Analysis Preservation in STAR

Ankush Kanuganti, Jerome Lauret, Eric Lancon

13 August 2026

@BrookhavenLab

Analysis reproducibility in a STAR analysis

- As a part of the DAP effort, one question has been "are STAR analysis notes sufficient to reproduce an analysis as documented?" and the **aim is to preserve the analysis code and the workflow** before the expert knowledge is lost
- **Identify gaps: missing files, environments, expert knowledge?**

- Observations from my earlier presentation :
  - For very old analyses, authors have referenced some code that was not documented well, but I was able to find it by looking around
  - Documentation between the code repo README and the analysis note was leaving a gap i.e., one would have to refer both to get a complete picture
  - Software, OS, and build environments drift; transitions could create additional complications (CVS, AFS, Git,...)
  - Some jargon used in analysis notes (example picoDst) is not obvious to a beginner, but the Scibot could come to the rescue

Analysis selection

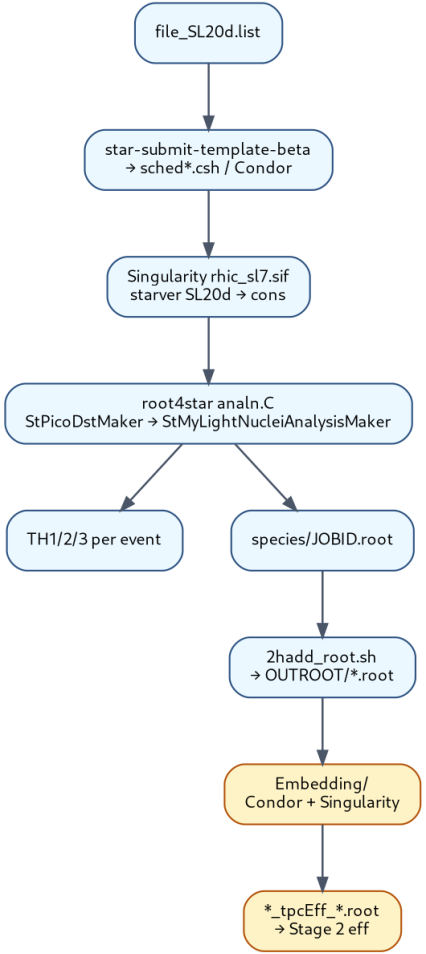
- One STAR analysis was chosen from 2023, which had well documented analysis steps in code repository README and also private STAR analysis note.
- At the time the analysis was performed, the documentation was well documented for 2023, but with the changes in the computing infrastructure, a few tweaks had to be made in order to run the analysis code
- Primary author’s motive at the time of the analysis was always to get the physics results, but not to preserve the analysis structure for long-term use

End-to-end pipeline (3 stages)

- **Stage 1**: raw picoDst dataset processing → TH2/TH3 histograms
  - uses specific star library version SL20d
  - batch processed using STAR Meta-Scheduler
    (datasets are forced to be described in XML format submit scripts)
- **Stage 2**: histograms → corrected yields, efficiencies, spectra
  - Lightweight and independent of the star library, one can even run this locally on a laptop with basic ROOT software.
  - In parallel, embedding NTuples → TPC efficiency histos fed into this stage
- **Stage 3**: physics results (not dependent on the star library) → publication figures
- Same picoDst filelist, separate jobs per species / cut variation

Stage 1: From raw data (picoDst) to histograms

- Read collision events from STAR picoDst files
- Apply event and track selection, particle identification (TPC + TOF)
- Fill histograms for each particle species (p, d, t, ^{3}He, ^{4}He)
- Run many parallel batch jobs, then merge into one ROOT file per species
- Repeat with varied cuts for systematic uncertainty estimates
- Separate embedding sample used to measure detector tracking efficiency
- Output: intermediate histogram files “not final physics results yet” → Stage 2



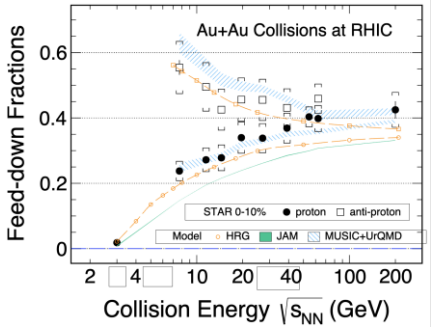
Stage 2: signal, efficiency, spectra

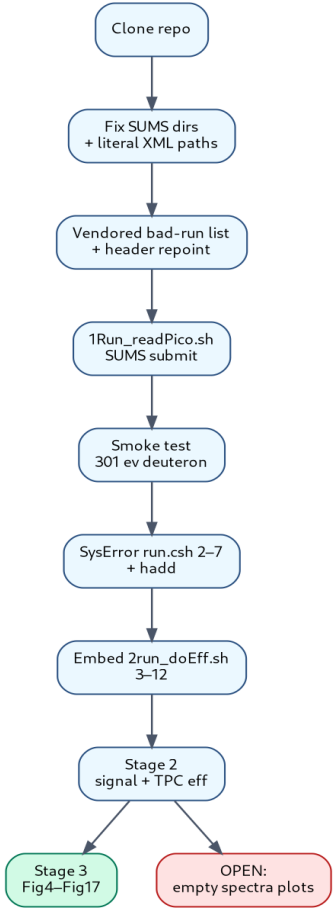
- Signal: mass² vs pT fits (deuteron_SL20d/Signal/)
- TPC efficiency from embedding (deuteron/Efficiency/TPCEff/): done
- TOF efficiency needs full Stage 1 OUTROOT → done
- Spectra/dN/dy macros (pt_spectra/) → done
- Proton comparison & feed-down: done after include-path fixes
- Two deuteron folder trees (deuteron_SL20d vs deuteron/) have a confusing legacy layout, mismatch with README instructions

Subfolder	Input	Output	Status
deuteron_SL20d/Signal/	Stage 1 histos	mass² vs pT fits, signal PDFs	✓ done
deuteron/Efficiency/TPCEff/	embed histos	tracking efficiency ROOT	✓ done
deuteron/Efficiency/TOFEff/	embed + data histos	TOF matching eff	done
deuteron_SL20d/pt_spectra/	signal + eff	pT, dN/dy, ⟨pT⟩ PDFs	done
proton/Comparison/	proton histos	cross-check plots	✓ done
proton/Feeddown/	proton + deuteron	feed-down correction	✓ (include fixed)

Reproduction path to Stage 3

- Input: rootfile.zip (author provided) from Stage 2
- Macros Fig4.C … Fig17.C → pdffile/*.pdf
- Plotting only → no new physics extraction
- Straightforward once Stage 2 outputs exist; completed in bring-up
- Example: Fig 4 reproduced from paper





Environment: Running the analysis today SL7 → containers

- Original analysis was built for Scientific Linux 7 (SL7) + specific star library (SL)
- Current BNL nodes (Alma 9) no longer natively support SL7 code
- Good news: STAR Singularity containers (and also docker images) let us run the SL7-era code on Alma 9 today
- Pin the same software version the paper used (SL20d)
- Started small: test run on a few events to confirm everything works, then scale up: full dataset, all particle species, batch submission

Takeaway: old analyses are revivable, but environment must be documented and pinned

What broke in the old setup and why it matters?

- The code assumed the old SL7 world, relative path instead of absolute path, sl7 batch system, and author-specific file locations baked in
  - XML and STAR Meta-Scheduler (SUMS) protects users from changes in the batch system
- SUMS submits native SL7 jobs. Container needs small changes (not a showstopper but one needs to know)
  - Having a structured batch condor job script template in XML with STAR Meta-Scheduler helps to make these changes easily with a schema transformation and automate the transition.
- External inputs (example bad run lists) were tied to user's account.
  - I were able to find out the list in a presentation linked in the analysis note.
- Some batch jobs that look complete by exit code. For example, the author ensured that all his jobs ran 100% manually
  - Lesson learned: user code needs to be intentional i.e. wrong input → returns error code, which is not always the case. Standardization of end of workflow could be improved.

Consequence: documentation of *environment* matters as much as documentation of *physics cuts*

Challenges reproducing older analysis

- Hardcoded paths to original author's home (or data) areas.
- Some of the folders have file protections with 600 permissions.
    - convention MUST be that once an analysis is published, there cannot be protections such as 600. it must assume anyone can redo it (and have access to all input for every step)
- From the analysis README, the relative paths were confusing and for someone who did not perform the analysis, a small search had to be performed to locate the scripts referenced.
- This analysis had a bad coding practice of duplicating a package for multiple species, it could have been avoided
- Most of the above fixes are easy. Once a pattern is found across many analyses, it can be captured as a SKILL.md (that can be ported to multiple analysis)
- Original batch submission was perfect for the original analysis, only a couple of lines are required to run on new computing environment

Recommendations for similar STAR analyses

- Structure: one repo root Config (REPO_ROOT), one StRoot/, species/cuts as params not directory copies
- Ship external/ with all headers and small test picoDst or test list
- Add scripts/smoke_test.sh with PASS/FAIL before any batch submit
- Use Singularity in submit.xml from day one if using STAR Meta-Scheduler
- Separate stages clearly: histogram production (ROOT5) vs physics (ROOT6) vs plotting
- Add .gitignore for sched*, csh/, log/, error/, *.package/, *.root
- Write ENVIRONMENT.md: starver tag, container bind mounts, ROOT version per stage

Summary & takeaways

- Answering our question: "Are STAR analysis notes sufficient to reproduce an analysis as documented?”
    - Yes, with some minor changes to XML submit file to be able to use old sl7 containers, which could be easily fixed to be able to run on the current computing environment
- Gaps identified: documentation could be incomplete in the code README or analysis note (but is complete when complemented), the XML submission script is well structured and uniform for all analyses, but still there is some flexibility on how the user chooses datasets to be parsed
- Three-stage design (histo → physics → plot) is sound: invest in Stage 1 reproducibility first
- For future analyses: pinned env, no absolute user paths, best practices on how to handle external dependencies (bad run list for example).
- Next step: Check if the skills learned here are applicable to another analysis
- Long term plan is to use AI agents to drive containerization and capture the workflow