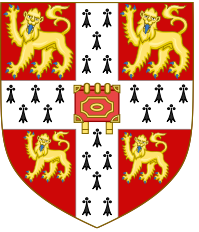


CERN openlab

Numerical study of floating-point precision within the PEPPER event generator

CERN openlab Summer Student Programme Lightning Talks

Yucheng Zhang

Supervisor: Dr. Enrico Bothmann & František Stloukal

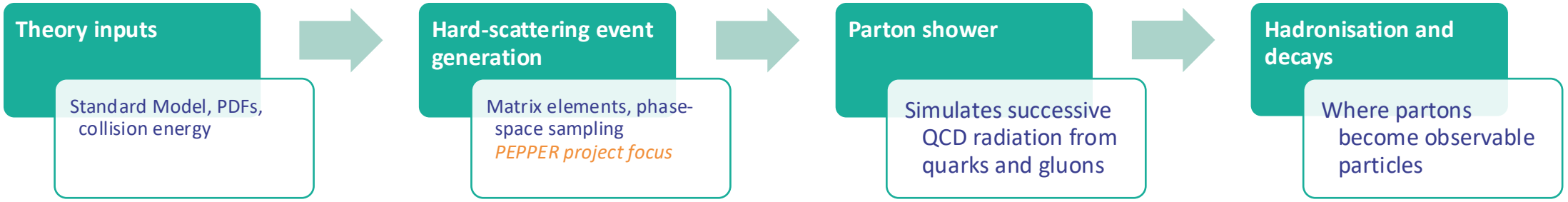
18/08/2026

Introduction

Background

- **Event generation** simulates possible particle-collision outcomes predicted by theoretical models.
- Simulated data are compared with experimental data.
- These simulated events are essential for modelling signals and backgrounds, designing analyses and interpreting experimental measurements.
- From the current LHC programme to the future **HL-LHC**[1], there is a growing demand in larger and more precise simulations.

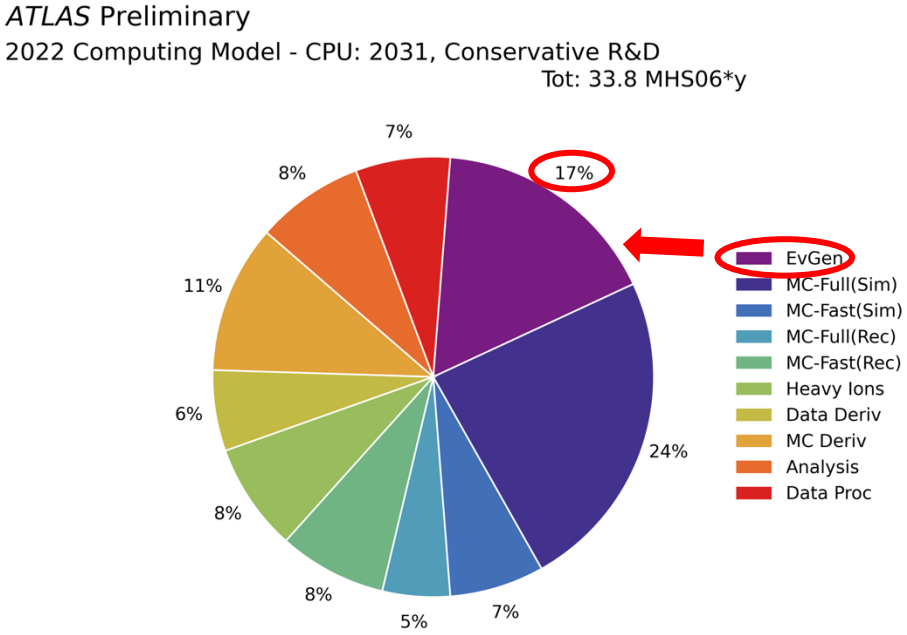
Event generator workflow (simplified)



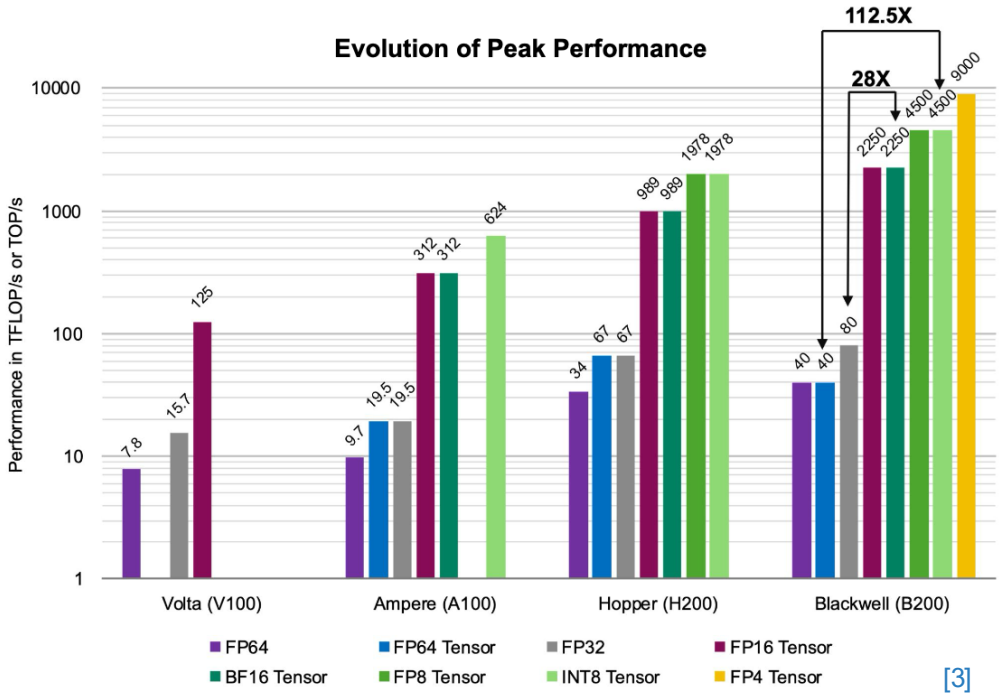
Introduction

Motivation

- Event generation is projected to account for around **17% of ATLAS CPU demand by 2031**.
- Growing costs motivate more efficient approaches such as using **GPUs and reduced precision**.
- Lower precision can improve performance, but may introduce numerical instabilities; this project investigates **where precision is lost and whether lower precision can be used reliably in PEPPER**.



[2]



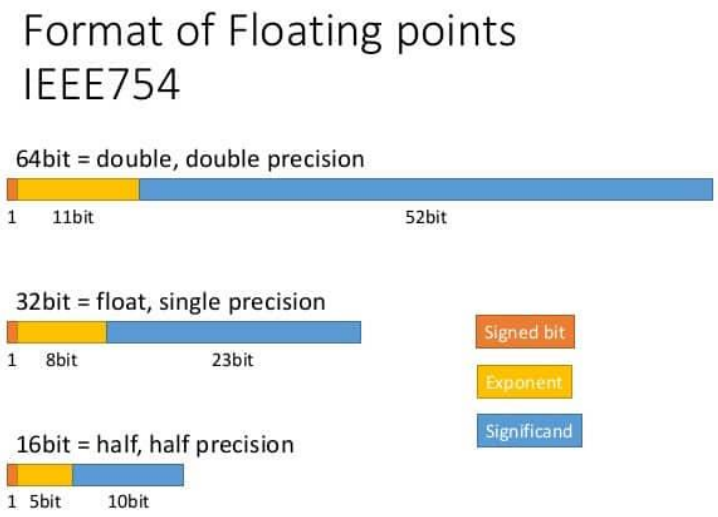
[3]

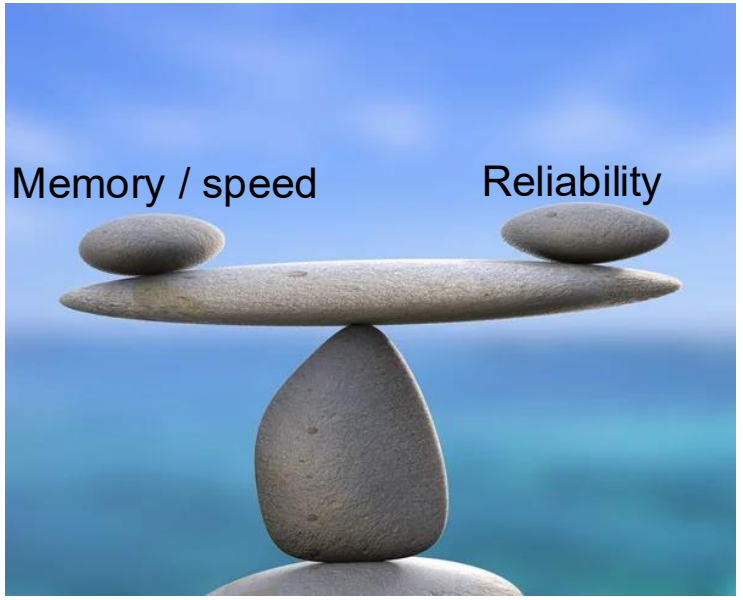
Project Aims

Floating-Point Precision

- Computers approximate real numbers using floating-point formats, most commonly single precision **FP32** (7 digits aka Float) and double **FP64** (16 digits aka Double).
- FP32 requires less memory and can be faster on GPUs, but represents numbers with fewer digits than FP64.
- Rounding errors and numerical cancellations can propagate through event generation and produce unreliable results.
- Eg: Big number - (Big number+1) =

Small number – (Small number – 10^{-20}) also =





Spicy Warnings

PEPPER [1]

- **PEPPER** (**P**ortable **E**ngine for the **P**roduction of **P**arton-level **E**vent **R**ecords) is a parton-level event generator designed for efficient execution on both CPUs and GPUs.
- PEPPER generates events in parallel batches, which makes it a useful platform for investigating faster event generation on modern hardware.
- However, PEPPER was originally designed for **FP64**, with many numerical and statistical variables explicitly stored as [double].
- It cannot fully exploit the higher FP32 performance available on modern GPUs.
- This project makes PEPPER's numerical precision configurable, by introducing **FP32 (single) and other diagnostic FP64/32 modes (float_st & double_st)**.

Methodology

CADNA (***C**ontrol of **A**ccuracy and **D**ebugging for **N**umerical **A**pplications*)[4]

- Is a numerical-validation library to detect numerical instabilities such as: @.0 which indicates that a result contains **no reliable significant digits**.

CADNA Workflow

Perform three stochastic evaluations

CADNA stores 3 versions of each value and varies the roundings

Result 1: 2.345671
Result 2: 2.345669
Result 3: 2.345670

Compare the 3 results

Since these values agree closely, CADNA concludes that most leading digits are reliable.

Estimate significant digits

The smaller the variation between the results, the more significant digits CADNA reports.

5 reliable digits

Detect unstable calculations

Consider subtracting two nearly equal numbers. Different rounding may produce:

Result 1: 0.000000012
Result 2: -0.00000008
Result 3: 0.00000003

@.0

Precision Debug

Workflow demonstration

- **Failure:** The lepton-antilepton pair production test resulted in @.0 (no reliable significant digits), this nightmare propagates.
- **Localisation:** Days of searching for the root of problem by printing more detailed logs progressively.
- **Numerical issue:** Inside Z-fermion coupling, there is a representative complex number $z = a + ib$, PEPPER evaluates:

$$\sqrt{z} = \sqrt{\frac{r+a}{2}} + i\ \mathrm{sgn}(b)\sqrt{\frac{r-a}{2}}, \quad r = \sqrt{a^2+b^2}$$

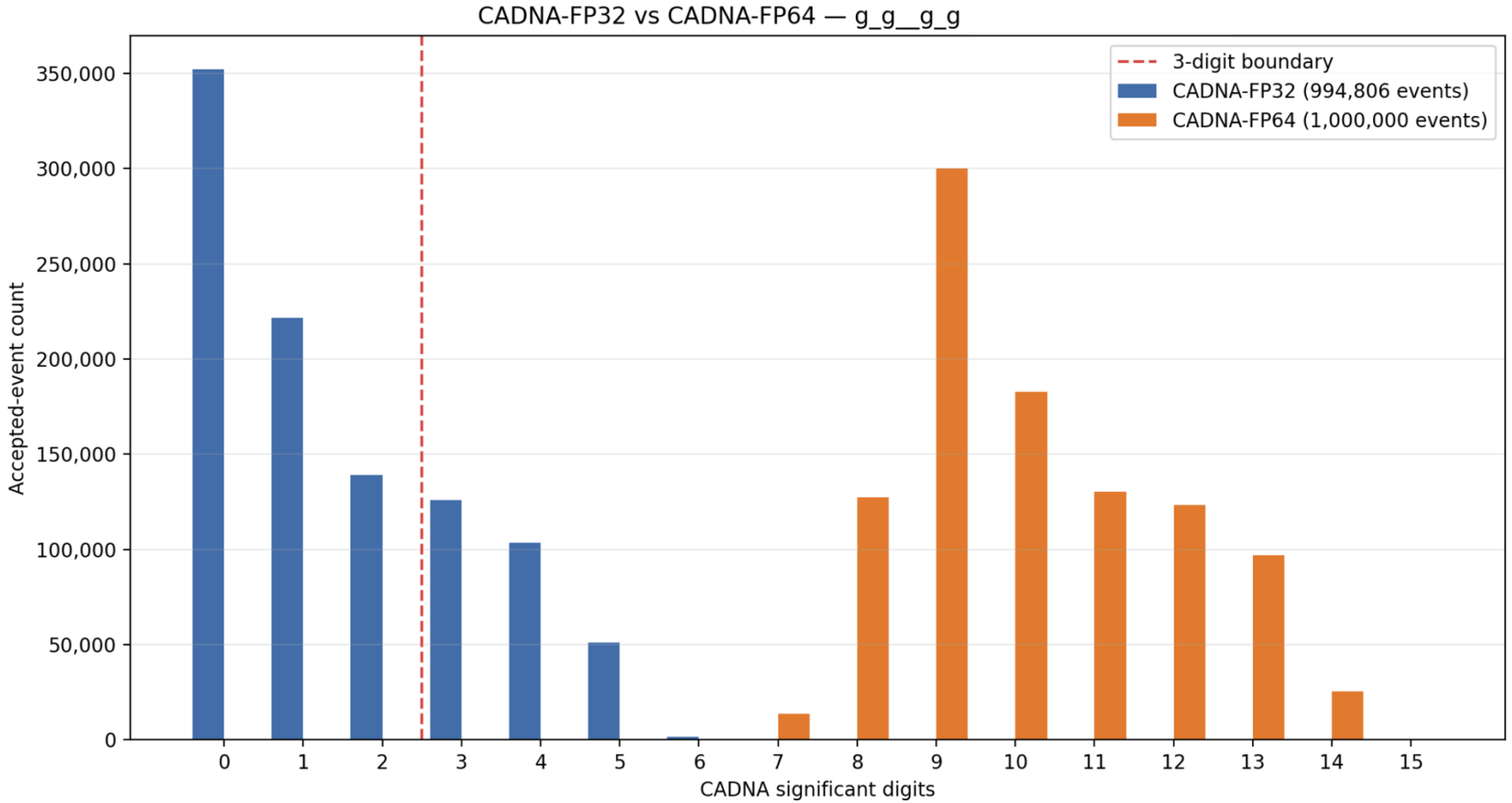
- For case $r \approx a$; $(r-a)$ subtracts two nearly equal numbers and can lose all significant digits.
- **Targeted fix:** Detect negligible cancellation relative to the input scale and replace it with an exact zero:

$$r - a < \varepsilon \times \frac{r+a}{2} \Longrightarrow r - a = 0$$

- with $\varepsilon = 10^{-5}$ for CADNA-FP32 and (10^{-12}) for CADNA-FP64.
- **Outcome:** The expected zero imaginary component was represented as an exact zero instead of @.0, tests passed.

Results and Analysis

- Successful diagnostic run (blue) with benchmark (orange) for 4g process
- Identified problematic regions (ME2) and prints digits for the **matrix element squares**
- The massless gluons might be the reason why there's a massive failure in FP32



Conclusion and Future Work

Conclusion

- Successfully upgraded PEPPER so that it supports multiple floating-point types with diagnostic modes.
- Tested and run upgraded PEPPER to show numerical instabilities when switching to lower precision types.
- Identified key regions of numerical failures and some functions that are failing.

Future work

- Import improved arithmetic or other libraries to solve the numerical instabilities
- Final implementation of FP32 in PEPPER with GPU usage

Last slide
Thank you for your attention

References

[1] **E. Bothmann et al**, “A portable parton-level event generator for the high-luminosity LHC,” *SciPost Physics* **17**, 081 (2024). DOI: 10.21468/SciPostPhys.17.3.081.

[2] **ATLAS Collaboration**, *ATLAS Software and Computing HL-LHC Roadmap*, CERN-LHCC-2022-005, LHCC-G-182 (2022), Figure 3.

[3] J. Rommes, *Sparse Days 2025*; performance data from NVIDIA GPU specifications

[4] **F. Jézéquel and J.-M. Chesneaux**, “CADNA: a library for estimating round-off error propagation,” *Computer Physics Communications* **178**(12), 933–955 (2008). DOI: 10.1016/j.cpc.2008.02.003

Back Ups

Why HL-LHC needs faster event generation?

High-Luminosity LHC

- The HL-LHC upgrade will increase the LHC's integrated luminosity by approximately a factor of ten.
- ATLAS and CMS expect around 140–200 proton–proton interactions per bunch crossing and about 3,000 fb^{-1} of data.
- The larger dataset enables studies of rare processes and more precise measurements—but requires much larger and more accurate simulated samples.

Sources: CERN HiLumi LHC; ATLAS Software and Computing HL-LHC Roadmap, CERN-LHCC-2022-005; HSF Generator WG, CSBS 5, 12 (2021); Bothmann et al., SciPost Phys. 17, 081 (2024).

Event Gen I

A simulated event is one randomly sampled possible collision. Millions of events are generated to reproduce physical probabilities, cross sections, and observable distributions.

Physics model

→ Hard scattering — PEPPER

→ Parton shower

→ Hadronization and decays

→ Detector simulation

→ Reconstructed ATLAS objects

1. Define the collision

Select the process, the collision energy, particle masses, couplings, PDFs, scales and analysis cuts.

Because protons are composite, the colliding quarks or gluons carry fractions (x_1) and (x_2) of the proton momenta.

2. Generate the hard scattering — PEPPER's scope

Random numbers are mapped onto a phase-space point: a set of outgoing four-momenta satisfying energy–momentum conservation and particle mass constraints.

3. Calculate event weights and the cross section

The matrix element is combined with PDFs, flux factors, couplings, phase-space Jacobians and selection cuts.

Each sampled point receives an event weight.

Adaptive Monte Carlo integration concentrates samples in important regions of phase space.

The average of the weights estimates the total cross section; the distribution of weights predicts differential observables.

Event Gen II

4. Produce events for later simulation
Weighted events: every event retains its numerical weight.
Unweighted events: an accept–reject procedure converts events into approximately equal-weight samples.
Unweighted events are more convenient for showering and detector simulation, but inefficient when the event weights vary greatly.
5. Build the complete particle collision
Parton shower: models initial- and final-state QCD radiation.
Underlying event: adds additional parton interactions and beam remnants.
Hadronization: converts coloured quarks and gluons into colour-neutral hadrons.
Particle decays: unstable particles decay into observable final states.
6. Simulate and reconstruct the detector response
Stable particles are transported through the ATLAS detector material.
Their interactions produce detector hits, which are converted into simulated electronic signals.
Pile-up collisions can be overlaid before reconstruction.
Reconstruction turns these signals into tracks, jets, electrons, muons and missing transverse momentum.
References for the slide notes
E. Bothmann et al., “A portable parton-level event generator for the high-luminosity LHC,” *SciPost Physics* **17**, 081 (2024), DOI: 10.21468/SciPostPhys.17.3.081.
HSF Physics Event Generator Working Group, “Challenges in Monte Carlo event generator software for High-Luminosity LHC,” *Computing and Software for Big Science* **5**, 12 (2021), arXiv:2004.13687.
C. Bierlich et al., “A comprehensive guide to the physics and usage of PYTHIA 8.3,” *SciPost Physics Codebases* **8** (2022), arXiv:2203.11601.
Geant4 Collaboration, “Geant4—a simulation toolkit,” *Nuclear Instruments and Methods A* **506**, 250–303 (2003).

PEPPER brings parton-level event generation to modern hardware

Why is it special?

Designed for parallelism

Traditional generators commonly process one event at a time.

PEPPER processes entire batches simultaneously, allowing many CPU threads or GPU cores to work on different events.

Performance-portable

PEPPER uses the Kokkos framework so that one maintainable C++ codebase can target CPUs and NVIDIA, AMD or Intel GPUs.

MPI support enables execution across multiple processors, GPUs and computing nodes.

Hardware-aware data layout

Event properties are stored in a structure-of-arrays layout, placing similar data together in memory.

This improves memory access and allows one event to be processed per GPU thread.

A complete parton-level pipeline

PEPPER parallelizes phase-space generation, cuts, PDF evaluation, helicity sampling, matrix-element calculation, unweighting and event filtering.

Only accepted events are transferred for storage, reducing unnecessary movement of data between GPU and CPU memory.

Why this matters for HL-LHC

HL-LHC analyses require larger and more complex simulated samples. PEPPER aims to exploit modern heterogeneous computing systems without maintaining a separate event generator for every hardware architecture.

References for the slide notes

E. Bothmann et al., “A portable parton-level event generator for the high-luminosity LHC,” *SciPost Physics* **17**, 081 (2024), DOI: 10.21468/SciPostPhys.17.3.081, arXiv:2311.06198.

PEPPER Collaboration, “Portable Engine for the Production of Parton-level Event Records,” official source repository: https://gitlab.com/spice-mc/pepper

Contents

Floating point (FP), Pepper, Cadna and so on

1. Introduction: Background and Motivations
2. Project aims: Floating point and PEPPER
3. Methodology: CADNA, PROMISE and Workflow demo
4. Results and Analysis
5. Conclusion and future work