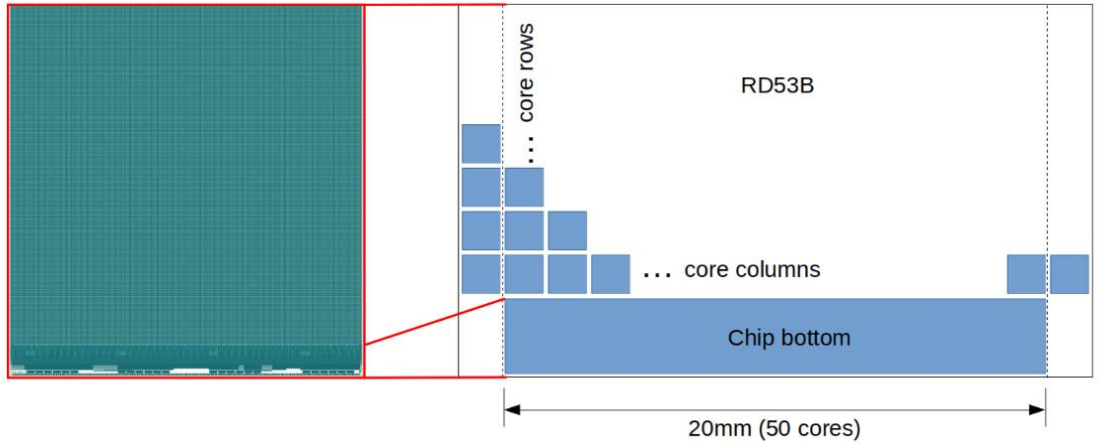
Athena

- Main ATLAS software framework (Athena Documentation)
- Contains the code to run track reconstruction (Reco_tf.py)
- Example C230 track reconstruction job on **simulated ITk detector output**:

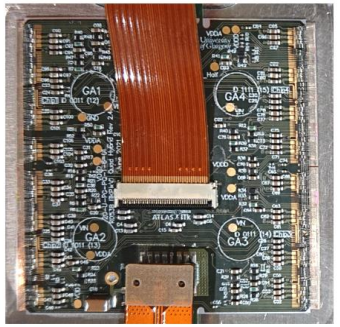
```
Reco_tf.py --CA \
--multithreaded True \
--maxEvents '-1' \
--preInclude
'InDetConfig.ConfigurationHelpers.OnlyTrackingPreInclude,ActsConfig.ActsCIFlags.actsWorkflowF
lags' \
--postInclude 'ActsConfig.ActsPostIncludes.ACTSClusterPostInclude' \
--preExec "flags.Tracking.doITkFastTracking=True; from ActsConfig.ActsConfigFlags import
SeedingStrategy;flags.Acts.SeedingStrategy=SeedingStrategy.Gbts2;flags.Tracking.doPixelDigita
lClustering=True; " \
--steering 'doRAWtoALL' \
--inputRDOFile ${Input_RDO} \
--outputAODFile 'OUTPUT.AOD.pool.root'
```

Types of Defect

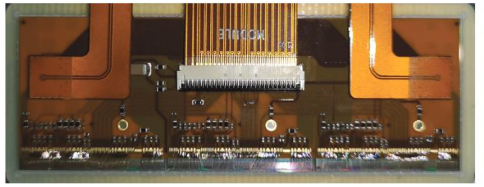
- Have to account for different types of defect that may occur in the detector
- `PixelDefect:` defect in a pixel (the fundamental building block of ITk pixel detector)
- `FrontEndCCDefect:` a core column has 8x384 = 3072 pixels.
- `CornerDefect:` defect in the corners of a chip.
- `ModuleDefect:` chips make up modules. 9164 modules in ITk pixel detector
- `Noise:` pixel registers random hit that isn't the result of a particle passing through.



There are 50 core columns in a chip



Quad module: 1 quad-size sensor tile ($\sim 4 \times 4\,\mathrm{cm}^2$) and 4 frontend (FE) chips



Triplet module: 3 $\times$ single bare modules each with 1 single 3D sensor tile on 1 FE chip

There are 3 or 4 chips per module

Source: Lingxin Meng

Simulating defects

- Simulate ITk defects using extra flags

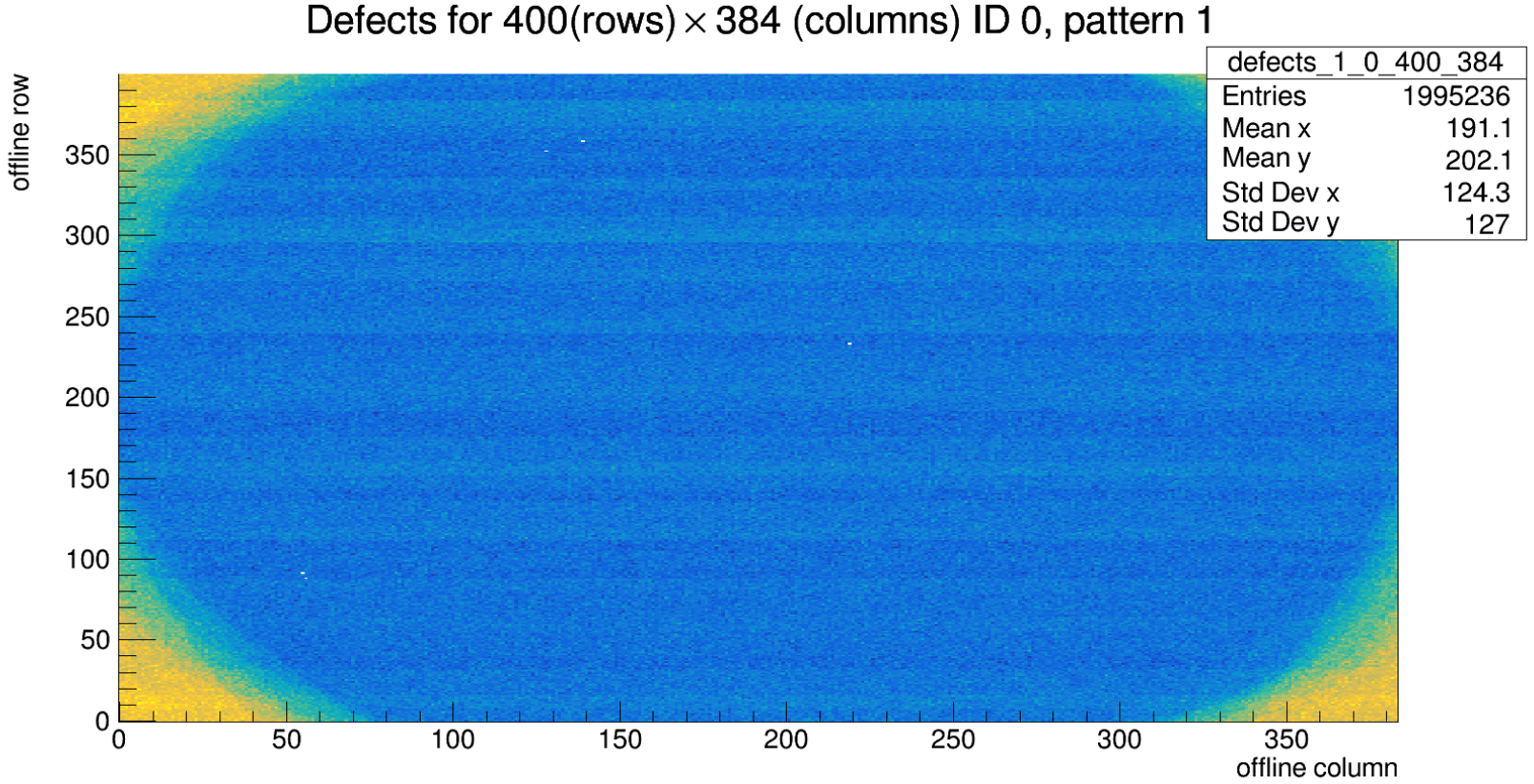
```
Reco_tf.py --CA \
--multithreaded True \
--maxEvents '-1' \
--preInclude
'InDetConfig.ConfigurationHelpers.OnlyTrackingPreInclude,ActsConfig.ActsCIFlags.actsWorkflowFlags' \
--postInclude 'ActsConfig.ActsPostIncludes.ACTSClusterPostInclude' \
--preExec "flags.Tracking.doITkFastTracking=True; from ActsConfig.ActsConfigFlags import
SeedingStrategy;flags.Acts.SeedingStrategy=SeedingStrategy.Gbts2;flags.Tracking.doPixelDigitalCluste
ring=True; " \
--postExec "from InDetDefectsEmulation.PixelDefectsEmulatorPostInclude import
emulateITkPixelDefectsDefault;
emulateITkPixelDefectsDefault(flags,cfg,HistogramFileName='itk_pixel_defects.root',FrontEndCCDefectP
rob=1e-1,PixelDefectProb=1e-2,ModuleDefectProb=1e-2,CornerDefectProb=15e-2,NoiseProb=1e-
5,PropagateDefectsToStatus=True);" \
--steering 'doRAWtoALL' \
--inputRDOFile ${Input_RDO} \
--outputAODFile 'OUTPUT.AOD.pool.root'
```

Simulating defects (cont.)

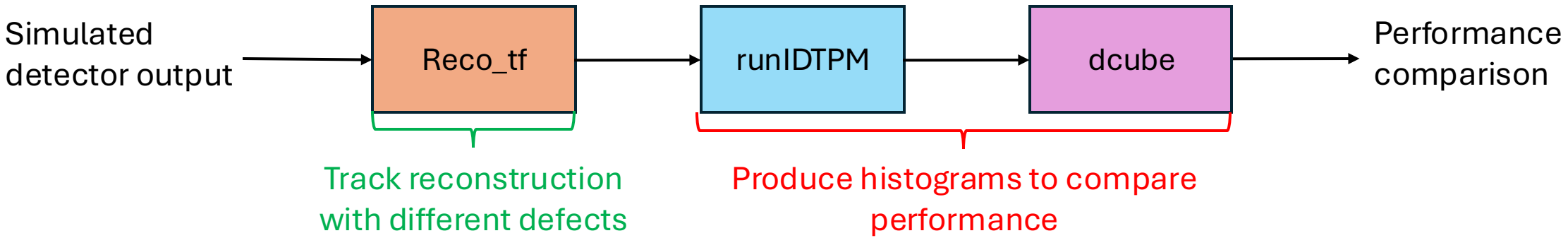
- Use default defect parameters:

```
FrontEndCCDefectProb=0.1
PixelDefectProb=0.01
ModuleDefectProb=0.01
CornerDefectProb=0.15

+

NoiseProb=0.00001
```

- Test performance with no defects and with defects for both pipelines



Performance Comparison Pipeline



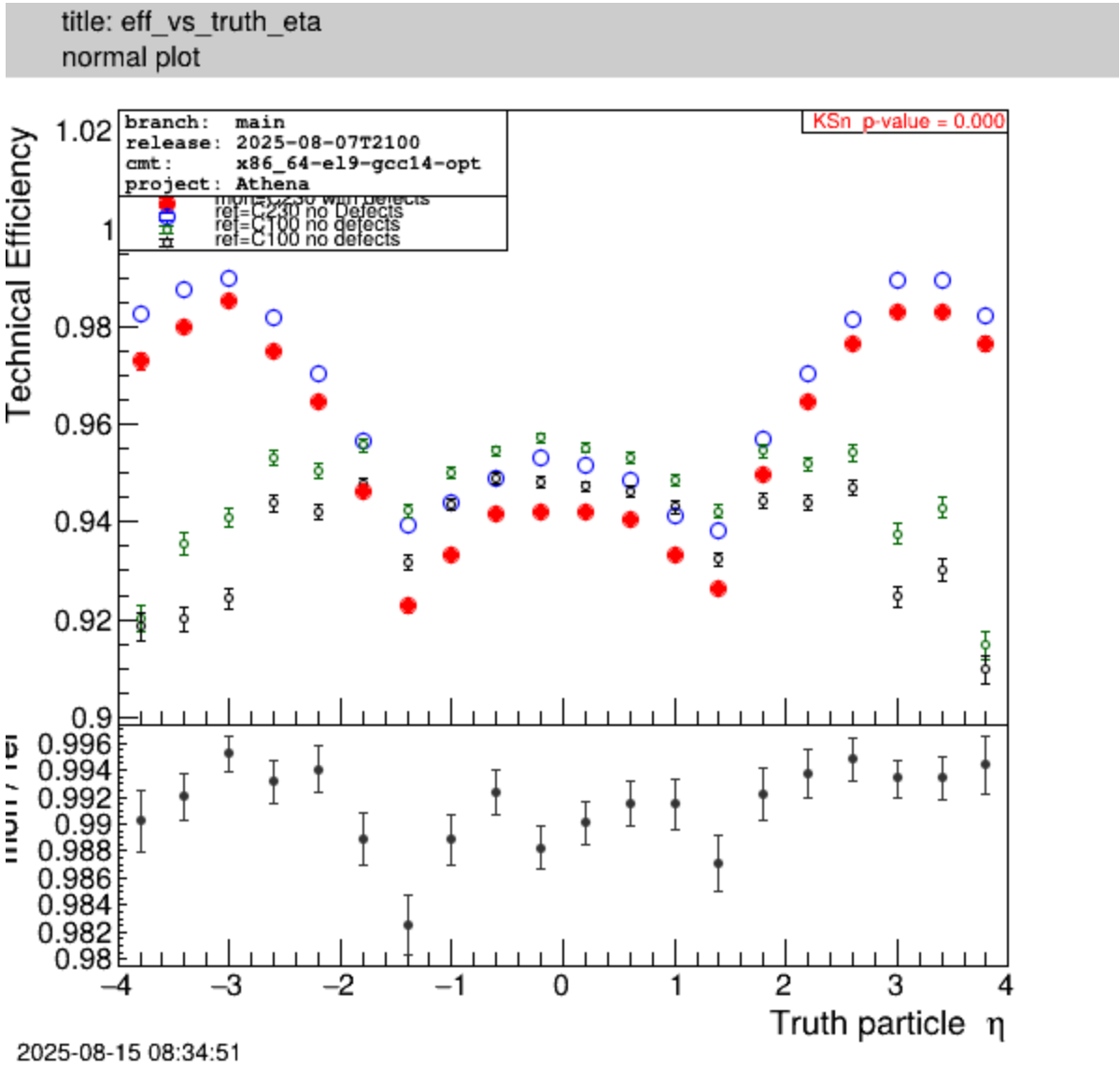
- Ran with 10000 ttbar events, pileup 0 from **mc21_14TeV.601229.PhPy8EG_A14_ttbar_hdamp258p75_SingleLep.recon.RDO.e8514_s4422_r16128**
- Ran with Athena release dated 2025-08-07
- Ran on RAL CPUs (mercury machine - 006) **Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz**

Results (All Defects)

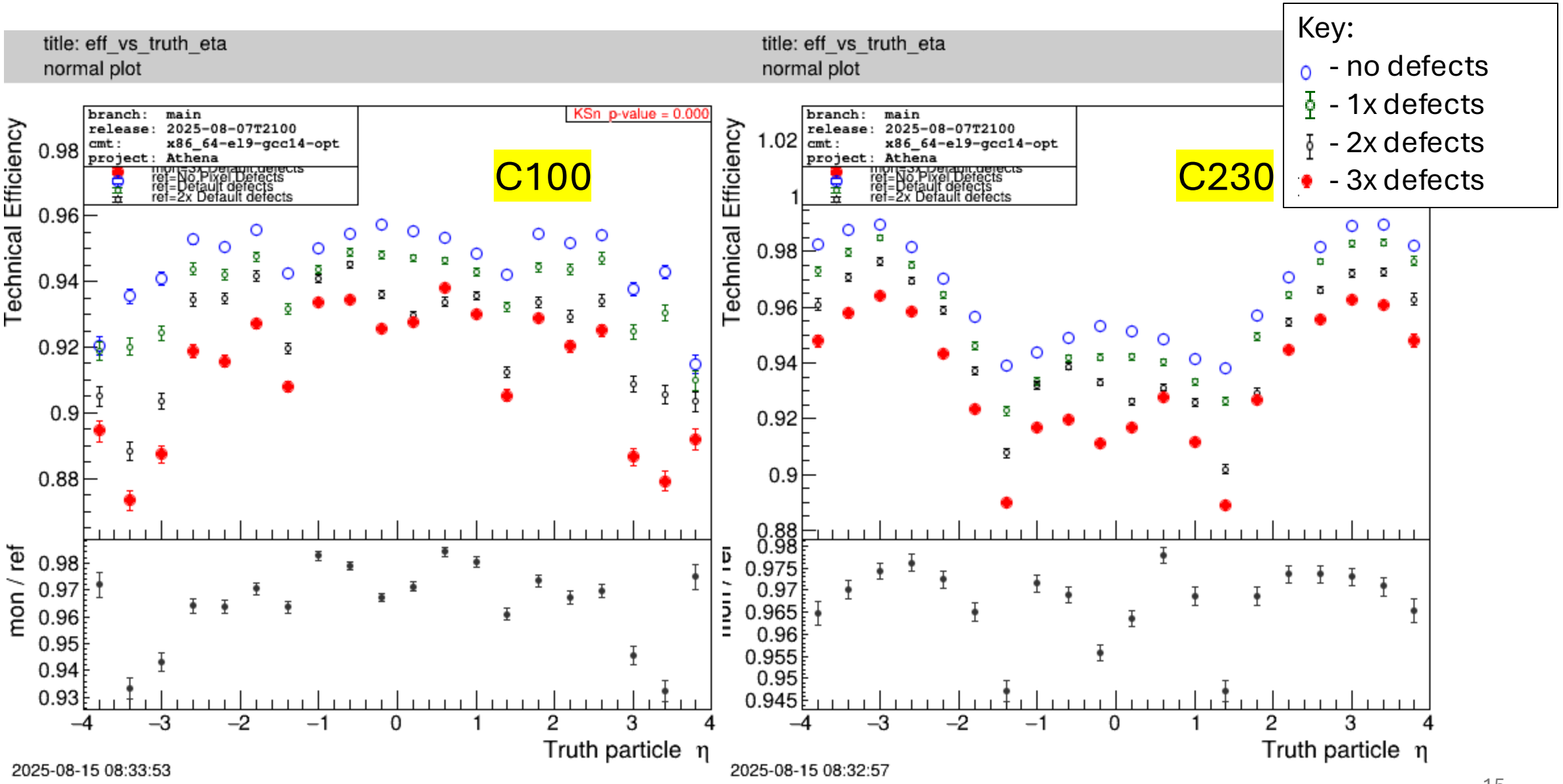
- Defects reduce efficiency and resolutions get worse
- Plots available here

Key:
- - C230 with defects
- - C230 no defects
- - C100 with defects
- - C100 no defects

Pipeline	Inclusive Efficiency
C230 with defects	(94.903 ± 0.031) %
C230 no defects	(95.790 ± 0.028) %
C100 with defects	(94.074 ± 0.033) %
C100 no defects	(94.916 ± 0.031) %



Results (Technical Efficiency – All Defects)



Results (Summary - All Defects)

- Both pipelines show similar overall drop off
- C230 drop off worse in barrel
- C100 drop off worse in endcap
- Plots other metrics for tracking performance available in More Results section

Pipeline	Inclusive Efficiency
C230 no defects	(95.790 ± 0.028) %
C230 1x defects	(94.903 ± 0.031) %
C230 2x defects	(93.861 ± 0.034) %
C230 3x defects	(92.570 ± 0.040) %

Pipeline	Inclusive Efficiency
C100 no defects	(94.916 ± 0.031) %
C100 1x defects	(94.074 ± 0.033) %
C100 2x defects	(92.890 ± 0.040) %
C100 3x defects	(91.960 ± 0.040) %

Dead Modules

- There is functionality to disable specific modules for track reconstruction
- Tests only module defects, but can control exactly where the dead modules are
- Example C230 job with dead modules:

```
Reco_tf.py --CA \
--multithreaded True \
--maxEvents '-1' \
--preInclude
'InDetConfig.ConfigurationHelpers.OnlyTrackingPreInclude,ActsConfig.ActsCIFlags.actsWorkflowFlags' \
--postInclude 'ActsConfig.ActsPostIncludes.ACTSClusterPostInclude' \
--preExec "flags.Tracking.doITkFastTracking=True; from ActsConfig.ActsConfigFlags import
SeedingStrategy;flags.Acts.SeedingStrategy=SeedingStrategy.Gbts2;flags.Tracking.doPixelDigitalCluster
ing=True; flags.ITk.JsonPathPixelModuleVeto=\"$jsonFile\"" \
--steering 'doRAWtoALL' \
--inputRDOFile ${Input_RDO} \
--outputAODFile 'OUTPUT.AOD.pool.root'
```

- Use the scripts at https://gitlab.cern.ch/atlas/athena/-/tree/main/InnerDetector/InDetExample/InDetDetDescrExample/tools to create JSON of modules to mask

Dead Modules Bug

- Running Reco_tf with dead modules mask and ACTS flags with Athena release from 2025-07-23 gave an error message

```
ActsTrackFindingAlg                                    ERROR Measurement not
clustered by identifierHash / geometryId. New measurement 197511 with geo
Id vol=8|lay=58|sen=21 type = 1 idHash=0 but already recorded for this
geo ID the range : [0]0 .. 65
```

- Tried various "hacks" in the underlying ACTS C++ backend code to fix this issue, unable to fix myself

Dead Modules fix

ATLAS EF Tracking / EFTRACK-903

Reco_tf modules veto incompatibility with ACTS

Edit | Add comment | Assign | More | Open

Details

Type:	Bug	Resolution:	Unresolved
Priority:	Minor		
Component/s:	None		
Labels:	ACTS Tracking		

Description

Ran Reco_tf with dead modules as described by https://gitlab.cern.ch/atlas/athena/-/tree/main/InnerDetector/InDetExample/InDetDetDescrExample/tools with ACTS. The full command looks like

```
Reco_tf.py --CA \
    --multithreaded 'True' \
    --maxEvents ${n_events} \
    --preInclude "InDetConfig.ConfigurationHelpers.OnlyTrackingPreInclude,ActsConfig.ActsCIFlags.actsWorkflowFlags" \
    --preExec "flags.ITk.JsonPathPixelModuleVeto=\"$jsonFile\"" \
    --postInclude 'ActsConfig.ActsPostIncludes.ACTSClusterPostInclude' \
    --athenaopts="-l DEBUG" \
    --steering 'doRAWtoALL' \
    --inputRDOFile  ${input_rdo} \
    --outputAODFile 'output.AOD.pool.root'
```

where the input RDO is **RDO.33629020._000047.pool.root.1** from dataset **mc21_14TeV.601229.PhPy8EG_A14_ttbar_hdamp258p75_SingleLep.recon.RDO.e8481_s4149_r14700** and my json file of modules (the whole inner barrel) to mask is attached.
Produces error:

```
ActsTrackFindingAlg                             ERROR Measurement not clustered by identifierHash / geometryId. New measurement
197511 with geo Id vol=8|lay=58|sen=21 type = 1 idHash=0 but already recorded for this geo ID the range : [0]0 .. 65
```

Error seems to occur in https://acode-browser1.usatlas.bnl.gov/lxr/source/athena/Tracking/Acts/ActsTrackReconstruction/src/detail/TrackFindingMeasurements.cxx

- Opened JIRA ticket EFTRACK-903

Dead Modules fix (cont.)

- Update in Athena MR81793 fixes this issue

Fix Acts clusterization with bad modules.

Merged **Goetz Gaycken** requested to merge `goetz/athena:main_fix_ac…` into `main` 1 week ago

Overview 4 | Commits 1 | Pipelines 1 | Changes 2

Add at least an empty cluster collection per RawDataCollection, because this is the expectation, and cluster collections might be associated to the wrong detector element otherwise.

Supposed to address EFTRACK-903

Edited 1 week ago by Goetz Gaycken

👍 0 👎 0 🚀 1

- Have since run **successfully** with Athena release from 2025-08-10

Pipeline #12616770 passed

Pipeline passed for `beac2efc` on `goetz:main_fix_acts…` 1 week ago

Approval is optional — Assign reviewers

Merged by Thomas Strebler 1 week ago — Revert — Cherry-pick

Merge details

- Changes merged into `main` with 11955229 (commits were squashed).
- Did not delete the source branch.
- Mentions issue EFTRACK-903

Barrel vs Endcap

- Ran with same set of 10000 ttbar events, pileup 0 from **mc21_14TeV.601229.PhPy8EG_A14_ttbar_hdamp258p75_SingleLep.recon.RDO.e8514_s4422_r16128**

- Ran with defects in different parts of detector (barrel and endcap defects)

- Ran with Athena release from 2025-08-10

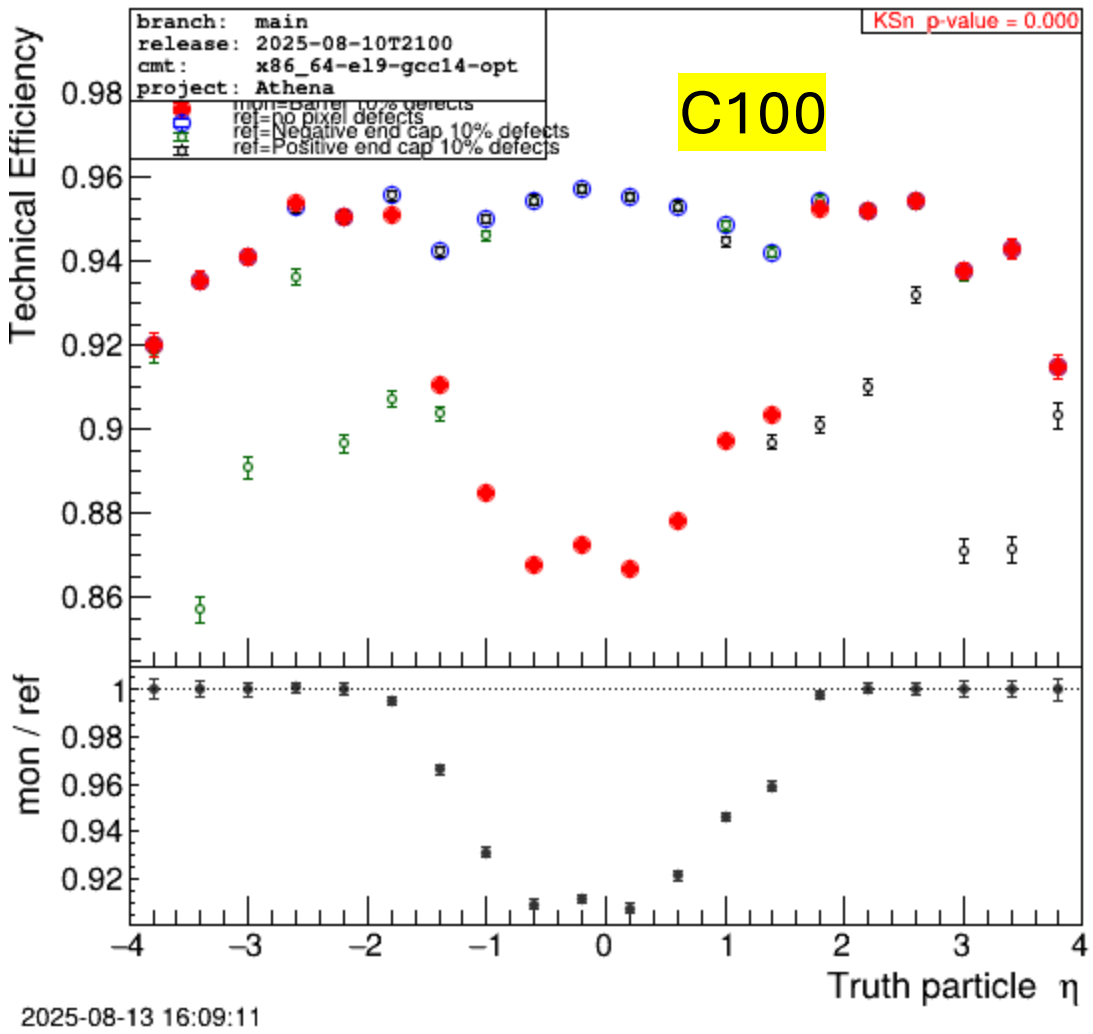
Available at:
https://hepunx.rl.ac.uk/webtempfiles/jasonn/dcube_store_C100_barrel_endcap/ (C100)

https://hepunx.rl.ac.uk/webtempfiles/jasonn/dcube_store_C230_barrel_endcap/ (C230)

Results (Barrel vs Endcap)

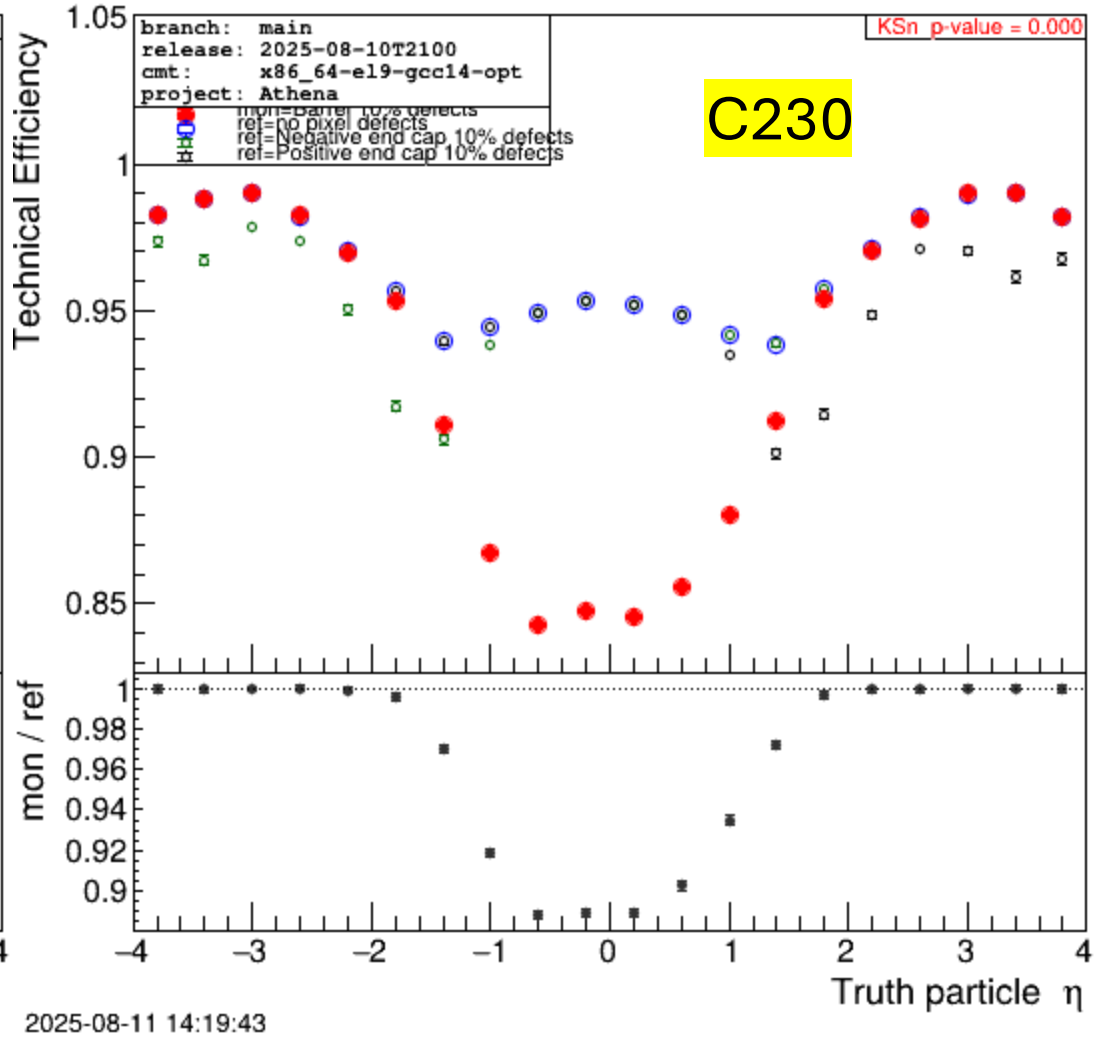
Key:
- - no defects
- - 10% negative endcap defects
- - 10% positive endcap defects
- - 10% barrel defects

title: eff_vs_truth_eta
normal plot



2025-08-13 16:09:11

title: eff_vs_truth_eta
normal plot



2025-08-11 14:19:43

Results (Summary - Barrel vs Endcap)

- Barrel defects seem more impactful to both pipelines
- Seems to confirm earlier intuition that:

C230 drop off worse in barrel

C100 drop off worse in endcap

- Plots other metrics for tracking performance available in More Results section

Pipeline	Inclusive Efficiency
C230 no defects	(95.790 ± 0.028) %
C230 10% barrel defects	(91.070 ± 0.040) %
C230 10% positive endcap defects	(95.006 ± 0.031) %
C230 10% negative endcap defects	(95.121 ± 0.030) %

Pipeline	Inclusive Efficiency
C100 no defects	(94.916 ± 0.031) %
C100 10% barrel defects	(90.870 ± 0.040) %
C100 10% positive endcap defects	(93.702 ± 0.034) %
C100 10% negative endcap defects	(93.782 ± 0.034) %

Summary

- Compared C230 and C100 performance with various kinds of defects
- General trend: C230 drop off worse in barrel, C100 drop off worse in endcap
- Found a bug in ACTS tracking pipeline (later fixed by Goetz Gaycken)
- Ways to improve C230 performance with defects currently under investigation (work in progress)
- Documentation page at CodiMD link

Personal Reflection

- Would not advise on commuting to RAL from London (2 hours 40 minutes either way)
- Always check where your train is going (see photo)
- PPD Public Access Day was a highlight!

Backup

Definitions of Performance Metrics

- "The technical efficiency is calculated by additionally requiring that primary truth particles satisfy the same hit criteria used in track reconstruction. This ensures that only reconstructable truth particles are considered, effectively removing inefficiencies from hadronic interactions with detector material, which largely impact the physics efficiency of the tracking algorithms" [3]

Definition 5.4: *Tracking efficiency*

Tracking efficiency is defined as the fraction of primary truth particles matched with a track passing a quality selection. The tracking efficiency can be also measured with respect to offline tracks through a double-ratio measurement in each bin (ratio of efficiency with respect to truth over offline efficiency with respect to truth).

Definition 5.6: *Resolution on track parameters*

The resolution on a track parameter is defined as the RMS of the core distribution of the residual difference between the reconstructed and the matched-truth values of the parameter. If more than one track matches, the one with the highest P_{match} is taken as reference.

[3] The ATLAS Collaboration, *ATLAS TDAQ Phase-II Upgrade: EF Tracking C-100/230 Pipeline Report*, https://cds.cern.ch/record/2939310/

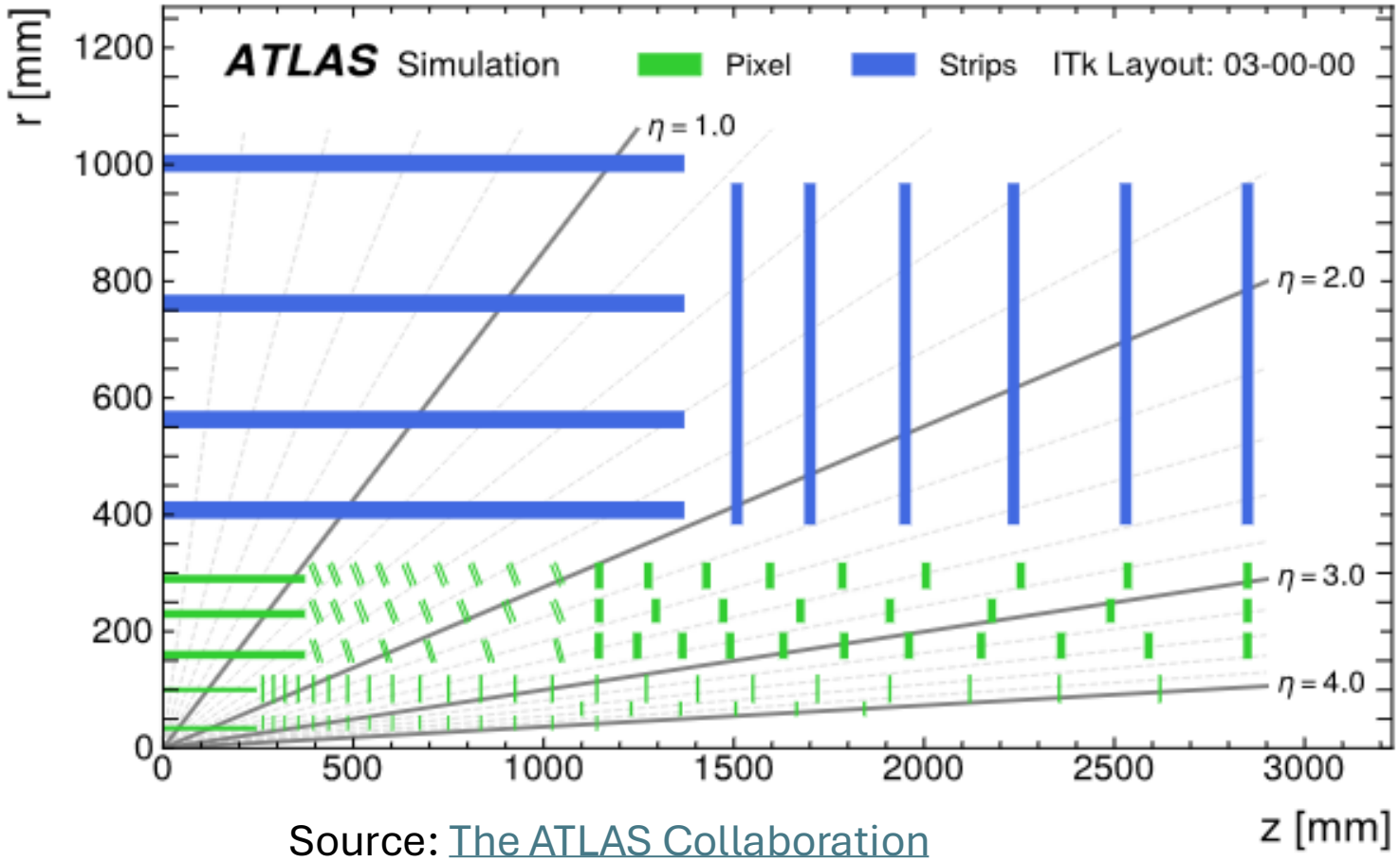
Pseudorapidity η

- Defined as $\eta = -\ln \tan(\theta/2)$

where θ is measured from positive z axis

- Can see how different values of η correspond to different parts of ITk

- Negative value for the other endcap



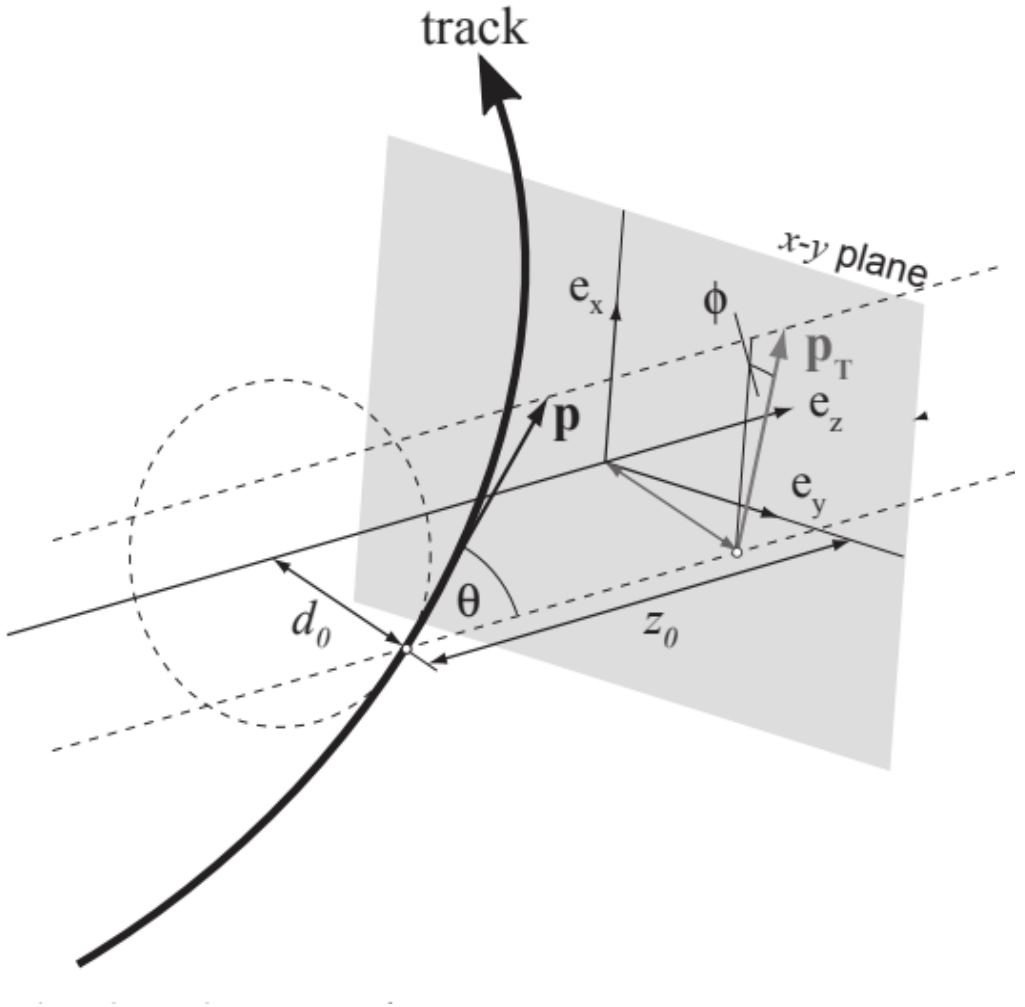
Source: The ATLAS Collaboration

Perigee Representation

- Tracks represented by 5 parameters

Global track parameters e.g. wrt. perigee

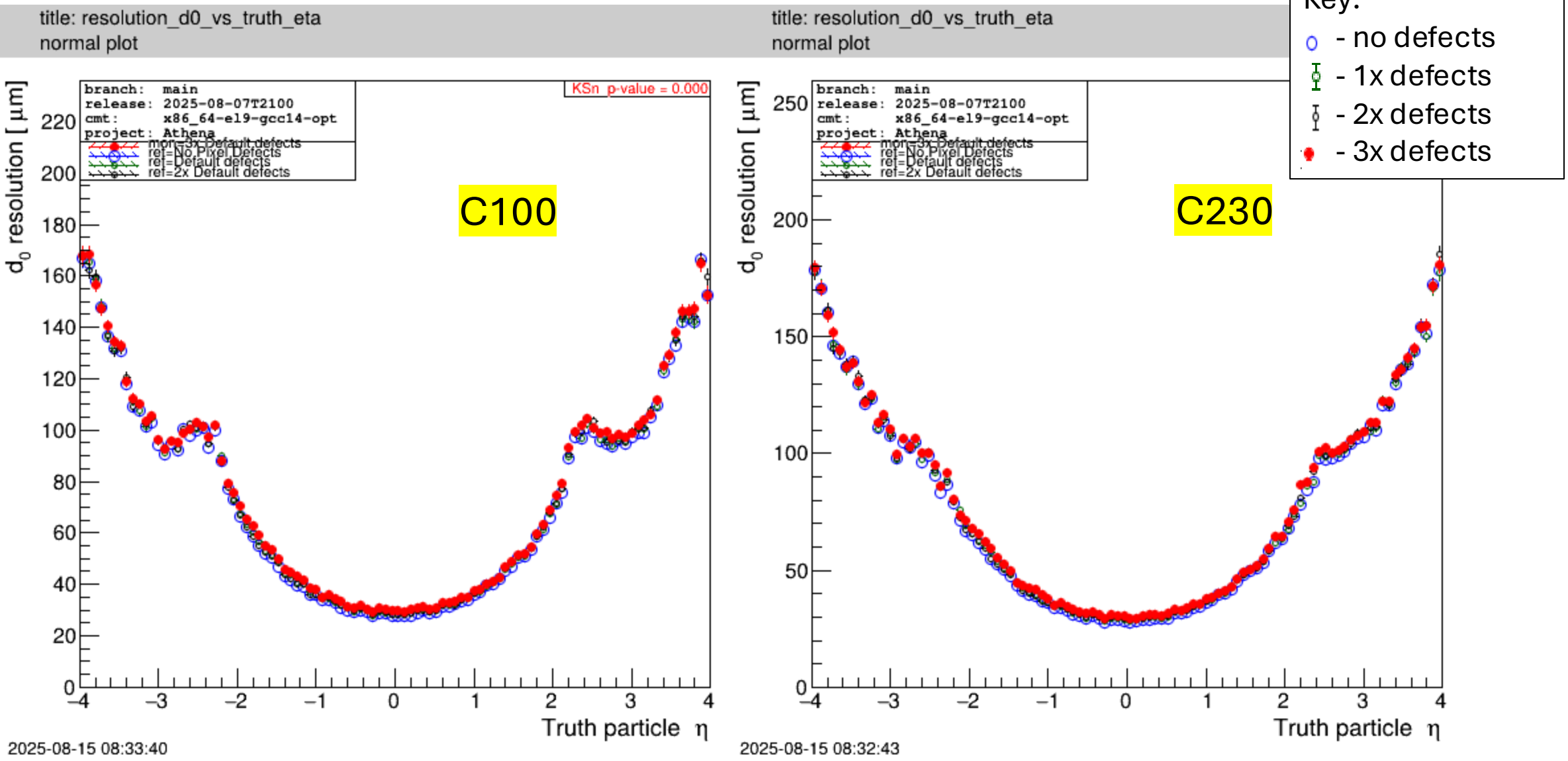
$$\left(d_0, z_0, \phi, \theta, \frac{q}{p}\right)$$



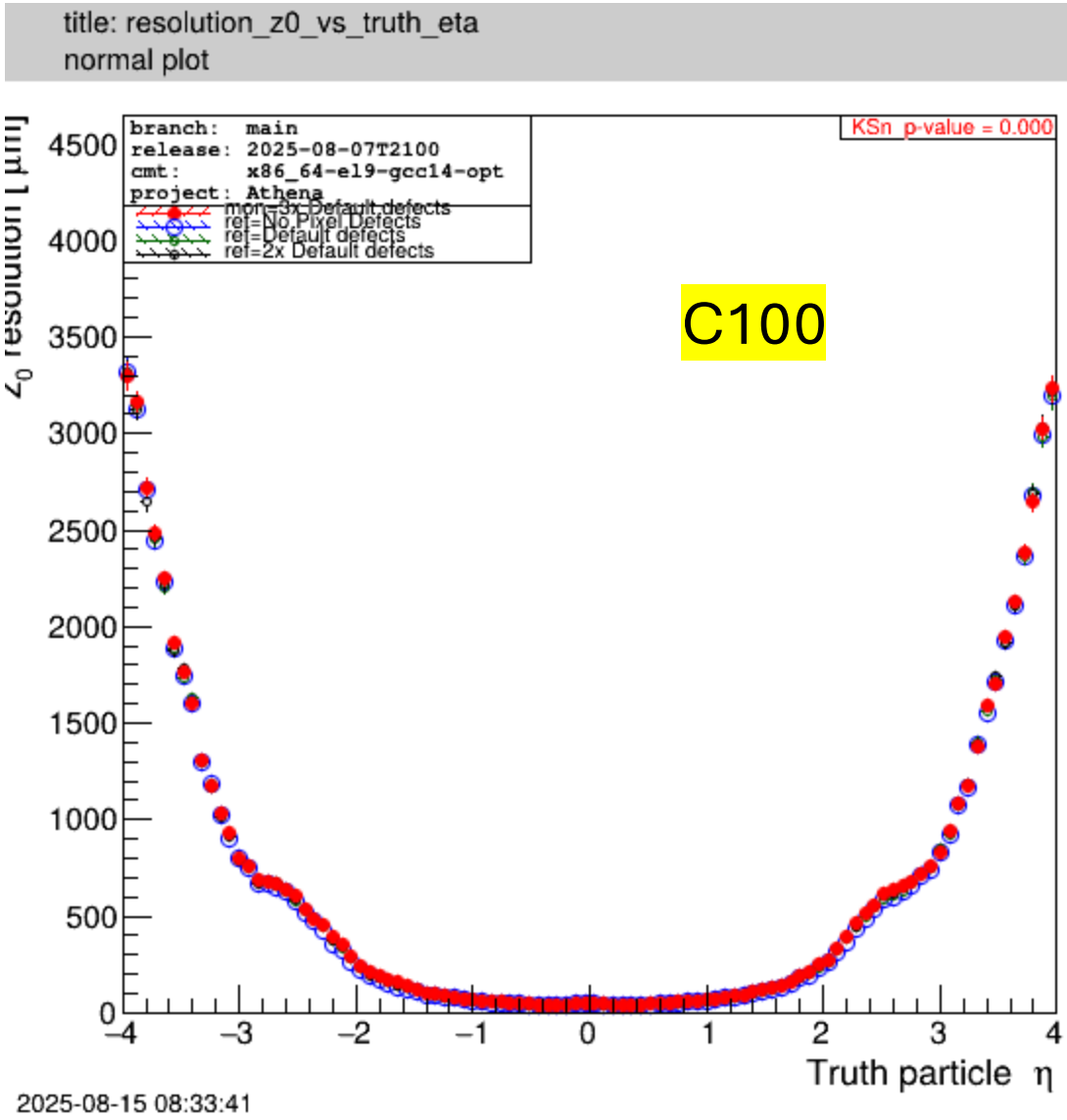
Source: ATLAS Software Documentation (Goetz Gaycken)

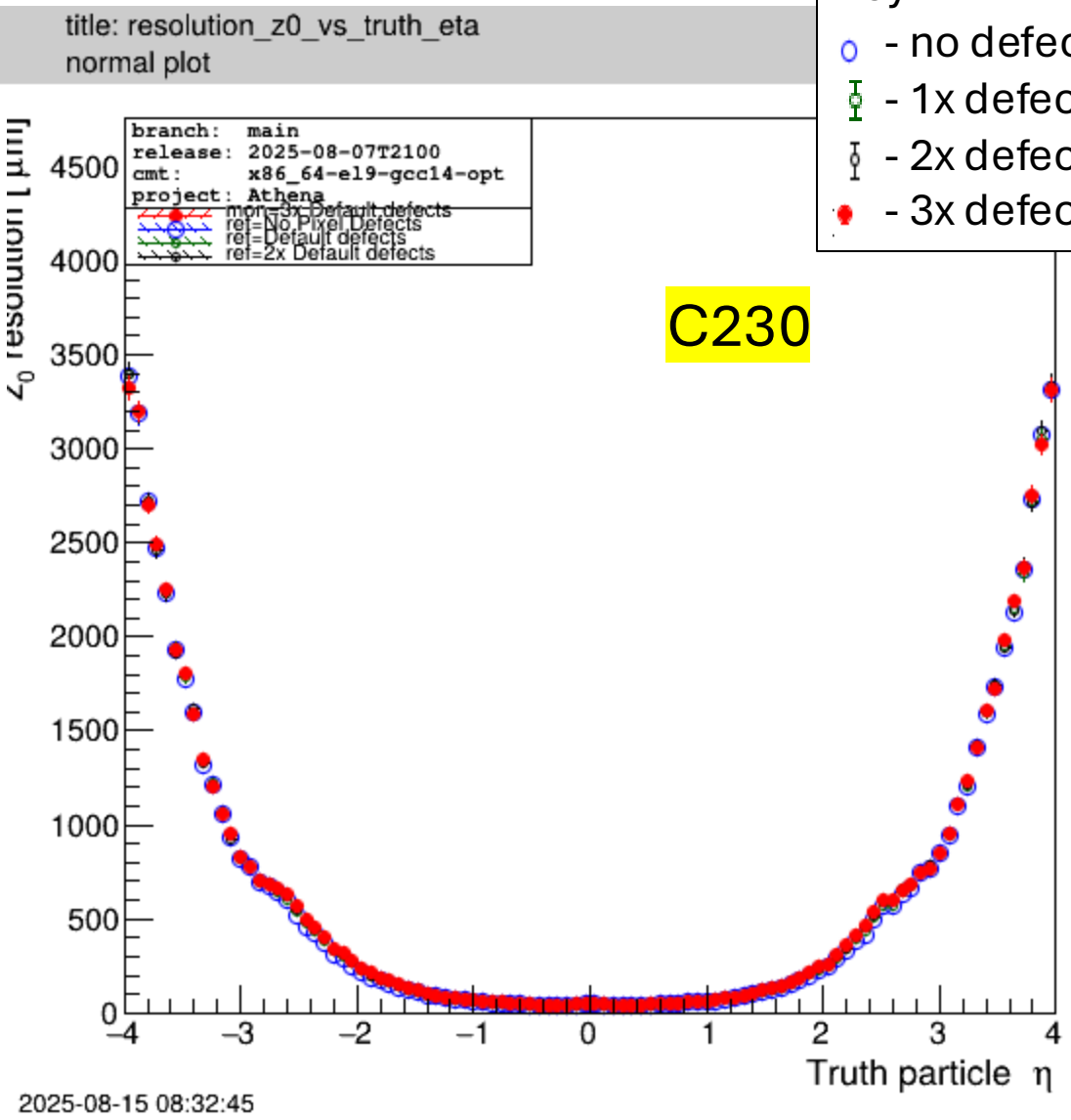
More results

Results (d0 resolution – All Defects)



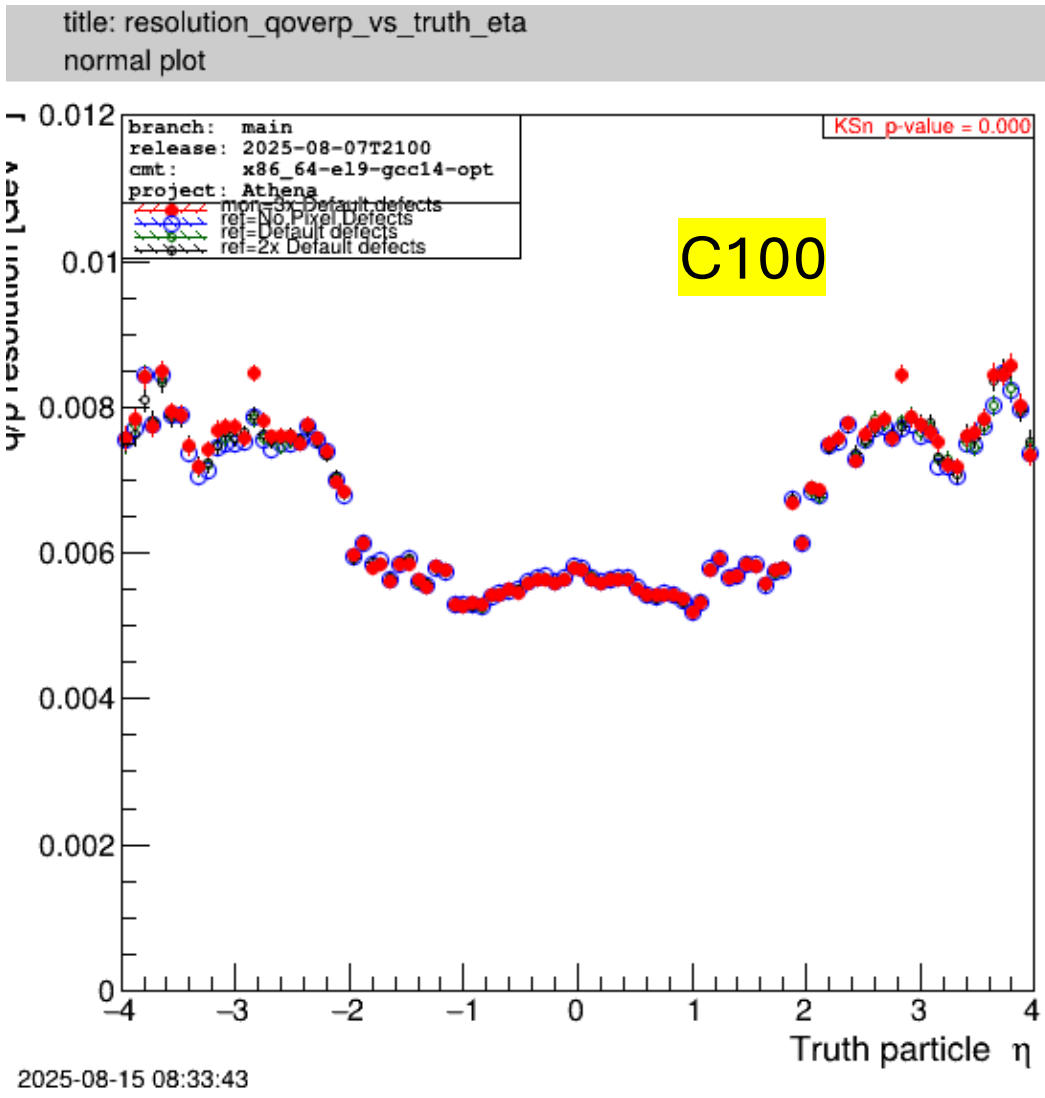
Results (z0 resolution – All Defects)

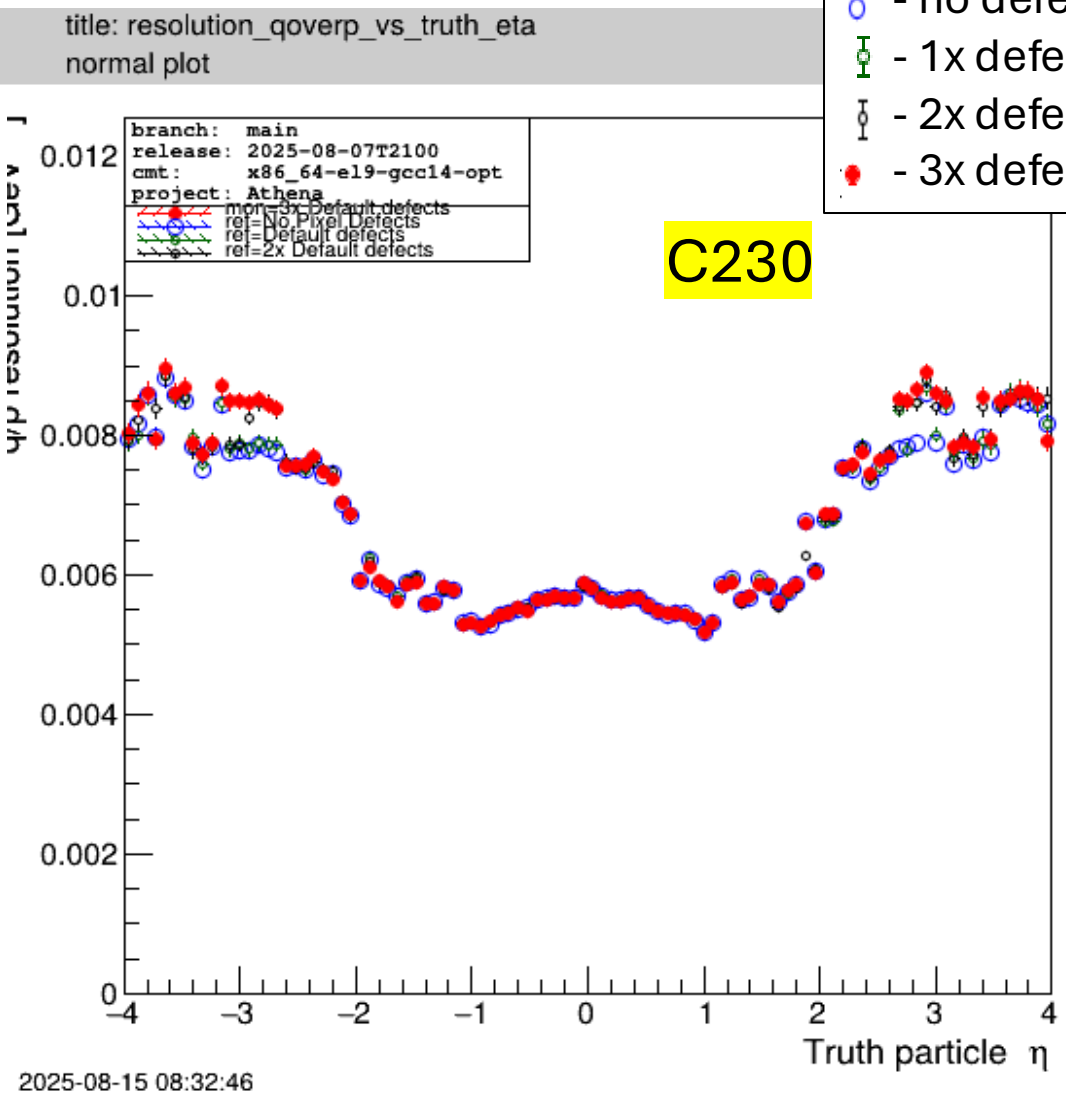




Key:
- no defects
- 1x defects
- 2x defects
- 3x defects

Results (q/pT resolutions – All Defects)





Key:
- no defects
- 1x defects
- 2x defects
- 3x defects

Results (Seeding Only – All Defects)

- Can look at the results of seeding only

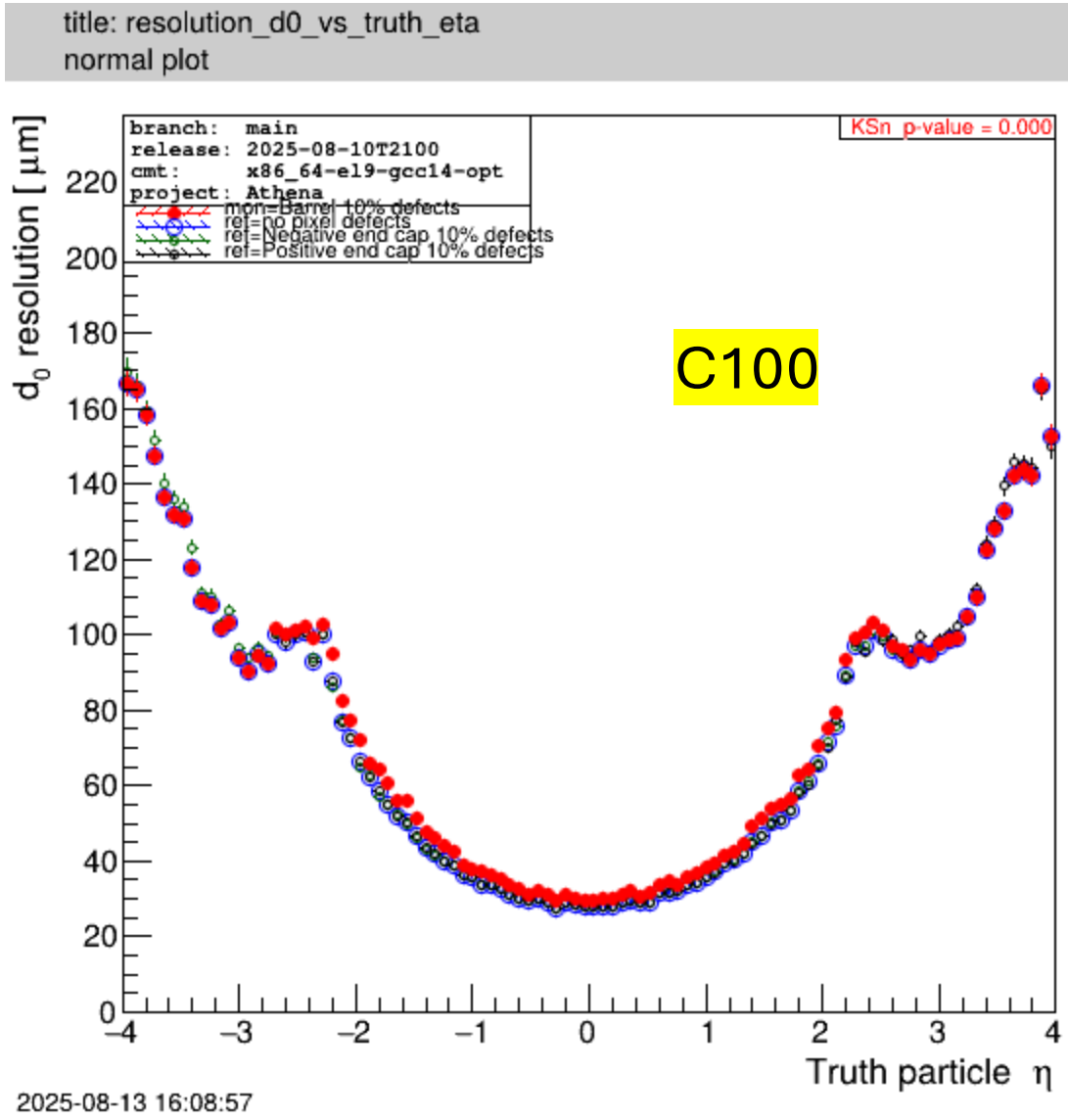
Pipeline	Spacepoints	Tracklets	Inclusive Efficiency
C230 no defects	39356290	1281000	(95.790 ± 0.028) %
C230 1x defects	536204655	1285860	(94.903 ± 0.031) %
C230 2x defects	1009943930	1690577	(93.861 ± 0.034) %
C230 3x defects	1465504208	3153308	(92.570 ± 0.040) %

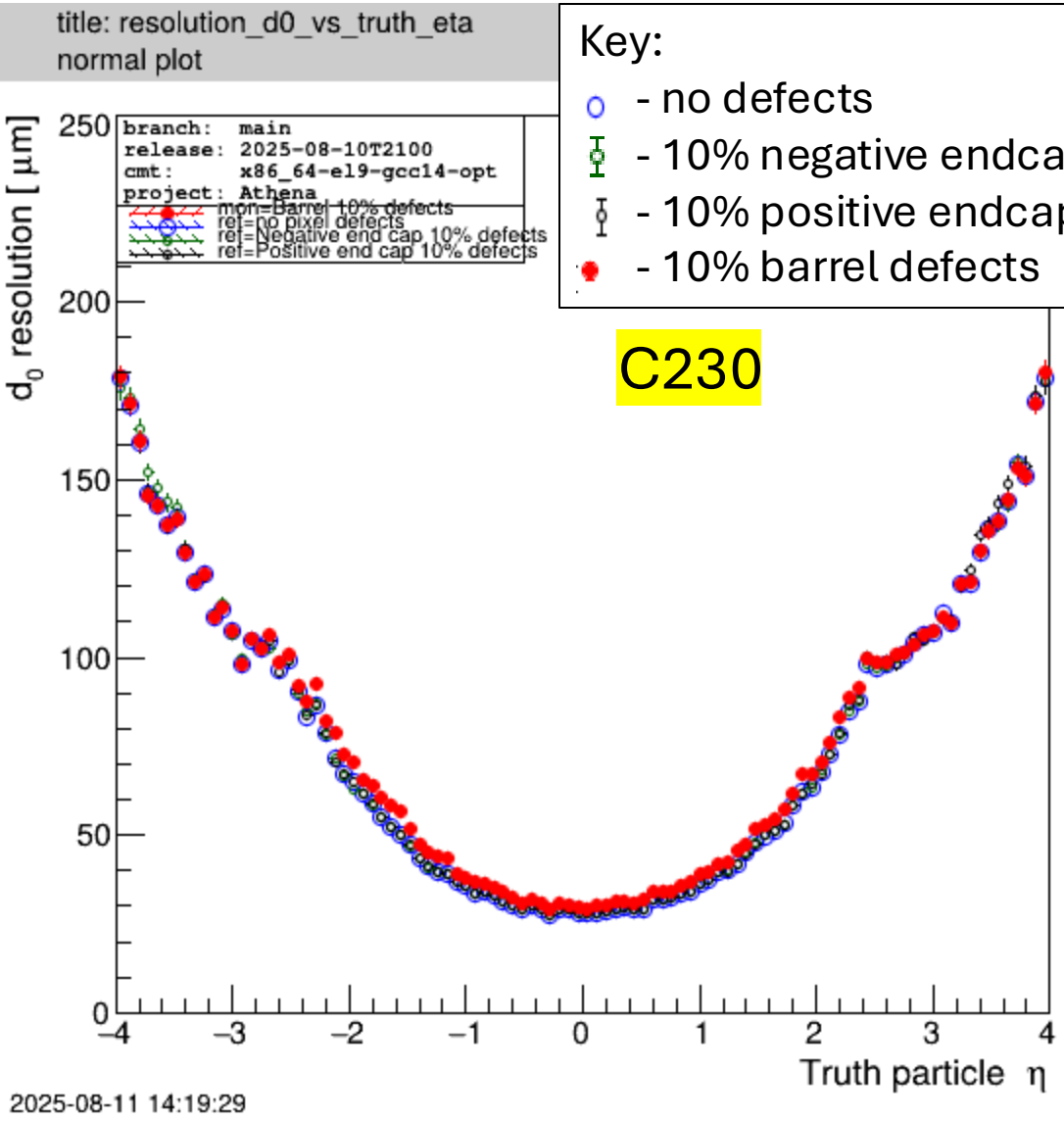
Pipeline	Spacepoints	Seeds	Inclusive Efficiency
C100 no defects	39356290	2978154	(94.916 ± 0.031) %
C100 1x defects	536204655	3676331	(94.074 ± 0.033) %
C100 2x defects	1009943930	8581953	(92.890 ± 0.040) %
C100 3x defects	1465504208	21977642	(91.960 ± 0.040) %

- Spacepoints increase by orders of magnitude

- Possible if you lose enough hits from a cluster, the clustering algorithm then splits clusters into smaller clusters

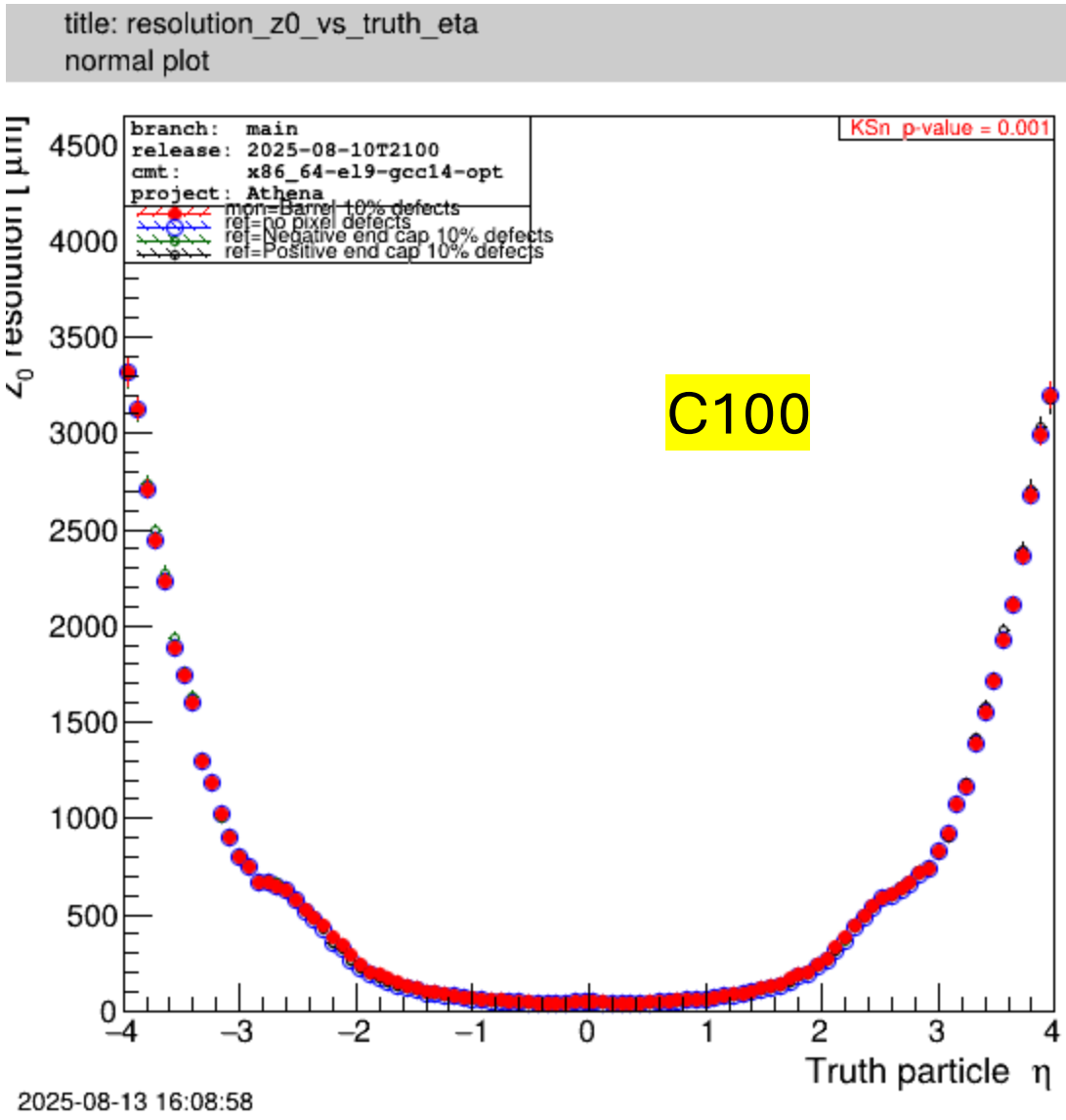
Results (d0 resolution – Barrel vs Endcap)

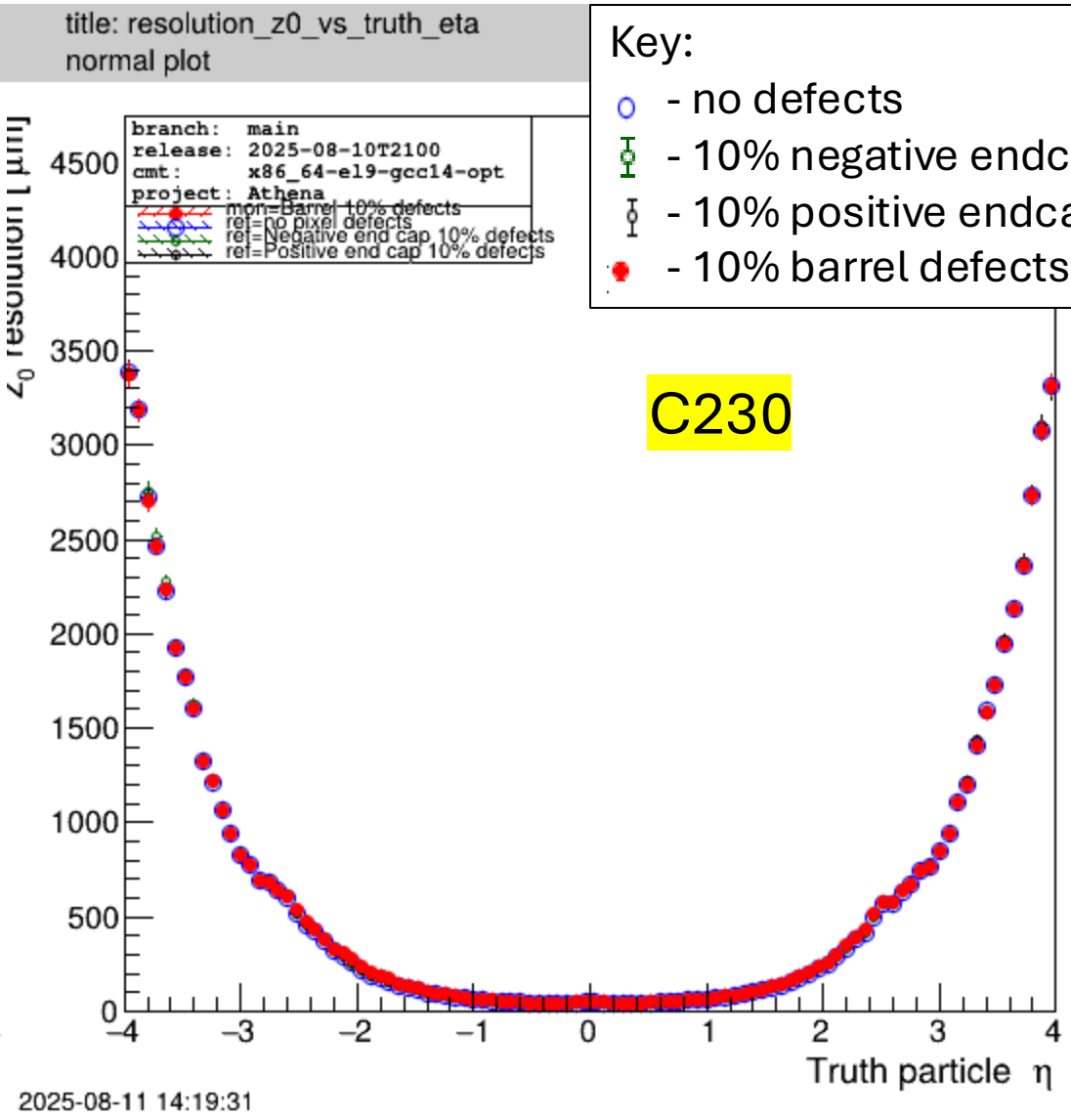




Key:
- no defects
- 10% negative endcap defects
- 10% positive endcap defects
- 10% barrel defects

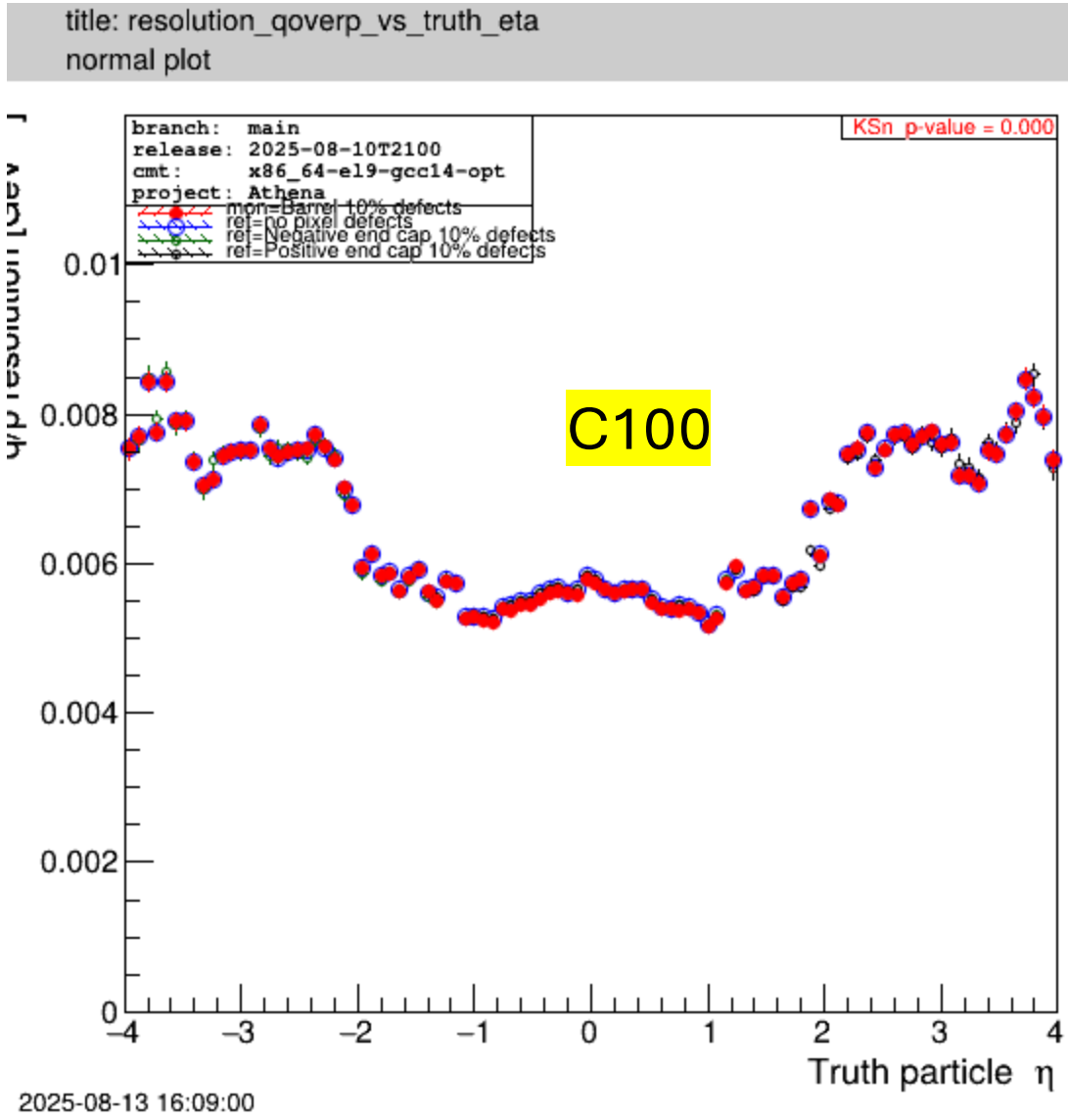
Results (z0 resolution – Barrel vs Endcap)

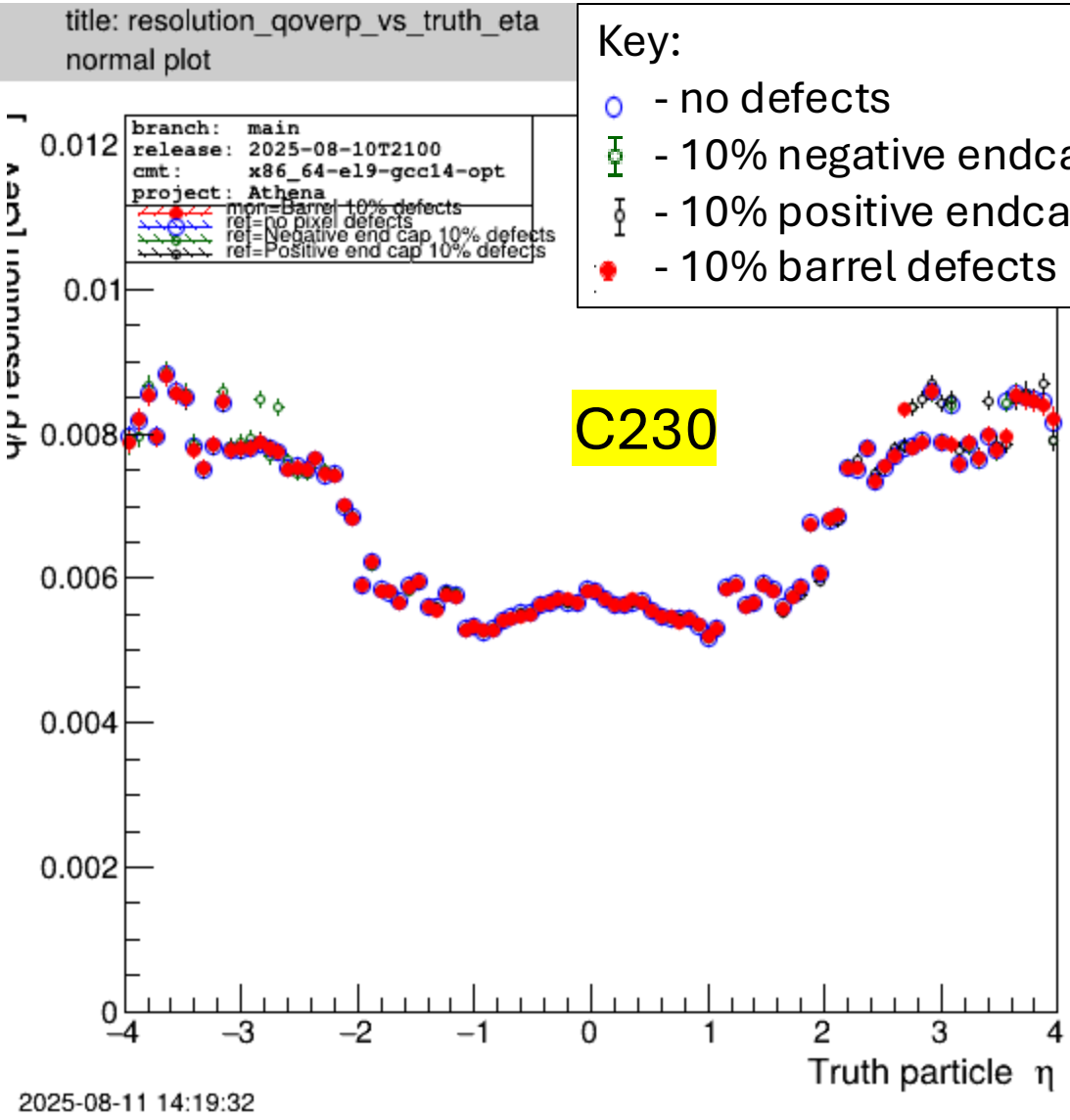




Key:
- no defects
- 10% negative endcap defects
- 10% positive endcap defects
- 10% barrel defects

Results (q/pT resolutions – Barrel vs Endcap)





Key:
- no defects
- 10% negative endcap defects
- 10% positive endcap defects
- 10% barrel defects

Results (Seeding Only – Barrel vs Endcap)

Pipeline	Spacepoints	Tracklets	Inclusive Efficiency
C230 no defects	39356290	1281000	(95.790 ± 0.028) %
C230 10% barrel defects	38570391	1211485	(91.070 ± 0.040) %
C230 10% positive endcap defects	37820458	1207215	(95.006 ± 0.031) %
C230 10% negative endcap defects	37754428	1206990	(95.121 ± 0.030) %

Pipeline	Spacepoints	Seeds	Inclusive Efficiency
C100 no defects	39356290	2978154	(94.916 ± 0.031) %
C100 10% barrel defects	38570391	2733927	(90.870 ± 0.040) %
C100 10% positive endcap defects	37820458	2800919	(93.702 ± 0.034) %
C100 10% negative endcap defects	37754428	2806450	(93.782 ± 0.034) %

- Lose seeds/tracklets due to module defects (module defects affect bigger area than cluster so probably losing entire clusters)

- Barrel defects seem more impactful to both pipelines

Percentage masks

- Ran with same set of 10000 ttbar events, pileup 0 from **mc21_14TeV.601229.PhPy8EG_A14_ttbar_hdamp258p75_SingleLep.recon.RDO.e8514_s4422_r16128**

- Ran with 1%, 5%, 10% and 15% pre-made modules masks already created for performance testing

- Ran with Athena release from 2025-08-10

Available at:
https://hepunx.rl.ac.uk/webtempfiles/jasonn/dcube_store_C100_percentages/ (C100)

https://hepunx.rl.ac.uk/webtempfiles/jasonn/dcube_store_C230_percentage_masks/ (C230)

Results (Percentage Masks)