



Production Pipeline Aurora

Wirecell Meeting

Gen-2 1000-event benchmark · WCT non-determinism: production version and fix validation

Avinay Bhat
10/01/2026

Gen-2 Aurora production benchmark is ready for review

Official production configuration

- SBND Gen-2 BNB Light Run-1 MC
- sbndcode v10_14_02_03
- stock WCT v0_32_1 (not overridden)
- official FCL chain:
 - GEN → G4 → DetSim → Reco1 → Reco2 → CAF
- SBN-ALCF Parsl workflow (Thomas), adapted for retained outputs
- one Aurora node, capacity queue

Benchmark result — PBS 8875659

5000 → 1328

GEN attempts → accepted events
(50 subruns × 100 attempts)

26.56%

acceptance

50 / 50

G4, Reco1 and flat CAF files
(each product complete)

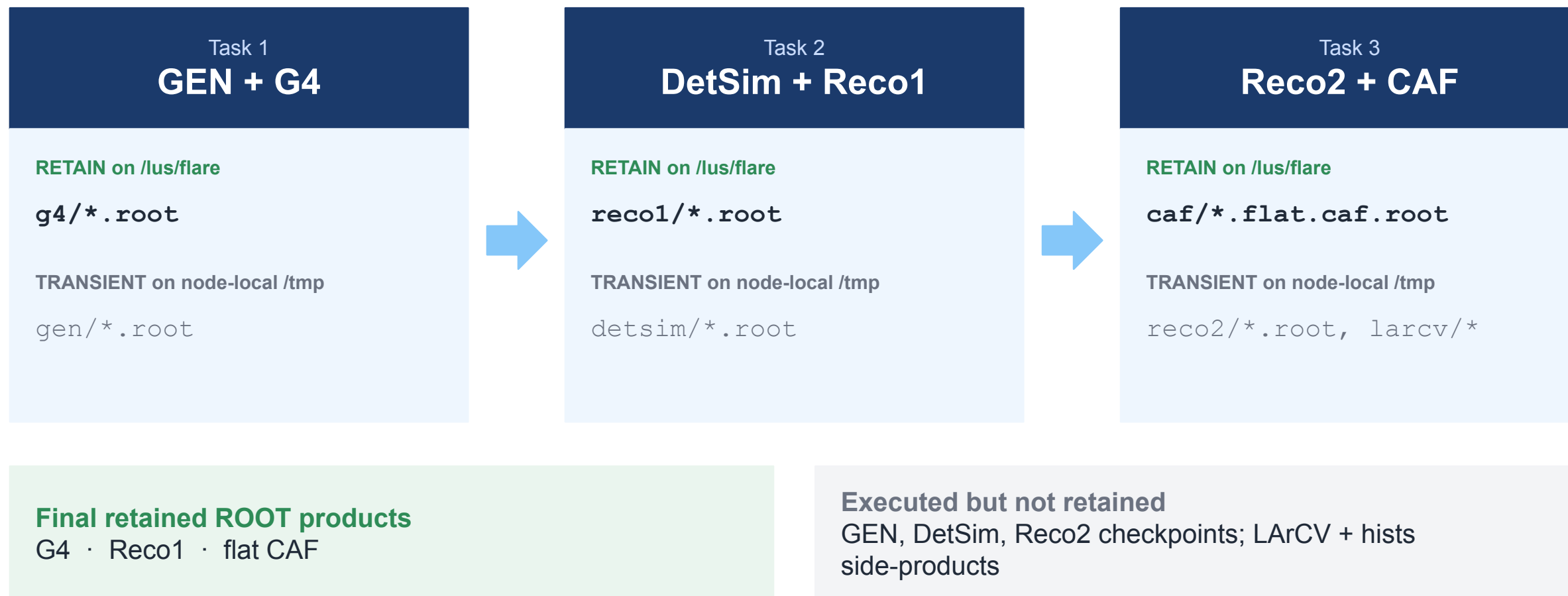
0

TASK_FAILED · 0 retries

Wall time 3:51:56 of 8:00:00 allocated. **PBS Exit_status 271** = SIGTERM during Parsl executor teardown, after all tasks had completed and all 150 outputs were written.

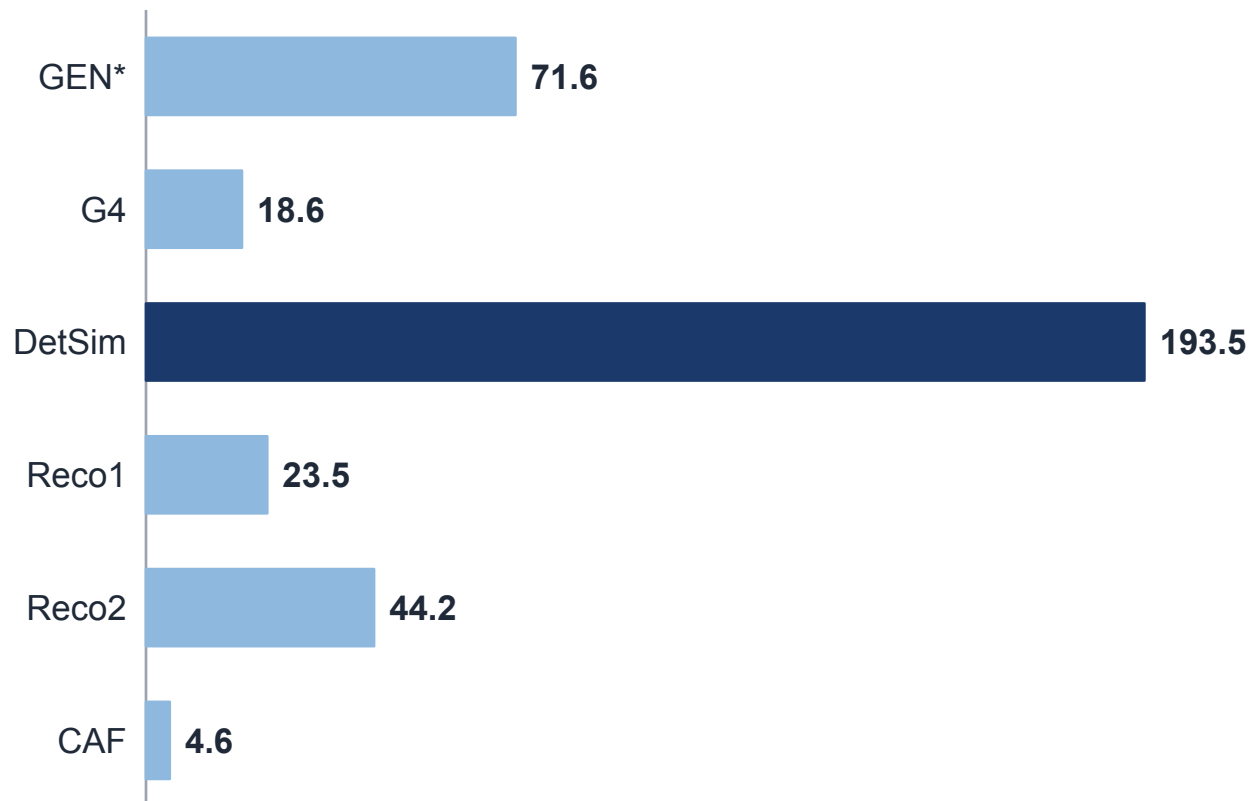
Retained-output Parsl workflow

Three combined Parsl tasks per subrun · each task = two sequential lar invocations sharing node-local /tmp · same physics chain, less persistent storage



Benchmark performance and bottlenecks

Average art Real time per final accepted event (s)



*GEN = 19.0 s per GEN attempt; 71.6 s per final event after the 26.56% acceptance. Averages over 50 subruns.

343.5

final accepted events / node-hour
(1328 events ÷ 3.866 h, one node)

~2911

node-hours per 1M final events
at measured single-node throughput

6.35 GiB

DetSim avg VmHWM / worker (6822 MB)
no OOM at the concurrency used

Conservative estimate: the node was not saturated (≤ 50 active subrun chains on 102 worker slots), so wall time $\approx$ latency of the slowest chain.

1M-event projection

3.765M

GEN attempts for 1M accepted events
(1 / 0.2656)

~2911

node-hours at measured throughput

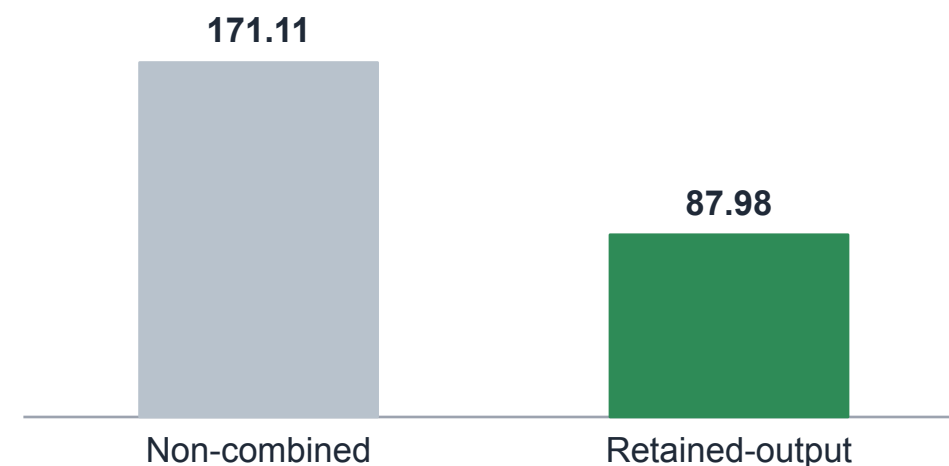
87.98 TiB

retained storage for 1M events

Retained product	MiB / event	TiB / 1M
G4 ROOT	64.21	61.24
Reco1 ROOT	27.37	26.10
flat CAF ROOT	0.68	0.65
Total retained	92.25	87.98
Non-combined baseline (all ROOT)	179.42	171.11

Baseline: 169.5 GB over 901 accepted events from the earlier non-combined run (34 subruns; includes hist, LArCV, DetSim, Reco2). Final retained bytes only — not total /lus/flare write traffic.

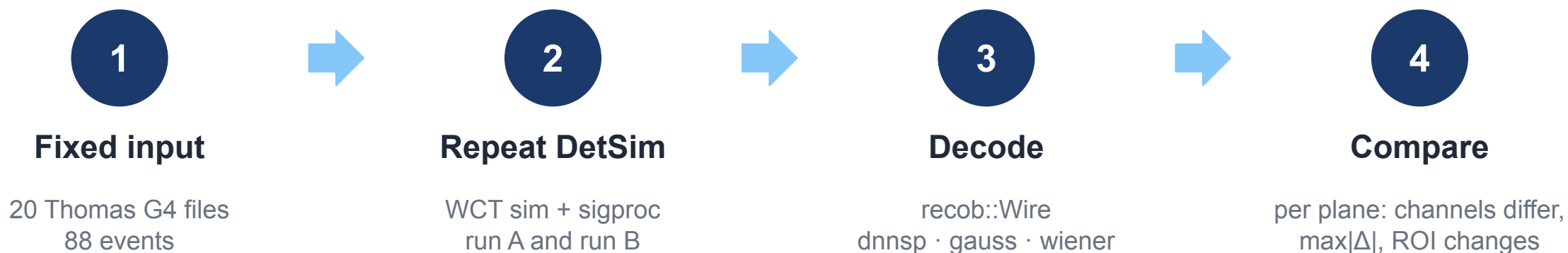
Final retained storage, 1M events (TiB)



-83.13 TiB saved per 1M events (≈48%)

WCT non-determinism: why and how we tested

Motivation Haiwang, Joseph, Lynn, Thomas and others observed Wire-Cell run-to-run differences. We asked whether a residual effect exists in the Gen-2 production chain, and where it lives.



WCT versions tested

- v0_32_1 — production version (stock)
- old master 7a85e5c5
- new master 51b5a1fc — includes Haiwang's fix

Relevant WCT commits

- 47d16673 FFTW plan cache · 36489a20 c_data_afterfilter init (both masters)
- 68d3a99d + 98846706 stale contained_rois cleanup (new master only)

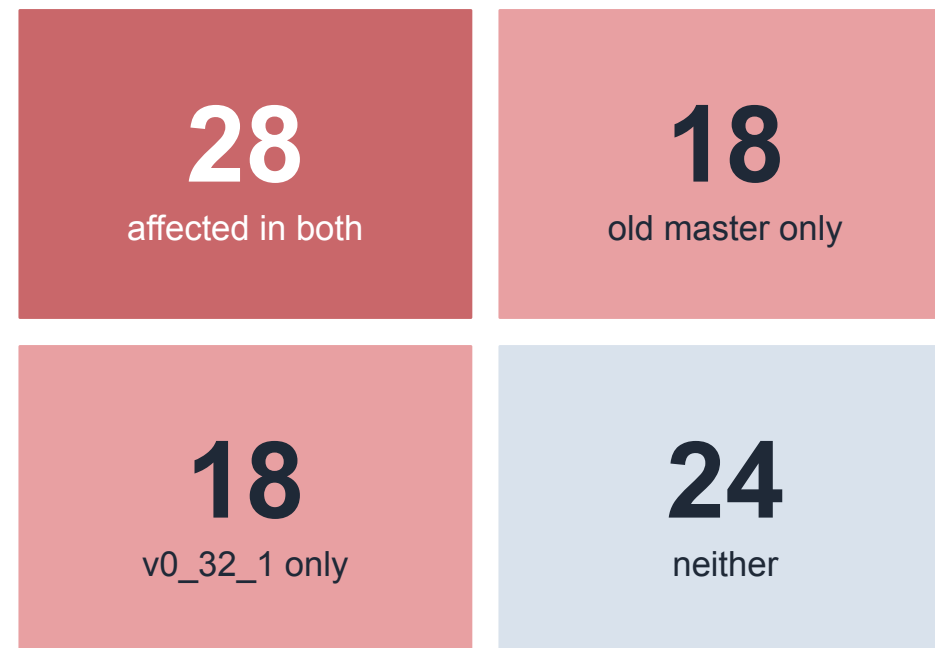
Population result: old master fixes dnnsp, not gauss/wiener

88 events from 20 fixed G4 files · events with any A/B difference

Branch	WCT v0_32_1	old master 7a85e5c5
dnnsp	16 / 88	0 / 88 ✓
gauss	46 / 88	46 / 88
wiener	46 / 88	46 / 88

- Old master removes the observed dnnsp A/B variation
- Traditional gauss/wiener non-determinism remains
- Same incidence in both versions: 46 / 88

But which events are affected changes



Jaccard overlap = 0.4375 (28 / 64)

Gauss and wiener differences are perfectly correlated

Event-level contingency, 88 events (identical for old master and v0_32_1)

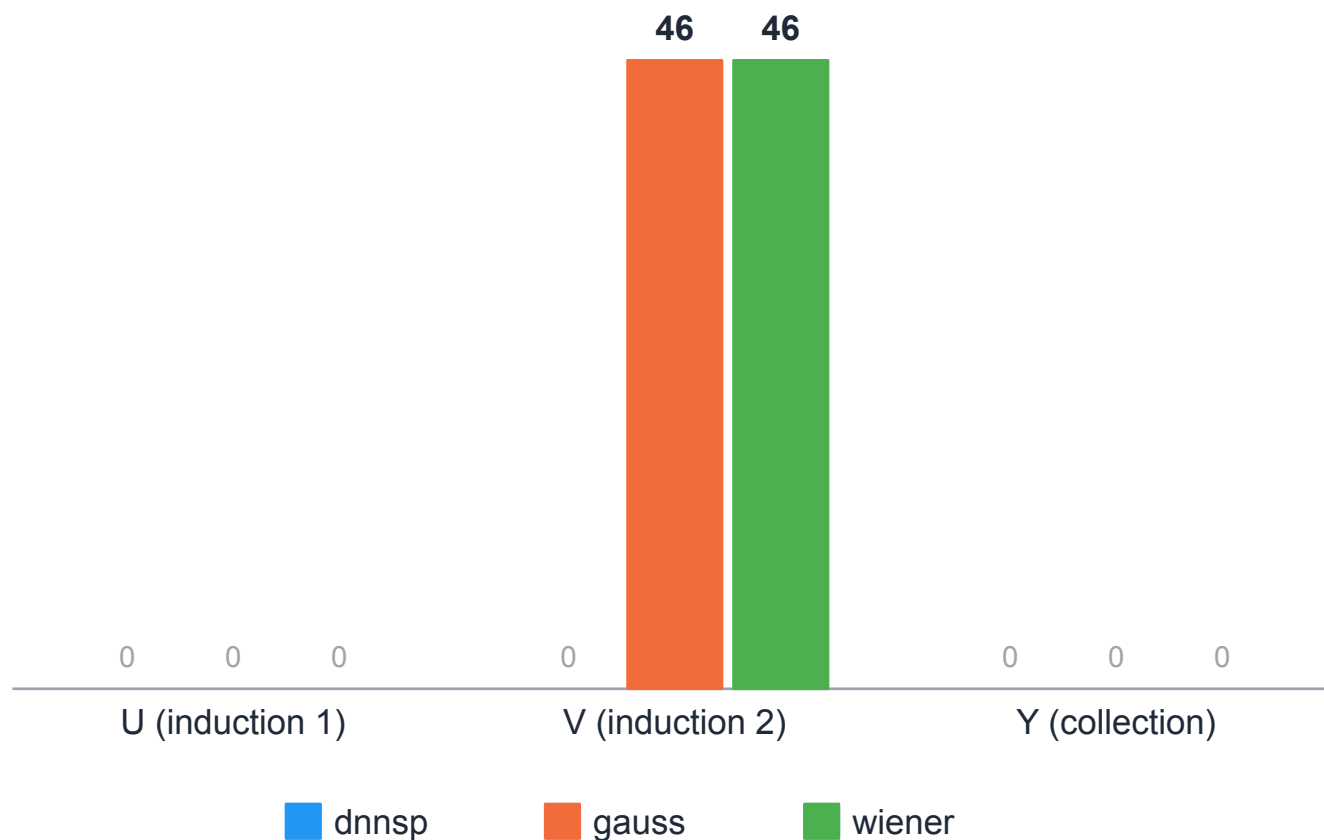
	wiener differs	wiener same
gauss differs	46 both differ	0 gauss only
gauss same	0 wiener only	42 neither

Interpretation

- Co-occurrence is exact at event level, in both versions
- Consistent with a shared source upstream of both filter outputs
- Not two independent effects in the Gaussian and Wiener filters
- Pointed the search at the common induction-ROI path — where the bug was found

Plane breakdown: old master residual is V-plane only

Events with differences per plane — old master 7a85e5c5, run A vs run B



195 / 195

differing channel-events on V
(gauss; identical for wiener)

0

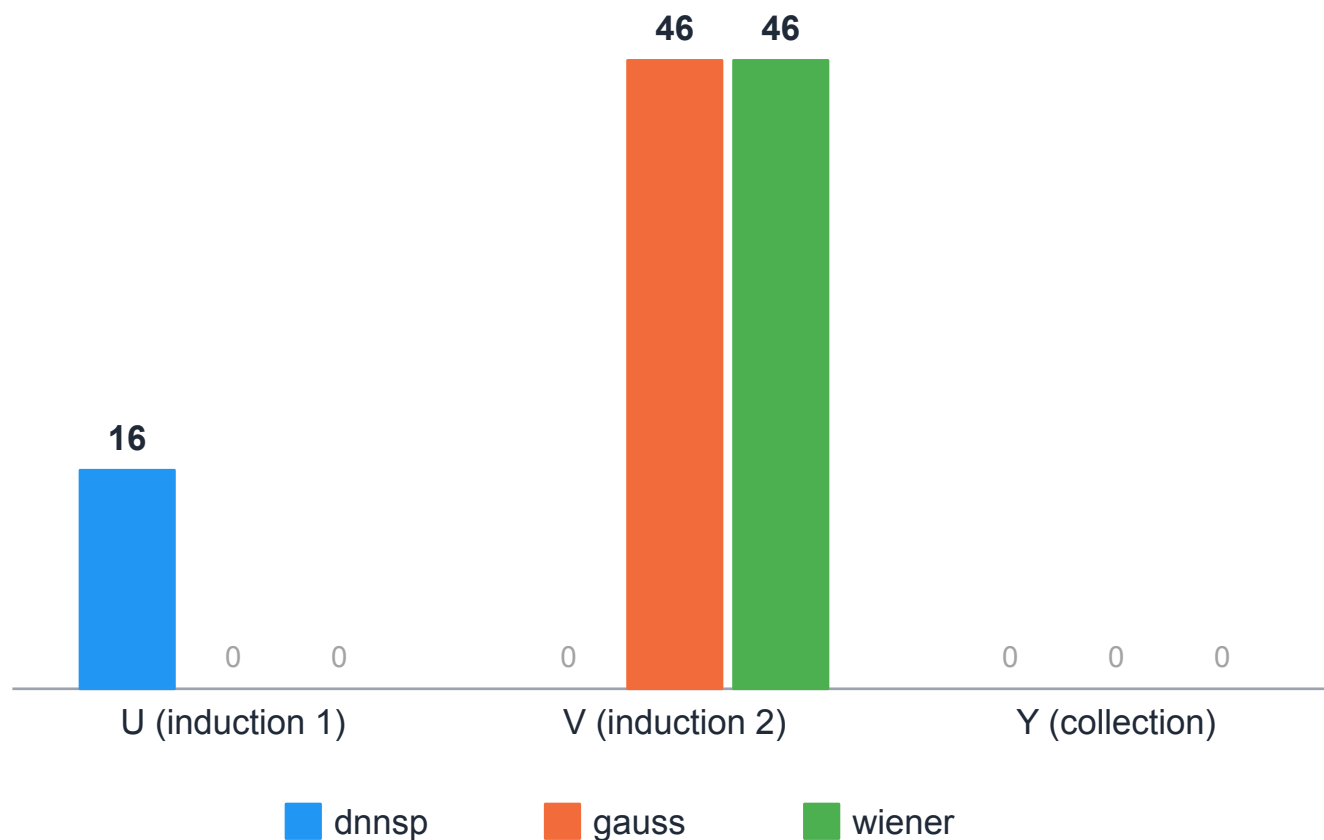
differences on U or Y, any branch

Takeaway

U is also an induction plane, so the effect is not in a path shared by all induction planes. Plane assigned with `recob::Wire::View()`.

Production version v0_32_1: plane breakdown

Events with differences per plane — v0_32_1, run R1 vs run R2



Two plane-disjoint effects

- dnns: U only**
16 events · 55 channel-events
- gauss / wiener: V only**
46 events · 256 channel-events · 243 unique channels
- Y (collection): nothing**
zero differences in any branch

Consistent with two independent sources: the dnns source (U, fixed by earlier commits) and the gauss/wiener source (V, persisted in old master).

ADC scale: v0_32_1 vs old master

Branch	v0_32_1 events	v0_32_1 max $ \Delta $ (ADC)	old master events	old master max $ \Delta $ (ADC)
dnnsp	16 / 88 (U)	11.59	0 / 88	0
gauss	46 / 88 (V)	19.86	46 / 88 (V)	18.32
wiener	46 / 88 (V)	22.12	46 / 88 (V)	22.28
V ch-events (gauss / wiener)	256 / 256		195 / 195	

Largest dnnsp difference (v0_32_1)

11.59 ADC

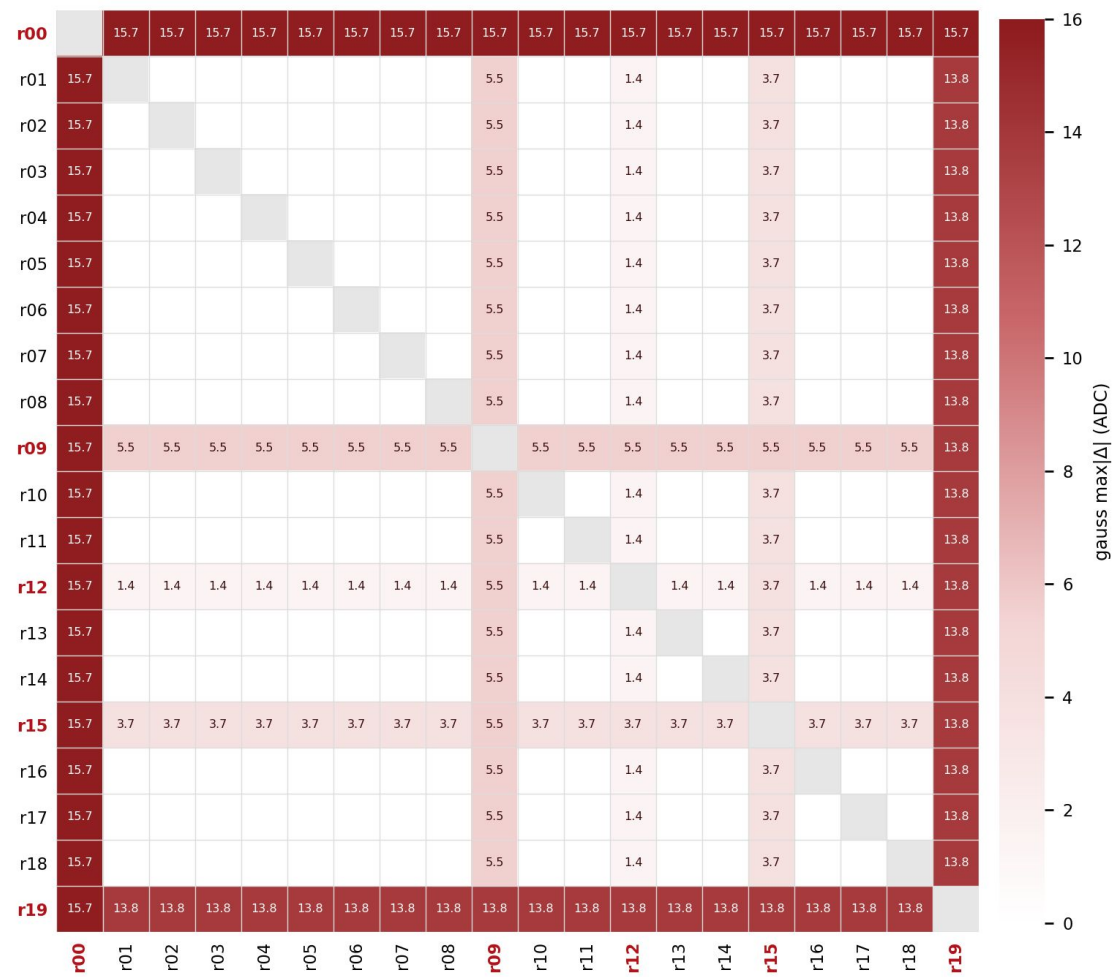
f356-90c1-0215-e140, event 2, channel 6530 (U)

Largest gauss / wiener difference (v0_32_1)

19.86 / 22.12 ADC

199e-ec8c-cfaf-a30e, event 3, channel 9304 (V)

Repeated exemplar (old master): discrete run-dependent states



g4-gen-199e-ec8c-cfaf-a30e.root, event 1 · 20 old-master DetSim runs · 190 pairs · gauss max|Δ| per pair (red = singleton states)

Pairs that differ (of 190)

0

dnns

85

gauss

85

wiener

State	Run	gauss max Δ	wiener max Δ
S0	run_00	15.70	21.00
S19	run_19	13.76	17.88
S9	run_09	5.50	8.80
S15	run_15	3.66	7.78
S12	run_12	1.36	3.70
C (common)	15 runs	0	0

1 common + 5 singleton states = 6 distinct decoded outputs. 85 = $5 \times 15 + C(5,2)$. All affected channels on V. max|Δ| vs common state (ADC).

Root cause (Haiwang): stale ROI entry, reused heap address

sigproc/src/OmnibusSigProc.cxx · CleanUpInductionROIs()

1

U-plane tight ROI is stored

in contained_rois[u_roi] during U loose-ROI processing

2

U loose ROI is deleted

but its key u_roi stays in contained_rois (stale)

3

V ROI reuses the heap address

new V-plane ROI pointer == old u_roi

4

Lookup hits the stale entry

contained_rois[v_roi] returns U-plane tight-ROI data

5

V-plane processing corrupted

affects both gauss and wiener outputs

6

Reuse depends on run

heap layout varies with TBB task ordering → discrete states

Fix: 68d3a99d erases the contained_rois entry when a loose ROI is deleted; 98846706 makes this the default (r_erase_stale_contained = true). Merged in master 51b5a1fc.

Fix validation: 88-event A/B comparison

Same 20 G4 files, same FCL, same container, same comparison — only the WCT commit changes

gauss

46 → 0

events affected / 88 (old → new master)

wiener

46 → 0

events affected / 88 (old → new master)

dnnsnp

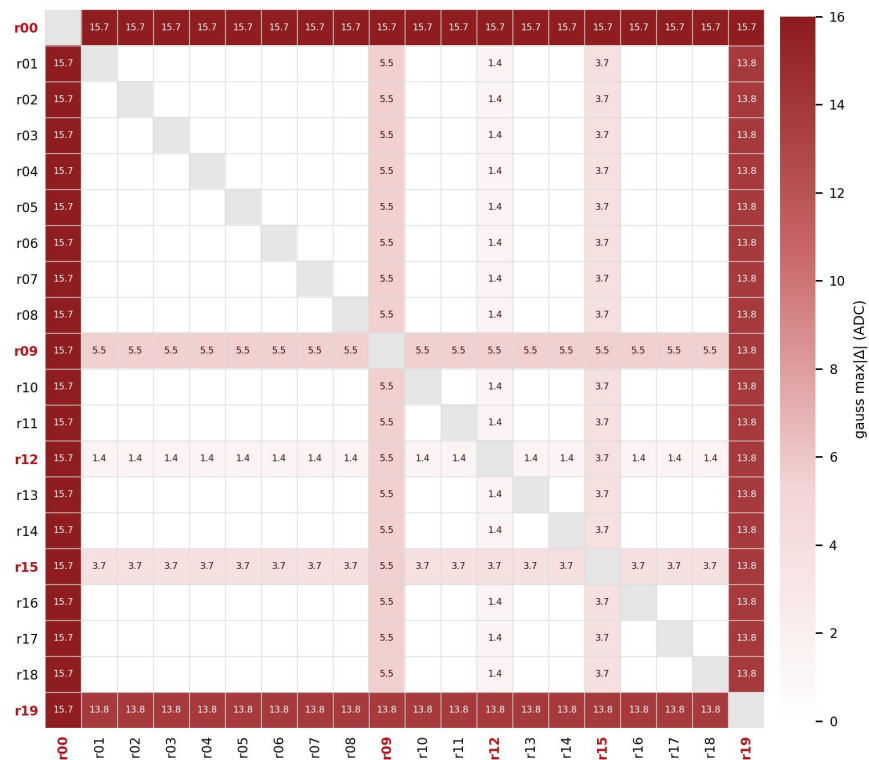
0 → 0

events affected / 88 (old → new master)

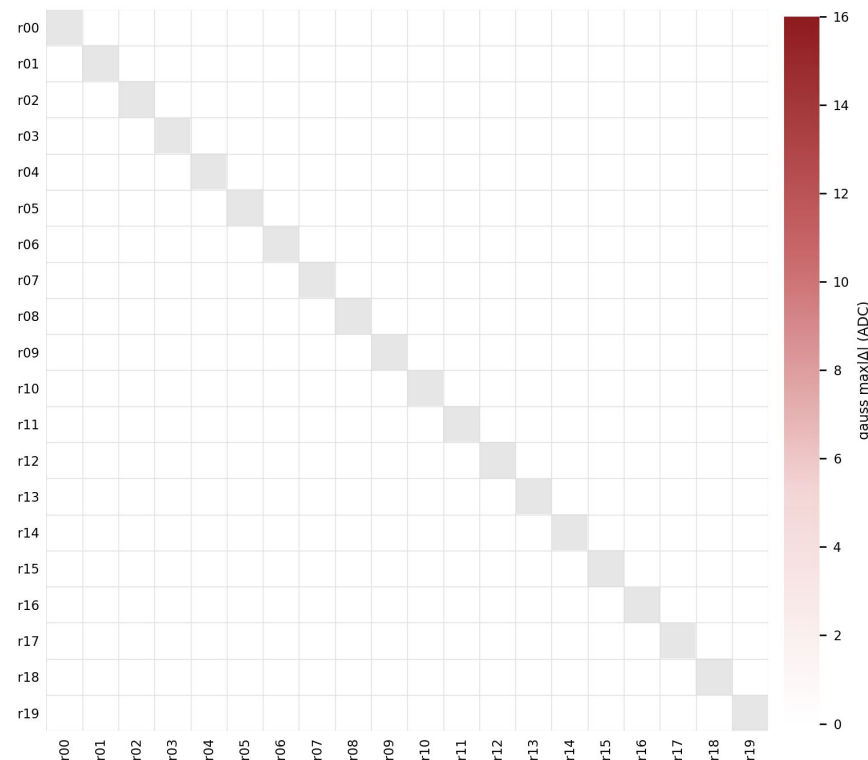
Metric (88 events)	old master 7a85e5c5	new master 51b5a1fc
gauss channel-events differ	195	0
gauss global max $ \Delta $ (ADC)	18.32	0.00
wiener channel-events differ	195	0
wiener global max $ \Delta $ (ADC)	22.28	0.00
Differences on U / V / Y (any branch)	V only	none

Fix validation: 20-run repeated event

Old master 7a85e5c5



New master 51b5a1fc



Pairs that differ / 190

gauss

85 → 0

wiener

85 → 0

dnns

0 → 0

gauss max|Δ| per run pair · g4-gen-199e-ec8c-cfaf-a30e.root, event 1 · white = identical

6 states → 1 state

How this squares with Joseph / Lynn

Joseph / Lynn

- older, Gen-1-like chain, WCT v0_30_3
- fixed SimEnergyDeposits
- WCT sim + sigproc rerun many times
- dramatic downstream variation in special events
- predates Haiwang's simulation-ordering fix

This study (Gen-2)

- fixed Gen-2 G4 input; v0_32_1, old and new master
- dnns (U): fixed by earlier commits
- gauss/wiener (V): stale contained_rois entry
- new master 51b5a1fc: no observed differences
- mechanism identified (Haiwang); fix validated here

Open: does Joseph's v0_30_3 event involve the same stale-ROI path, the older simulation-ordering issue, or both?

Simplest bridge test now: rerun Joseph's fixed-SED event repeatedly with a build that has both the simulation-ordering fix and 51b5a1fc.

What this means for official production

The 1000-event benchmark used stock WCT v0_32_1 — the version characterised on slides 10–11

Benchmark still demonstrates

- Workflow execution and pipeline correctness
- Output completeness and acceptance
- Memory, throughput and scaling estimates
- Retained-storage estimates

Benchmark does not demonstrate

- Bitwise-reproducible recob::Wire output
- Under v0_32_1: dnnsp 16/88 (U), gauss/wiener 46/88 (V)
- Differences up to ~22 ADC, with ROI boundary shifts
- A property of WCT — not of the Gen-2 config or of Aurora

Decision for the group: produce the 1M sample with stock v0_32_1, or with a WCT that includes 51b5a1fc? The second option needs a WCT build usable with sbndcode v10_14_02_03 and a short re-run of the benchmark to confirm nothing else changes.

Summary and next steps

What we know

- Production workflow ready for review
- v0_32_1: dnnsp U-only, gauss/wiener V-only, Y clean
- Old master fixed dnnsp; V effect persisted
- Root cause: stale contained_rois entry + heap reuse
- New master 51b5a1fc: 0/88 events, 0/190 pairs differ

Still open

- Is new-master output equal to the old common state?
- Does 51b5a1fc change anything else vs v0_32_1?
- Does the fix resolve Joseph/Lynn's event?
- Which WCT version for the 1M campaign?

Next steps

- Compare new-master output to old-master common state and to v0_32_1, event by event
- Run Joseph's event with the fixed build
- If adopted: build for v10_14_02_03, re-run short benchmark
- Then staged scale-up toward 1M

Requests for the group

1 Production

- Confirm retained outputs are sufficient: G4, Reco1, flat CAF
- Review 1M projections (≈ 88 TiB, ≈ 2911 node-hours)
- Decide the WCT version for the 1M campaign: stock v0_32_1 or with 51b5a1fc
- Should a packed-node test come before sizing?

2 WCT (Haiwang, Lynn, Joseph)

- Confirm our reading of the fix mechanism
- Is a v0_32-series release with the fix planned for sbndcode?
- Share Joseph's fixed-SED event for a fixed-build test
- Should bitwise reproducibility be a production acceptance check?

Thanks to Haiwang for tracking down and fixing the V-plane source.

Back Up Slides

Backup: per-stage resources and combined-unit overhead

Per-stage averages (50 subruns)

Stage	VmHWM (MB)	Real (s)	s / final evt
GEN	2353	1903	71.6
G4	2739	493	18.6
DetSim	6822	5140	193.5
Reco1	4834	624	23.5
Reco2	2176	1174	44.2
CAF	3180	121	4.6

VmHWM in base-10 MB from art MemReport. GEN processes 100 attempts/subrun; downstream stages process accepted events only (26.56/subrun avg). 50 workers × 6.35 GiB ≈ 318 GiB < 410 GiB (80% of 512 GiB).

Combined-unit wall time per subrun

Unit	Wall (s)	art Real (s)	Overhead
GEN + G4	3061	2396	665 s
DetSim + Reco1	5789	5764	25 s
Reco2 + CAF	1682	1295	387 s

Overhead = tarball extraction, container start, FCL preparation, staging, cleanup (per-step breakdown not isolated). At 1M events: ~37,650 subruns × 3 tasks ≈ 112,950 Parsl task executions, so this fixed cost matters.

Backup: idealised scale-out at measured throughput

Nodes	Wall time (h)	Wall time (days)
1	2911	121
10	291	12.1
32	91	3.8
64	45.5	1.9
128	22.7	0.95
256	11.4	0.47

- Linear scaling, same occupancy as the benchmark
- Assumes no filesystem or scheduler degradation
- Per-node throughput is conservative: ≤ 50 active subrun chains on 102 slots
- Retained-storage saving per subrun: 48.4% (2.57 GB vs 4.99 GB)

Backup: event severity ranking (old master A/B)

Rank	G4 file UUID	evt	Score	gauss max Δ	wiener max Δ	n ch
1	199e-ec8c-cfaf-a30e	1	135.75	18.32	22.28	1
2	8b63-0aca-23d7-0ab0	1	104.08	14.52	17.64	13
3	8b63-0aca-23d7-0ab0	0	99.44	17.42	20.64	5
4	199e-ec8c-cfaf-a30e	3	92.61	14.78	20.70	6
5	4f08-780d-177d-1e84	0	83.23	15.54	19.34	1
6	1846-ed4b-b660-a83f	1	83.08	13.52	16.92	7
7	17b2-7a19-0173-eaa0	2	76.21	11.84	14.36	7
8	4f08-780d-177d-1e84	5	71.32	8.50	20.68	3

Composite score

$$2 \times (g_max + w_max) + 0.5 \times (g_nch + w_nch) + g_roi + w_roi + 50 \times (g_mean + w_mean)$$

Rank 1 holds the global max|Δ| for both gauss and wiener in old master, with an ROI structural change. Selected for the 20-run study.

Backup: provenance of the WCT studies

v0_32_1 characterisation (R1 vs R2)

Parameter	Value
WCT	v0_32_1 (scisoft install)
sbndcode	v10_14_02_05
FCL	standard_detsim_sbnd.fcl, no WCT override
Inputs	20 Thomas G4 files, 88 events
Container	slf7.sif + larsoft-v10_14_02 + sbnd-v10_14_02_05_delta

Production benchmark uses sbndcode v10_14_02_03; both releases link the same WCT v0_32_1 binary install.

Fix validation (old vs new master)

Parameter	Value
Old WCT	7a85e5c5b2a4...
New WCT	51b5a1fc0acb...
Fix commits	68d3a99d, 98846706 (ancestry verified)
Injection	LD_LIBRARY_PATH prepend + LD_PRELOAD taginfo shim
Build	cmake 3.21.4, GCC 12.1.0, Release, C++17, CentOS 7

All other inputs identical to the old-master study: sbndcode v10_14_02_05, same FCL, G4 inputs and container.

09/22: SBND Gen-2 production on Aurora

- Production framework: SBN-ALCF sbn_parsl
- Target production release: sbndcode v10_14_02_03
- Full chain demonstrated successfully:
 - GEN → G4 → DetSim → Reco1 → Reco2 → flat CAF
- Initial Parsl test:
 - 100 GEN attempts
 - 30 accepted physics events
 - all 30 propagated successfully through CAF
- Successful production exercised:
 - standard SBND Gen-2 FCLs
 - Aurora container environment
 - Parsl task dependencies
 - intermediate ROOT outputs
 - LArCV / calibration outputs
 - flat CAF generation
- Critical Aurora container issues were identified and documented
 - bind-mount requirements are release-dependent
 - v10_14_02_03 and v10_14_02_05 require different handling

09/22: Comparison against Thomas Wester's production

- Used the same 20 Thomas G4 files
- 88 total physics events
- Same downstream release:
 - sbndcode v10_14_02_05
- Same standard:
 - DetSim
 - Reco1
 - Reco2
- Our Parsl chain additionally produced flat CAFs
- Agreement: 88 / 88 events reproduced
- Identical persisted schemas:
 - DetSim: 59 / 59 branches
 - Reco1: 73 / 73 branches
 - Reco2: 231 / 231 branches
- File sizes agree within approximately 0.2%
- All deterministic G4 / truth pass-through products are bit-identical
- No missing products or unexpected structural differences

09/22: Readiness for a 1M-Event Production Campaign

- DetSim remains the dominant CPU stage
- Typical process memory remains modest:
 - peak approximately 6–7 GiB
- Reco1 output: approximately 28 MiB/event
 - about 27 TiB for 1M events
- Reco2 output: approximately 31 MiB/event
 - about 30 TiB for 1M events
- Flat CAF: approximately 5 MiB/event
 - about 5 TiB for 1M events
- Keeping all major intermediate stages would require O(100 TB) of storage
- Production strategy
 - Restore the official v10_14_02_03 Gen-2 configuration
 - Freeze the Parsl code, TOML, FCLs and container setup
 - Run a staged scale-up: small validation → O(10k) → O(100k) → 1M
- At each step verify:
 - success / retry rates
 - events/hour/node
 - memory
 - storage growth
 - metadata integrity
 - CAF completeness