Understanding discrepancies in tracking efficiencies

Martin Tat

Heidelberg University

13th August 2025

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Introduction

Recap from last time:

- Had a look at tracking efficiencies using TrackCalib2
- Fits to MC
- Studied fit biases
  1. In some bins the uncertainties were underestimated
  2. Improve by changing background parameterisation
  3. $(N_{\mathrm{tot}}, \epsilon_{\mathrm{track}}) \rightarrow (N_{\mathrm{matched}}, N_{\mathrm{failed}})$ for background yields
- Today:
  1. Some developments to TrackCalib2
  2. Further tweaks to stabilise fits and reduce fit biases
  3. Fits to data

Tag-and-probe method

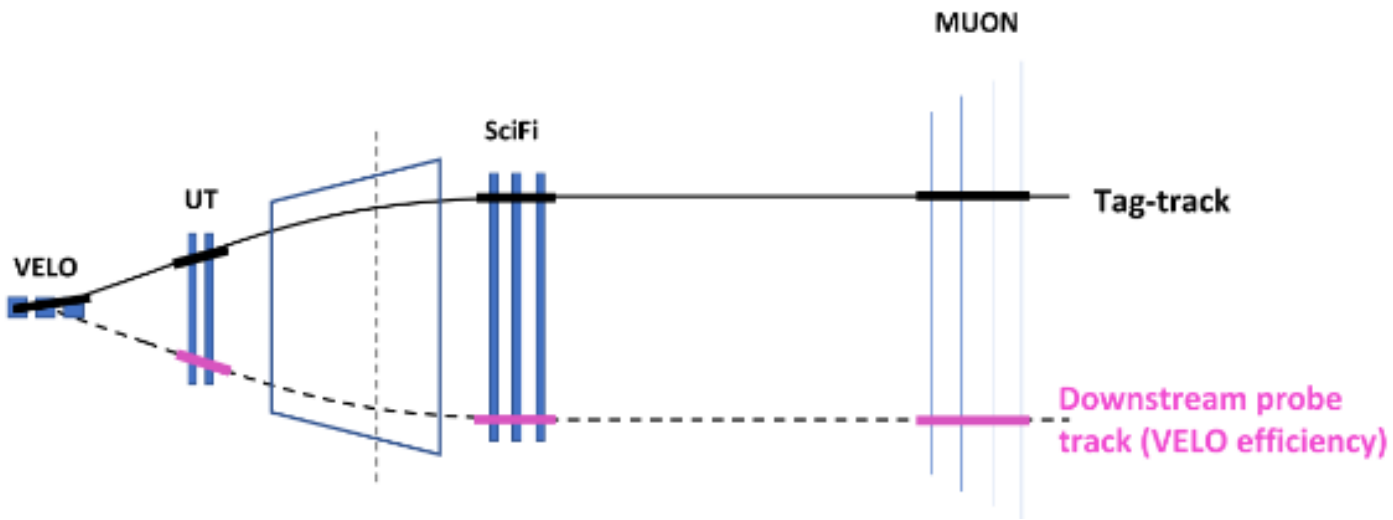


Figure from Rowina's thesis

- Fully reconstruct one muon from $J/\psi \to \mu^+\mu^-$
- Partially reconstruct the other muon
- Match hits in specific sub-detector with partially reconstructed track

$$\epsilon_{\text{track}} = \frac{N_{\text{matched}}}{N_{\text{matched}} + N_{\text{failed}}}$$

Tweaks to TrackCalib2

- TrackCalib2 works well out-of-the-box
- But I wanted to make a few changes, after my studies on fit biases
  - TrackCalib2 developments in the `mtat/dev/2024` branch
  - TCFit developments still local, no push access yet
- Improvements are mainly to stabilise the fit and ensure good convergence

TrackCalib2 tweak 1

Changed background yield parameterisation

- $(N_{\mathrm{tot}}, \epsilon_{\mathrm{track}}) \rightarrow (N_{\mathrm{matched}}, N_{\mathrm{failed}})$
- Some efficiencies and uncertainties have moved slightly
  - Particularly important in bins where the signal yield in the failed sample is very small

TrackCalib2 tweak 2

Change from exponential to Chebyshev polynomial for background shape

- Exponential shape doesn't describe background well, especially in data
- TrackCalib2 by default uses Chebyshev polynomials of order 3, with separate shapes for matched and failed samples
  - $\rightarrow$ Change to second order for failed sample
- Improves fit quality, but also makes it slower and perhaps more unstable because it adds 4 additional parameters

TrackCalib2 tweak 3

Change parameterisation of the widths σ

- Matched signal shape is two Crystal Ball functions with different σ
- Failed signal shape also has two Crystal Ball functions, where one of the σ are shared with the matched shape
- In total 3 different σ parameters $\rightarrow$ Can lead to degeneracies and unstable fits, especially in data
- Instead I changed it to $\sigma' = R\sigma$ where the ratio R is floating
  - Can also fix R in the fit to data, so that only a single σ parameter is floated in the fit to data

TrackCalib2 tweak 4

Bug fix in Crystal Ball shape

- Two independent bugs in TCFit and RooFit
  - RooFit bug was only revealed because of TCFit bug!
- TCFit bug: Exponent n in the power-law tail was forced to be $n > 1$
  - This restricts the shape to have very small tails, but tails can be large
  - Wikipedia wrongly states that $n > 1$, but this is only true if the range is infinite, for finite ranges we are only restricted by $n > 0$!

TrackCalib2 tweak 4

Bug fix in Crystal Ball shape

- RooFit bug: In a range $n - 1 \in [-10^{-5}, 10^{-5}]$, an approximation must be used in the normalisation integral (with $b = (n/\alpha)^n - \alpha$):

$$\begin{aligned}
&\left(\frac{n}{\alpha}\right)^n \times \frac{(b - z_{\min})^{1-n} - (b + z_{\max})^{1-n}}{1 - n} \\
\approx &\left(\frac{n}{\alpha}\right)^n \times \left(\ln(b - z_{\min}) - \ln(b + z_{\max})\right)
\end{aligned}$$

- This is wrong!

TrackCalib2 tweak 4

Bug fix in Crystal Ball shape

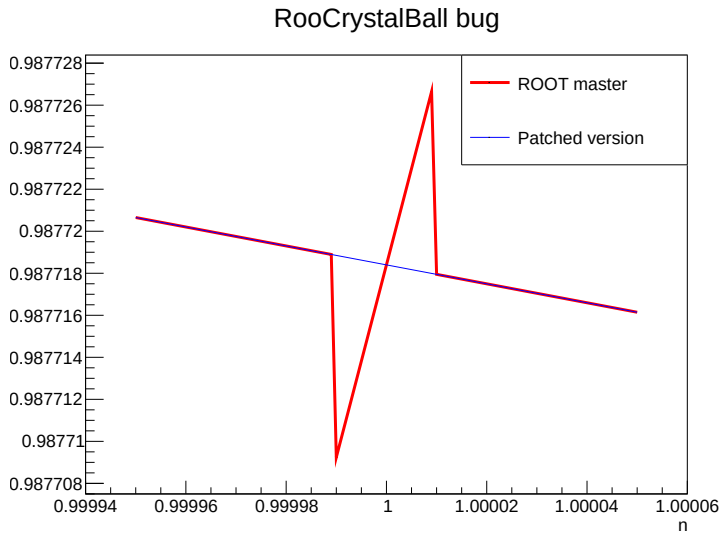
$$\left(\frac{n}{\alpha}\right)^n \times \frac{(b - z_{\min})^{1-n} - (b + z_{\max})^{1-n}}{1 - n}$$
$$\approx \left(\frac{n}{\alpha}\right)^n \times \left(\ln(b - z_{\min}) - \ln(b + z_{\max})\right)$$

- Prefactor $(n/\alpha)^n$ can be expanded to first order in $n - 1$
- So numerator must be expanded to second order

$$\left(\frac{n}{\alpha}\right)^n \times \left(\ln(b - z_{\min}) - \ln(b + z_{\max}) \right.$$
$$\left. + \frac{1}{2}(1 - n)(\ln(b - z_{\min})^2 - \ln(b - z_{\max})^2)\right)$$

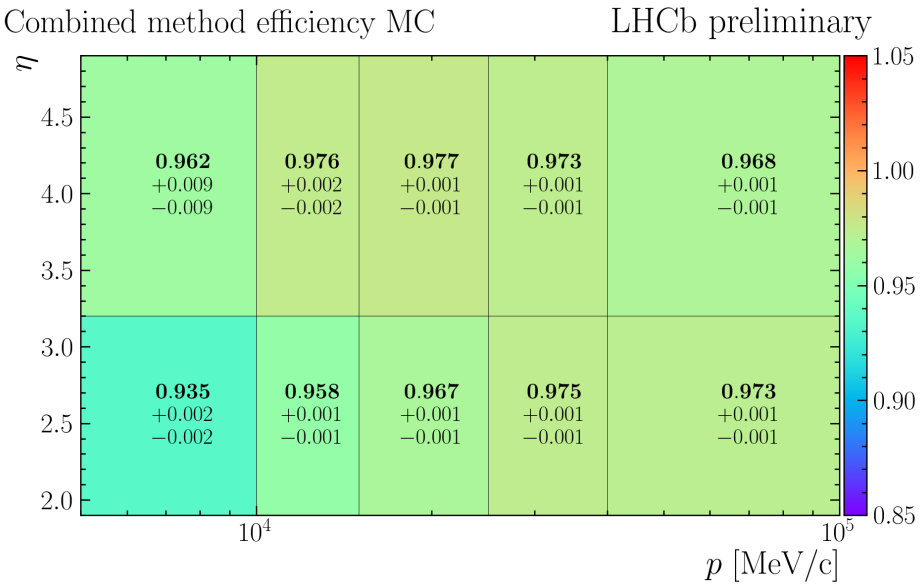
TrackCalib2 tweak 4

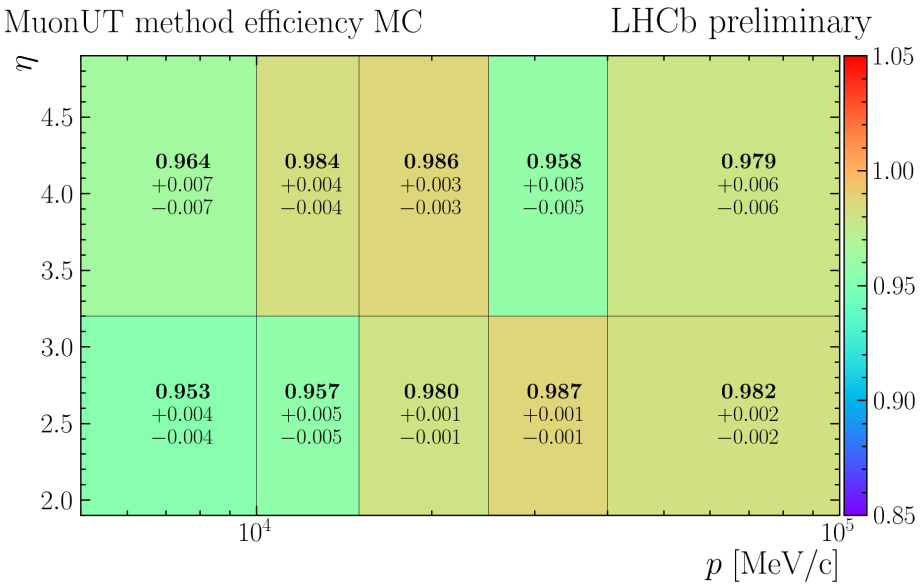
Bug fix in Crystal Ball shape



- This bug can trick the fit into converging near $n = 1$ with a very small uncertainty
- I assume it's highly unlikely, and people probably just changed the starting parameters when that happened in the past...
- ... but with the TCFit, it resulted in weird behaviour near $n = 1$
- Opened a pull request to the ROOT project here: !19602

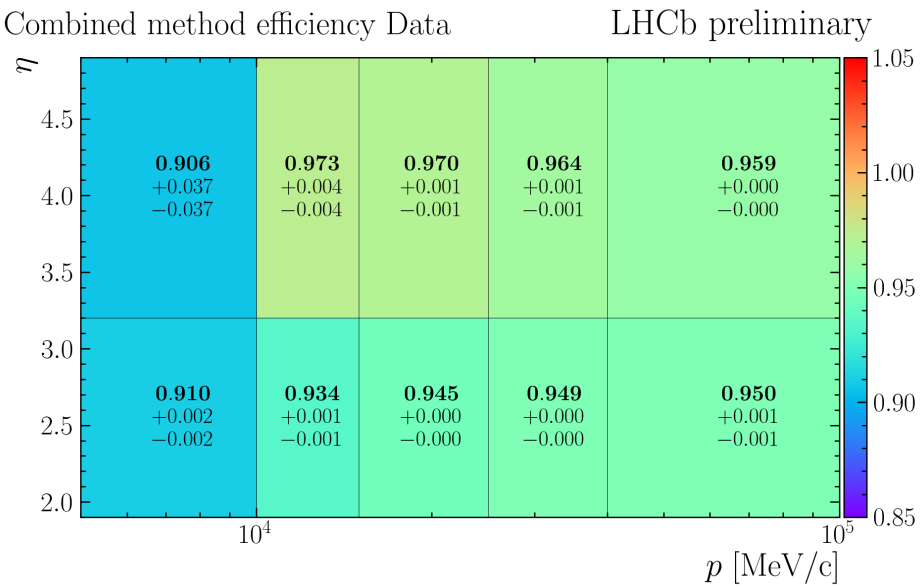
Tracking efficiencies

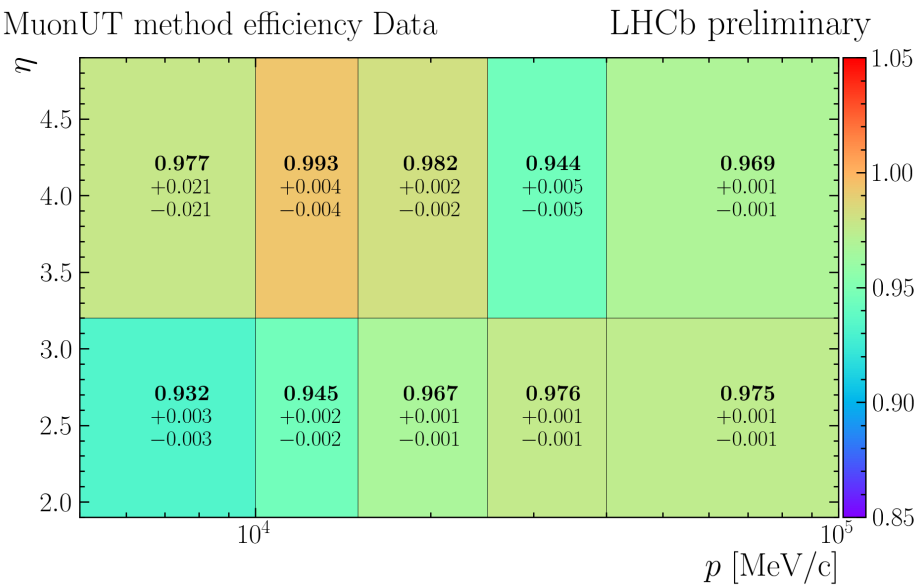




- Discrepancies are roughly 1% between Combined and MuonUT methods
- Bin $(0, 0)$ has a $(2.2 \pm 0.4)\%$ discrepancy

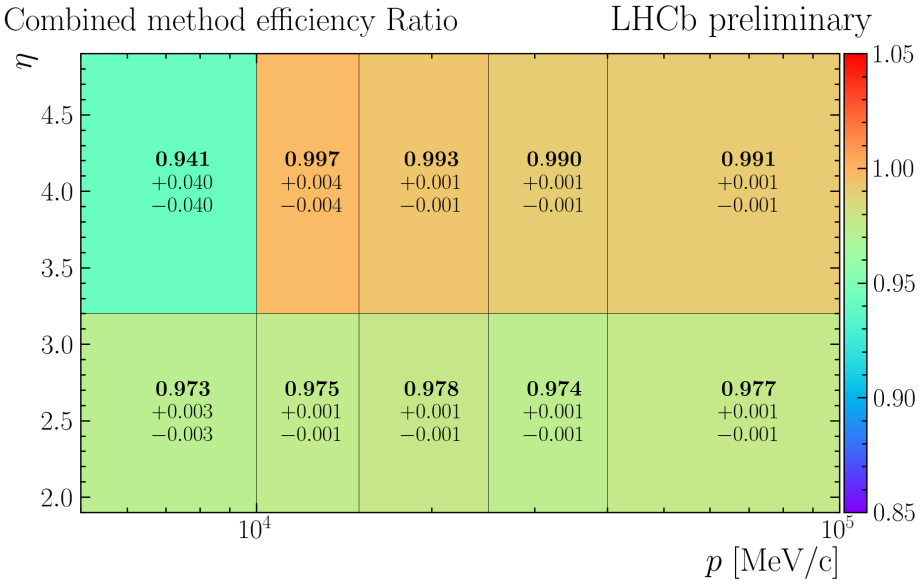
Tracking efficiencies

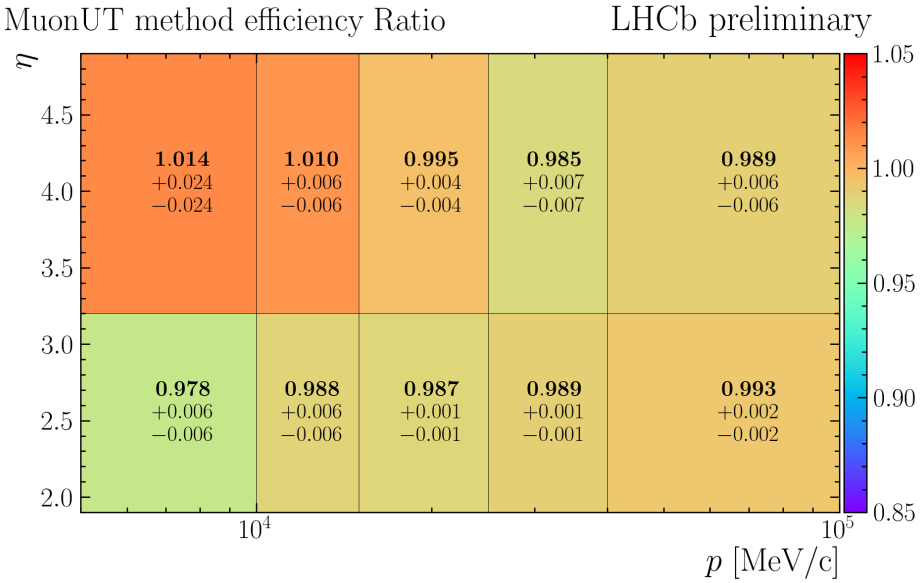




- Discrepancies are a lot larger in data than in MC, around 2%
- Bin $(0, 1)$ has a large discrepancy, but so are the (statistical) uncertainties

Tracking efficiencies





- However, when looking at the ratio, the discrepancies are mostly around 1%
- Bin $(0, 1)$ has a large discrepancy due to large uncertainties...
- ... but apart from this bin, the largest discrepancy is 1.5%

Summary and next steps

- Tweaks to TrackCalib2 to address fit biases/stability
- Fit performed on both data and MC
- Combined and MuonUT discrepancies are around 1–1.5%
- Discussion: Are these discrepancies significant?
  1. I agree there are systematic differences between the Combined and MuonUT methods
  2. But the bins with largest differences also have huge statistical uncertainties...
  3. ... so assigning a systematic of 1.0% or 1.5% seems reasonable to me
- Next steps:
  1. Float-point precision issues in unbinned fits
  2. More thinking about other effects that can bias the efficiencies

Thanks for listening!