ACAT 2024

Line Segment Tracking

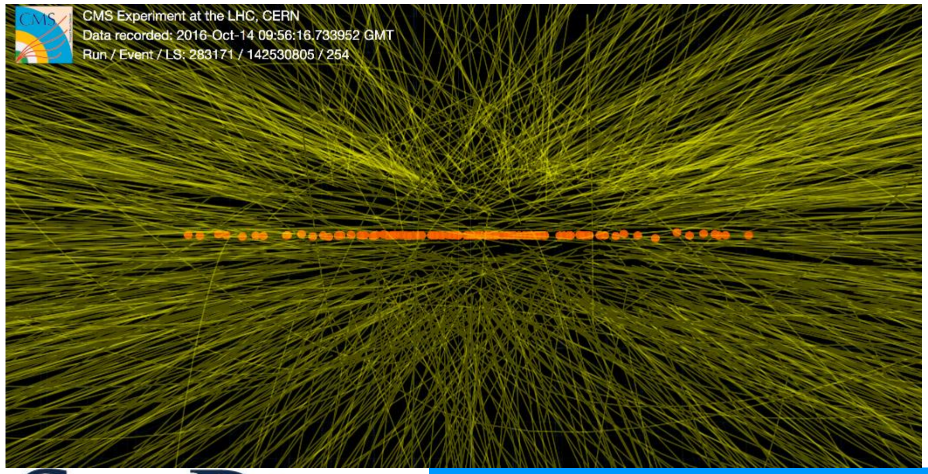
Improving the Phase 2 CMS High Level Trigger Tracking with a Novel, Hardware-Agnostic Pattern Recognition Algorithm

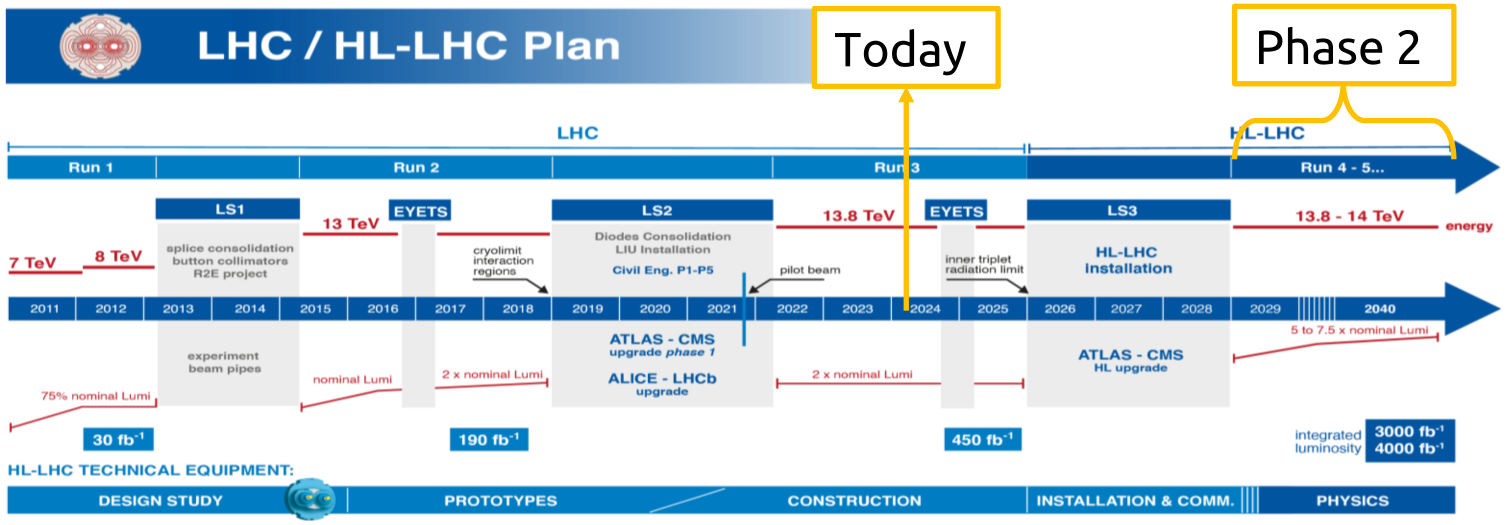
Emmanouil (Manos) Vourliotis
on behalf of the CMS Collaboration

The Challenges of Phase 2 Tracking

- The High Luminosity LHC (HL-LHC) brings new opportunities:
  A new phase for the LHC experiments…
- To realize new opportunities, new challenges surface:
  - High luminosity ➔
    Large number of concurrent collisions (pileup or PU): Up to 200 ➔
    Large number of tracks.
  - Combinatorial nature of pattern recognition algorithms (tracking) ➔
    Superlinear increase of computational complexity with increasing input.
  - Combining the above:
    - **Increased timing**: Will the (online) algorithms be able to keep up with the rate at which data arrive?
    - **Increased cost**: Increasing the processing power to keep up drastically increases the budget.

Event at “only” 100 PU



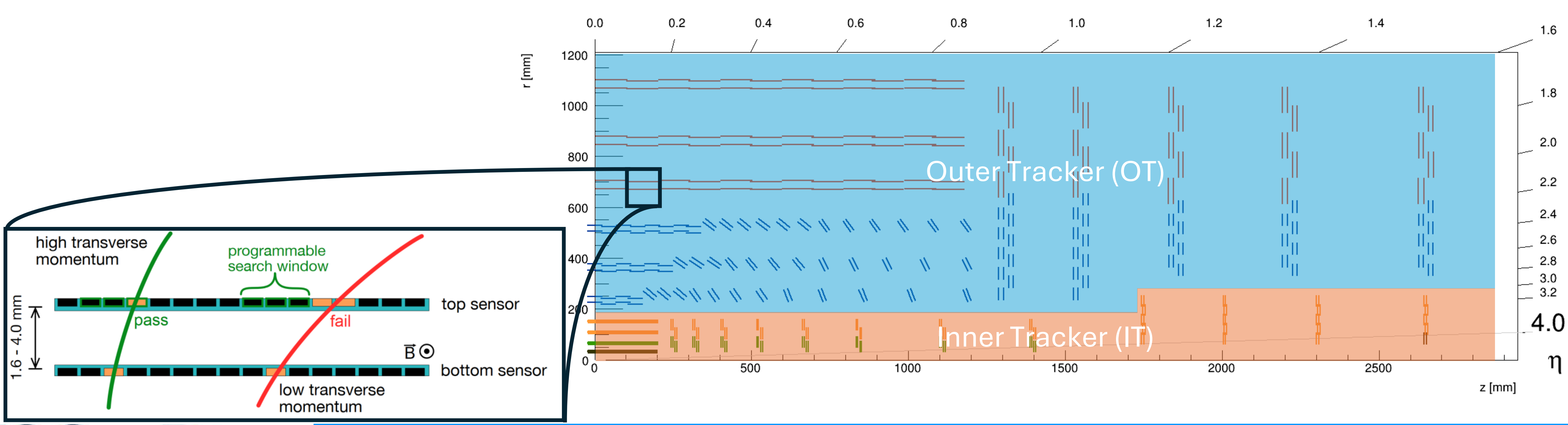


The To-Do List for Future Tracking

- Prerequisites:
  - Timing:
    - Parallelization ➔ Vectorized algorithms (mkFit).
    - Complement CPUs with hardware well suited for parallelization ➔ Heterogeneous computing using GPUs (Patatrack).
  - Physics performance:
    - Retain or even improve the already covered phase space ➔ Machine learning techniques.
    - Extend the probed phase space ➔ Displaced tracks.
- Enter **Line Segment Tracking (LST)**:
  - Moves away from sequential pattern recognition ➔ Designed for parallelization.
  - Leverage GPU performance for parallel tasks ➔ Hardware agnostic implementation (Alpaka framework): CPU and GPU variants with common codebase.
  - Machine learning to improve pattern recognition.
  - Extend acceptance to displaced tracks.

The CMS Phase 2 Outer Tracker

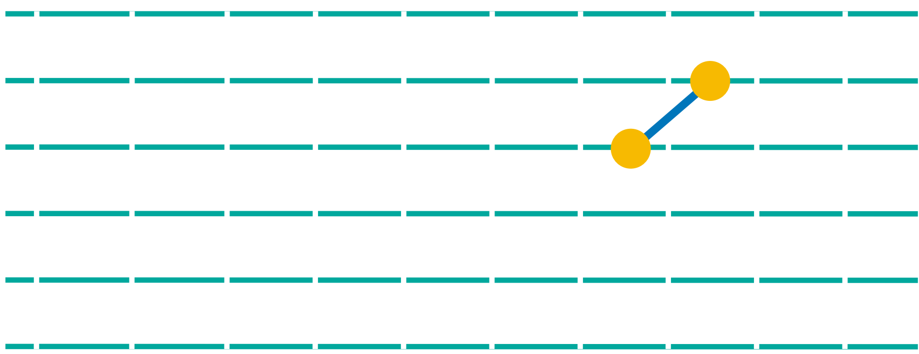
- Key characteristic of the CMS Phase 2 Outer Tracker (OT):
  Each layer comprises 2 closely-spaced silicon sensors.
- **MiniDoublets (MDs)**:
  Linked pair of hits in sensors of the same layer.
  - Reduce combinatorics.
  - Can be locally reconstructed ➔ Allow for parallelization.
  - Elementary building block for tracks.
- Further combinatorics reduction:
  Tune the search window for hit pairs ➔ p_T threshold (0.8 GeV for LST).



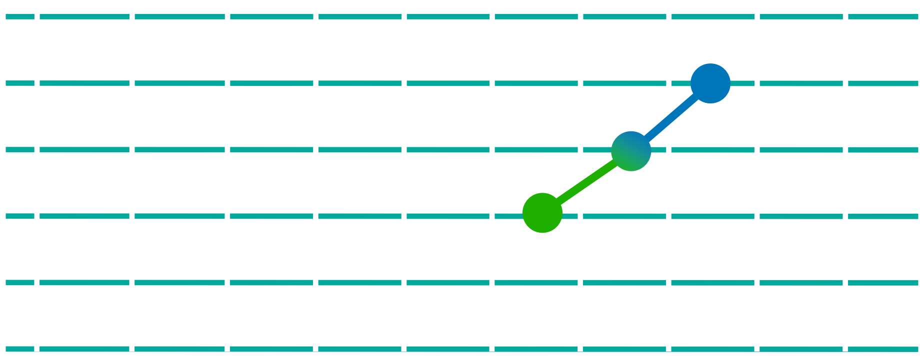
LST Logic and Objects

- LST logic: Link short “tracks” to longer tracks.
  - Line Segment (**LS**)
  - Triplet (**T3**)
  - Quintuplet (**T5**)
  - Inner Tracker (IT) tracks (**pLS**) linked OT objects:
    - pLS + T3 (**pT3**)
    - pLS + T5 (**pT5**)
- Each object independent of others ➔ Massive parallelization in object creation.

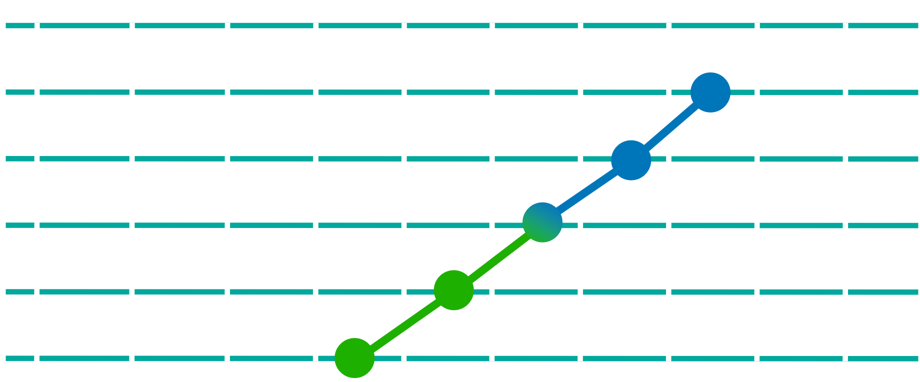
MD + MD = LS



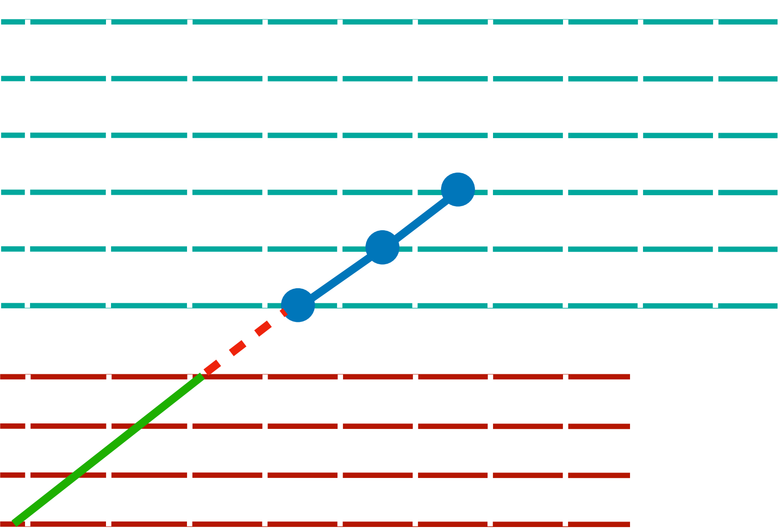
LS + LS = **T3**



T3 + T3 = **T5**

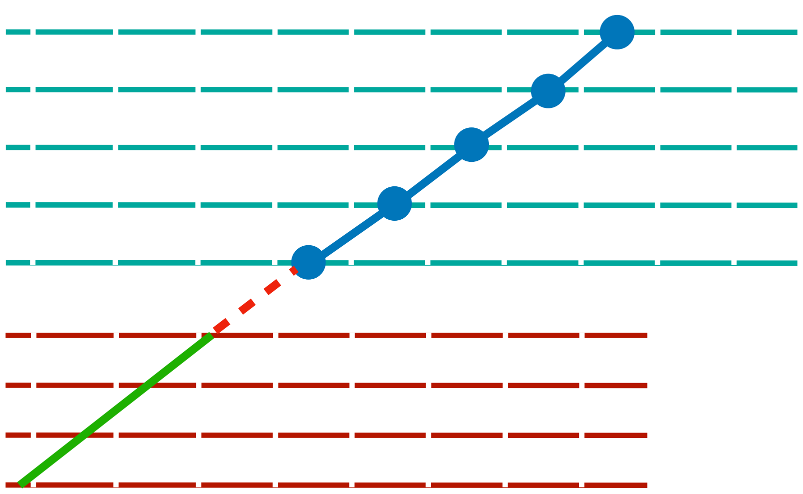


pLS + T3 = **pT3**



pixel LS (pLS)

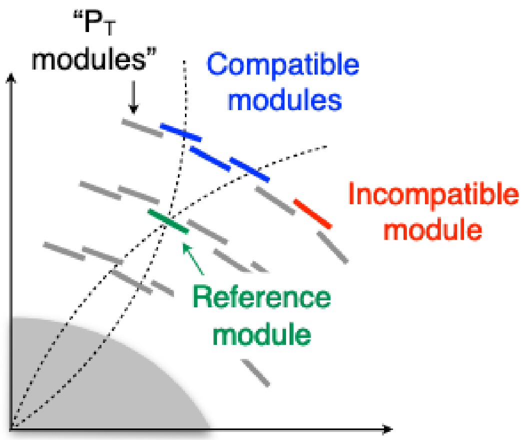
pLS + T5 = **pT5**

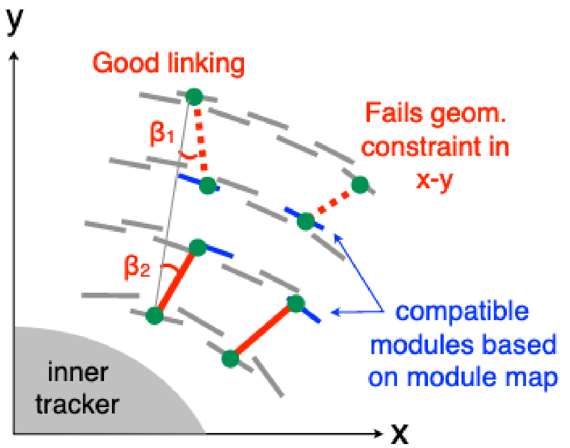


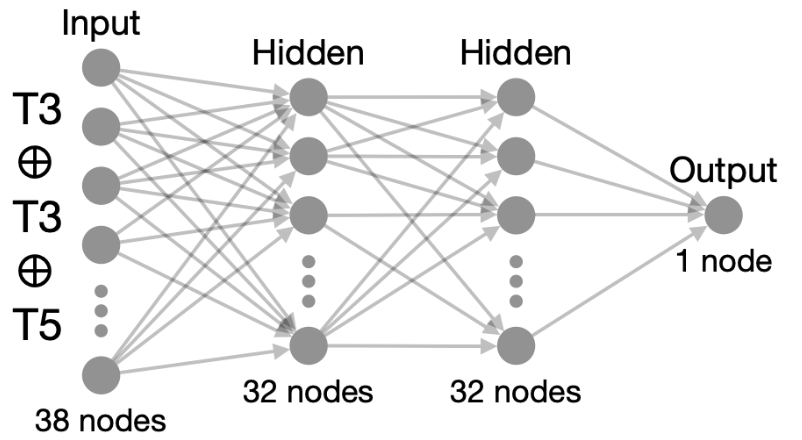
Inner Tracker

LST Selection in a Nutshell

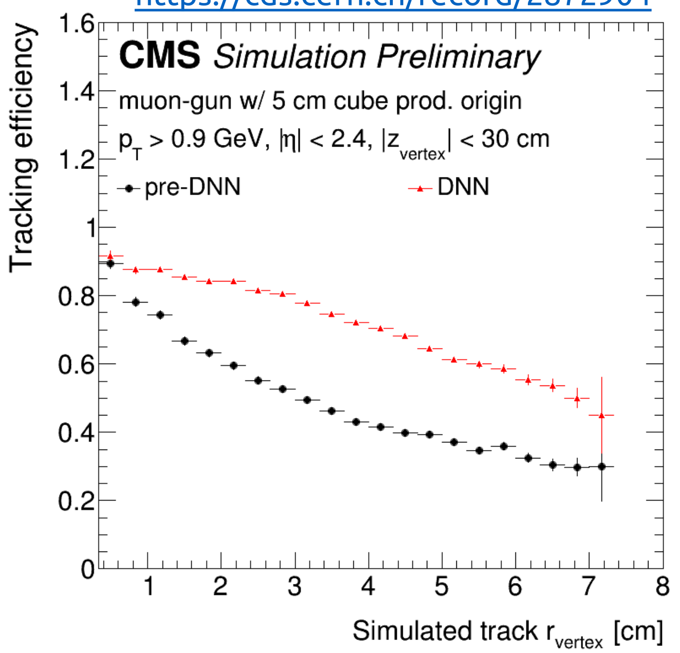
- LST selection for object creation relies on:
  - Precomputed connection maps
    - OT and IT+OT
  - Geometric criteria
- Longer objects get more complicated ➔ Opportunity for machine learning to do better!
  - Simple **DNN implemented to select T5s**.
  - No effect on the timing.
  - Significant reduction in fakes and duplicates.
  - Important gains in efficiency for displaced tracks.
- Final LST output objects:
  - **pT5s**: Longest tracks ➔ Efficiency driver.
  - **pT3s**: Efficiency recovery.
  - **T5s**: OT-only object ➔ Efficiency for displaced tracks.
  - **Unlinked pLSs with ≥4 hits**: Efficiency for high |η| & low p_T.







https://cds.cern.ch/record/2872904



HLT setup

- CMS Phase 2 High Level Trigger (HLT) tracking (Base CKF):
  Reconstruction of tracks with p_T>0.9 GeV in 2 iterations
  with different set of initial track estimations (**track seeding**):
  - initialStep:
    Tracks from pixel seeds with ≥ 4 hits (quads) from the Patatrack algorithm.
  - highPtTripletStep:
    Tracks from pixel seeds with 3 hits (triplets) from the legacy pixel seeding algorithm.
- Pattern recognition (**track building**) with the usage of the Combinatorial Kalman Filter algorithm (CKF):
  - Inherently sequential.
  - Implemented only on CPU.
- Built tracks (collection of hits from the same track) undergo:
  - Track fitting to extract final track parameters.
  - Selection based on track parameter requirements (**tracking ID**):
    - highPurity ID applied ➔ Good efficiency with low fake and duplicate rate for prompt tracks.

LST in HLT setup

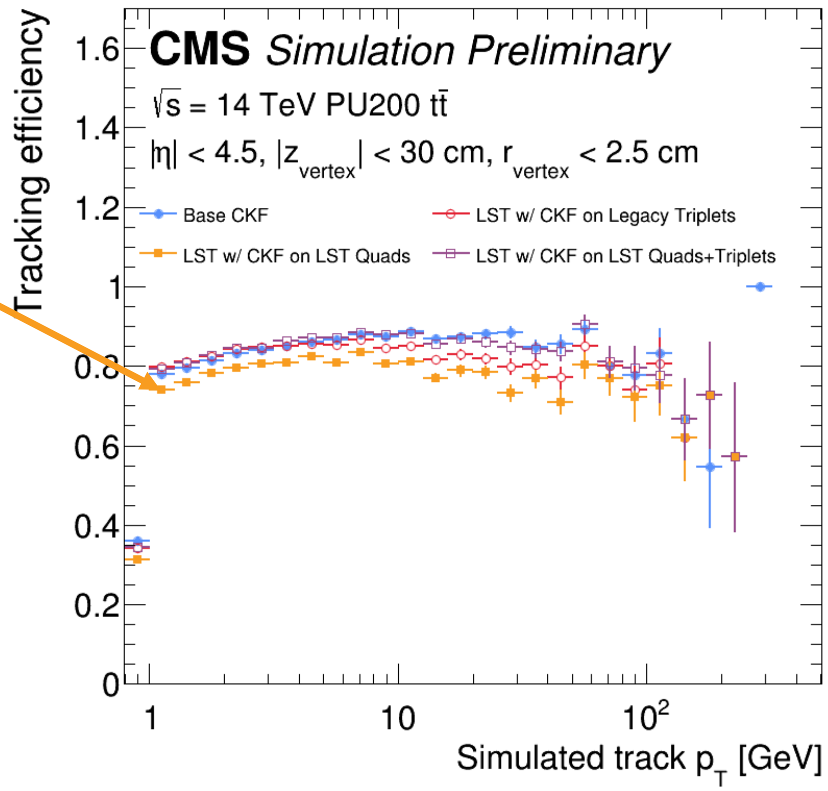
https://cds.cern.ch/record/2890677

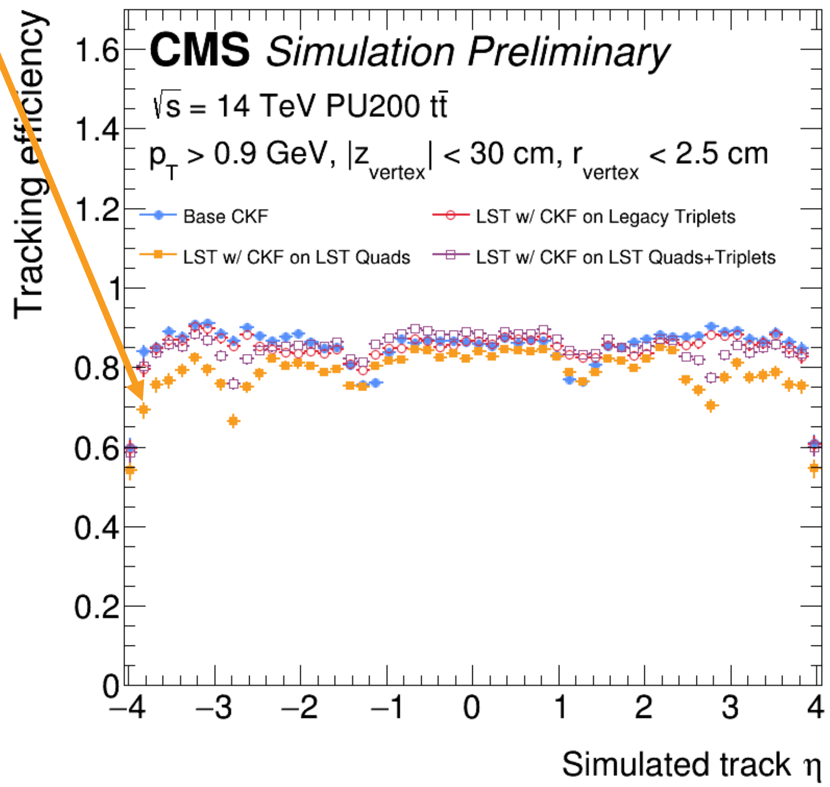
- **LST** to replace track building for initialStep:
  - Using pixel seeds with ≥ 3 hits as pLSs.
- Different tracking ID applied to different LST output objects:
  - No selection (apart from the LST one) applied on T5s ➔ High efficiency for displaced tracks.
- LST does not build tracks for |η|>2.5 (out of OT acceptance) ➔ Run **CKF** on different sets of seeds in highPtTripletStep to recover efficiency:
  - Legacy triplets.
  - LST pLSs quads or quads+triplets ➔ **LST can also be used as a seeding algorithm**!

Iteration	Procedure	Base CKF	LST w/ CKF on Legacy Triplets	LST w/ CKF on LST Quads	LST w/ CKF on LST Quads+Triplets
Initial Step	Seeding	Patatrack quads	Patatrack quads + Legacy triplets	Patatrack quads + Legacy triplets	Patatrack quads + Legacy triplets
	Building	CKF	LST	LST	LST
	Tracking ID	highPurity	highPurity (pT3, pT5, pLS) None (T5)	highPurity (pT3, pT5) None (T5)	highPurity (pT3, pT5) None (T5)
HighPtTriplet Step	Seeding	Legacy triplets	Legacy triplets	LST pLS quads	LST pLS quads+triplets
	Building	CKF	CKF	CKF	CKF
	Tracking ID	highPurity	highPurity	highPurity	highPurity

Physics Performance wrt. Base CKF

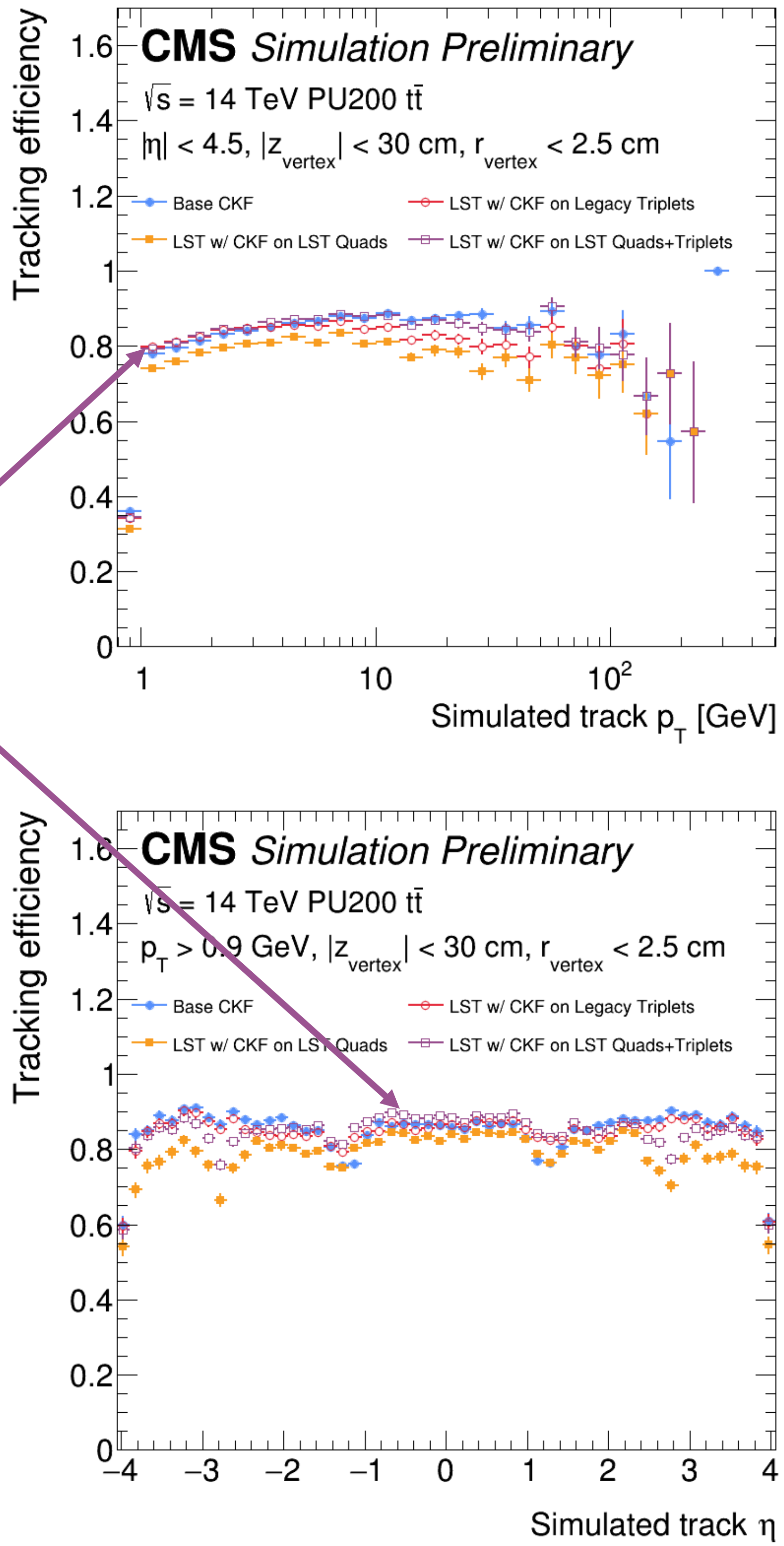
- Lower efficiency when triplets are not built ➔ Mostly from the endcaps ➔ Triplets important in current setup. Alternatives:
  - Use triplets from the Patatrack algorithm.
  - Improve quad reconstruction.





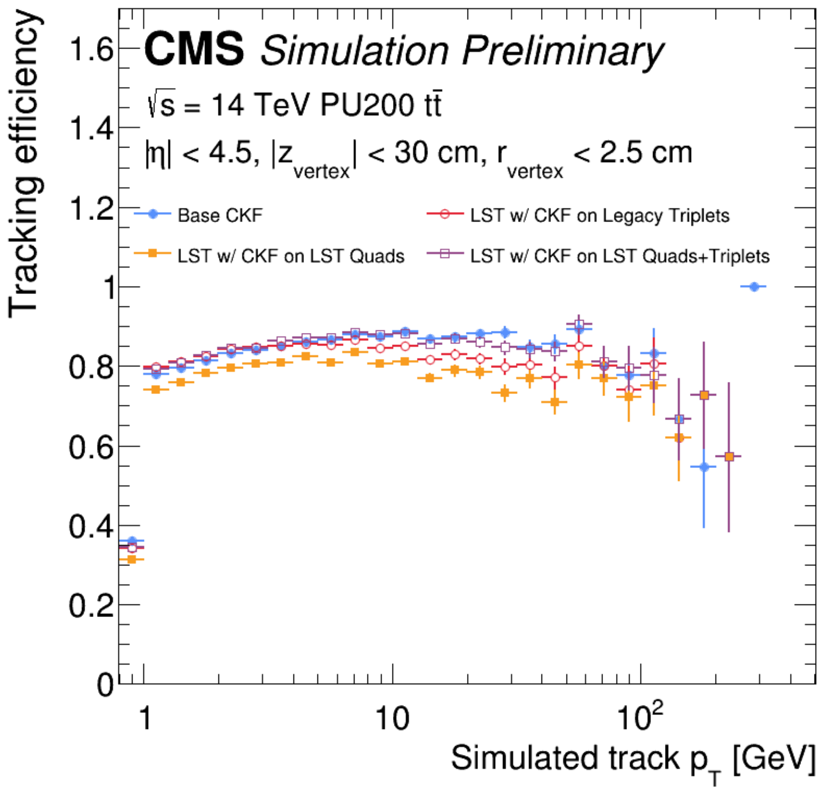
Physics Performance wrt. Base CKF

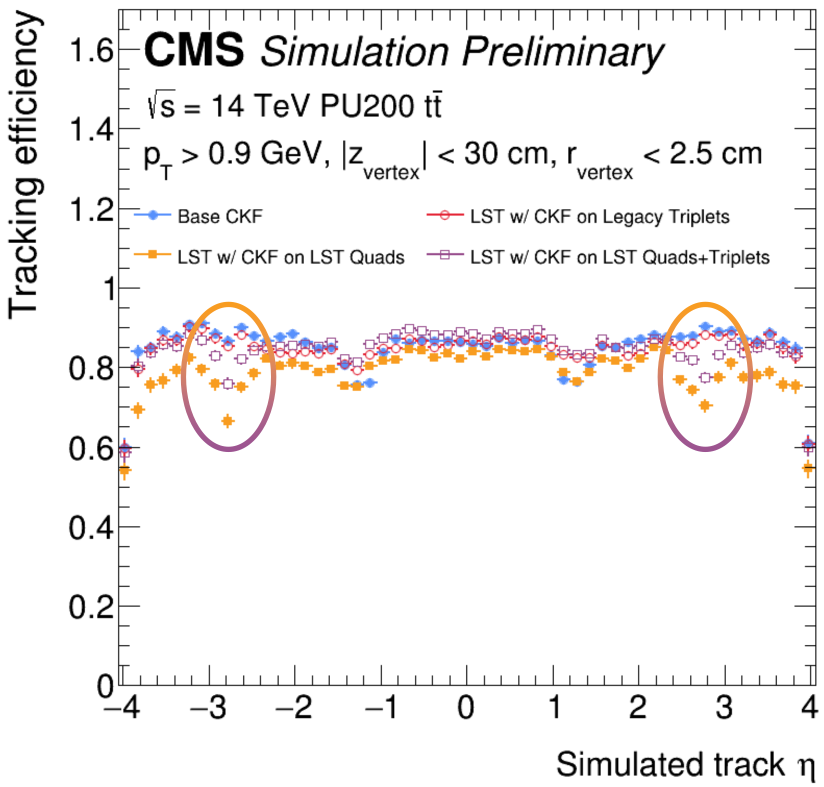
- Lower efficiency when triplets are not built ➔
  Mostly from the endcaps ➔
  Triplets important in current setup.
  Alternatives:
  - Use triplets from the Patatrack algorithm.
  - Improve quad reconstruction.
- Similar/higher efficiency when all LST pLSs built:
  - Efficiency improvement from p_T<5 GeV or $|\eta|$<1.
  - Highlights usefulness of LST as a seeding algorithm.



Physics Performance wrt. Base CKF

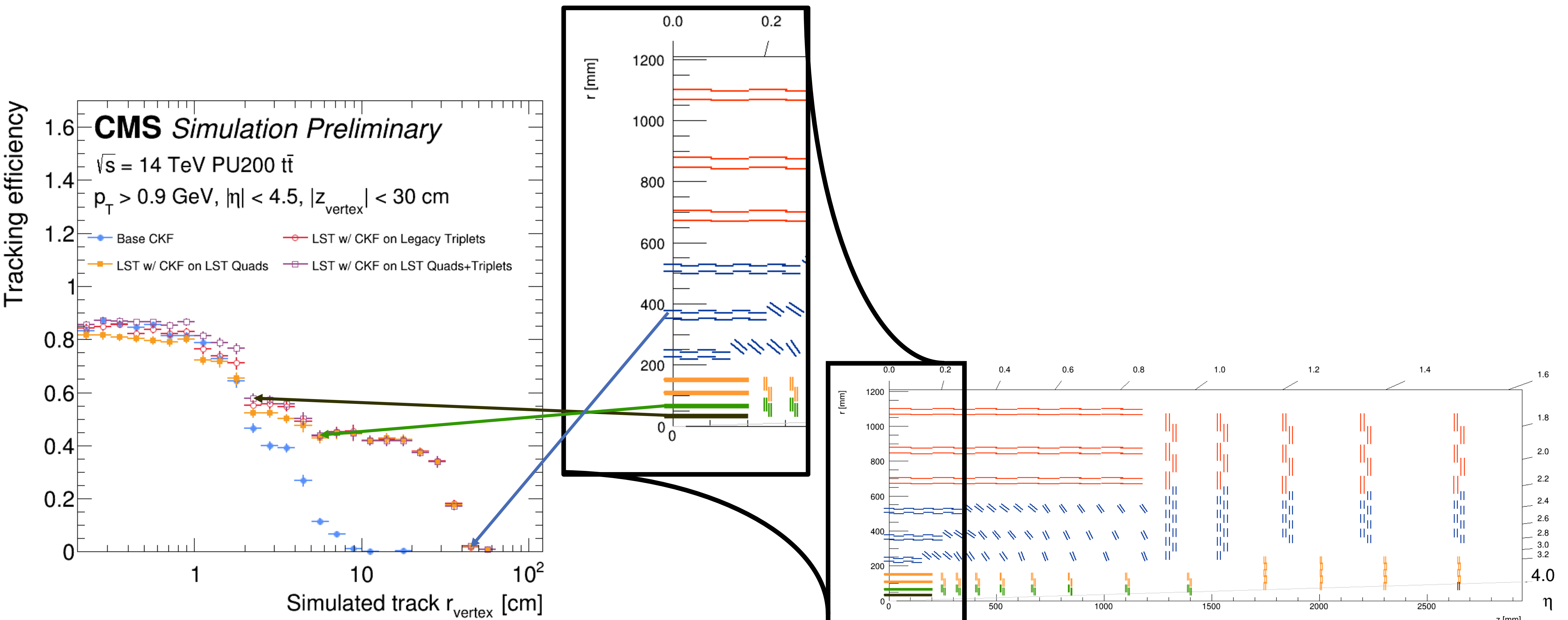
- Lower efficiency when triplets are not built ➔ Mostly from the endcaps ➔ Triplets important in current setup. Alternatives:
  - Use triplets from the Patatrack algorithm.
  - Improve quad reconstruction.
- Similar/higher efficiency when all LST pLSs built:
  - Efficiency improvement from p_T<5 GeV or |η|<1.
  - Highlights usefulness of LST as a seeding algorithm.
- Efficiency dip for LST seeding (orange, purple) for 2.5<|η|<3.0:
  - Room for improvement for LST reconstruction and selection in the OT-IT transition region.





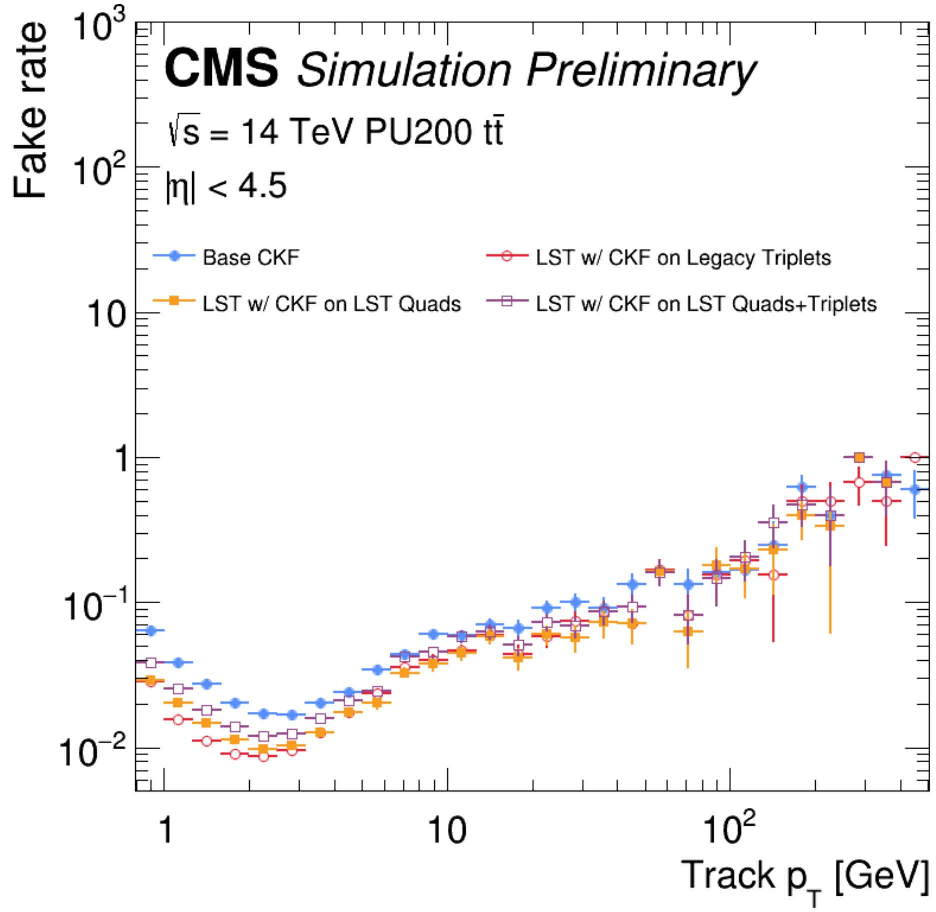
Physics Performance wrt. Base CKF

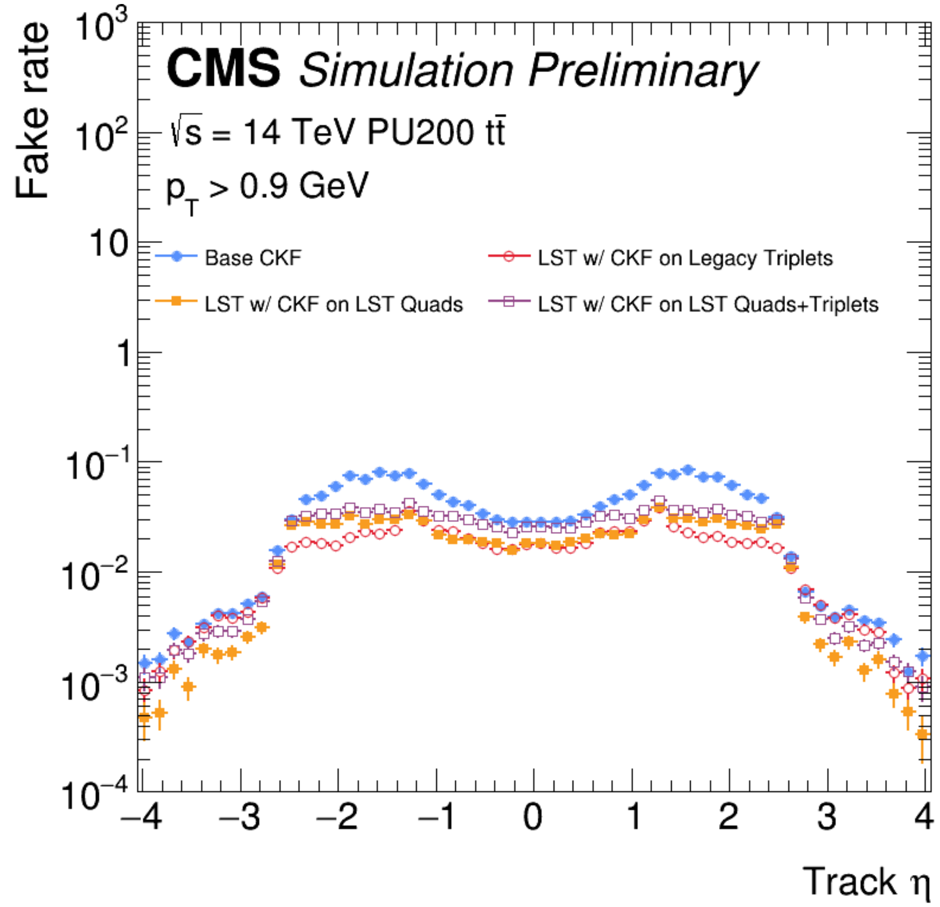
- Any configuration using LST for track building (red, orange, purple) allows for acceptance of displaced tracks (r_{vertex}>5 cm):
  - **Completely new feature for CMS HLT**!
- Efficiency drops roughly corresponding to tracker layers:
  - Endpoint: ~35cm ➔ Less than 4 layers available ➔ No T5 possible.



Physics Performance wrt. Base CKF

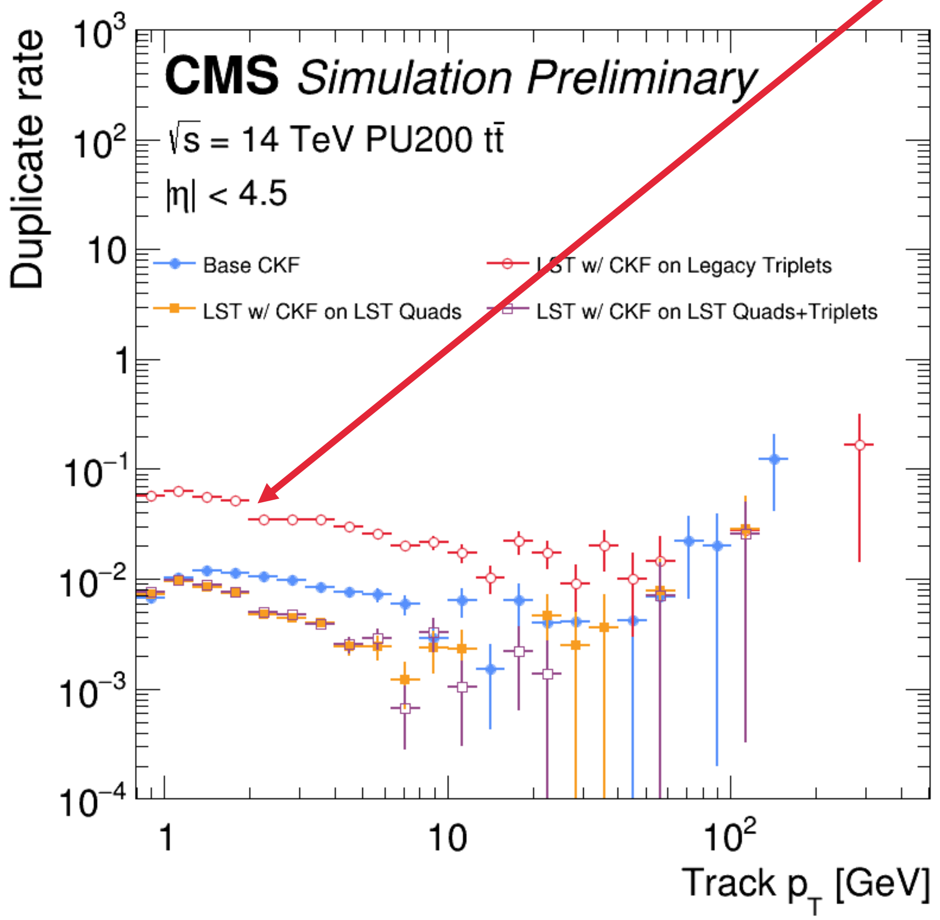
- Lower fake rate for any configuration using LST for track building (red, orange, purple):
  - Mostly for p_T<10 GeV, where the bulk of tracks are ➔ Significant computing reduction downstream.
  - Mostly for |η|<2.5, where LST builds tracks ➔ Implying effective selection for LST objects.

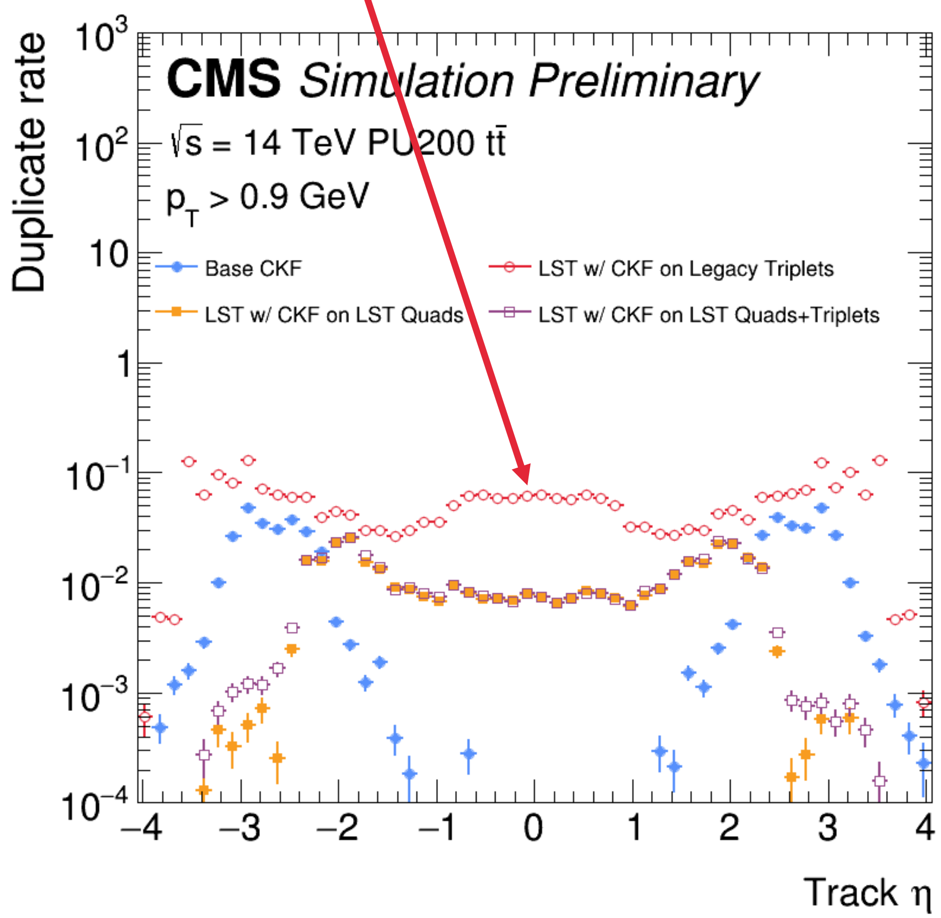




Physics Performance wrt. Base CKF

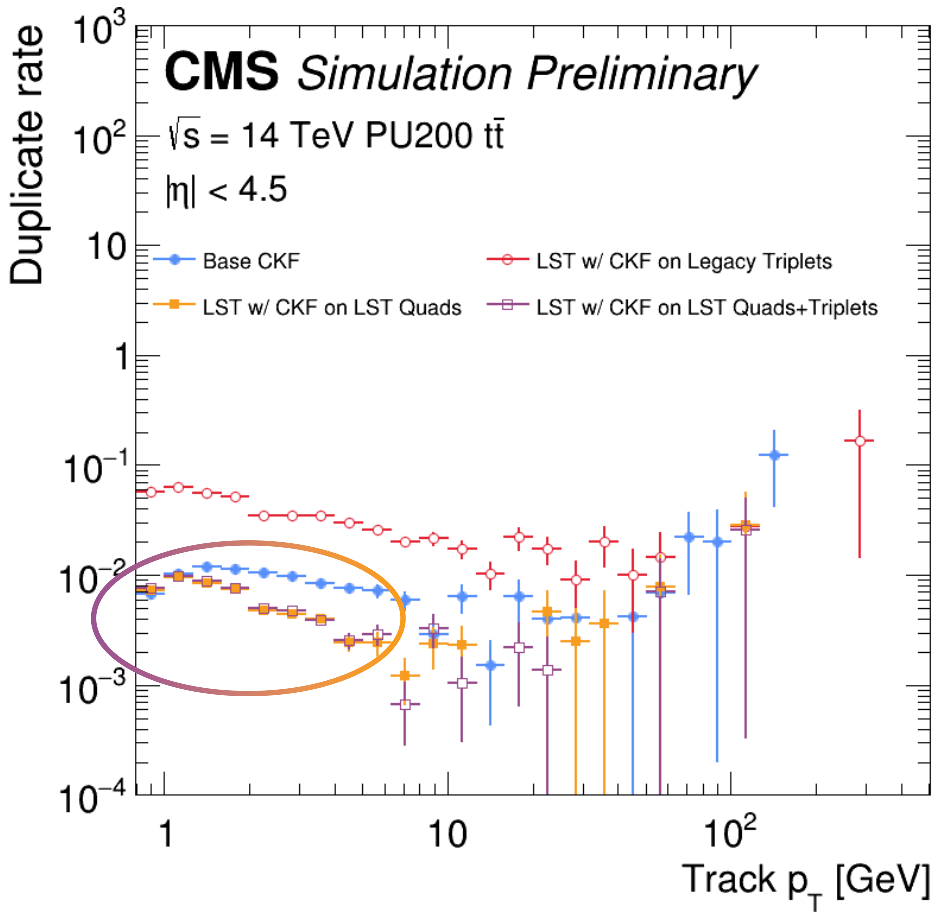
- Higher duplicate rate when CKF run on legacy triplets:
  - Duplicates between LST objects for |η|<2.5.
  - Duplicates between LST and CKF for |η|>2.5.

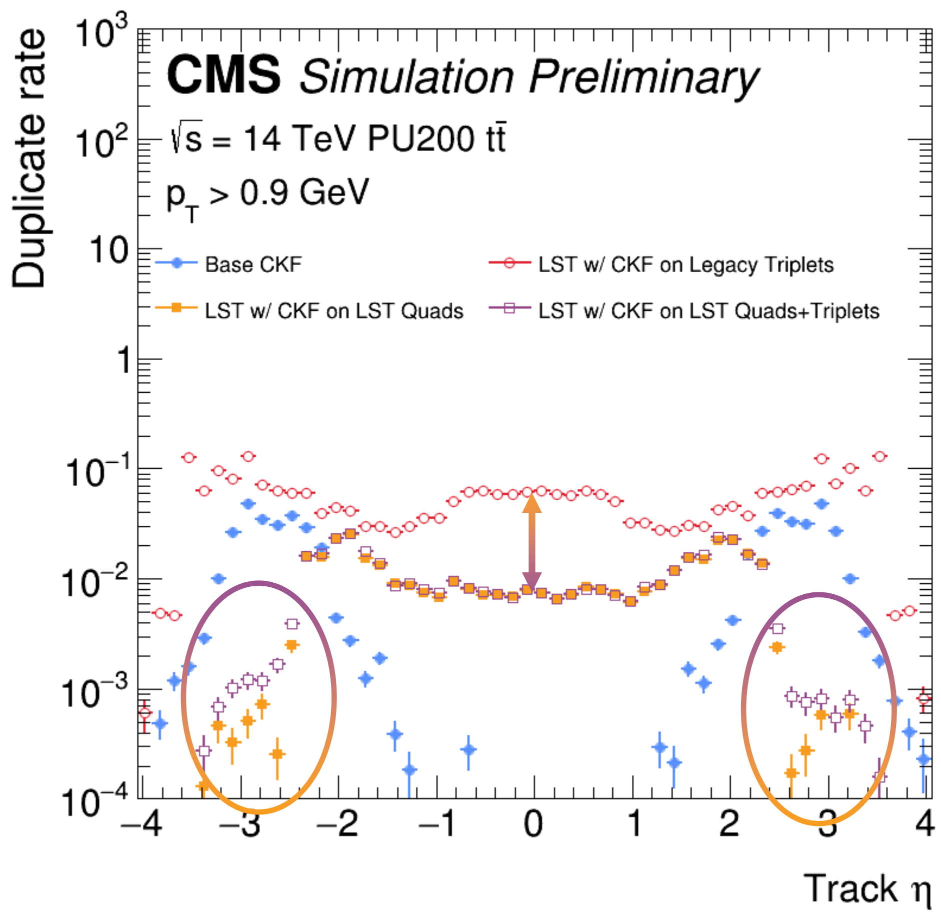




Physics Performance wrt. Base CKF

- Higher duplicate rate when CKF run on legacy triplets:
  - Duplicates between LST objects for |η|<2.5.
  - Duplicates between LST and CKF for |η|>2.5.
- Solution from LST seeding (orange, purple):
  - Better cross-cleaning for |η|<2.5.
  - Effective duplicate merging for |η|>2.5.





Throughput wrt. Base CKF

- A look at **Run 3 computational performance**:
  - Tracking: Complex task performed by serial algorithm ➔ Most time-consuming reconstruction step (offline & online).
  - Displaced tracking: 50% reduction of offline tracking reconstruction throughput ➔ Computationally-heavy task due to large combinatorics.
- **LST** configurations allows for:
  - displaced tracking,
  - with similar (red) or even better (purple) physics performance,
  - with marginal speed up or slowdown of HLT tracking.
- **LST on CPU** shows a slowdown up to 30%:
  - Still better than 50% slowdown expected from Run 3.
  - CPU implementation not optimized and not parallelized currently ➔ Room for improvement!
- **LST on GPU** shows a similar throughput with all the physics gains applied.
- **Majority of the time spent on the CKF iteration** to recover endcap efficiency:
  - Triplets from legacy pixel seeding algorithm ➔ Numerous and impure (compare orange vs. purple) ➔ Slow down for building…

	LST w/ CKF on Legacy Triplets	LST w/ CKF on LST Quads	LST w/ CKF on LST Quads+ Triplets
LST on CPU Throughput / Base CKF	0.72±0.07	0.86±0.07	0.70±0.09
LST on GPU Throughput / Base CKF	1.03±0.09	1.35±0.12	0.92±0.09

Outlook and Plans

- **First exploratory integration of the LST algorithm in the CMS Phase 2 HLT ➔** LST in HLT opens up the possibility for:
  - Displaced tracking at HLT at smaller-than-expected timing cost!
  - Offloading of the track building step on GPUs!
  - More modularity!
- Preliminary results – **Defining the way forward**:
  - On the **LST side**:
    - Refinements in LST object cleaning ➔ Recover efficiency at transition region and reduce duplicate rate.
    - Creation of additional objects targeting displaced tracks and more ML applications.
    - Optimizations on the CPU implementation of LST.
  - On the **additional building iteration**:
    - Refinements of the quad pixel seeding in the endcaps ➔ Rely less on the triplet seeds.
    - Replace triplets from legacy pixel seeding algorithm by triplets from Patatrack ➔ Purer collection ➔ Less computations downstream.
    - Replace CKF building by mkFit building ➔ Speed up of the highPtTripletStep building by up to 70%.

✅ Proof of principle for multiple improvements

SOON More developments to go even further – Faster and more efficient.