ROOT Q3 2024 Report

Danilo Piparo for the ROOT Project (CERN, EP-SFT)

24-09-2024

Executive Summary

- **ROOT 6.34.00 Development Release will be released in November**
  - Expose new features early to users, no patches after 6.36.00 is out
  - An even **more powerful RooFit**: automatic Differentiation through Clad
  - **An upgraded Pythonic interface**, e.g. Math, RooStats – a step towards a rewarding fully Pythonic interface
  - **LLVM18**
  - DistRDF: included stat inference in AGC example, API cleanup and upgraded interaction with C++
  - Training of ML models with zero-copy-reading from ROOT native datasets (RBatchGenerator)
  - **Target release for the RNTuple 1.0 binary format**
- 2025 proposed Release schedule: **May 6.36 for production**
  - November 2025: 6.38 for development
- **Training: a theme of Q3**
  - ~200 students and early career scientists trained, also in collaboration with HSF and IRIS-HEP
- Metrics:
  - **523 open issues (1539 in January) – 36% reduction**
  - 72 items PoW 44% complete - Q1 16%, Q2 28%, 2023 55% (55 items, including extra items)
  - **6h on average to provide feedback on posts on the Forum**

All statistics in this talk were gathered on September 15th

This Meeting

- **ROOT reporting every quarter to stakeholders,** third meeting of a series in 2024
- Usual questions:
  - Is the current Quarterly Report format useful to you?
  - What can we improve?
  - What should be preserved?
- Questions, comments, feedback are welcome at the end or during the talk!

ROOT's Strategic Goals

Provide a unified software package for the storage, processing, visualisation and analysis of scientific data that is reliable, performant, supported and sustainable, that is easy to use and obtain, and that minimises computing resources and scientists' time needed to achieve results.

The success of experiments and all ROOT users at large is our priority

R&D Projects

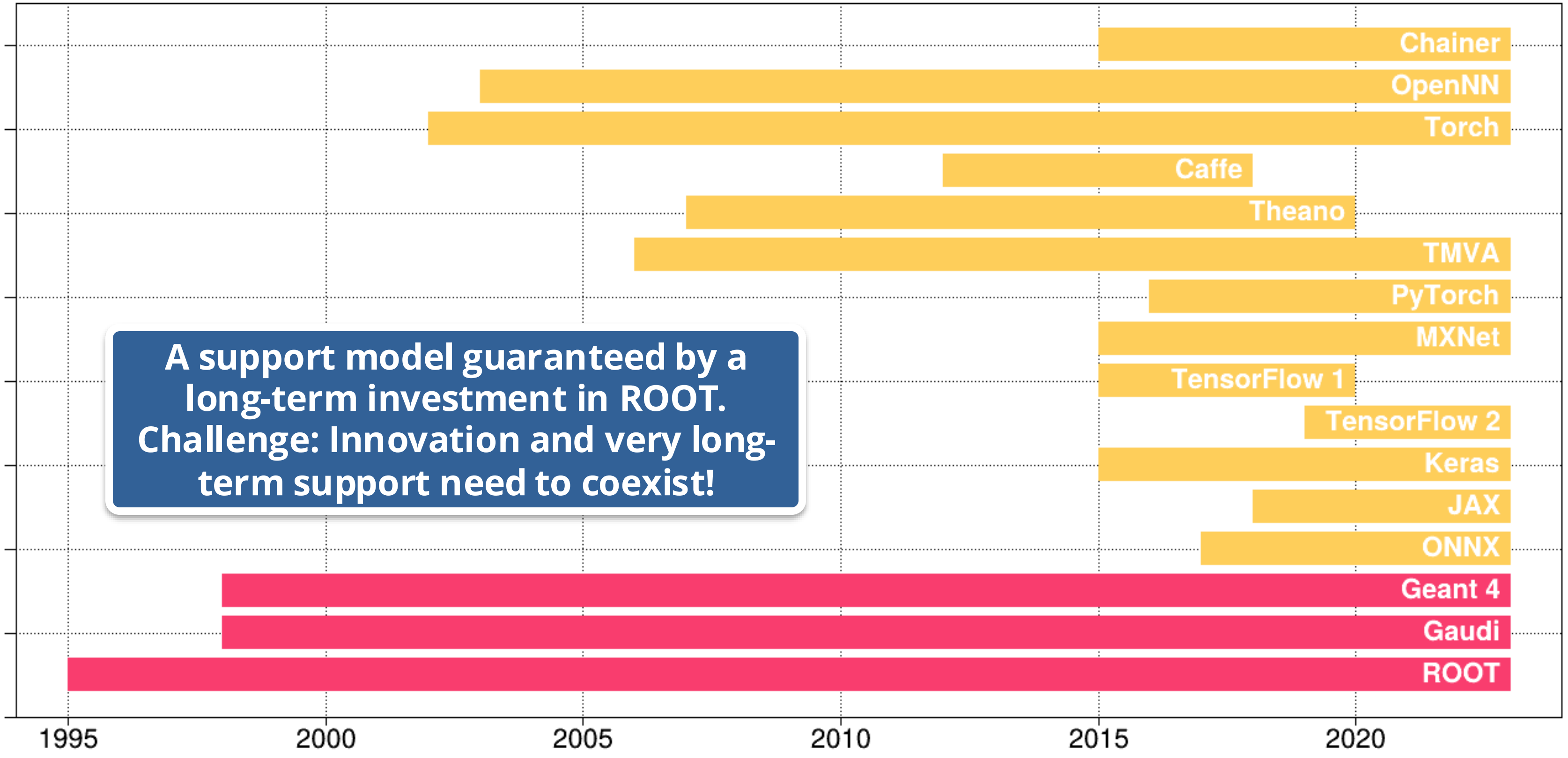
- HighLO: Project **High** Energy Physics Tools in **L**imit **O**rder Book Analysis (HighLO) applies particle physics methods and tools to financial market data.
- SYCLOPS (EU): Advancing AI/data mining for extremely large and diverse data for Europe and beyond, by democratizing its acceleration through open standards and a healthy, competitive, and innovating ecosystem.
- CERN Experimental Physics Department R&D (EP E&D): innovative I/O formats (RNTuple), Python-C++ interoperability, Analysis at scale, key4HEP
- Next Generation Triggers: Common software developments for heterogeneous architectures, efficient interfaces to Machine Learning inference engines to minimize data movements and execution latencies

And internally, other projects, e.g.

- Cling (C++ interpreter) automatic differentiation in RooFit (see ICHEP talk),
- RNTuple (the TTree successor) – 6y of R&D, now transitioning to production,
- New thread friendly histograms.

ROOT: a project where result-oriented R&D activities can blossom and reach thousands of scientists.

HEP Software Support Timeline



A support model guaranteed by a long-term investment in ROOT. Challenge: Innovation and very long-term support need to coexist!

Plot inspired by [M. Mazurek]

Who is ROOT?

- **ROOT is its user community, contributors, and developers**
- **ROOT is an international collaboration**
  - Institutional responsibilities, but also precious contributions coming from the user community!

Team from https://root.cern , PhD students onwards

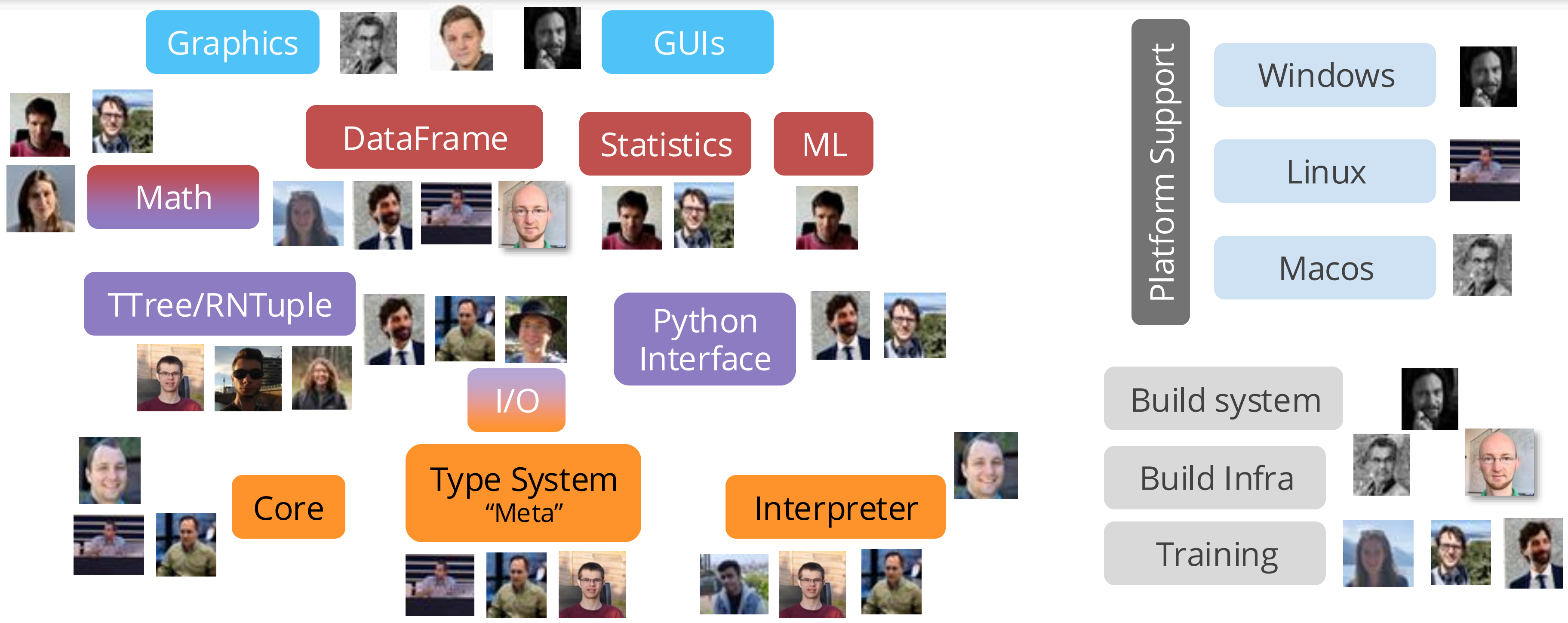
From left to right, starting from the first row (the affiliation is CERN if not specified):

1. Devajit Valaparambil, Bertrand Bellenot, Danilo Piparo, Florine de Geus, Jakob Blomer
2. Jonas Hanfeld, Jonas Rembser, Marta Czurylo, Olivier Couet, Philippe Canal **(FNAL)**,
3. Vasil Vassilev **(Princeton)**, Lorenzo Moneta, Monica Dessole, Vincenzo Padulano, Serguei Linev **(GSI)**
4. Jack Parolini, Stephan Hageböck

Plus students, working with us a few months, up to ~1 year

Not everybody in this slide is 100% dedicated to one project, but most are.

The Team and the Focus Areas



Team members are encouraged to be involved in more than one ROOT component, as well as to take part to baseline work.

The structure of the project, components and people's focus has been greatly simplified for this slide: to be taken with a grain of salt!

Highlights and Releases

Students during Summer

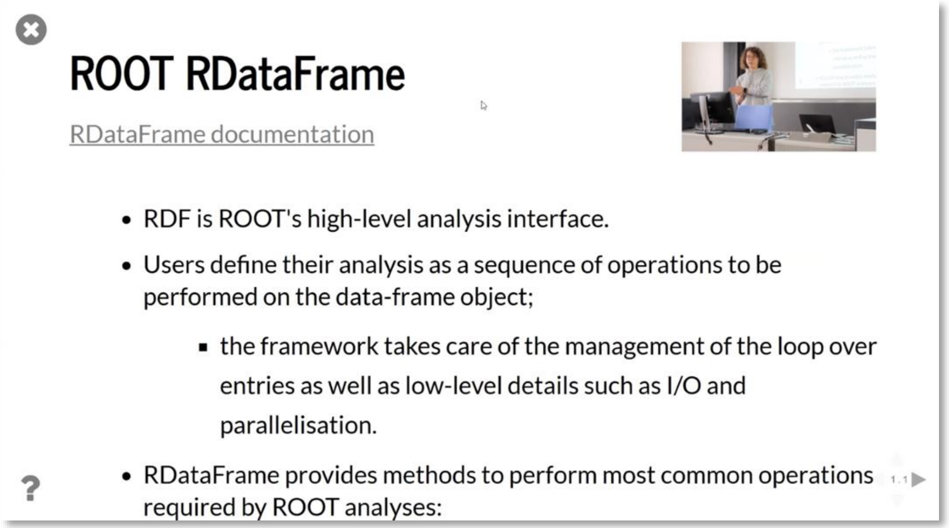
Very successful programs (summary presentations linked)

- CERN Summer Students
  - N. Subsa-Ard, Fast and efficient data pipelines for training ML models
  - K. Nobel, Batching histogram computations on GPUs
  - I. Caspary, Explorative data layout analysis of ne xt-generation High Energy Physics datasets
  - R. Syring, New statistical analysis techniques in RooFit
- CERN Openlab Students
  - A. Mehrabi, Evaluation of HPC Storage Systems for HEP Analyses
  - A. Ola Mejicanos, Boosting user experience of the ROOT data analysis interface
- IRIS-HEP Fellows
  - Valerii Kholoimov, Statistical analysis in the Analysis Grand Challenge with RooFit, Supervisors: J. Rembser, A. Held

Summer Trainings

- Scale up the training of early career colleagues
  - About 200 during 7 events
- Two categories of events:
  - *HSF/IRIS-HEP/SFT Python for Analysis* trainings an
  - *CERN Summer Student Workshops*
- ROOT Python interface and Notebooks
- Additional results:
  - [ROOT Video Course on CDS]
  - Complete surveys: the feedback received will be incorporated in future events
  - Several ROOT devs involved: we trained to train!

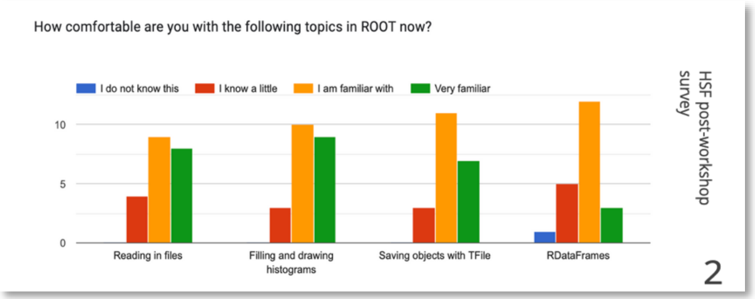
Check out [the blog post] on the ROOT website!



ROOT RDataFrame

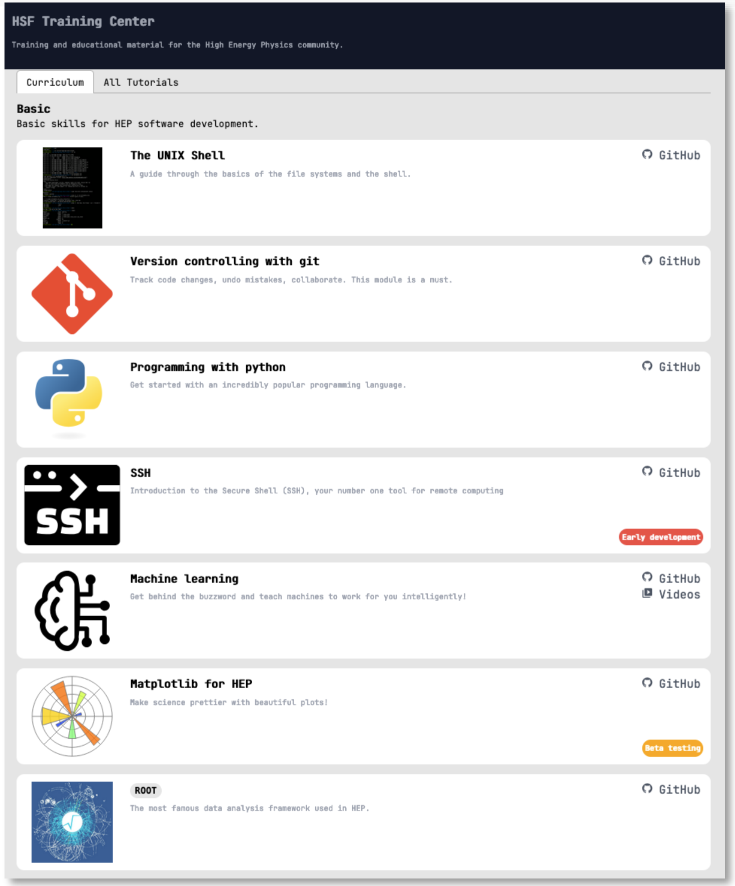
RDataFrame documentation

- RDF is ROOT's high-level analysis interface.
- Users define their analysis as a sequence of operations to be performed on the data-frame object;
  - the framework takes care of the management of the loop over entries as well as low-level details such as I/O and parallelisation.
- RDataFrame provides methods to perform most common operations required by ROOT analyses:



HSF post-workshop survey

Training: Interlude



HSF Training Center

Training and educational material for the High Energy Physics community.

Curriculum | All Tutorials

Basic

Basic skills for HEP software development.

The UNIX Shell — GitHub
A guide through the basics of the file systems and the shell.

Version controlling with git — GitHub
Track code changes, undo mistakes, collaborate. This module is a must.

Programming with python — GitHub
Get started with an incredibly popular programming language.

SSH — GitHub
Introduction to the Secure Shell (SSH), your number one tool for remote computing
Early development

Machine learning — GitHub, Videos
Get behind the buzzword and teach machines to work for you intelligently!

Matplotlib for HEP — GitHub
Make science prettier with beautiful plots!
Beta testing

ROOT — GitHub
The most famous data analysis framework used in HEP.

HSF Training Centre updated with the new material!

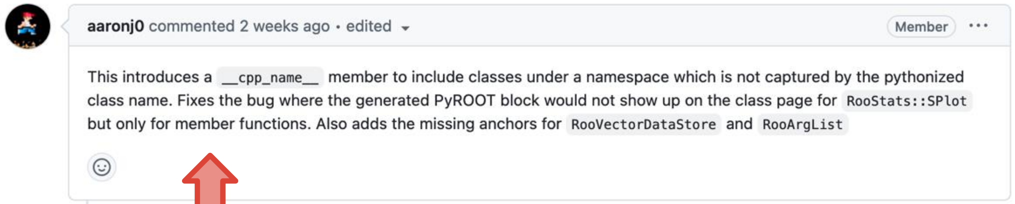
ROOT and Particle physics methods courses
https://hsf-training.org/training-center

Pythonic Interface

```python
import ROOT
import numpy as np
import math

def pyf_tf1_coulomb(x, p):
    return p[1] * x[0] * x[1] / (p[0]**2) * math.exp(-p[2] / p[0])

rtf1 = ROOT.TF1("my_func", pyf_tf1_coulomb, -10, 10, ndims = 2, npars = 3)
# x dataset: 5 pairs of particle charges
x = np.array([
    [1.0, 10, 2.0],
    [1.5, 10, 2.5],
    [2.0, 10, 3.0],
    [2.5, 10, 3.5],
    [3.0, 10, 4.0]
])
params = np.array([
    [1.0],      # Distance between charges r
    [8.99e9],   # Coulomb constant k (in N·m²/C²)
    [0.1]       # Additional factor for modulation
])
# Slice to avoid the dummy column of 10's
res = rtf1.EvalPar(x[:, ::2], params)
```


aaronj0 commented 2 weeks ago · edited
Member

This introduces a `__cpp_name__` member to include classes under a namespace which is not captured by the pythonized class name. Fixes the bug where the generated PyROOT block would not show up on the class page for `RooStats::SPlot` but only for member functions. Also adds the missing anchors for `RooVectorDataStore` and `RooArgList`

This means better documentation

```python
# Previously:
sData = ROOT.RooStats.SPlot("sData", "An SPlot", data, massModel, [zYield,
qcdYield], ROOT.RooStats.RooCmdArg("Strategy", 0))

# No longer require RooCmdArg instances
sData = ROOT.RooStats.SPlot("sData", "An SPlot", data, massModel, [zYield,
qcdYield], Strategy=0)
```

All this accompanied with a reduction of patches in cppyy down to 7

Providing a powerful and Pythonic interface is a priority

LLVM 16 → LLVM 18

- The Cling interpreter is based on LLVM
- ROOT 6.32 releases based on LLVM version 16 (LLVM16)
- **Upgraded to the new version for 6.34: LLVM18 (released in March)**
  - A veritable sprint, started even before the official llvm18 release
  - Profit from all functionality of this compiler version
  - Keep Cling and ROOT up to date with latest software technologies
  - Allows to work with very early stages (pre-release candidate period) of future LLVM version, e.g. by upstreaming in clang-repl (~a sort of "Cling from Scratch") features which are key for us and will therefore not be broken during the evolution of llvm.

New Colour Scheme and Fonts

Accessible Color Schemes

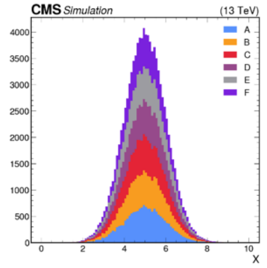
DP-2024-040

Recently standardized CMS *recommended* color scheme

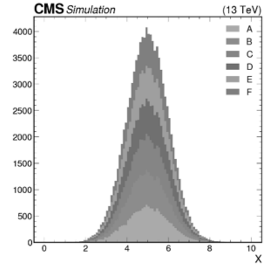
Chosen for **colorblind accessibility** (joint effort with **CMS Diversity Office**)

- From M. Petroff arxiv:2107.02270 – petroff6 or petroff10

Now also used by ATLAS



Greyscale conversion is also acceptable



Hex Code	Color Preview
#3f90da	🟦
#ffa90e	🟧
#bd1f01	🟥
#94a4a2	⬜
#832db6	🟪
#a96b59	🟫
#e76300	🟧
#b9ac70	🟨
#717581	⬛
#92dadd	🟦

Hex Code	Color Preview
#5790fc	🟦
#f89c20	🟧
#e42536	🟥
#964a8b	🟪
#9c9ca1	⬜
#7a21dd	🟪

07/19/2024 | Andrzej Novak   ICHEP 2024 (Prague)   16

"Common Analysis Tools in CMS", A. Nowak, ICHEP24

- ► New accessible colour scheme
- ► Adopted both by ATLAS and CMS
- ► Provided natively in 6.34: PR #1648
  - Code approved by CMS CAT (Common Analysys Tools) management
- ► For 2D plots, the current palette continues to be adopted

ROOT's active engagement with experiments' analysis tools teams continues

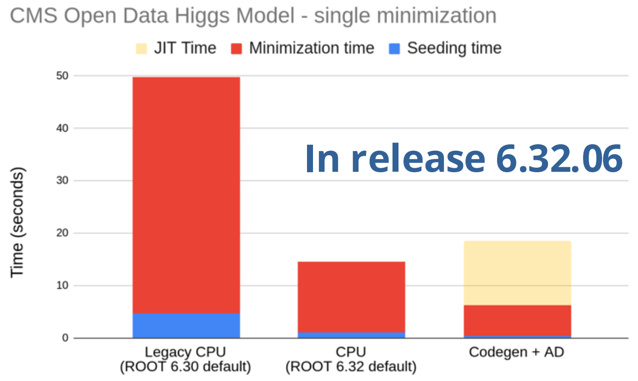
- ► Tex Gyre Heros (like Helvetica but open source) already in 6.32.06 (PR #14841)
- ► Honouring a request received by the CAT

Image from A. Nowak's talk

RooFit Achievements

- Significant advances in **Automatic Differentiation (AD) in RooFit with Clad**, resulting in the ICHEP presentation
  - Performance measurements with ATLAS toy benchmarks and open likelihoods from CMS
  - **Clad release 1.6** enabled this workflow, which requires differentiating through numerical integrals and other non-trivial functions
- New tutorials on **Simulation Based Inference (SBI)** in RooFit (*example tutorial rf615*)
  - Showcase the usage of ML classifiers trained with Python libraries in the context of a RooFit analysis
  - Benefit from generalized binding of Python functions for RooFit via RooFit::bindFunction()
- Implementation **of the Analysis Grand Challenge statistical analysis in RooFit, following up on the existing RDataFrame implementation**
  - Inspired by this project: improved HistFactory interfaces and updated tutorial

R&D became production grade, now available to the HEP Community

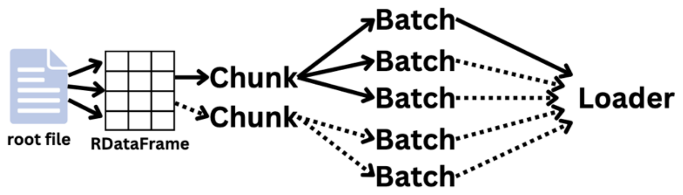


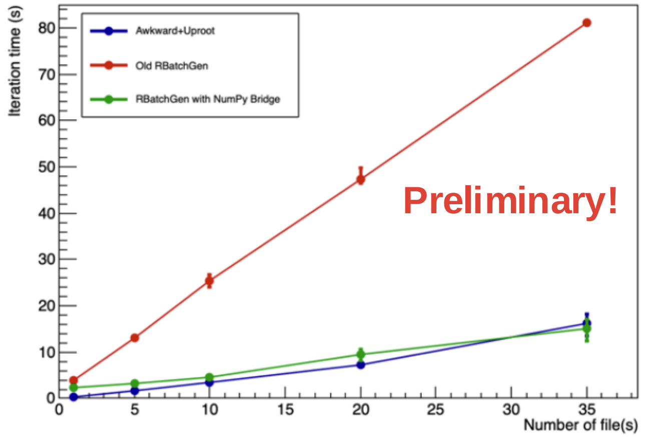
In release 6.32.06

RooFit AD for the example of the CMS Higgs Discovery likelihood: using the analytic gradient can significantly speed up minimization time, even more than the new vectorizing backend of RooFit in ROOT 6.32. One still has to pay for JITting the gradient code, which we aim to speed up further in the future.

RBatchGenerator

- Read data in ROOT format with RDataFrame
- Zero-copy creation of tensors (np.ndarray, torch.Tensor, tensorflow.Tensor)
- Pass data chunks to Torch for model training
- Advantages
  - No conversion of your training input datasets (1 set of files)
  - Profit from ROOT optimized (remote) reading capabilities



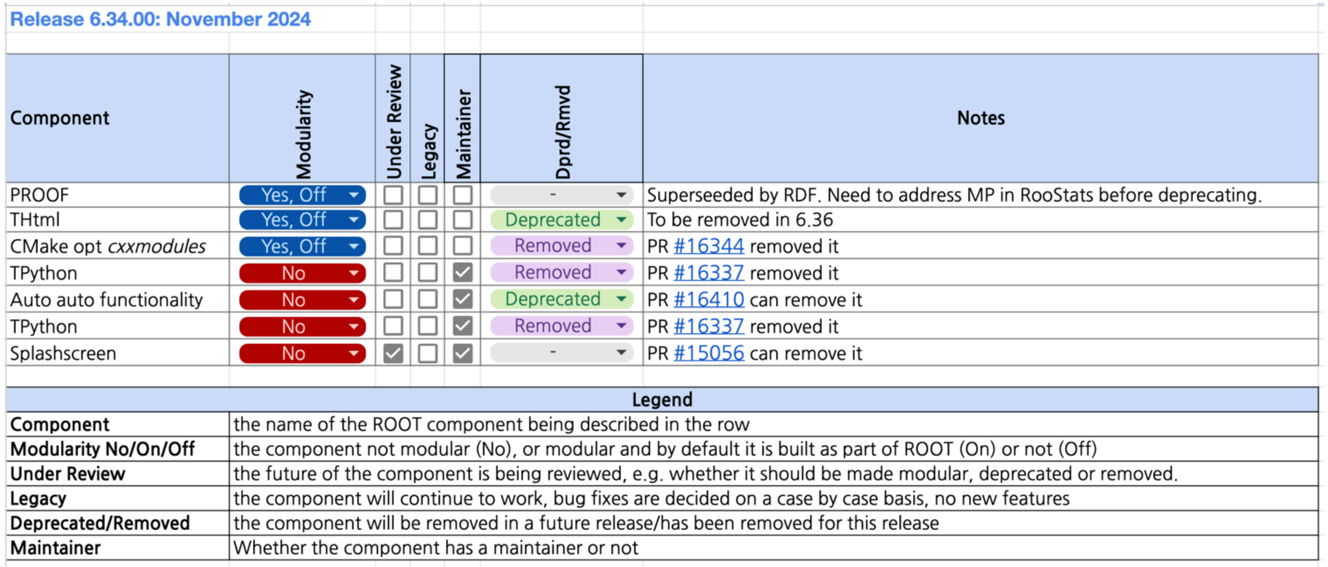


RBatchGenerator: a scalable, performant and sustainable solution for ML training?

THAI-CERN

N. Subsa-Ard

Managing Evolution: "ROOT Components Table"



Release 6.34.00: November 2024

Component	Modularity	Under Review	Legacy	Maintainer	Dprd/Rmvd	Notes
PROOF	Yes, Off	☐	☐	☐	-	Superseeded by RDF. Need to address MP in RooStats before deprecating.
THtml	Yes, Off	☐	☐	☐	Deprecated	To be removed in 6.36
CMake opt *cxxmodules*	Yes, Off	☐	☐	☐	Removed	PR #16344 removed it
TPython	No	☐	☐	✓	Removed	PR #16337 removed it
Auto auto functionality	No	☐	☐	✓	Deprecated	PR #16410 can remove it
TPython	No	☐	☐	✓	Removed	PR #16337 removed it
Splashscreen	No	✓	☐	✓	-	PR #15056 can remove it

Legend	
Component	the name of the ROOT component being described in the row
Modularity No/On/Off	the component not modular (No), or modular and by default it is built as part of ROOT (On) or not (Off)
Under Review	the future of the component is being reviewed, e.g. whether it should be made modular, deprecated or removed.
Legacy	the component will continue to work, bug fixes are decided on a case by case basis, no new features
Deprecated/Removed	the component will be removed in a future release/has been removed for this release
Maintainer	Whether the component has a maintainer or not

A Contribution to the streamlining of ROOT's modernisation

The process is new, and will be refined in the months to come.

- **Currently internal to the ROOT project**: once the process is better established, include users and stakeholders
- One table per release series: **track the evolution of the modularisation, legacy, deprecation and removal**
- **Regularly discussed by ROOT developers and contributors**
- Not a trivial task: need to increase sustainability of the code base, avoiding to break users' experiments' code

Proposed Releases in 2024-2025

Recap of 2024

- ROOT 6.32.00 Production Release in May 2024
- ROOT 6.34.00 Development Release in November 2024
  - Short support cycle: fixes backported until 6.36.00 is cut
  - Two main goals:
    1. Expose the many features which were added to ROOT since March
    2. Gather feedback by users to improve the 6.36 cycle: both its stability and feature set

Proposal for 2025: same timeline for releases in 2024

- ROOT 6.36.00 Production Release in May 2025
  - Data taking release, long support cycle
- ROOT 6.38.00 Development Release in November 2025
  - Mainly targeting analysis and exposing new features for all users
  - The following production Release will depend on LS3 plans of experiments (Legacy Processing campaigns, large MC productions, RNTuple commissioning…)

RNTuple: Flagship Development in 2024

- We are on track to finalize RNTuple 1st binary format production version, 1.0, this year
- **RNTuple successfully integrated in the ATLAS, CMS, and LHCb frameworks**
  - e.g. to read and write ATLAS AODs and DAODs and CMS MiniAOD and NanoAOD formats
  - Successful large-scale tests with the Analysis Grand Challenge were performed on a multi-petabyte EOS cluster.
- Backwards and forward compatibility:
  - **ROOT commits to backwards compatibility for data written as of v1.0**
  - **Compatibility with previous, experimental versions will be deliberately broken**
  - **Forward-compatibility: not guaranteed in the upcoming years – the format may be amended, e.g. new compression algorithms that work better for our data**
- Matching the release schedule:
  - **RNTuple 1.0 binary format target: release 6.34 (November)**
  - 1st production RNTuple APIs 6.36 release (May), incorporating the US HEP-CCE2/SOP Review feedback

RNTuple production release: the first major I/O upgrade after 25+ years of TTree serving HEP. It provides significant improvements to data size, read and write speed, and robustness in preparation of experiment data storage in the 2030s. It makes available to users the result of about 6 years of R&D.

HEP-CCE/SOP RNtuple API Review

- HEP-CCE **(Center for Computational Excellence) drives a RNtuple API Review**
- Objective: not only achieve computational performance (aim to provide *the best IO system for HEP*), but also **ergonomics, coding best practices and sustainable software**
- Mid-term report delivered to the ROOT Project on September 4th
- **Informative report, in-depth analysis carried out, useful recommendations**
- **Work already started to address them**
- Meeting with the reviewers scheduled for tomorrow

We thank reviewers for their hard work, which is highly valuable for the ROOT Project

U.S. DEPARTMENT OF ENERGY | Office of Science

Argonne NATIONAL LABORATORY   BROOKHAVEN NATIONAL LABORATORY   Fermilab   BERKELEY LAB Bringing Science Solutions to the World

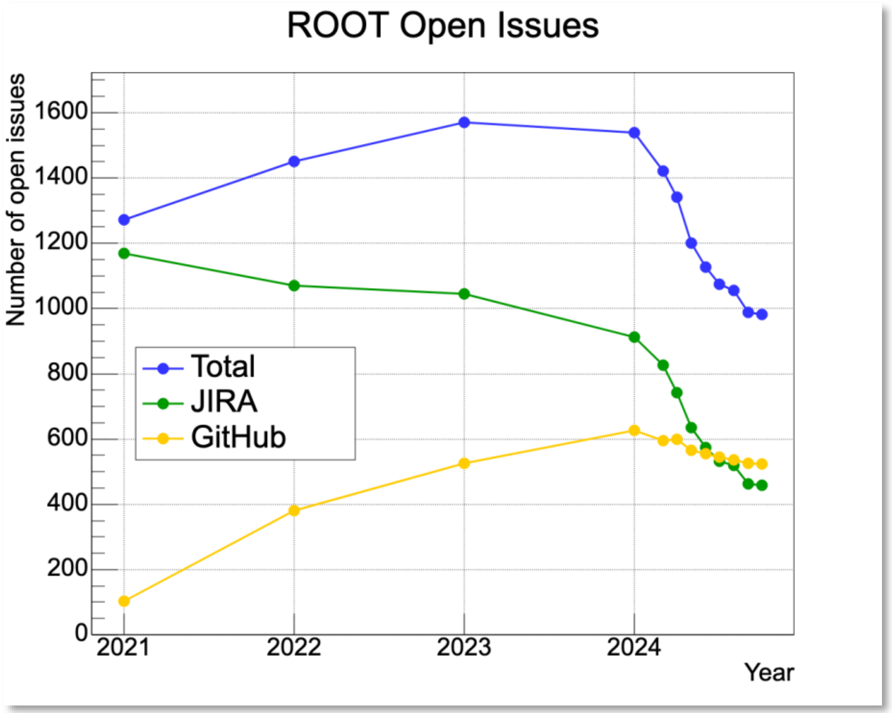
Q3 Metrics

Number of Open Issues

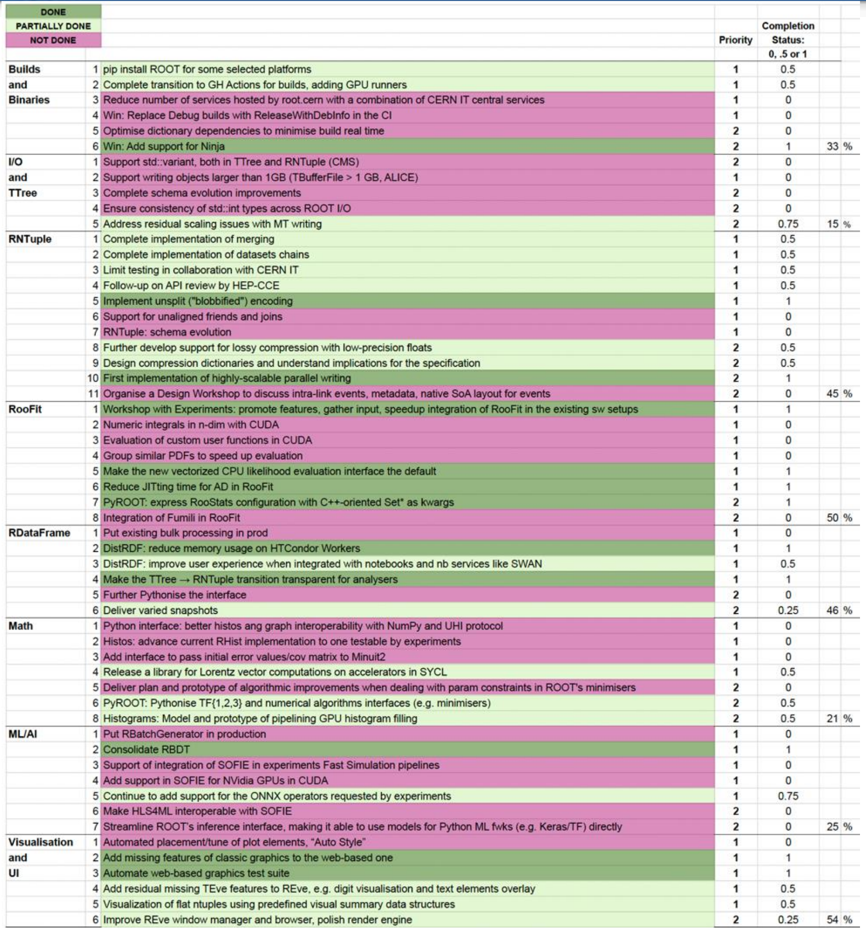
- Strong focus on reducing number of open issues
- Backlog reduction is implicitly *part of the PoW*
- 2024 so far:
  - 403 new issues created, of which 256 solved
  - **962 solved in total**
  - **36% reduction of # open issues**

Below the symbolic threshold of 1000 open issues

	JIRA	GitHub	Total	Notes
Dec-22	1045	525	1570	
Dec-23	912	627	1539	
Feb-24	826	596	1422	54 JIRA issues migrated to GH
Mar-24	739	601	1340	10 JIRA issues migrated to GH
Apr-24	635	566	1201	
May-24	573	555	1128	6.32.00 released
Jun-24	537	539	1076	1 JIRA issue migrated to GH
Jul-24	519	536	1055	2 JIRA issue migrated to GH
Aug-24	463	526	989	
Sep-24	458	523	981	

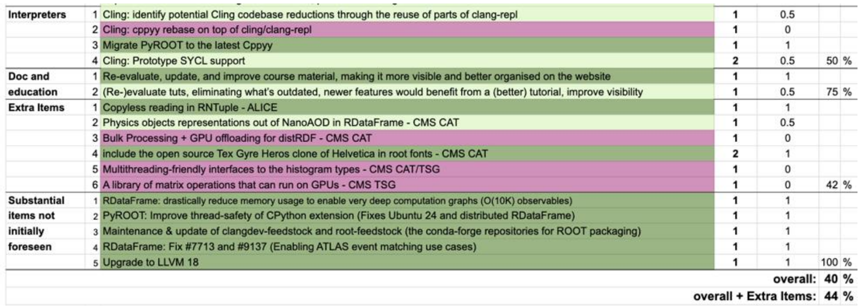


2024 PoW Completion



🟩	DONE			
🟩	PARTIALLY DONE			
🟥	NOT DONE			

			Priority	Completion Status: 0, .5 or 1	
Builds and Binaries	1	pip install ROOT for some selected platforms	1	0.5	
	2	Complete transition to GH Actions for builds, adding GPU runners	1	0.5	
	3	Reduce number of services hosted by root.cern with a combination of CERN IT central services	1	0	
	4	Win: Replace Debug builds with ReleaseWithDebInfo in the CI	1	0	
	5	Optimise dictionary dependencies to minimise build real time	2	0	
	6	Win: Add support for Ninja	2	1	33 %
I/O and TTree	1	Support std::variant, both in TTree and RNTuple (CMS)	2	0	
	2	Support writing objects larger than 1GB (TBufferFile > 1 GB, ALICE)	1	0	
	3	Complete schema evolution improvements	2	0	
	4	Ensure consistency of std::int types across ROOT I/O	2	0	
	5	Address residual scaling issues with MT writing	2	0.75	15 %
RNTuple	1	Complete implementation of merging	1	0.5	
	2	Complete implementation of datasets chains	1	0.5	
	3	Limit testing in collaboration with CERN IT	1	0.5	
	4	Follow-up on API review by HEP-CCE	1	0.5	
	5	Implement unsplit ("blobbified") encoding	1	1	
	6	Support for unaligned friends and joins	1	0	
	7	RNTuple: schema evolution	1	0	
	8	Further develop support for lossy compression with low-precision floats	2	0.5	
	9	Design compression dictionaries and understand implications for the specification	2	0.5	
	10	First implementation of highly-scalable parallel writing	2	1	
	11	Organise a Design Workshop to discuss intra-link events, metadata, native SoA layout for events	2	0	45 %
RooFit	1	Workshop with Experiments: promote features, gather input, speedup integration of RooFit in the existing sw setups	1	1	
	2	Numeric integrals in n-dim with CUDA	1	0	
	3	Evaluation of custom user functions in CUDA	1	0	
	4	Group similar PDFs to speed up evaluation	1	0	
	5	Make the new vectorized CPU likelihood evaluation interface the default	1	1	
	6	Reduce JITting time for AD in RooFit	1	1	
	7	PyROOT: express RooStats configuration with C++-oriented Set* as kwargs	2	1	
	8	Integration of Fumili in RooFit	2	0	50 %
RDataFrame	1	Put existing bulk processing in prod	1	0	
	2	DistRDF: reduce memory usage on HTCondor Workers	1	1	
	3	DistRDF: improve user experience when integrated with notebooks and nb services like SWAN	1	0.5	
	4	Make the TTree → RNTuple transition transparent for analysers	1	1	
	5	Further Pythonise the interface	2	0	
	6	Deliver varied snapshots	2	0.25	46 %
Math	1	Python interface: better histos ang graph interoperability with NumPy and UHI protocol	1	0	
	2	Histos: advance current RHist implementation to one testable by experiments	1	0	
	3	Add interface to pass initial error values/cov matrix to Minuit2	1	0	
	4	Release a library for Lorentz vector computations on accelerators in SYCL	1	0.5	
	5	Deliver plan and prototype of algorithmic improvements when dealing with param constraints in ROOT's minimisers	2	0	
	6	PyROOT: Pythonise TF{1,2,3} and numerical algorithms interfaces (e.g. minimisers)	2	0.5	
	8	Histograms: Model and prototype of pipelining GPU histogram filling	2	0.5	21 %
ML/AI	1	Put RBatchGenerator in production	1	0	
	2	Consolidate RBDT	1	1	
	3	Support of integration of SOFIE in experiments Fast Simulation pipelines	1	0	
	4	Add support in SOFIE for NVidia GPUs in CUDA	1	0	
	5	Continue to add support for the ONNX operators requested by experiments	1	0.75	
	6	Make HLS4ML interoperable with SOFIE	2	0	
	7	Streamline ROOT's inference interface, making it able to use models for Python ML fwks (e.g. Keras/TF) directly	2	0	25 %
Visualisation and UI	1	Automated placement/tune of plot elements, "Auto Style"	1	0	
	2	Add missing features of classic graphics to the web-based one	1	1	
	3	Automate web-based graphics test suite	1	1	
	4	Add residual missing TEve features to REve, e.g. digit visualisation and text elements overlay	1	0.5	
	5	Visualization of flat ntuples using predefined visual summary data structures	1	0.5	
	6	Improve REve window manager and browser, polish render engine	2	0.25	54 %



Interpreters	1	Cling: identify potential Cling codebase reductions through the reuse of parts of clang-repl	1	0.5	
	2	Cling: cppyy rebase on top of cling/clang-repl	1	0	
	3	Migrate PyROOT to the latest Cppyy	1	1	
	4	Cling: Prototype SYCL support	2	0.5	50 %
Doc and education	1	Re-evaluate, update, and improve course material, making it more visible and better organised on the website	1	1	
	2	(Re-)evaluate tuts, eliminating what's outdated, newer features would benefit from a (better) tutorial, improve visibility	1	0.5	75 %
Extra Items	1	Copyless reading in RNTuple - ALICE	1	1	
	2	Physics objects representations out of NanoAOD in RDataFrame - CMS CAT	1	0.5	
	3	Bulk Processing + GPU offloading for distRDF - CMS CAT	1	0	
	4	include the open source Tex Gyre Heros clone of Helvetica in root fonts - CMS CAT	2	1	
	5	Multithreading-friendly interfaces to the histogram types - CMS CAT/TSG	1	0	
	6	A library of matrix operations that can run on GPUs - CMS TSG	1	0	42 %
Substantial items not initially foreseen	1	RDataFrame: drastically reduce memory usage to enable very deep computation graphs (O(10K) observables)	1	1	
	2	PyROOT: Improve thread-safety of CPython extension (Fixes Ubuntu 24 and distributed RDataFrame)	1	1	
	3	Maintenance & update of clangdev-feedstock and root-feedstock (the conda-forge repositories for ROOT packaging)	1	1	
	4	RDataFrame: Fix #7713 and #9137 (Enabling ATLAS event matching use cases)	1	1	
	5	Upgrade to LLVM 18	1	1	100 %
				overall:	40 %
				overall + Extra Items:	44 %

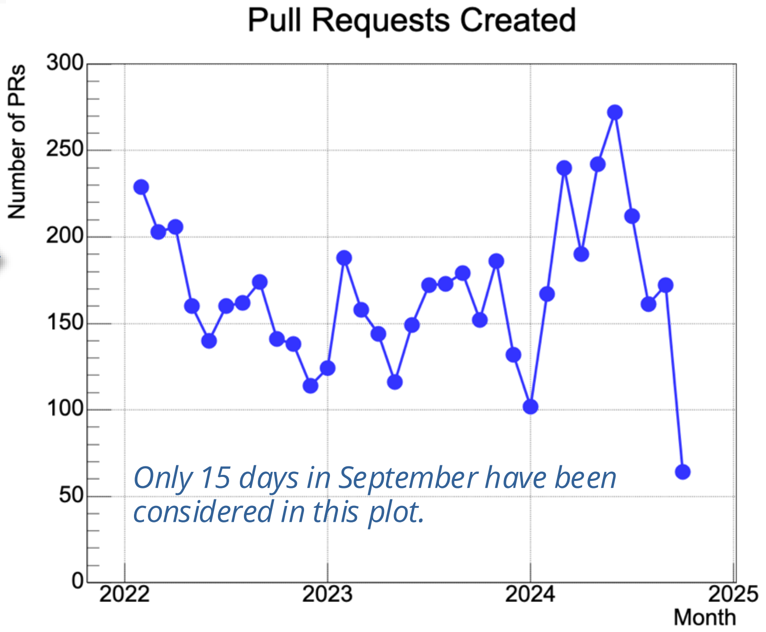
- A large PoW: 72 items
- **New arrivals helped, however they started later than initially planned**
- External help, i.e. ROOT community (e.g. experiments, but not only) can make the difference, too

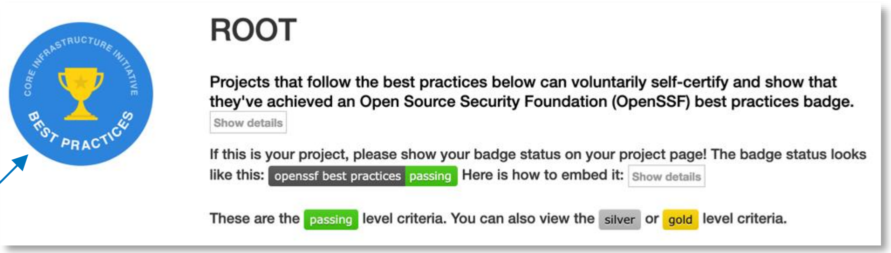
PoW is 45% complete if accounting for extra items

For comparison, 2023: 55 items (including extra items), 55.4% - All details at cern.ch/root-pow

ROOT Community and Development

- **ROOT is an open-source project, adopting an open-development model and open-planning, supported by a lively community**
- ROOT Forum: 7h on average to obtain a first response (20h in 2023) – 7.2k posts so far in 2024 (11.8k in 2023)
- **1720/1699 PR to the root repo created/closed, 1 day median to close a PR**
  - 1740/1698 in 2023





ROOT

Projects that follow the best practices below can voluntarily self-certify and show that they've achieved an Open Source Security Foundation (OpenSSF) best practices badge. Show details

If this is your project, please show your badge status on your project page! The badge status looks like this: openssf best practices passing Here is how to embed it: Show details

These are the passing level criteria. You can also view the silver or gold level criteria.

Since >3 years

Round Table

Written Input by CMS 1/3

- Questions
  - An update of ROOT in the CMSSW Integration Build that follows ROOT's master branch broke the build because ROOT::Minuit2::FCNBase::operator() API changed, and we have code inheriting from that class (moved from const reference of std::vector to std::span). Should we expect more of this kind of API breakage from modernization? If yes, could we get advance warnings? We would prefer an approach where both old and new API are available in one ROOT production release so that we could migrate to the new API in small chunks? For example, marking the old API deprecated e.g. with [[deprecated]] attribute would allow us to easily notice the deprecated API.

    This was not intendend and will be reverted. We apologise for the problem and thank you for the report.
  - How soon after the RNTuple on-disk format freeze can we expect a ROOT release?

    ROOT 6.34 will follow the freeze during Q4 2024.
- High priority
  - Schema evolution IORule not working for std::auto_ptr<T> -> std::unique_ptr<T> for a split TTree (cms-sw/cmssw#43923). This may be necessary for the planned re-MiniAOD processing of Run 2 UL MC on CMSSW_15_0_X.

    We could not deliver this, Philippe aware of the issue.
  - Inclusion of color-vision-deficiency-friendly palettes (root-project/root#16348, https://codimd.web.cern.ch/s/vdTvYZlSf#Inclusion-of-color-vision-deficiency-friendly-palettes)

    PR #16348 already available, will be in 6.34.
  - Inclusion of Tex Gyre Heros fonts. Tex Gyre Heros is an open source clone of Helvetica. This is particularly relevant for the rendering of "CMS" on plots. (https://codimd.web.cern.ch/s/vdTvYZlSf#Inclusion-of-Tex-Gyre-Heros-fonts)

    Already in release 6.32, PR #14841
- ► Release of conda packages. It would be desirable if the release of conda packages followed immediately the new releases. E.g. the latest package in conda is 6.32.02, while the latest release is 6.32.04. (https://codimd.web.cern.ch/s/vdTvYZlSf#Release-of-conda-packages)

  We take note of the importance, and will assign to this enough effort. Any help from CMS, or somewhere else, would be extremely appreciated.

Written Input by CMS 2/3

Medium priority

- More diagnostics from ROOT in “bytecount too large” errors, such as the branch causing the error. These errors have been see in prompt reconstruction, although the real culprit seems to have been a skim selecting much more events than expected (cms-sw/cmssw#45089, cms-sw/cmssw#40132)
- Mechanism to read files with std::auto_ptr<T> without relying on existence on std::auto_ptr<T> (cms-sw/cmssw#43422)
- RNTuple: Pattern for storing Structure-of-Array data structures
- RNTuple: schema evolution support
- Following three have workarounds
  - Schema evolution problem with a recursive class when reading a split TTree (cms-sw/cmssw#43768)
  - Schema evolution problem when removing an intermediate base class (cms-sw/cmssw#43516)
  - Towards a transient-persistent layer to read NanoAOD in RDataFrame (cms-sw/cmssw#45972

We could not deliver as much as we wanted in the medium priority level because of the load generated in other areas.

SoA: no news, but something to move forward at the focus days / hackathon (25-26-27 Nov)

Schema evolution: simple automatic cases work or have a draft PR (adding member, removing member, change member type to something else compatible, base class changes)
More complex automatic cases and manual rules on non-transient members are yet to complete
At the moment, the priority with CMS is the framework support (proper read & write of MiniAOD)

Written Input by CMS 3/3

- Low priority
  - Fix GetMethodWithPrototype returning incorrect function (root-project/root#7955, root-project/root#16232)
  - Fix for string cut parser sometimes not seeing all methods of an object when using ROOT reflection (cms-sw/cmssw#33084)
  - Infrequent crashes in Cling on x86 (cms-sw/cmssw#44659, cms-sw/cmssw#44438)
  - Crashes in Cling on ARM (cms-sw/cmssw#43802, cms-sw/cmssw#41961)
  - A case where rootcling crashes when generating dictionaries for type aliases (cms-sw/cmssw#44386)

It is worth checking if the the Cling related items on x86 are solved in master by the llvm18 upgrade

We are updating our CI with Linux arm nodes to be able to extend coverage for that platform and address this kind of problems *before* they reach experiments.

ROOT thanks CMS for coordinating the collection of this input from different groups and commits to work to solve the items brought to our attention