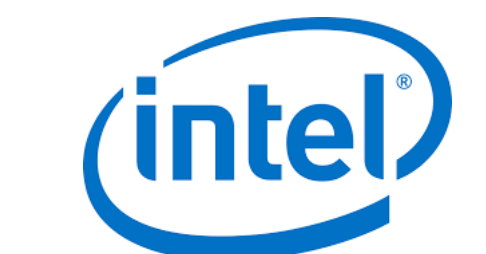


supported by



cooperations



A. Salzburger (CERN) for the ACTS project

# ACTS Developers Workshop

7–10 Nov 2023  
Europe/Zurich timezone



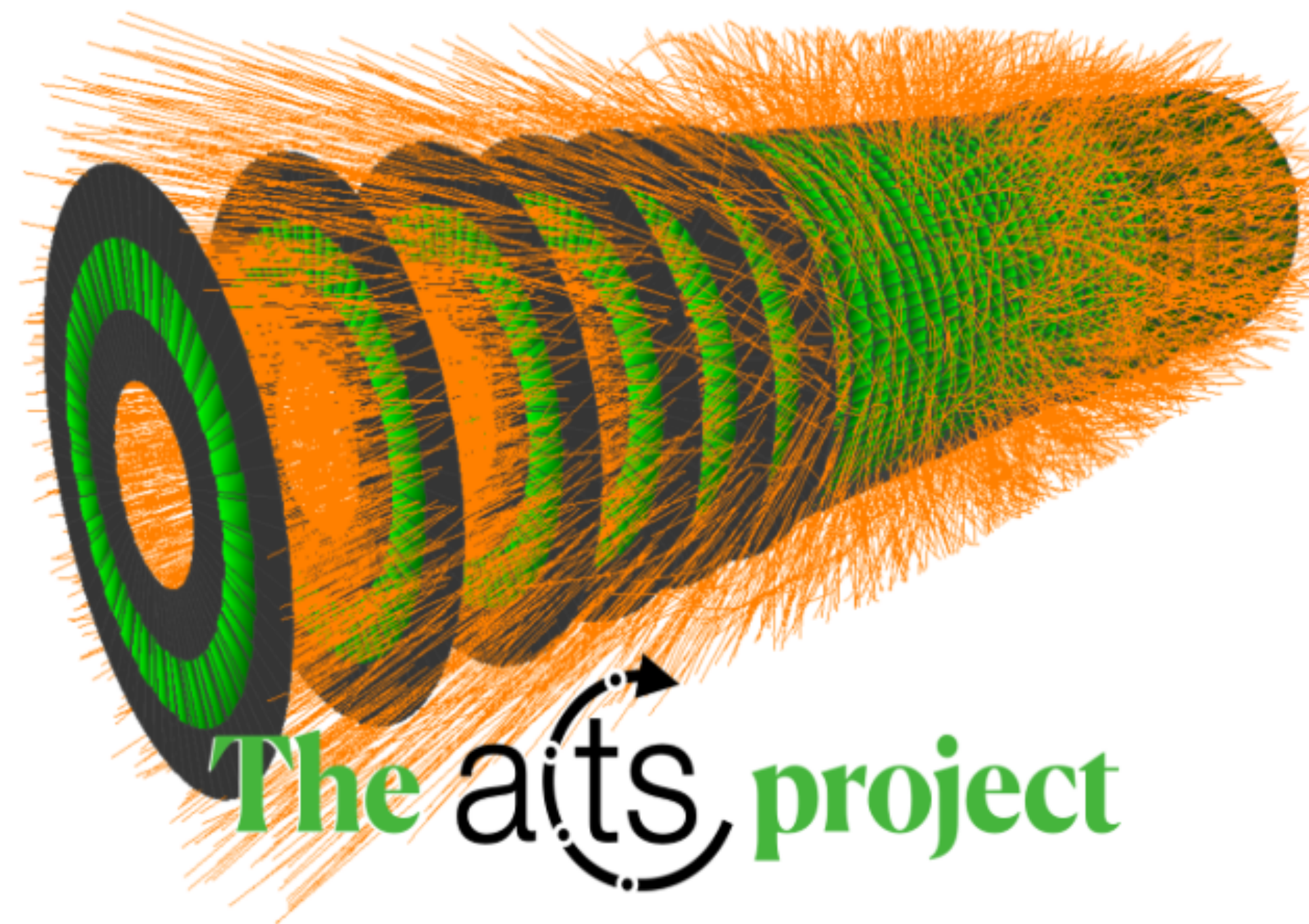
Overview

Timetable

Registration

Participant List

Travel to IJCLab Orsay



supported by



cooperations



The fourth **ACTS** developers workshop will take place at IJCLab (Orsay, France) from the 7/11 through 10/11, following [Berkeley \(2019\)](#), [DESY/Virtual \(2020\)](#) and [CERN \(2022\)](#)

<https://indico.cern.ch/event/1295479/>

# Development and R&D

## Core

[acts-developers@cern.ch](mailto:acts-developers@cern.ch)

CPU multi-threaded library of tracking reconstruction components

## R&D1

[acts-parallelization@cern.ch](mailto:acts-parallelization@cern.ch)

CPU/GPU “single source” demonstrator re-implementing the main Core chain

## R&D2

[acts-machinelearning@cern.ch](mailto:acts-machinelearning@cern.ch)

Machine learning and ML assisted modules for track reconstruction

## Core

[acts-developers@cern.ch](mailto:acts-developers@cern.ch)

CPU multi-threaded library of  
tracking reconstruction components



Project has **5** years of age  
More than **1500** merged PRs, more than **120** forks, **~80** stars  
**~50** different contributors



Mozilla Public Licence 2.0

# Core: the flagship project

## Main target & language

- x86/ARM64 **multithreaded** architectures, GPU development moved to R&D1 line
- C++17 standard (we started first steps to move to C++20)
- minimal core dependencies:  
CMake, Eigen, BOOST + **optional Plugins**

## Component library structure

- track & vertex reconstruction components that allow for assembling of a track reconstruction applications for different experimental setups
- designed to be integrated into an experiment software that provides scheduling, configuration, IO, conditions data, etc.

# Core concepts: multi threading and contextuality

## Built-in parallelisation support

```
namespace Acts {
  class Module {
    /// @param gctx the geometry context (e.g. alignment)
    /// @param input the input data
    OutputData geometricOperation(const GeometryContext& gctx, const InputData& input) const;
  };
}
```

Allows parallel execution of this operation (without explicit technology binding, such as **tbb**) within and across events, nested **State** structs are used for necessary caching operations

```
namespace Acts {
  class Module {
    /// Nested State struct
    struct State { ActsScalar cache = 0.; };
    /// @param state is a cache for this operation
    /// @param input the input data
    OutputData operationWithCache(State& state, const InputData& input) const;
  };
}
```

# Core concepts: multi threading and contextuality

Built-in parallelisation support and contextuality

```
namespace Acts {
  /// @param gctx the geometry context (e.g. alignment)
  /// @param input the input data
  OutputData geometricOperation(const GeometryContext& gctx, const InputData& input) const;
};
}
```

```
using GeometryContext = std::any;
```

ACTS allows you to pack your own contextual data into the context objects (geometry, magnetic, field) and will carry it through the code base (untouched)

```
auto Experiment::applyCorrection(const GeometryContext& gctx, const InputData& input) const {
  const Experiment::Payload& payload = std::any_cast<const Experiment::Payload&>(gctx);
}
```

# Core concepts: data driven, configuration & options

Design convention for data driven design, configuration and option

```
namespace Acts {  
  /// doxygen documentation  
  class Module {  
    /// @struct Config for this module,  
    struct Config {  
      ActsScalar globalParameter; ///  
    };  
  
    /// @struct Options for this module, changeable on call  
    struct Options {  
      ActsScalar callParameter; ///  
    };  
  
    /// @param cfg the configuration struct for this module  
    Module(const Config& cfg) : m_config(cfg){};  
  
    /// @param input the input data  
    OutputData operation(const InputData& input, const Options& opt) const;  
  
  };  
}
```



# Core concepts: configuration binding

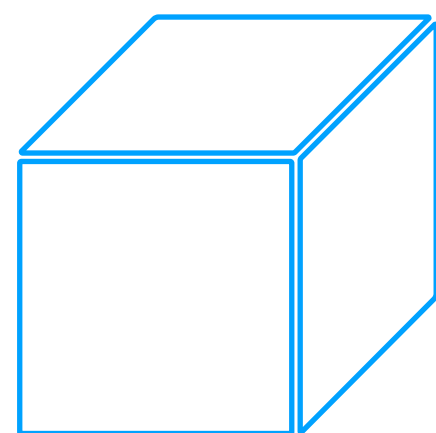
## Simple Config structs on ACTS side

```
namespace Acts {  
  /// doxygen documentation  
  class WorkHorse {  
    /// @struct Config for To  
    struct Config {  
      ActsScalar coatColor; ///  
      ActsScalar maxPath;    ///  
    };  
  };  
}
```

## Connection to experiment framework, e.g. Gaudi/Athena

```
/// feed from Framework into ACTS configuration  
declareProperty("CoatColor", m_cfg.coatColor);  
declareProperty("MaxPath", m_cfg.maxPath);
```

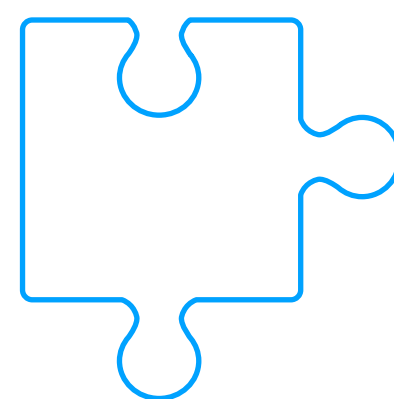
# Selected Core modules



## Geometry/Detector\*

(Surface based geometry)

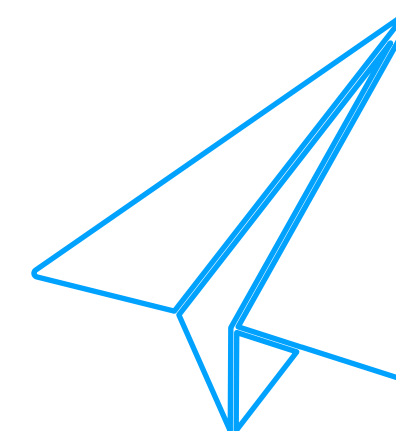
Plugins to DD4hep, TGeo, etc.



## Event Data Model

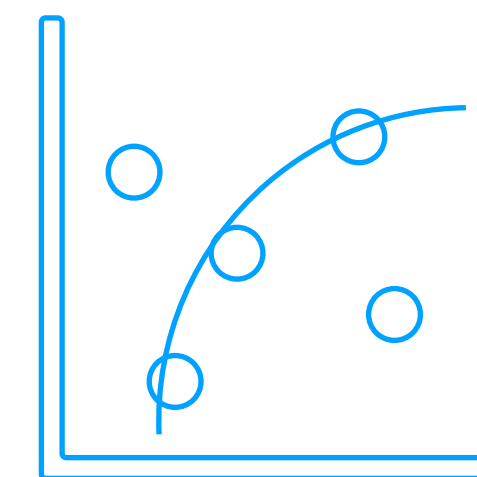
target track reconstruction

backend separation  
with different I/O models



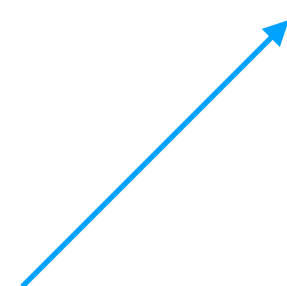
## Propagation

parameter + covariance  
transport through  
magnetic field



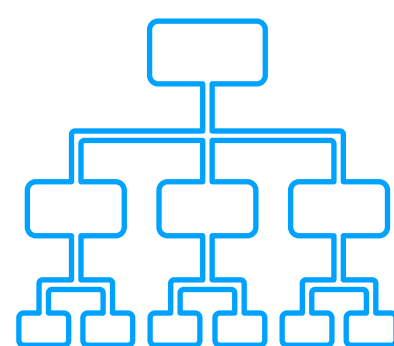
## Seeding

Seed finding with  
Triplet seeder,  
OrthogonalSeedFinder



## Track Fitting

parameter estimation  
with Kalman Filter,  
GSF, GX2F\*\*



## Combinatorial track finding

Combinatorial Kalman Filter  
for track finding



## Vertex finding + fitting

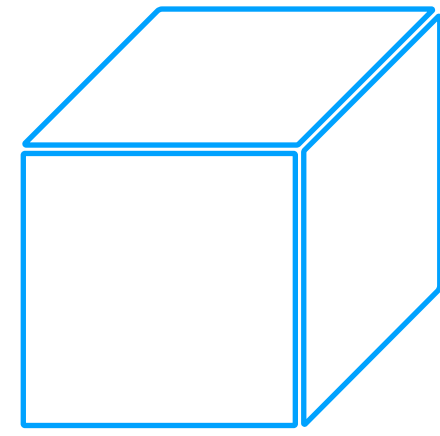
Iterative, multi variant  
primary vertex finders  
and fitters



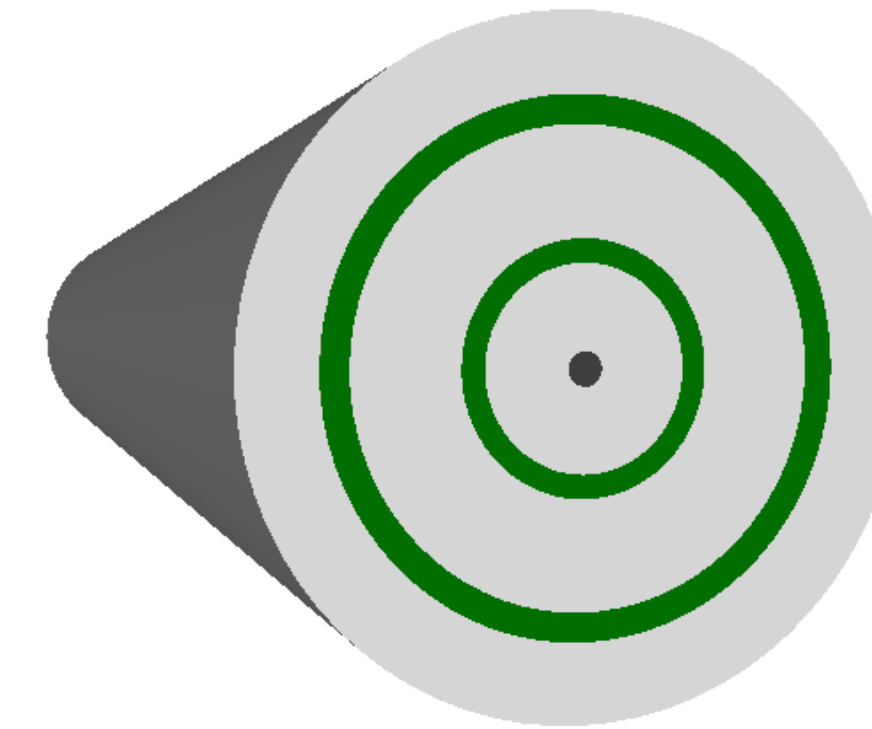
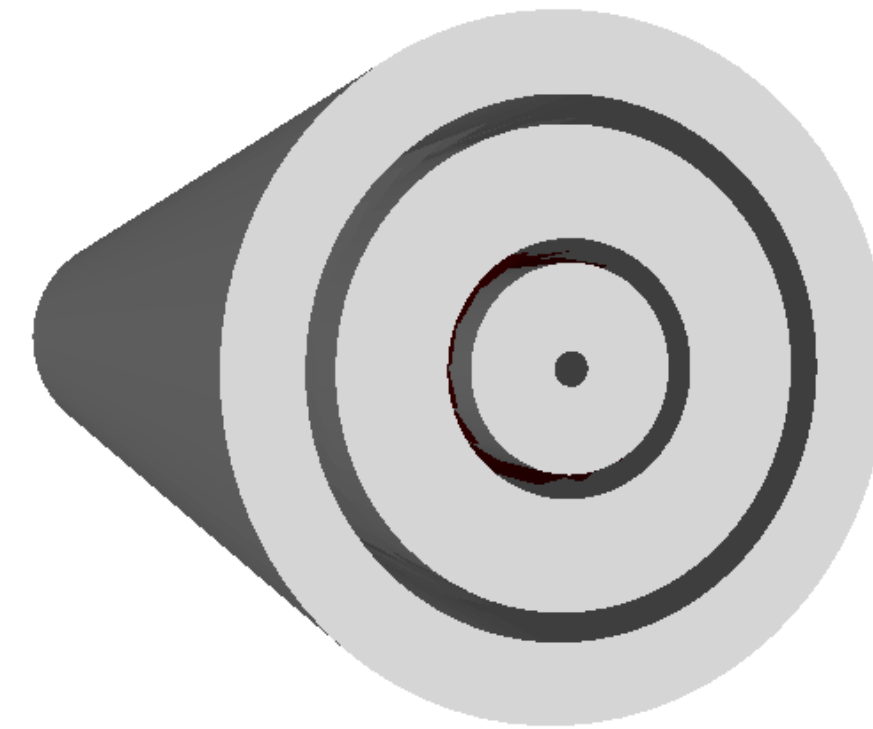
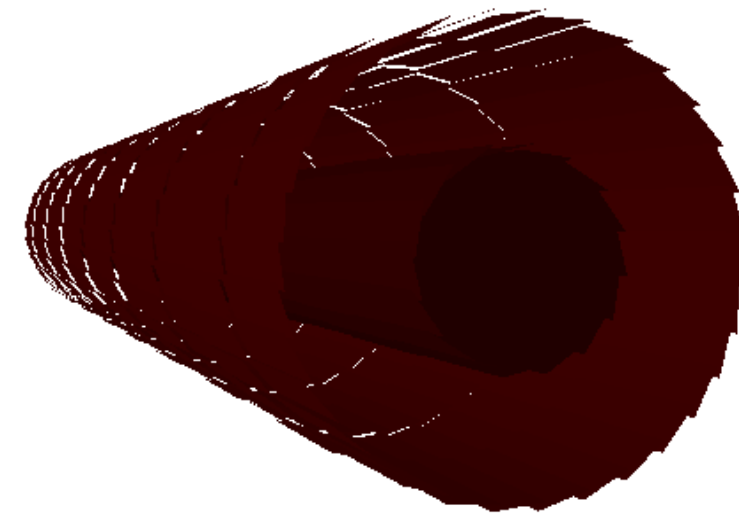
## Detector alignment

KF based alignment  
functionality

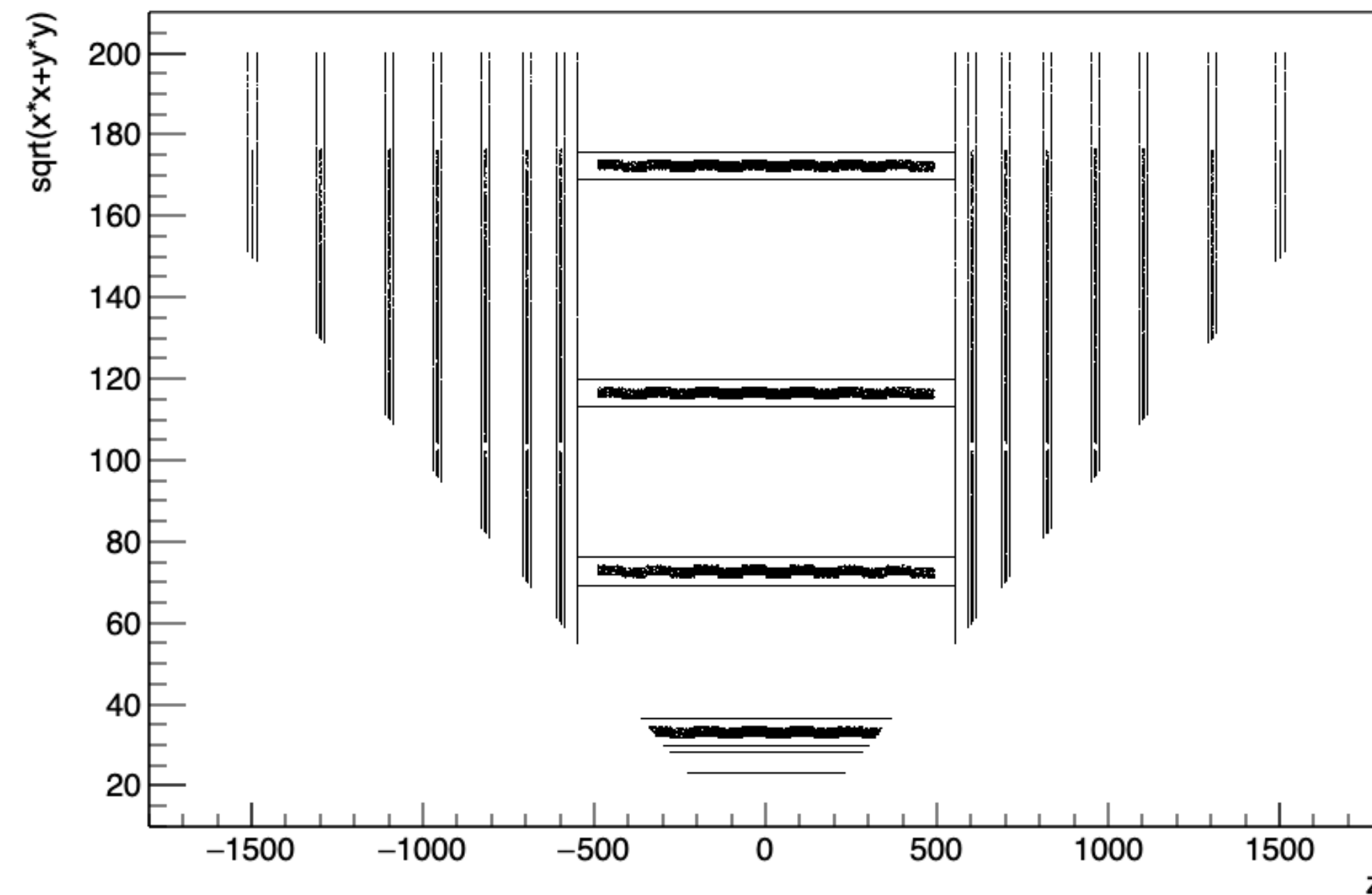
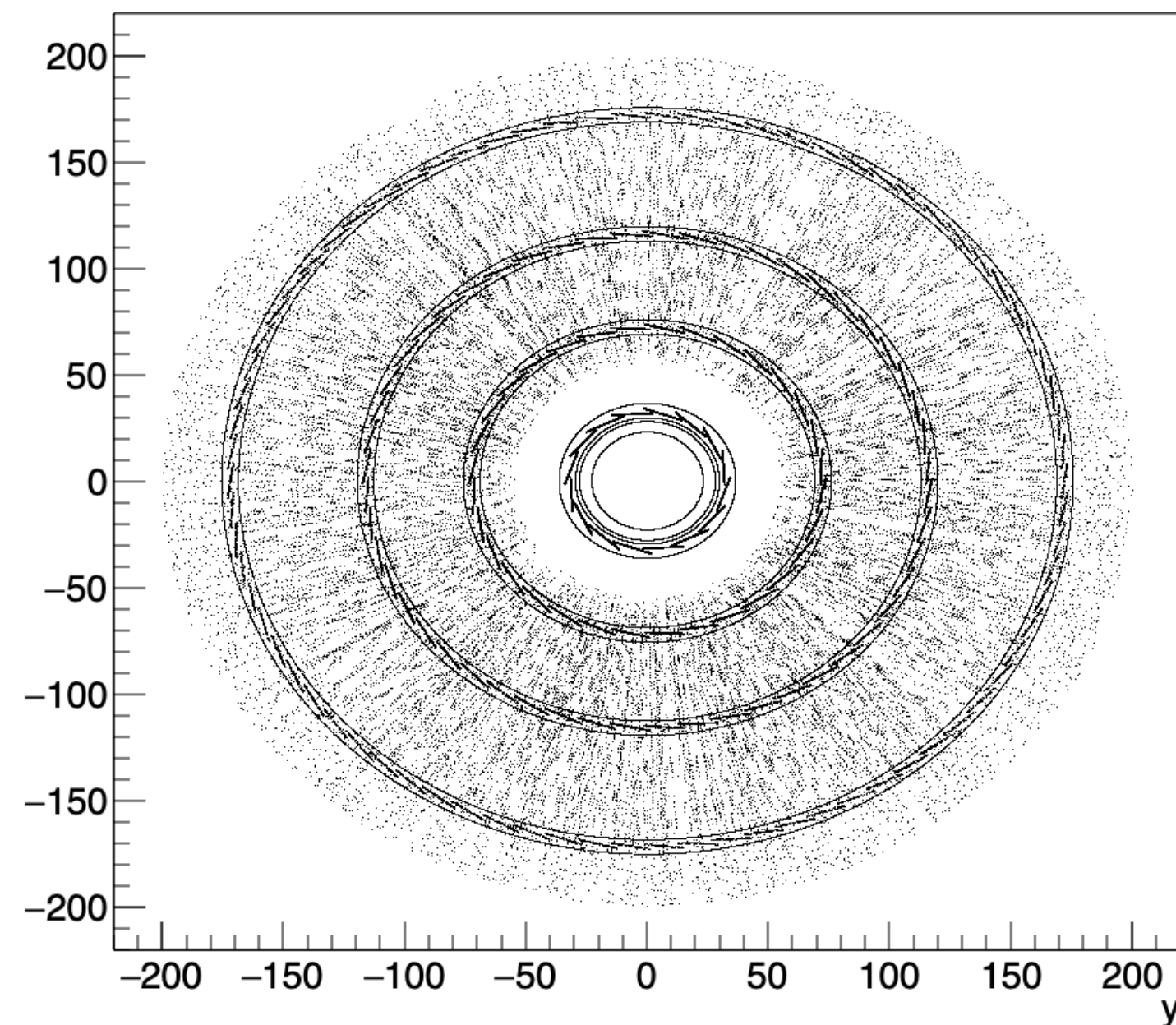
\*new geometry model to be deployed in 2023, \*\*GX2F prototype to come 2024

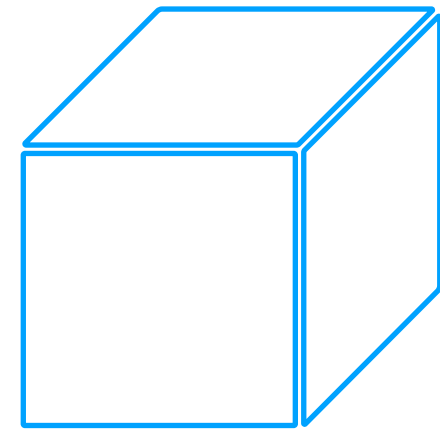


# Geometry + Navigation



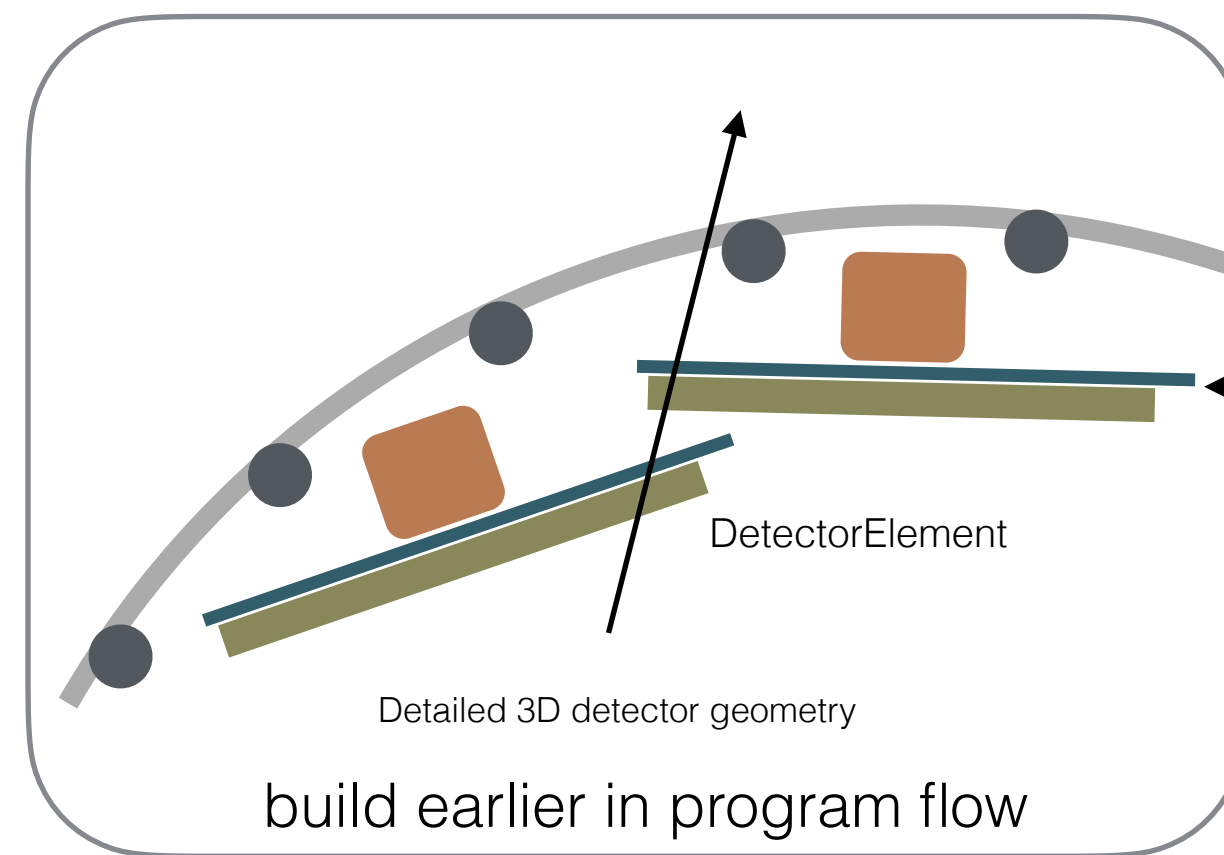
Surface based geometry model, optimised for reconstruction,  
can be **connected to standard HEP geometry modellers**



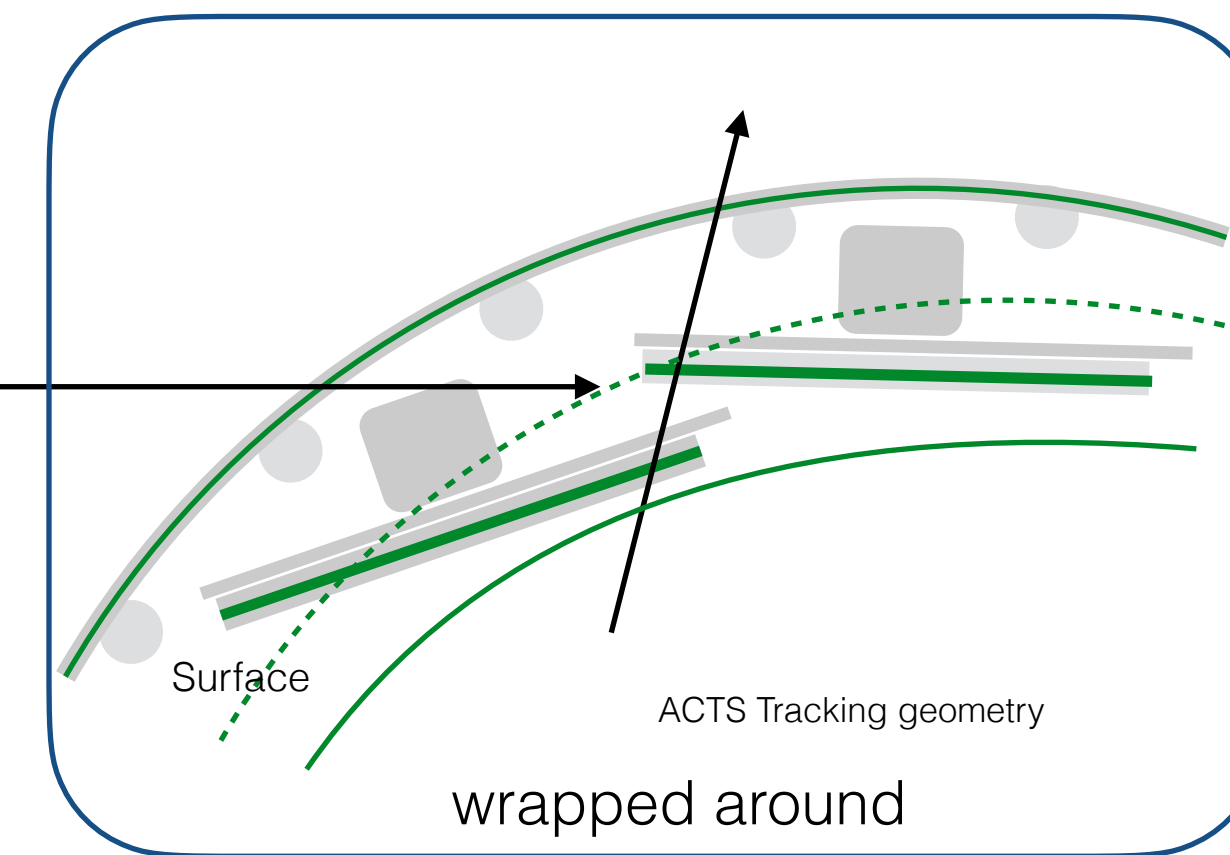


# Geometry + Interfacing

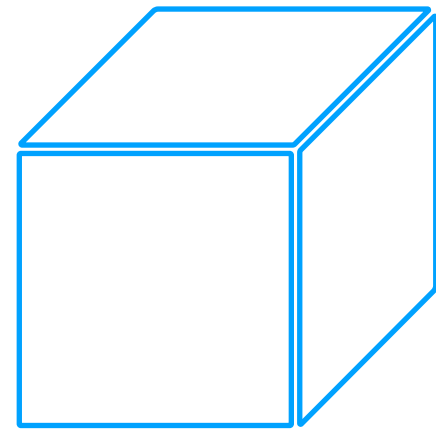
**connected to standard HEP geometry modellers**



**Detailed geometry model,**  
e.g. DD4hep, TGeo, GeoModel, etc.

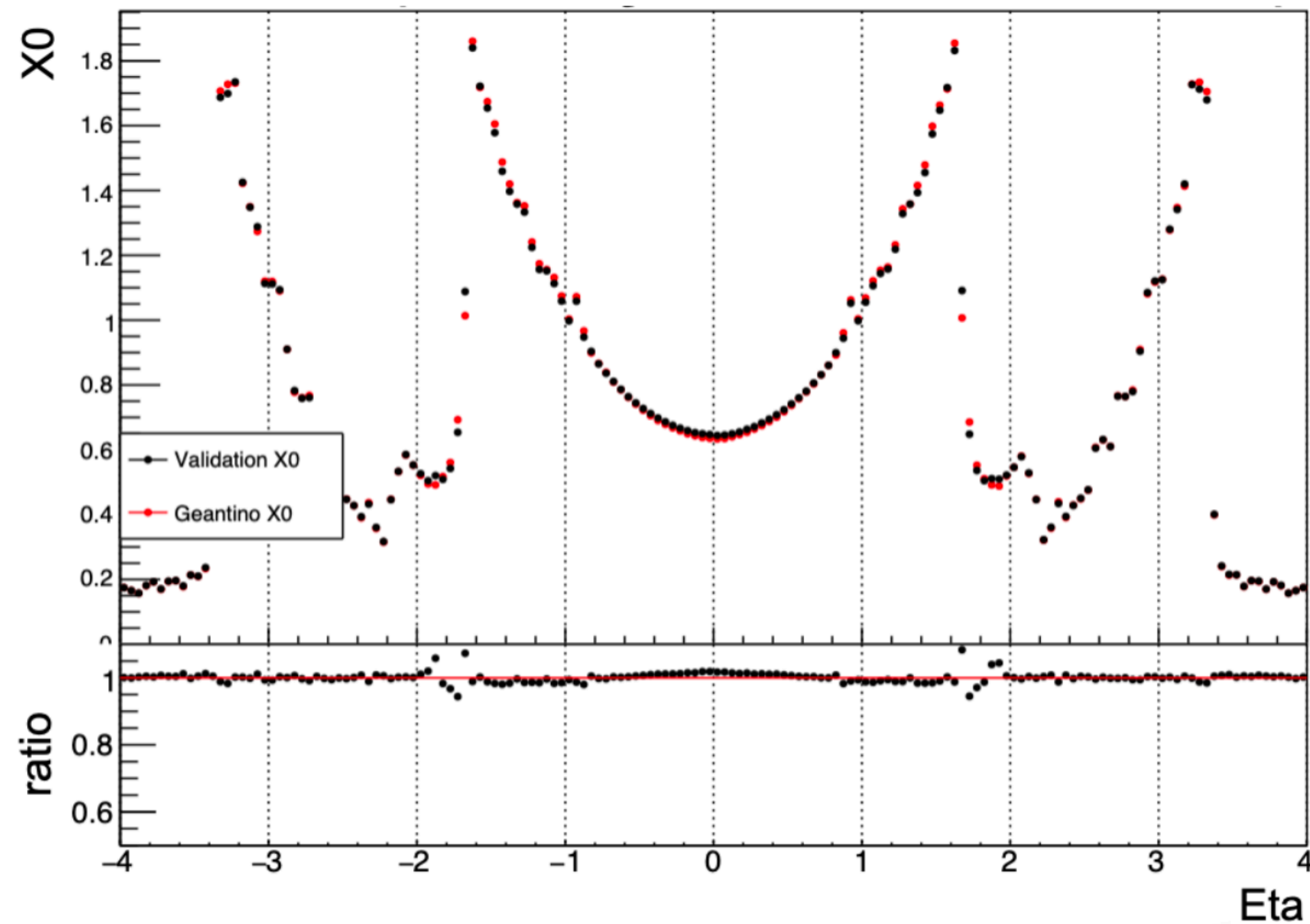


**ACTS geometry model**  
with builtin navigation

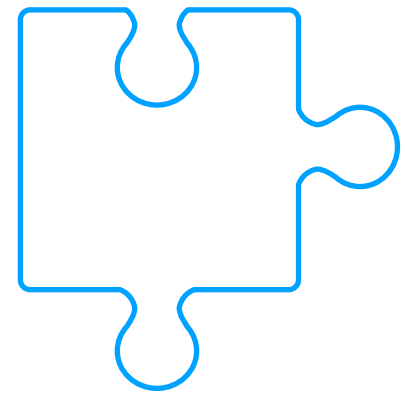


# Geometry + Material

**ACTS** ships with a **material mapping module** that allows to describe the detector material with close-to Geant4 precision



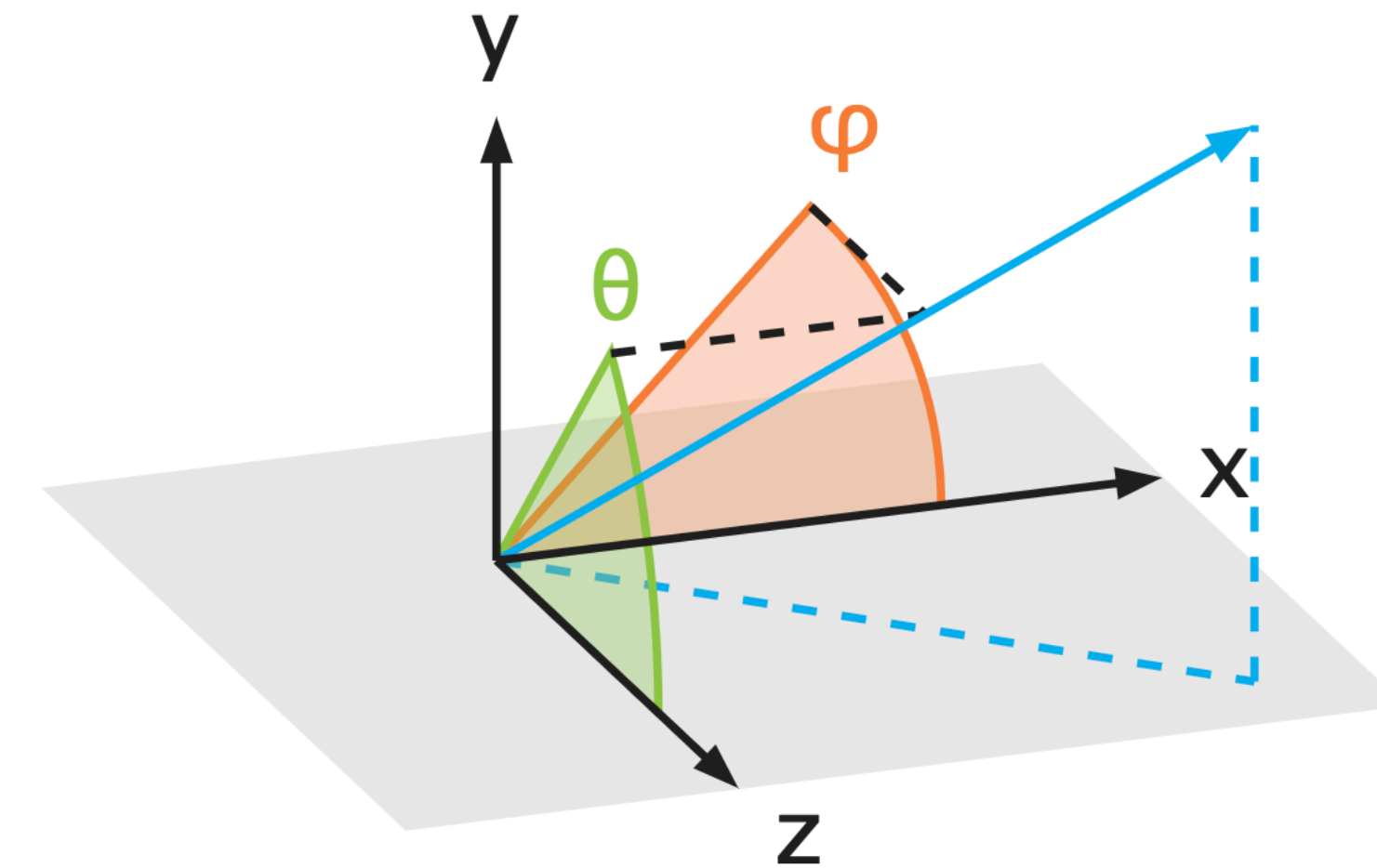
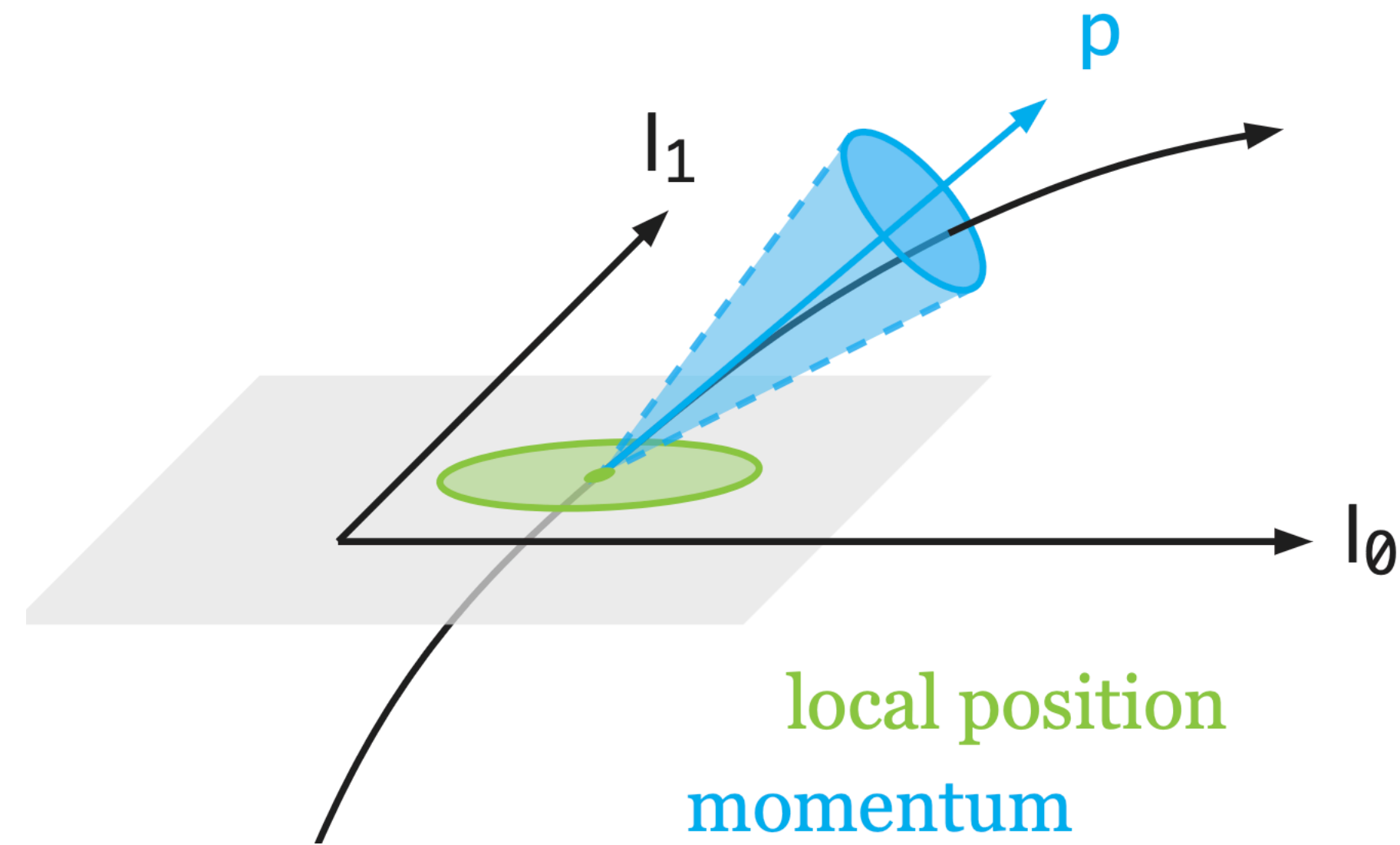
\* this can also be enhanced with a ML based material optimiser



# Event Data Model

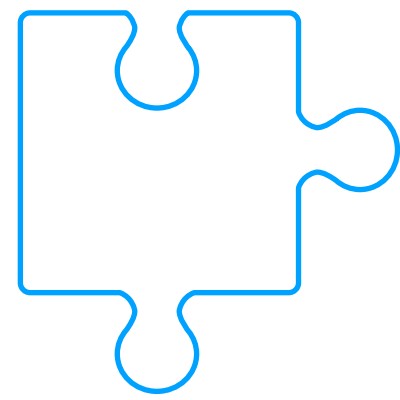
(Bound) track parameterisation is defined:

**local coordinates of the surface + global momentum**



$$\vec{x} = (l_0, l_1, \phi, \theta, q/p, t)^T$$

$$C = \begin{bmatrix} \sigma^2(l_0) & \text{cov}(l_0, l_1) & \text{cov}(l_0, \phi) & \text{cov}(l_0, \theta) & \text{cov}(l_0, q/p) \\ \cdot & \sigma^2(l_1) & \text{cov}(l_1, \phi) & \text{cov}(l_1, \theta) & \text{cov}(l_1, q/p) \\ \cdot & \cdot & \sigma^2(\phi) & \text{cov}(\phi, \theta) & \text{cov}(\phi, q/p) \\ \cdot & \cdot & \cdot & \sigma^2(\theta) & \text{cov}(\theta, q/p) \\ \cdot & \cdot & \cdot & \cdot & \sigma^2(q/p) \end{bmatrix}$$



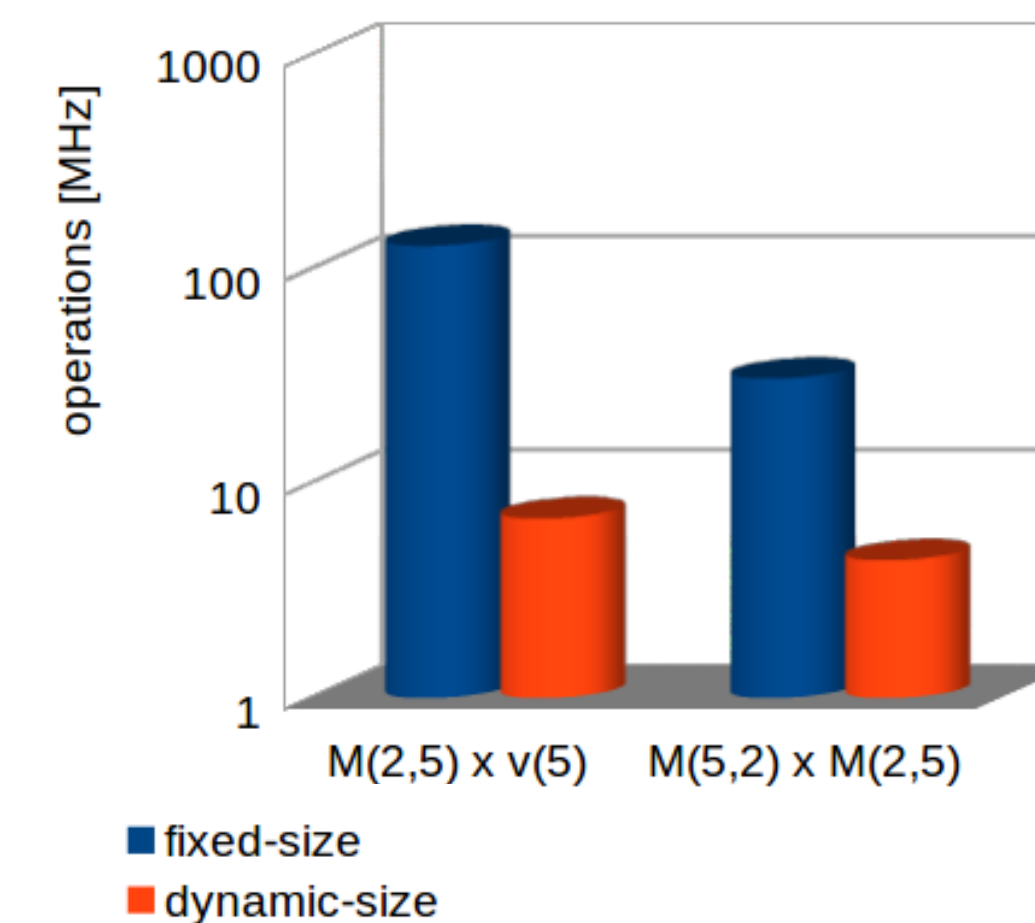
# Event Data Model

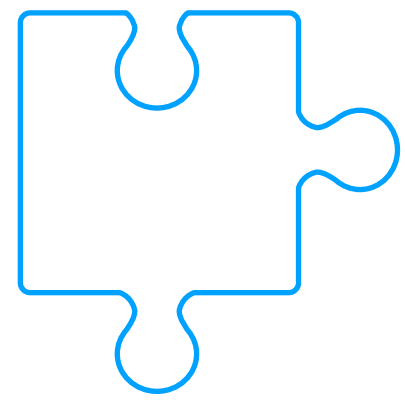
Parameter	$l_0$	$l_1$	phi	theta	q/p	t
Bound track parameters						
Pixel measurement						
Pixel measurement with time						
Strip measurement (along local x)						
Strip measurement (along local y)						
Drift time/circle measurement						
Track segment (straight line)						
...						

Measurements can be represented as subsets of the full bound parameter space.

**This is done at compile time to increase computing performance.**

Eigen-3.2.7, gcc-4.9.2, -O2, 4xi5-5200U @ 2.2GHz,

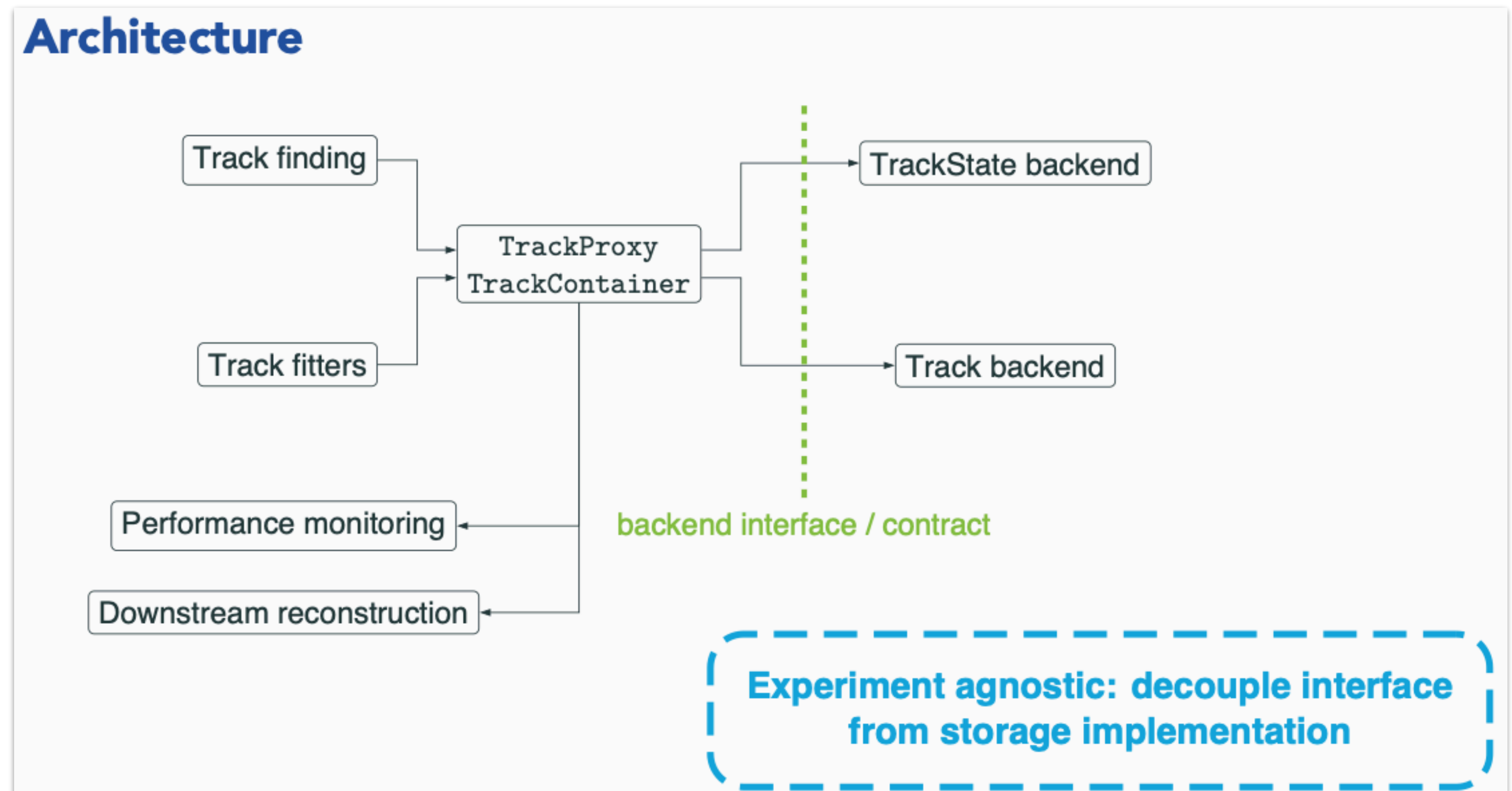




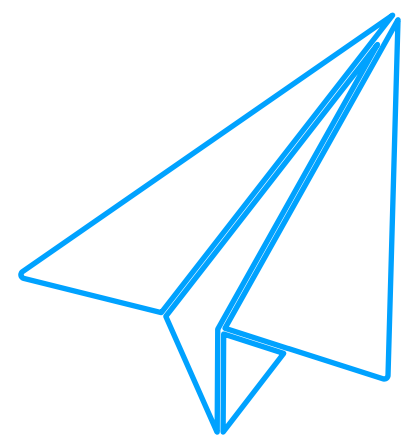
# Event Data Model + I/O backend

ACTS has an internal EDM optimised for track reconstruction.

- recent work to separate transient model from I/O backend
- demonstrator with PODIO established
- Non-optimised EDM4Hep version also available







# Propagation

Stepping loop  
(until max number of steps)

**Navigator::status:**

On a current surface?

Walk through actor list

Walk through aborter list

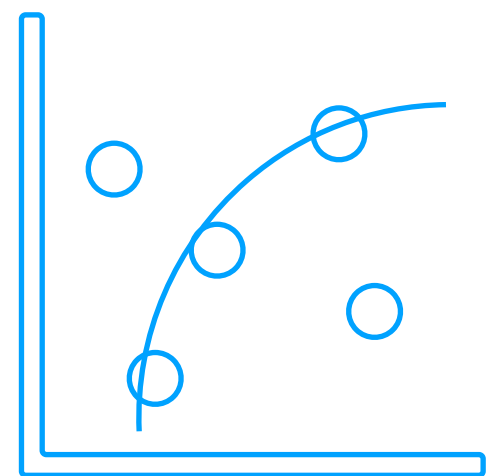
**Navigator::target**

Next candidate surface

```
// Pre-Stepping: abort condition check
if (!state.options.abortList(result, state, m_stepper)) {
    // Pre-Stepping: target setting
    m_navigator.target(state, m_stepper);
    // Stepping loop
    ACTS_VERBOSE("Starting stepping loop.");

    terminatedNormally = false; // priming error condition

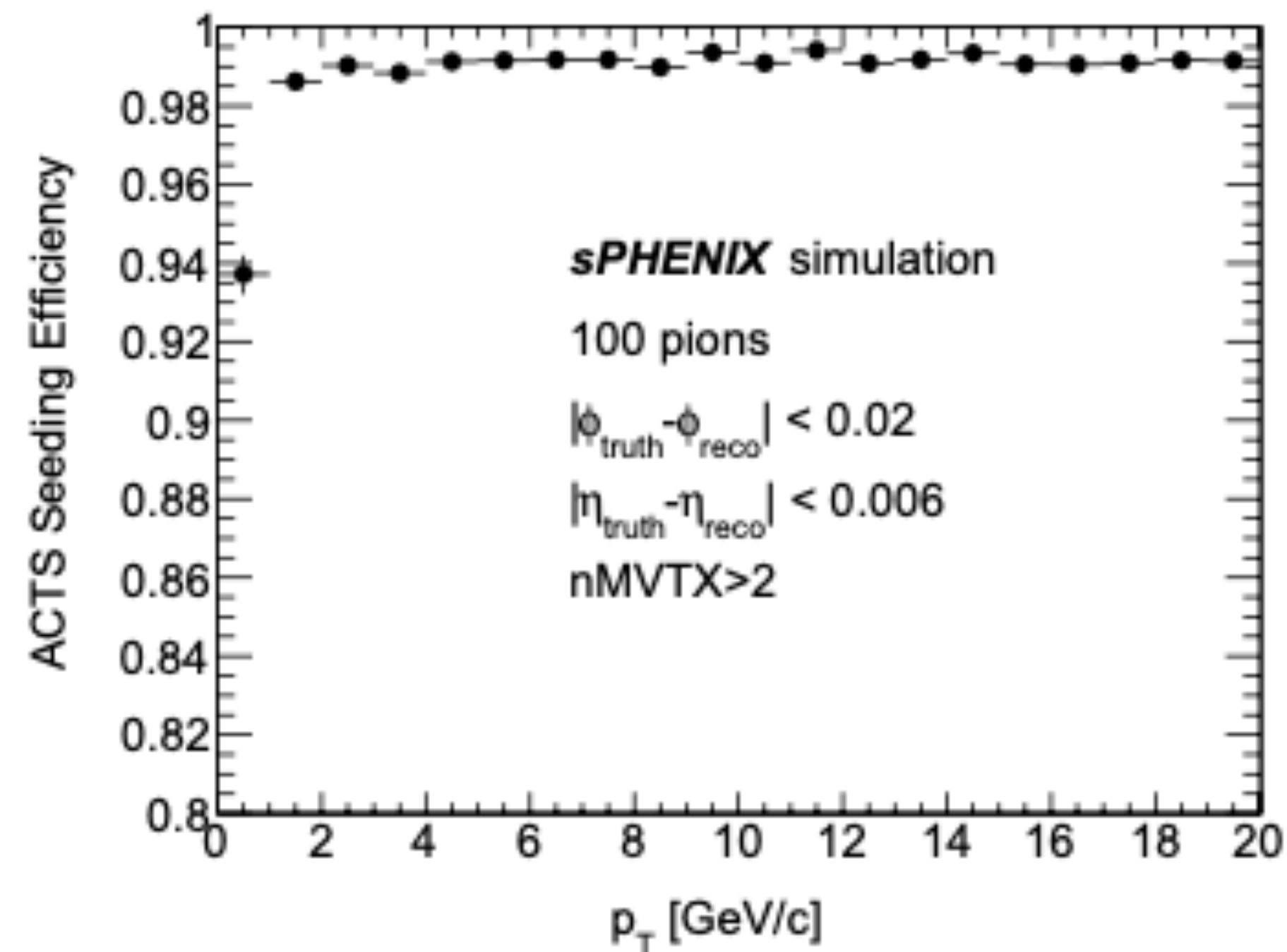
    // Propagation loop : stepping
    for (; result.steps < state.options.maxSteps; ++result.steps) {
        // Perform a propagation step - it takes the propagation state
        Result<double> res = m_stepper.step(state);
        if (res.ok()) {
            // Accumulate the path length
            double s = *res;
            result.pathLength += s;
            ACTS_VERBOSE("Step with size = " << s << " performed");
        } else {
            ACTS_ERROR("Step failed with " << res.error() << ": "
                << res.error().message());
            // pass error to caller
            return res.error();
        }
        // Post-stepping:
        // navigator status call - action list - aborter list - target call
        m_navigator.status(state, m_stepper);
        state.options.actionList(state, m_stepper, result);
        if (state.options.abortList(result, state, m_stepper)) {
            terminatedNormally = true;
            break;
        }
        m_navigator.target(state, m_stepper);
    }
} else {
    ACTS_VERBOSE("Propagation terminated without going into stepping loop.");
}
```



## Seeding

Triplet finding for initial track seeding,  
two alternative implementations:

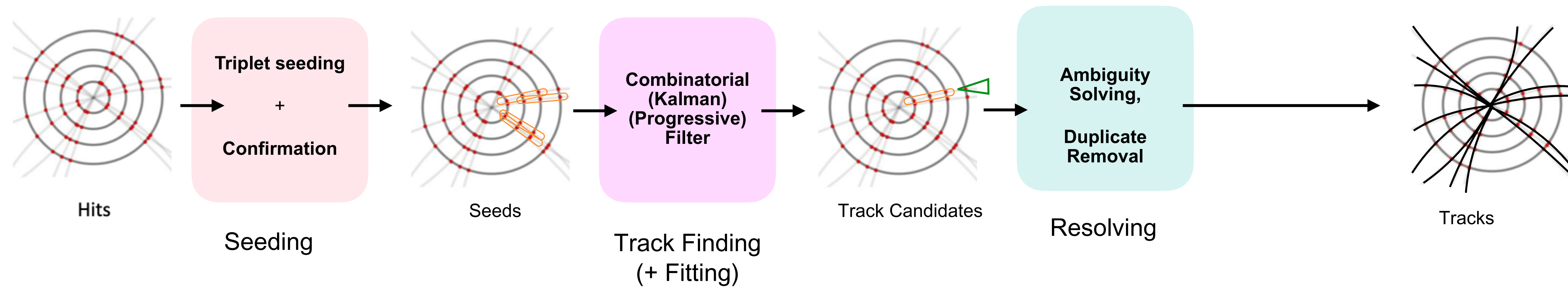
- ATLAS inspired mid-point seeding
- Orthogonal Seed finding



**Output of seeding is input to the Combinatorial Track finding module.**

# Full chain prototype implementation

## CKF chain



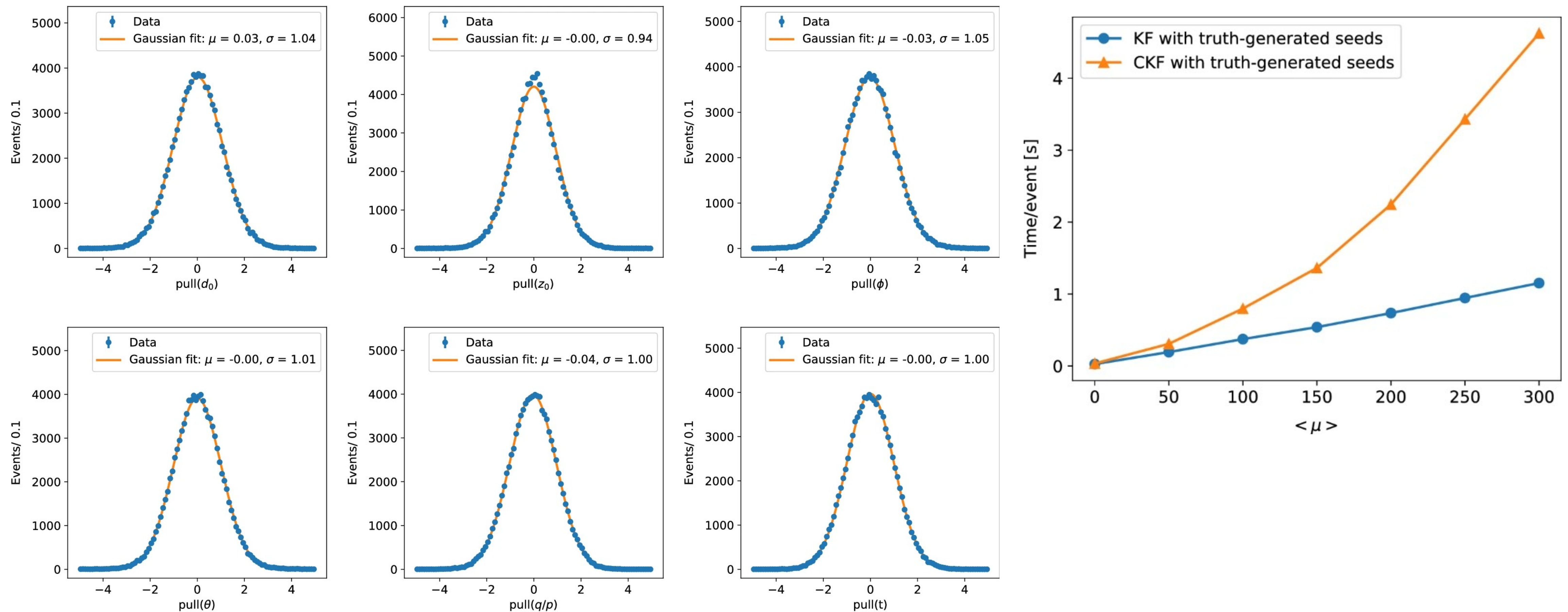
**This resembles the ATLAS track reconstruction strategy**

However, this is only one possible implementation put together by ACTS tools  
(That's the spirit of a component library)

# Full chain - with combinatorial track finding

A first full chain documented using the Open Data Detector / TrackML detector in:

<https://link.springer.com/article/10.1007/s41781-021-00078-8>



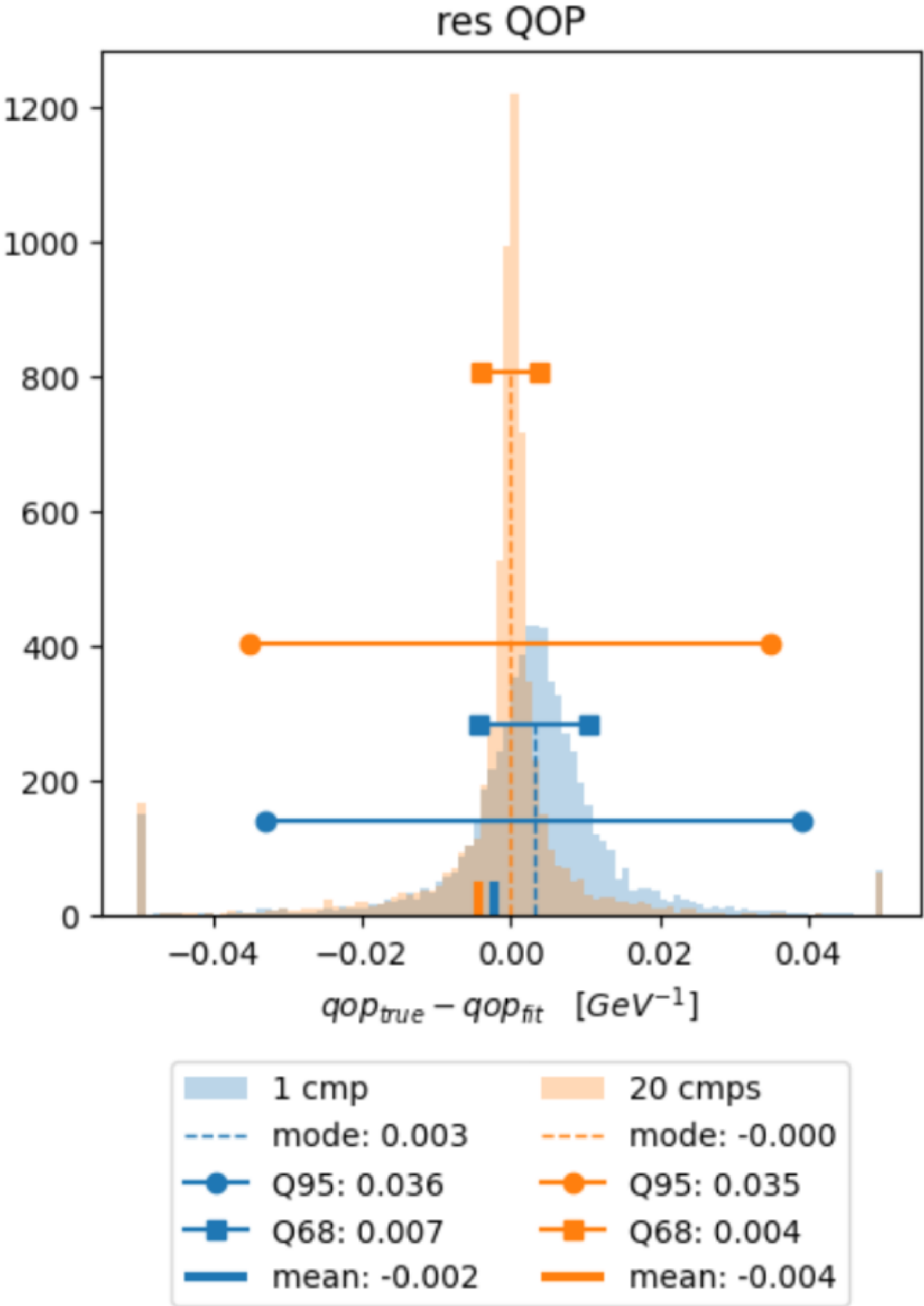
# Recently added functionality

First Gaussian Sum Filter (GSF) for electron reconstruction

The GSF is a multi-variant extension of the Kalman Filter to model non-Gaussian noise (i.e. Bremsstrahlung)

Global Chi2 Fitter currently in prototyping stage

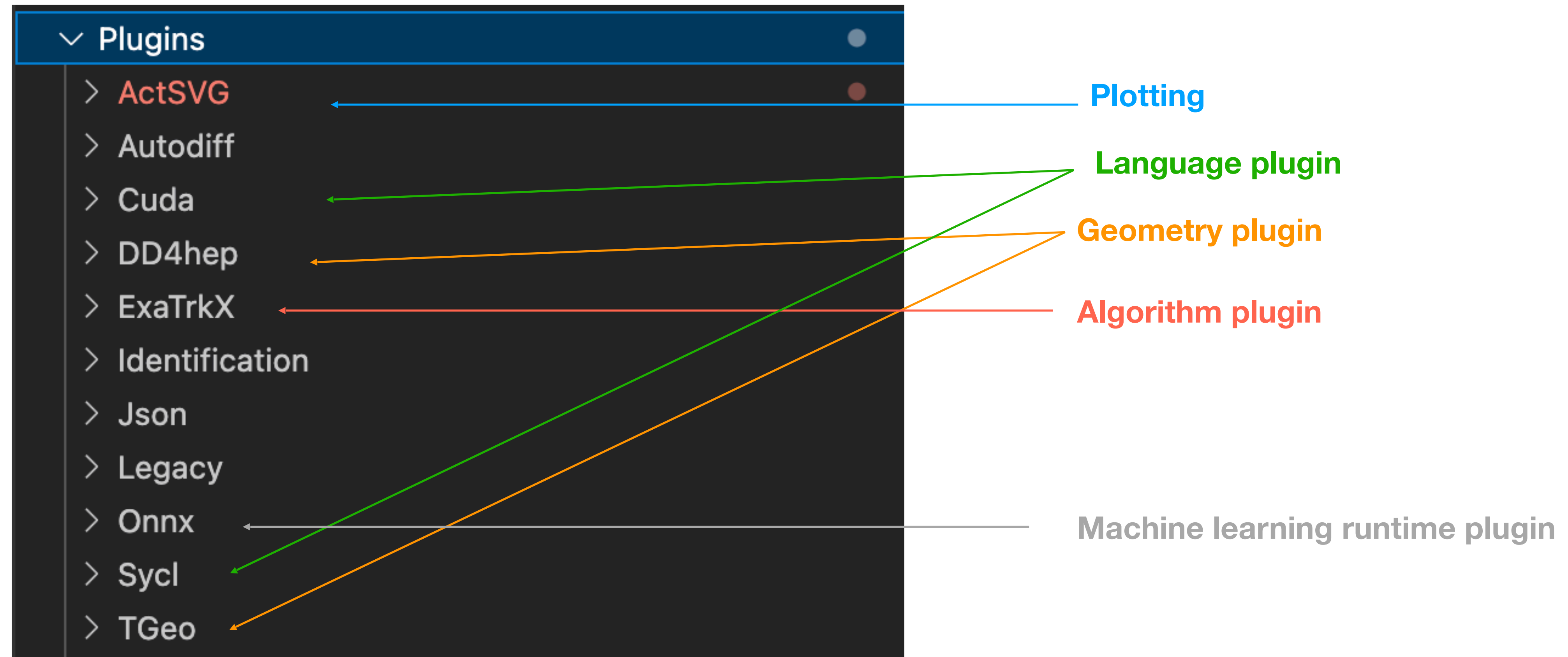
Aim is here to have interchangeable components.



# Core: Extensibility

## Plugin mechanism

- Library is extendable in functionality with several plugins
- Usually also pull in additional third party dependencies



# Core: testing

Comprehensive **Unit testing** is one of the main targets of our development model

- Best practise: write the code & tests together
- Small testable units/modules is key to this

Based on BOOST unit testing framework,  
Codecov (as part of CI) checks covering

**Codecov Report**

Merging #1551 (59d52ae) into main (f3b20f7) will decrease coverage by 0.00%.  
The diff coverage is 0.00%.

@@ ##	Coverage main	Diff #1551	Diff +/-	@@ ##
	48.47%	48.46%	-0.01%	
Files	381	381		
Lines	20699	20702	+3	
Branches	9503	9504	+1	
Hits	10034	10034		
- Misses	4112	4115	+3	
Partials	6553	6553		

```
namespace Acts {
using namespace detail;

namespace Test {

BOOST_AUTO_TEST_CASE(grid_test_1d_equidistant) {
    using Point = std::array<double, 1>;
    using indices = std::array<size_t, 1>;
    EquidistantAxis a(0.0, 4.0, 4u);
    Grid<double, EquidistantAxis> g(std::make_tuple(std::move(a)));

    // test general properties
    BOOST_CHECK_EQUAL(g.size(), 6u);
    BOOST_CHECK_EQUAL(g.numLocalBins().at(0), 4u);

    // global bin index
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{-0.3}})), 0u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{-0.}})), 1u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{0.}})), 1u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{0.7}})), 1u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{1}})), 2u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{1.2}})), 2u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{2.}})), 3u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{2.7}})), 3u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{3.}})), 4u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{3.9999}})), 4u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{4.}})), 5u);
    BOOST_CHECK_EQUAL(g.globalBinFromPosition(Point({{4.98}})), 5u);
}
}
```

# Core: collaboration & contributing

ACTS is Open Source and invites contributions, corrections, interactions



<https://github.com/acts-project/acts>

Clone:

<https://github.com/<username>/acts>

Develop & Make a PR



Make an Issue:

<https://github.com/acts-project/acts>



Ask on mattermost:

<https://mattermost.web.cern.ch/acts/channels/town-square>

Development, Exchange with Experts,  
Collaboration, Code review, CI testing



Discuss at the open develops meeting  
<https://indico.cern.ch/category/7968/>

Tuesday 17:00, CE(S)T



# Core: contributing

Pull requests come with a template that guides through a proper submission

**semantic naming: feat, doc, refactor, fix**

The screenshot shows a GitHub pull request interface. The title is "refactor!: MTJ stores measurement as jagged vector #1512", where "refactor!" is circled in blue. Below the title, it says "paulgessinger wants to merge 8 commits into acts-project:main from paulgessinger:refactor/mtj-jagged-meas".

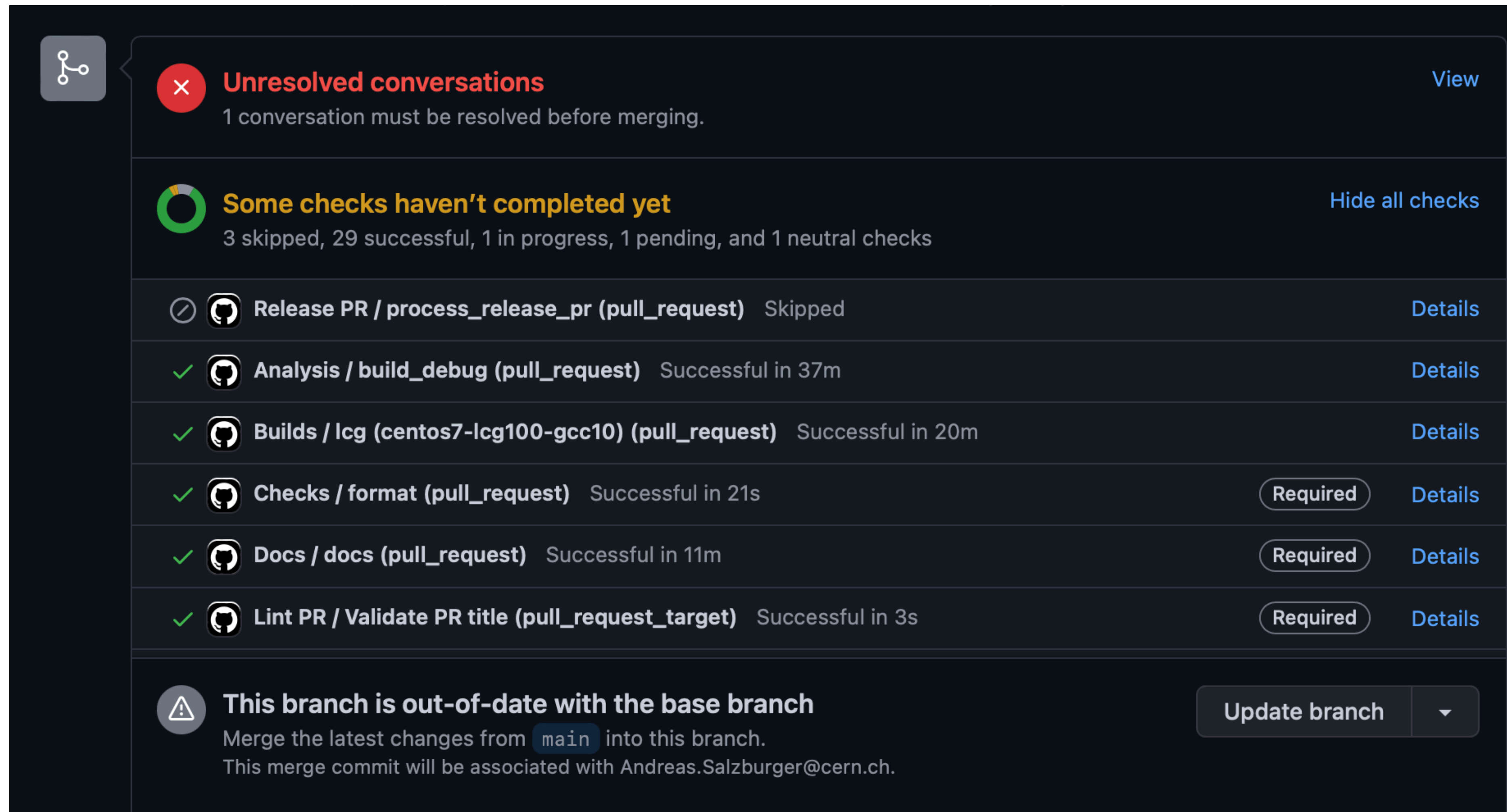
On the right side, there is a sidebar with the word "reviewers" in red. It lists two reviewers: "benjaminhuth" and "tboldagh", both circled in red. Below the reviewers, there are sections for "Assignees", "Labels", "Projects", and "Milestone". The "Milestone" section is circled in green and contains the word "milestone".

In the main content area, there is a comment from "paulgessinger" with a "meaningful description" in orange. The description includes a diagram of a jagged vector and a code diff. The code diff shows a breaking change in the `Acts::MultiTrajectory` measurement access methods. The diff is as follows:

```
- constexpr auto measurement(IndexType measIdx) const;  
+ template <size_t measdim>  
+ constexpr auto measurement(IndexType measIdx) const;  
  
and  
  
- constexpr auto measurementCovariance(IndexType covIdx)  
+ template <size_t measdim>  
+ constexpr auto measurementCovariance(IndexType covIdx)
```

# Core: contributing & testing

Pull requests run through a CI pipeline



The screenshot shows the GitHub Actions interface for a pull request. At the top, there is a 'View' link next to a red 'x' icon and the text 'Unresolved conversations' with a sub-message '1 conversation must be resolved before merging.' Below this is a green progress indicator and the text 'Some checks haven't completed yet' with a sub-message '3 skipped, 29 successful, 1 in progress, 1 pending, and 1 neutral checks' and a 'Hide all checks' link. The main section lists several workflow jobs:

- Release PR / process\_release\_pr (pull\_request)**: Skipped. Includes a 'Details' link.
- Analysis / build\_debug (pull\_request)**: Successful in 37m. Includes a 'Details' link.
- Builds / lcg (centos7-lcg100-gcc10) (pull\_request)**: Successful in 20m. Includes a 'Details' link.
- Checks / format (pull\_request)**: Successful in 21s. Includes a 'Required' badge and a 'Details' link.
- Docs / docs (pull\_request)**: Successful in 11m. Includes a 'Required' badge and a 'Details' link.
- Lint PR / Validate PR title (pull\_request\_target)**: Successful in 3s. Includes a 'Required' badge and a 'Details' link.

At the bottom, there is a warning icon and the text 'This branch is out-of-date with the base branch' with a sub-message 'Merge the latest changes from `main` into this branch. This merge commit will be associated with `Andreas.Salzburger@cern.ch`.' To the right of this message is an 'Update branch' button with a dropdown arrow.

# Core: collaboration & community

## Community-Supported Components: Acts

Weekly dev meeting with involvement of users at multiple experiments

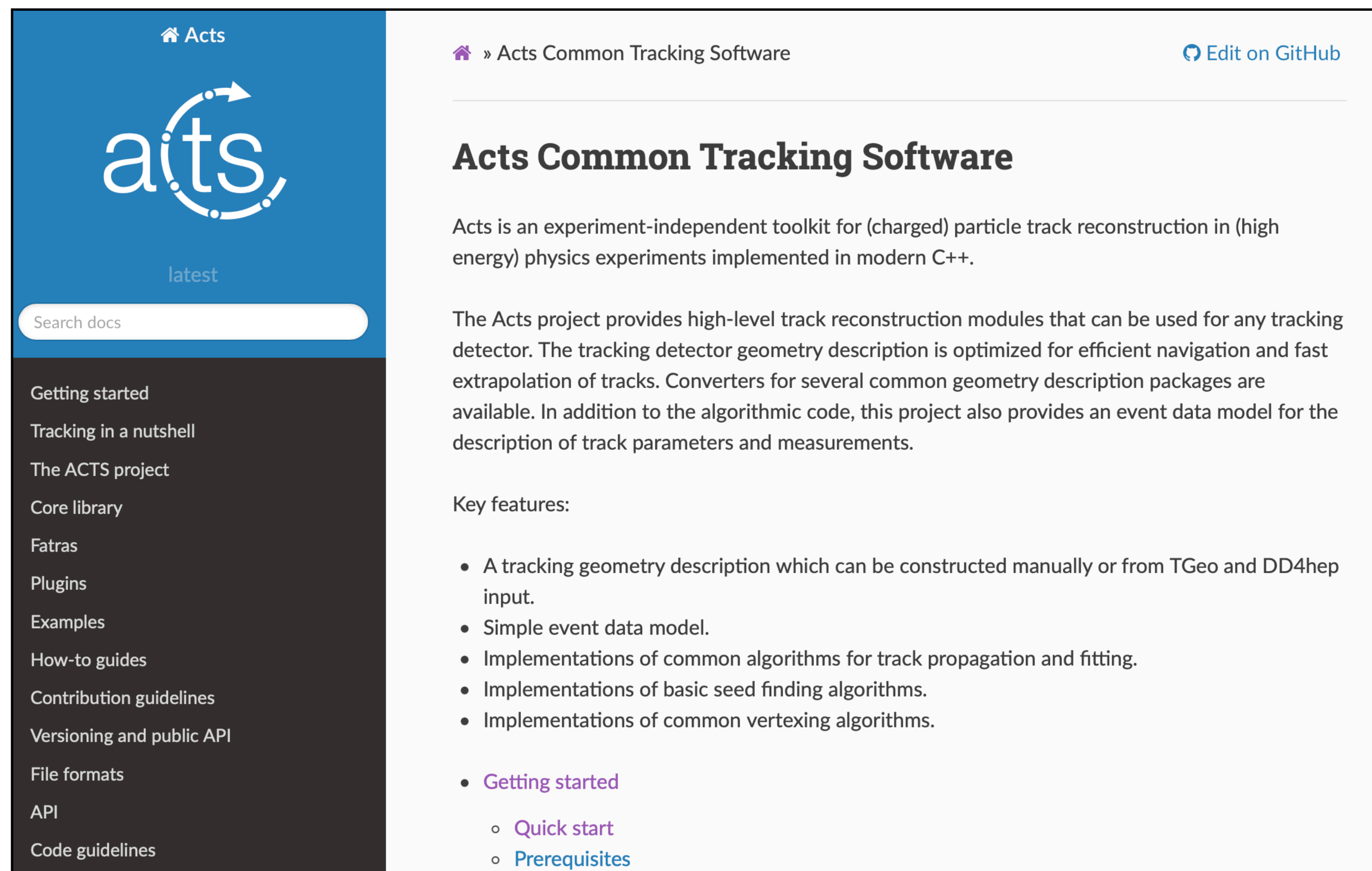
Status of work visibility through presentations

Example of agile in community software

<p>acts-project/acts: PRs merged between 2022-09-13 and 2022-09-20 II</p> <ul style="list-style-type: none"><li>docs: Update logging doc, add info on thresholds (PR#1520) by @paulgessinger, no assignee, merged on 2022-09-16</li><li>docs: update markdown cheatsheet (PR#1524) by @benjaminhuth, no assignee, merged on 2022-09-16</li><li>feat: Exa.TrkX with torchscript backend (PR#1473) by @benjaminhuth, no assignee, merged on 2022-09-16</li><li>docs: Gaussian Sum Filter (PR#1403) by @benjaminhuth, assigned to @benjaminhuth, merged on 2022-09-16</li><li>fix: Added missing return to seedfinder::CreateSeedsForGroup (PR#1521) by @guilhermeAlmeida1, no assignee, merged on 2022-09-16</li><li>refactor: Improve material mapping speed (PR#1458) by @Corentin-Allaire, assigned to @asalzbürger, merged on 2022-09-16</li><li>feat: Allow configurable particle selection and reproducible seeds for Geant4 (PR#1428) by @benjaminhuth, no assignee, merged on 2022-09-19</li><li>chore: Add priority merge label to kodiak config (PR#1532) by @paulgessinger, no assignee, merged on 2022-09-19</li></ul> <p>3/14</p>	<p>acts-project/acts: PRs merged between 2022-09-13 and 2022-09-20 III</p> <ul style="list-style-type: none"><li>chore: Add priority label to kodiak config (PR#1533) by @paulgessinger, no assignee, merged on 2022-09-19</li><li>docs: Contribution guidelines (PR#1525) by @paulgessinger, no assignee, merged on 2022-09-19</li><li>fix: ParticleSmearing options not setup in AMVF example exe (PR#1508) by @paulgessinger, no assignee, merged on 2022-09-19</li><li>refactor: improve full_chain_itk.py example (PR#1513) by @timadye, assigned to @andiwand, merged on 2022-09-19</li></ul> <p>4/14</p>
<p>acts-project/acts: Open PRs I</p> <ul style="list-style-type: none"><li>refactor: improve full_chain_odd.py example (PR#1538) by @andiwand, assigned to @timadye, updated on 2022-09-20</li><li>refactor: MTJ stores measurement as jagged vector (PR#1512) by @paulgessinger, no assignee, updated on 2022-09-20</li><li>feat: Hough Transform first implementation (PR#1305) by @jahreda, no assignee, updated on 2022-09-19</li><li>feat: Material Mapping Auto-tuning script with Orion (PR#1464) by @Corentin-Allaire, no assignee, updated on 2022-09-16</li><li>docs: Exa.TrkX (PR#1517) by @benjaminhuth, no assignee, updated on 2022-09-16</li><li>fix: Refactor and fix component merging for GSF (PR#1364) by @benjaminhuth, assigned to @asalzbürger, updated on 2022-09-13</li><li>feat: Add a tool for writing B-fields to disk in CSV format (PR#1470) by @stephenswat, assigned to @stephenswat, updated on 2022-09-07</li></ul> <p>5/14</p>	<p>acts-project/acts: Open PRs II</p> <ul style="list-style-type: none"><li>WIP ci: test exatrnx training ci (PR#1505) by @benjaminhuth, no assignee, updated on 2022-09-20</li><li>WIP feat: MultiTrajectory backends const version (PR#1496) by @paulgessinger, no assignee, updated on 2022-09-20</li><li>WIP feat: VectorMultiTrajectory memory statistics (PR#1511) by @paulgessinger, no assignee, updated on 2022-09-19</li><li>WIP refactor: Add macro to simplify algorithm binding (PR#1510) by @benjaminhuth, no assignee, updated on 2022-09-19</li><li>WIP docs: updates to the seeding documentation (PR#1476) by @LuisFelipeCoelho, no assignee, updated on 2022-09-16</li><li>WIP docs: adding Fatras description (PR#1402) by @asalzbürger, assigned to @asalzbürger, updated on 2022-09-14</li><li>WIP docs: polish tgeo plugin doc (PR#1397) by @niermann999, assigned to @niermann999, updated on 2022-09-14</li></ul> <p>6/14</p>

# Core: documentation

Submitted code should have `doxygen` documentation and `readthedocs` resources



The screenshot shows the documentation page for Acts Common Tracking Software. The left sidebar contains a navigation menu with the following items: Getting started, Tracking in a nutshell, The ACTS project, Core library, Fatras, Plugins, Examples, How-to guides, Contribution guidelines, Versioning and public API, File formats, API, and Code guidelines. The main content area features the Acts logo, a search bar, and the title "Acts Common Tracking Software". Below the title is a paragraph describing Acts as an experiment-independent toolkit for (charged) particle track reconstruction in (high energy) physics experiments implemented in modern C++. A "Key features:" section follows, listing several bullet points: A tracking geometry description which can be constructed manually or from TGeo and DD4hep input; Simple event data model; Implementations of common algorithms for track propagation and fitting; Implementations of basic seed finding algorithms; and Implementations of common vertexing algorithms. At the bottom of the list, "Getting started" is highlighted, with sub-links for "Quick start" and "Prerequisites". A "Home » Acts Common Tracking Software" breadcrumb and an "Edit on GitHub" link are visible at the top right of the main content area.

<https://acts.readthedocs.io/en/latest/>

# Core: some selected clients



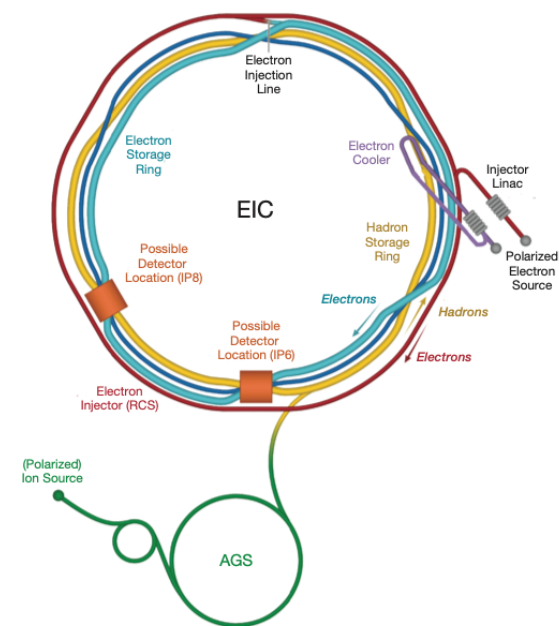
ACTS Vertex reconstruction in Run-3  
full ACTS powered reconstruction for Phase-2



full ACTS powered track reconstruction  
[\[ CSBS Article \]](#)



full ACTS powered track reconstruction  
[\[ FASER Preprint with ACTS mention \]](#)



Electron-Ion Collider (EIC) software stack,  
common track reconstruction software based  
on ACTS  
[\[ CHEP2023 keynote talk \]](#)



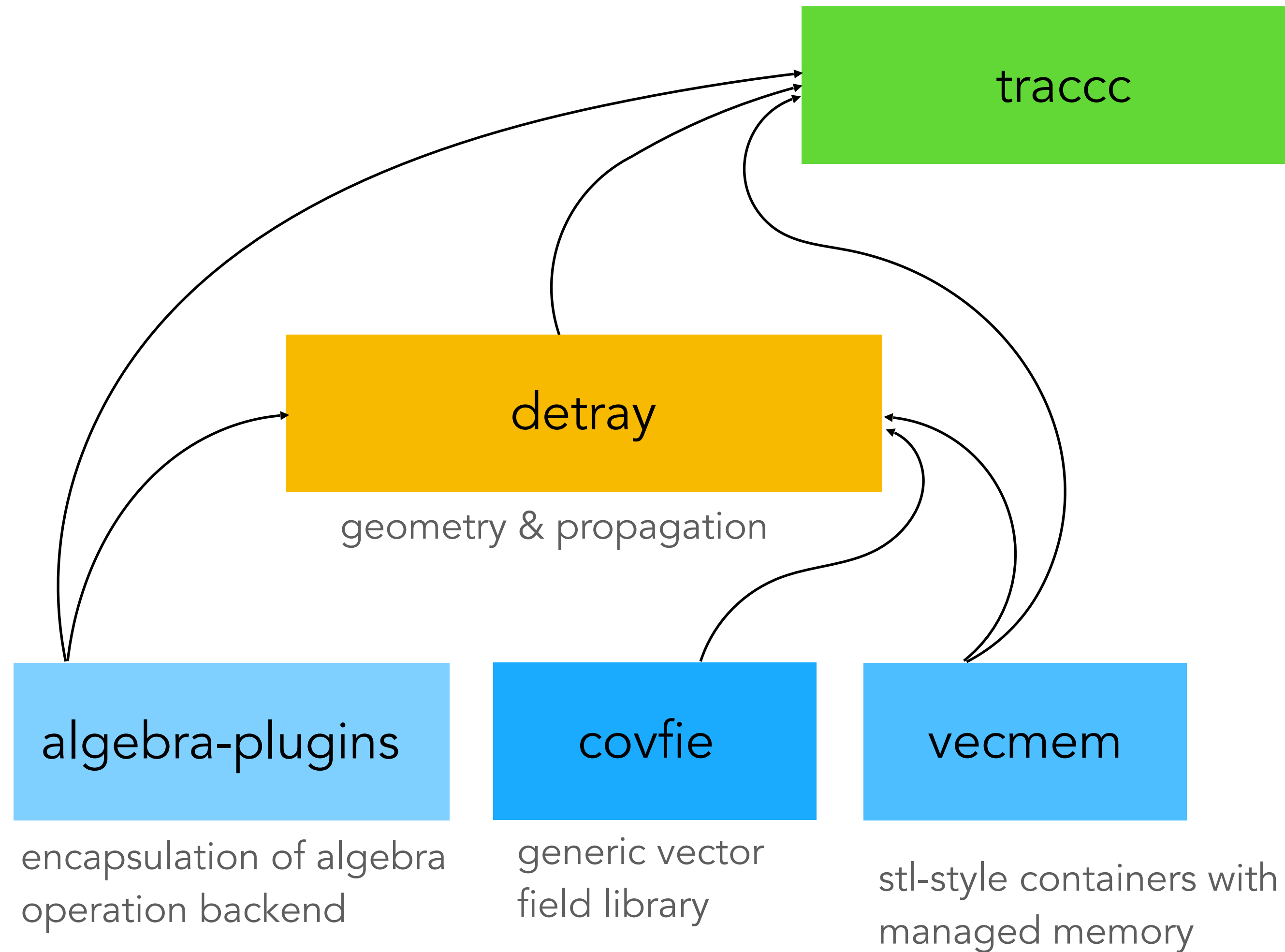
Test implementation for CEPC design study  
[\[ CEPC Concept paper \]](#)

**R&D1**

[acts-parallelization@cern.ch](mailto:acts-parallelization@cern.ch)

CPU/GPU “single source” demonstrator  
re-implementing the main Core chain

# R&D1: the **traccc** project



full scale demonstrator of an ATLAS-like track reconstruction chain for CPU/GPU

# R&D1: vecmem & algebra-plugins

## vecmem: memory management

- use of `std::pmr::memory_resource` to customize the allocation scheme in the host side
- Supports CPU, CUDA, SYCL, and HIP
- Provides STL-like containers for host side for convenience of HEP developers

`vecmem::vector`, `vecmem::jagged_vector` (vector of vector), `vecmem::array`

## algebra-plugins: encapsulation layer for algebra operations

- targeted at track reconstruction entirely
- dimensions up to 8 (needed for parameter propagation)
- supports device execution where possible and vecmem based backend
- can be used for algebra library benchmarking in realistic applications (instead of mockup benchmarks)

Backend	CPU	CUDA	SYCL
<code>cmath</code>	✓	✓	✓
<code>Eigen</code>	✓	✓	●
<code>SMatrix</code>	✓	●	●
<code>VC</code>	✓	●	●

✓: natively supported

●: natively supported, but not tested

●: no support

32





# R&D1: track reconstruction: [covfie](#)

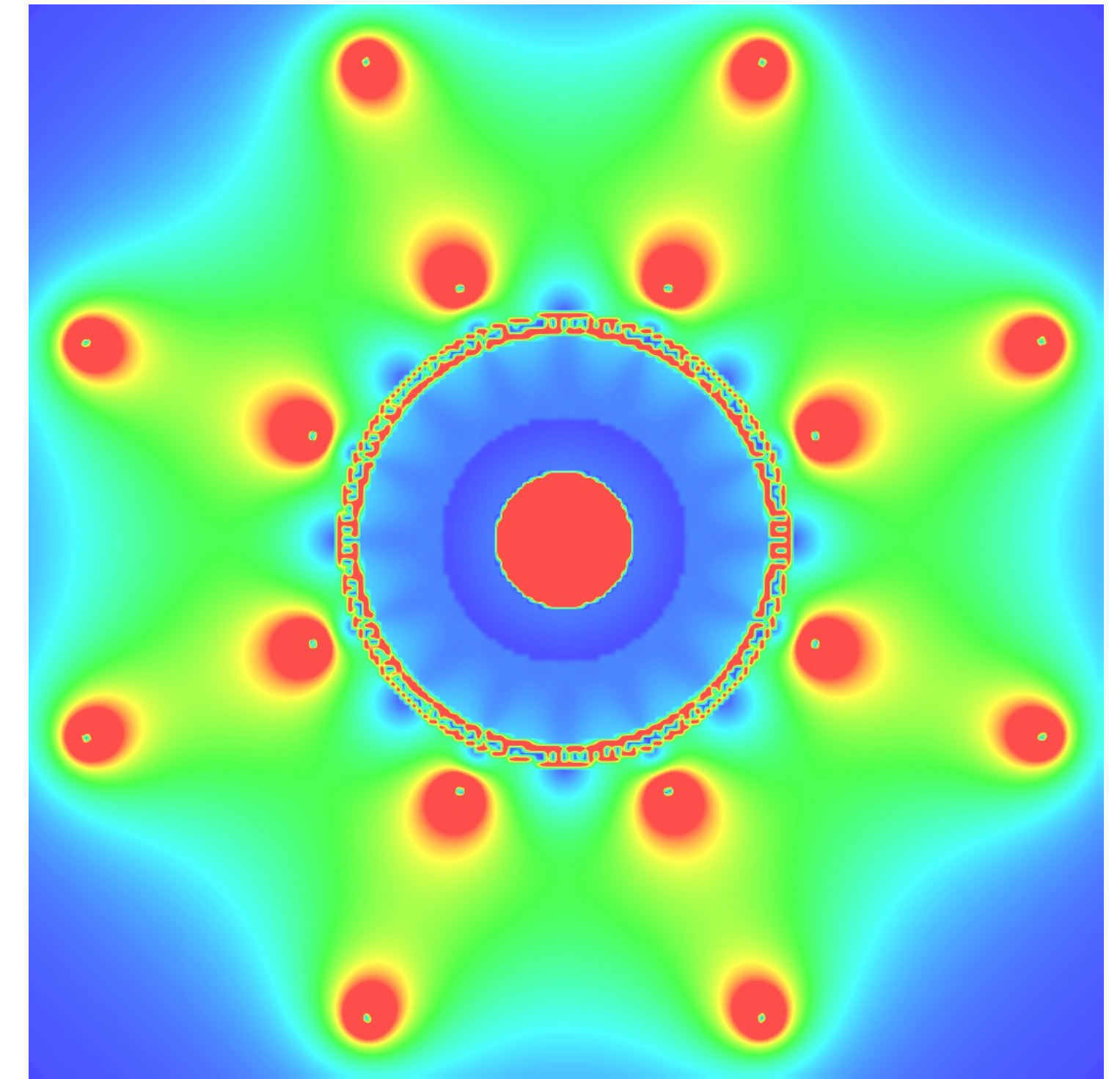
A generic vector field library based on composition design  
 - format, coordinate transform and storage at compile time

```
using field_t =
  covfie::field<covfie::backend::transformer::interpolator::linear<
    covfie::backend::layout::strided<
      covfie::vector::ulong2,
      covfie::backend::storage::array<covfie::vector::float2>>>>;
```

possible field on CPU

```
using cuda_field_t = covfie::field<covfie::backend::transformer::affine<
  covfie::backend::transformer::interpolator::linear<
    covfie::backend::layout::strided<
      covfie::backend::vector::input::ulong3,
      covfie::backend::storage::cuda_device_array<
        covfie::backend::vector::output::float3>>>>>;
```

possible field on GPU



ATLAS magnetic field slice at  $z=0$ ,  
entirely rendered on a GPU

	8192 X 8192 lookup time [ms]
CPU (Intel i5-7300U)	191719.2
GPU (GTX 1660 Ti)	90.4
GPU w/ texture memory	17.1

33

# R&D1: detray (1)

## Compile-time polymorphic geometry library

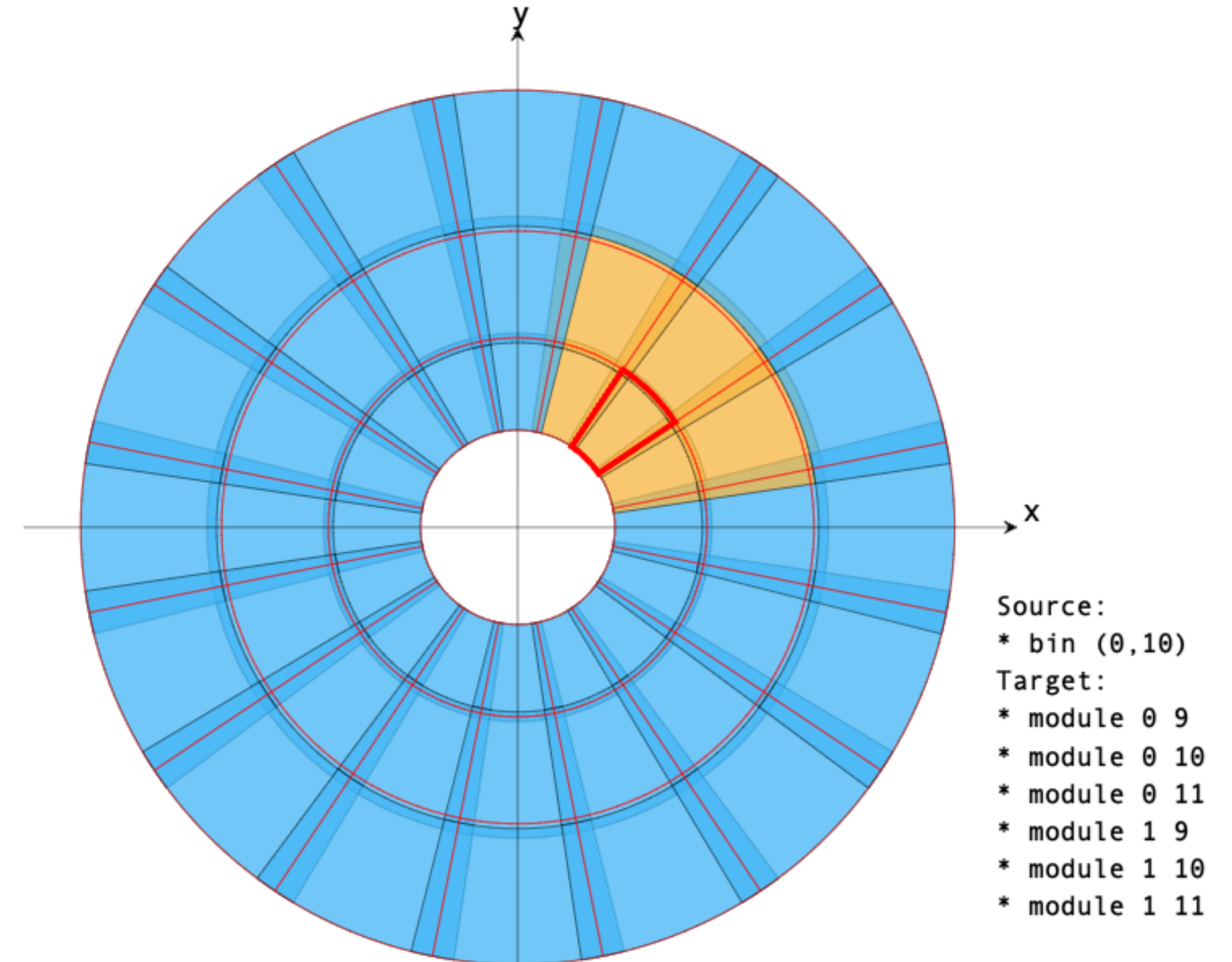
- bound surface type model and ACTS navigation
- polymorphism achieved by type unrolling
- device specialization through `vecmem`

```

/** The detector definition.
 *
 * This class is a heavy templated detector definition class, that sets the
 * interface between geometry, navigator and grid.
 *
 * @tparam metadata helper that defines collection and link types centrally
 * @tparam array_type the type of the internal array, must have STL semantics
 * @tparam tuple_type the type of the internal tuple, must have STL semantics
 * @tparam vector_type the type of the internal array, must have STL semantics
 * @tparam source_link the surface source link
 */
template <typename metadata,
          template <typename, std::size_t> class array_t = darray,
          template <typename...> class tuple_t = dtuple,
          template <typename...> class vector_t = dvector,
          template <typename...> class jagged_vector_t = djagged_vector,
          typename source_link = dindex>
class detector {

```

Endcap with templates



Local navigation resolving by  
grid binning in detray

34



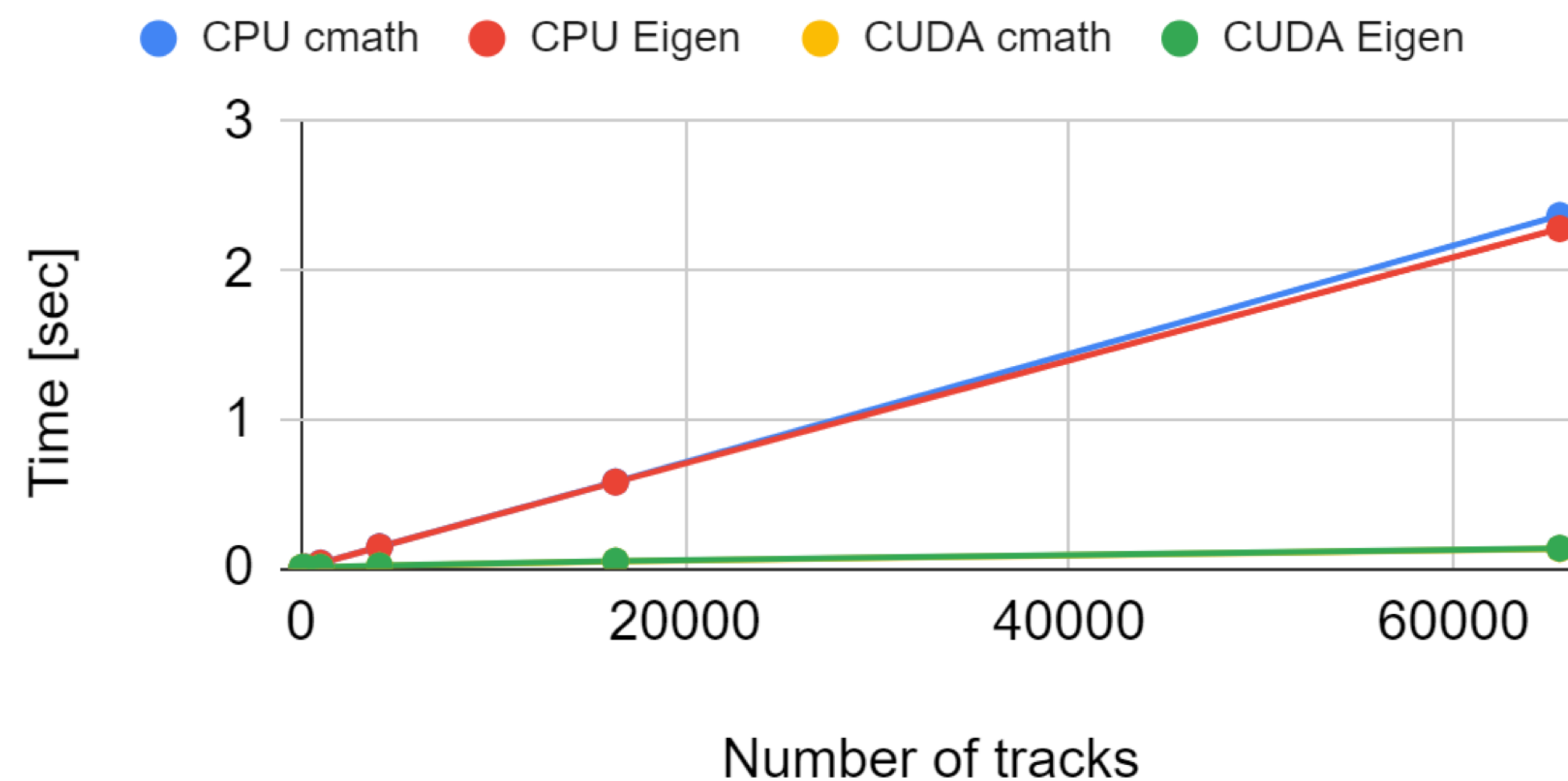
# R&D1: detray (2)

## Runge-Kutta propagation

- 4th order RKN propagator for parameters and covariances
- navigation and material effects integration using `detray::geometry`
- magnetic field access using `covfie`

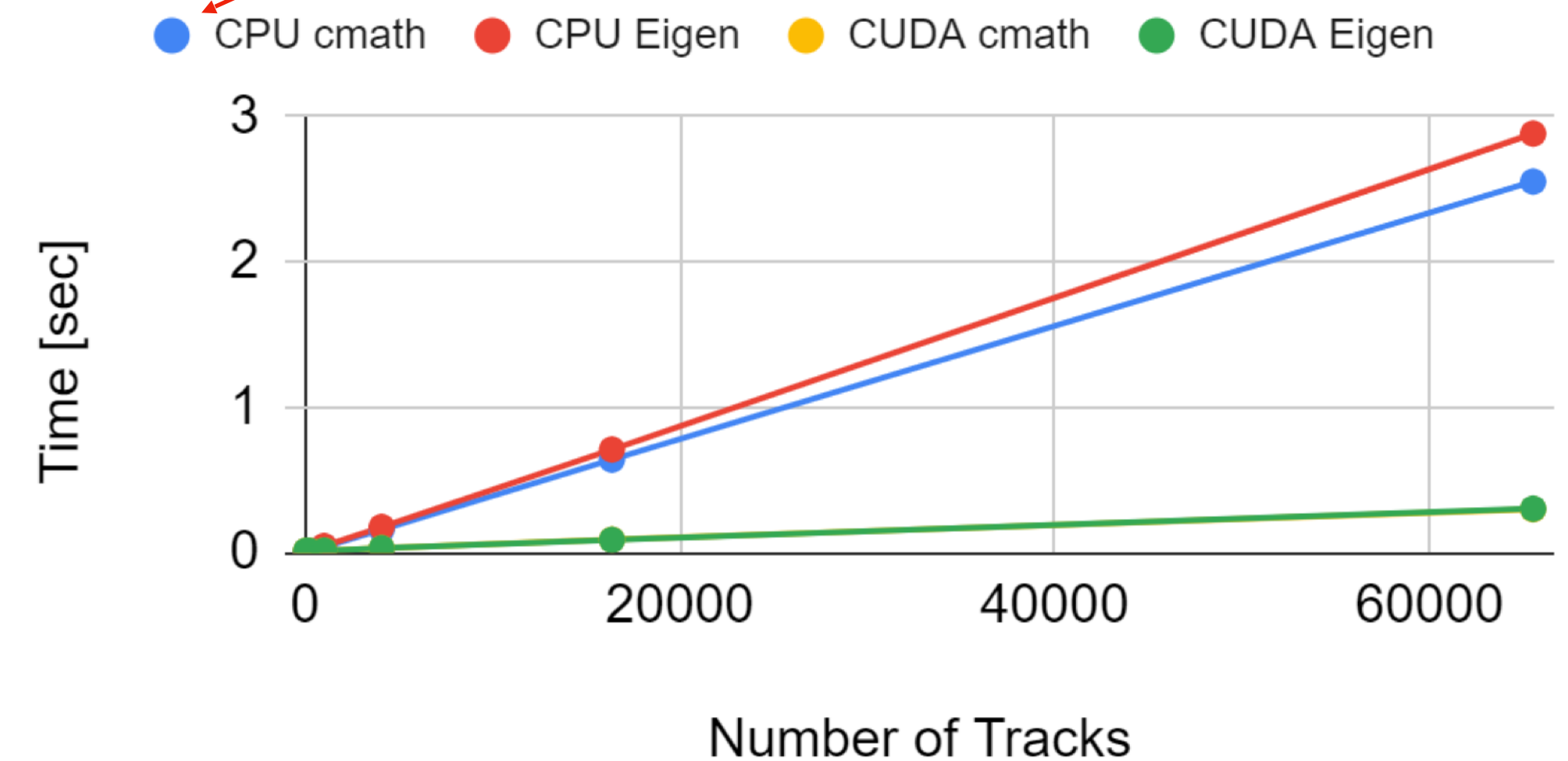
### Single Precision

CPU: i7-10750H / GPU: RTX 2070



### Double Precision

CPU: i7-10750H / GPU: RTX 2070

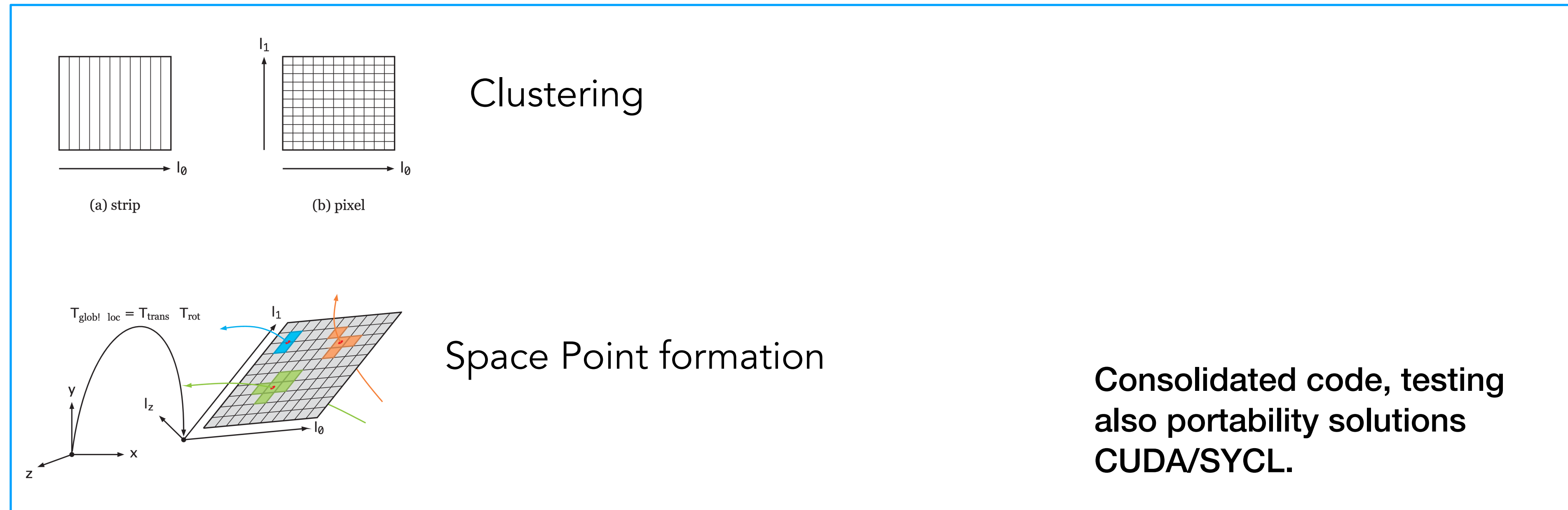


35

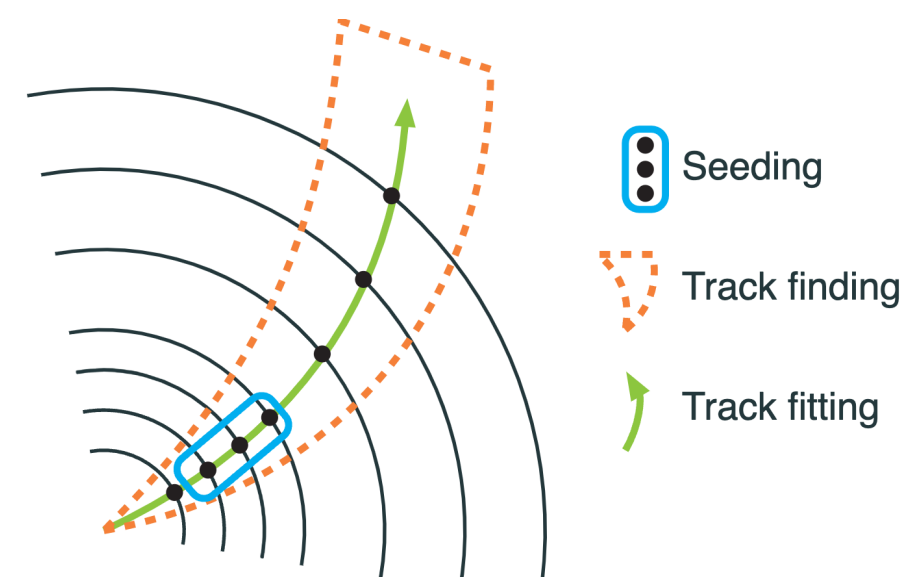


# R&D1: **traccc** (1)

Full chain demonstrator for track reconstruction on CPU/GPU



36



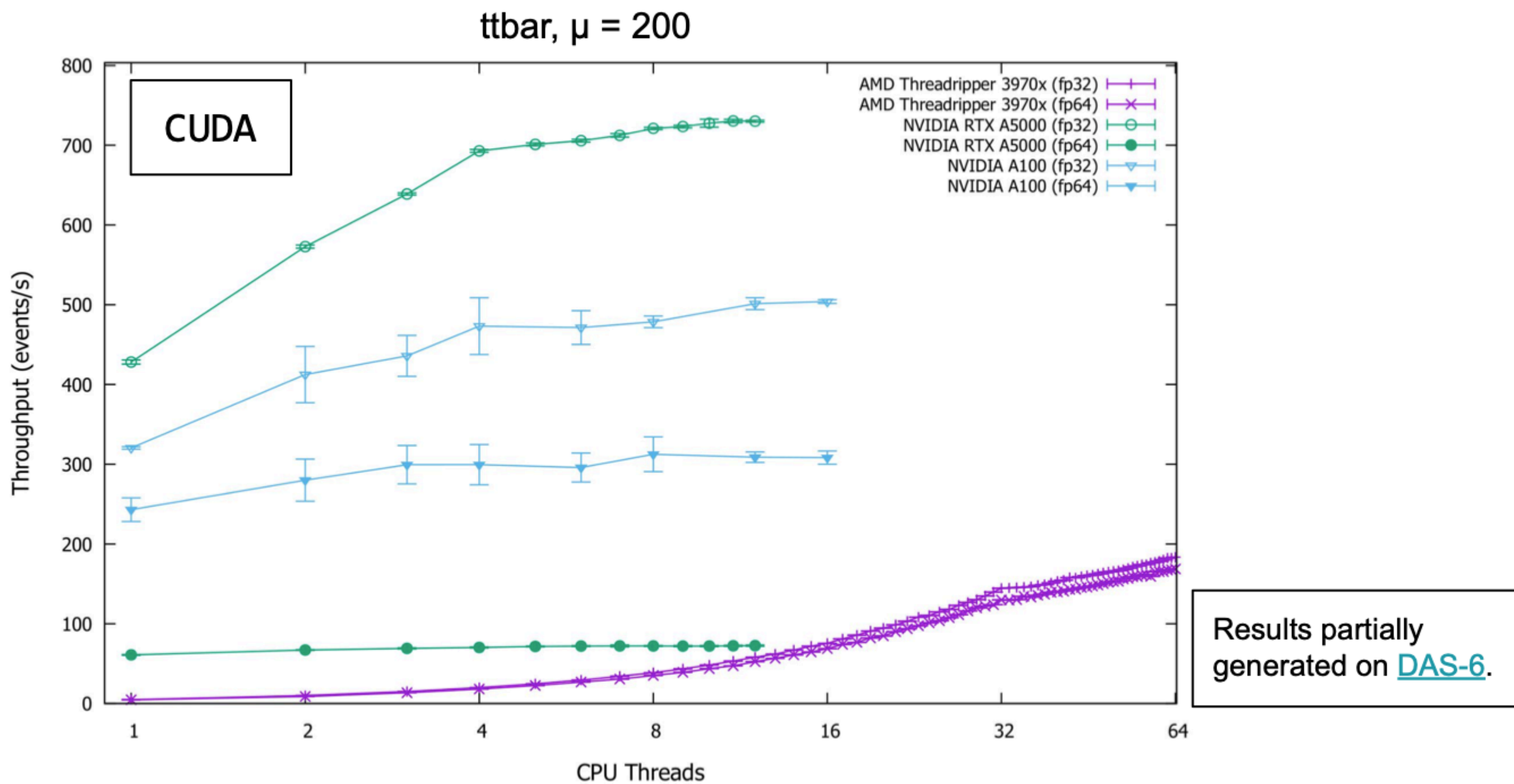
Track finding & fitting



[traccc](https://github.com/traccc) is an ideal playground to gather/share/exchange knowledge about code for heterogeneous systems.

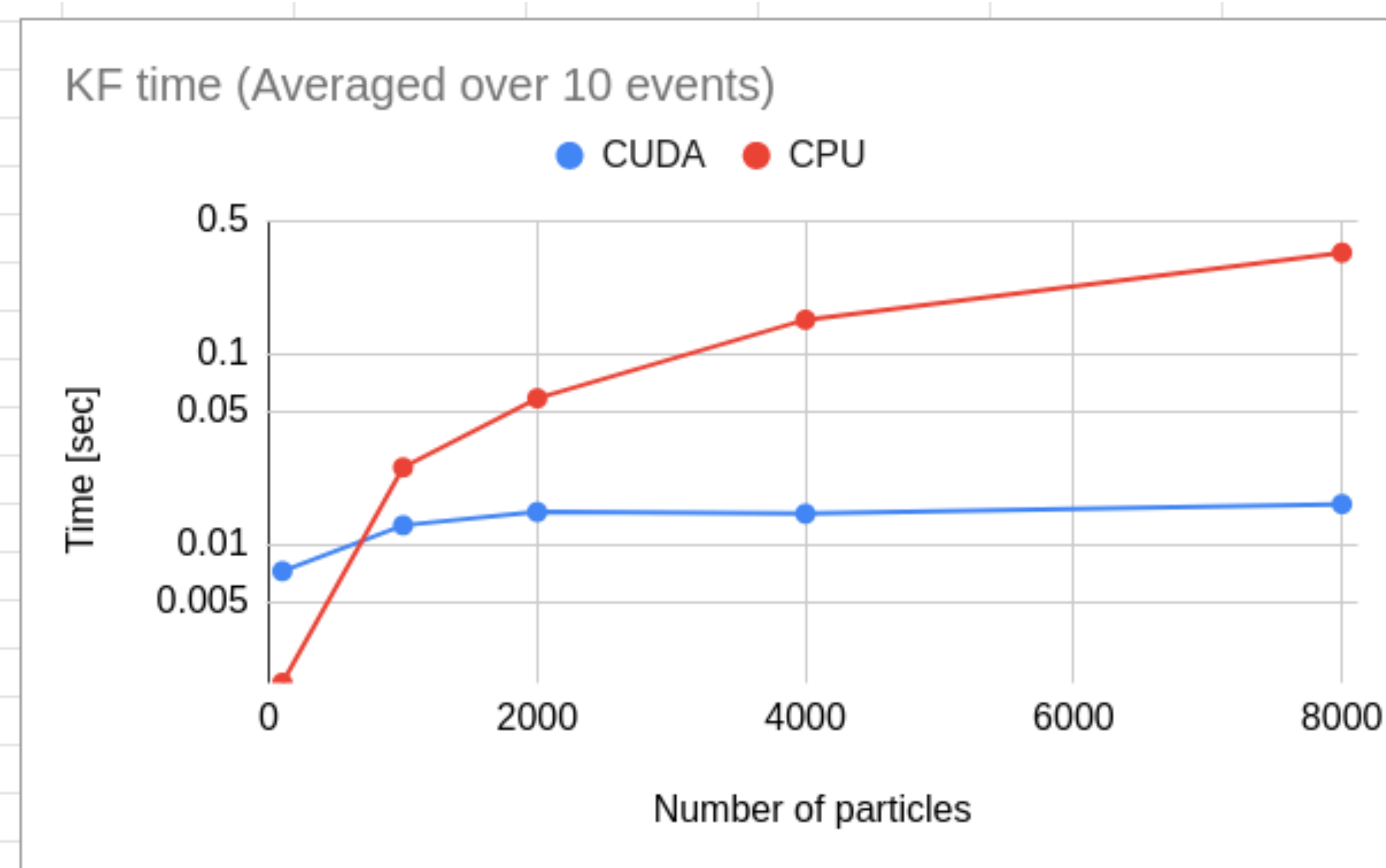
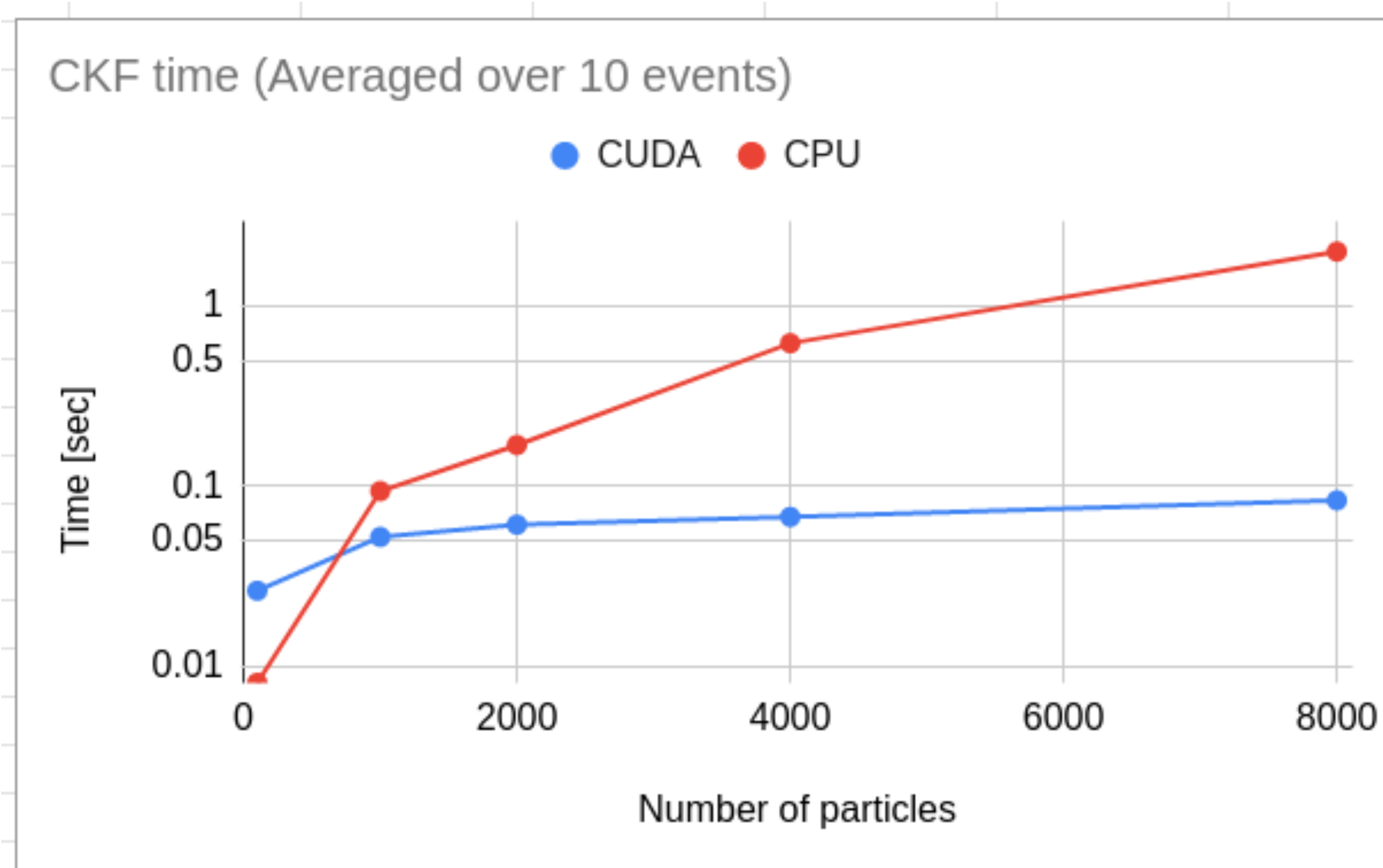
# R&D1n: **traccc** (2)

Full chain demonstrator for track reconstruction on CPU/GPU

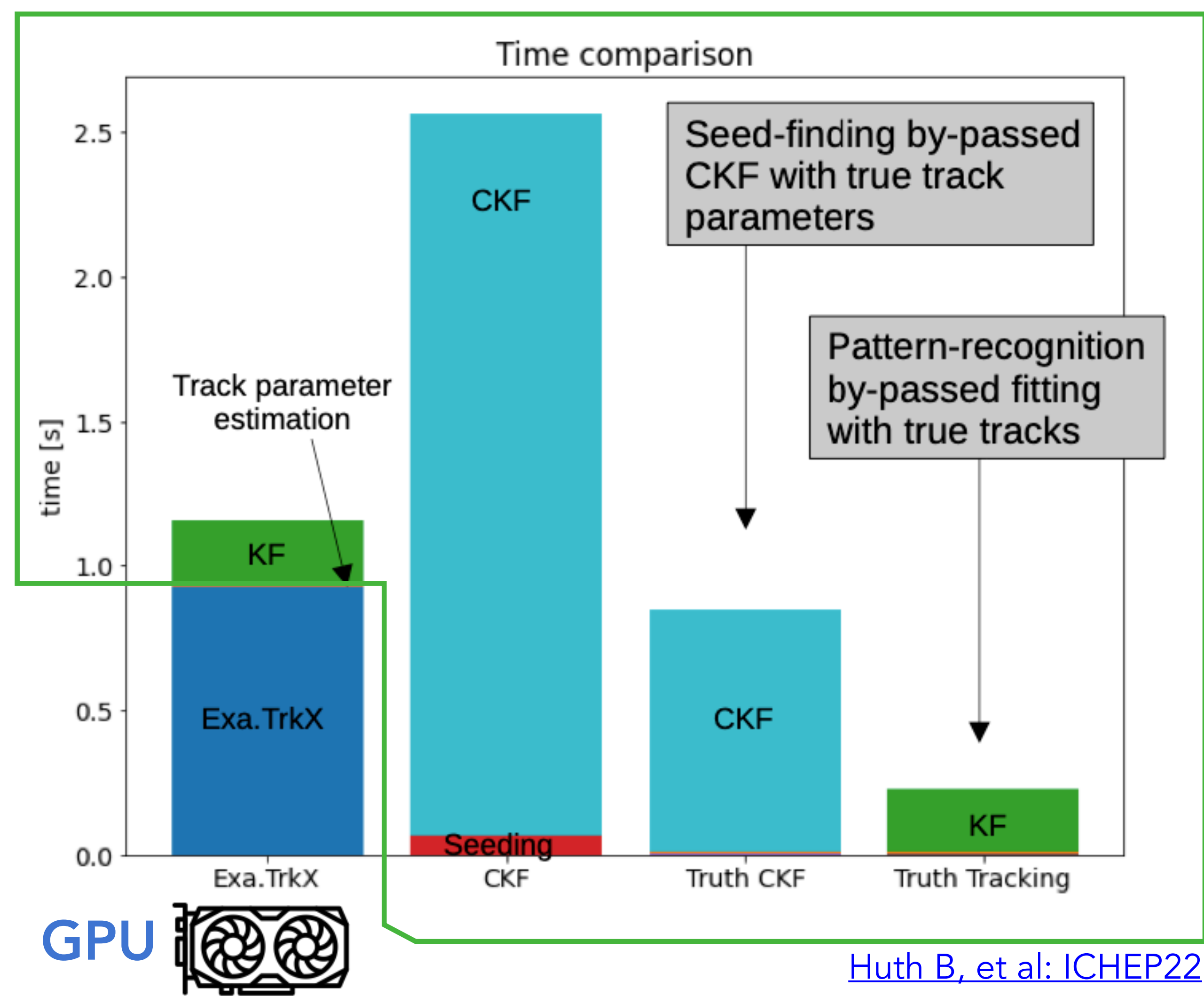


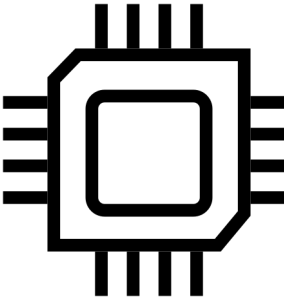
# R&D1n: **traccc (3)**

First results on CKF with detrayer - on toy detector (TrackML pixels)

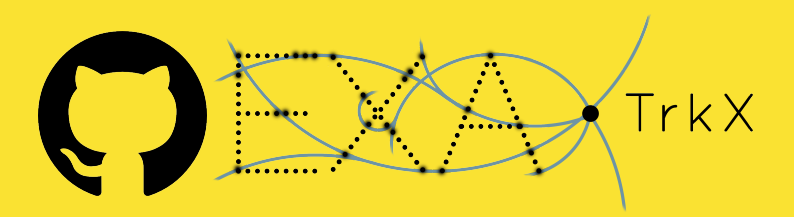
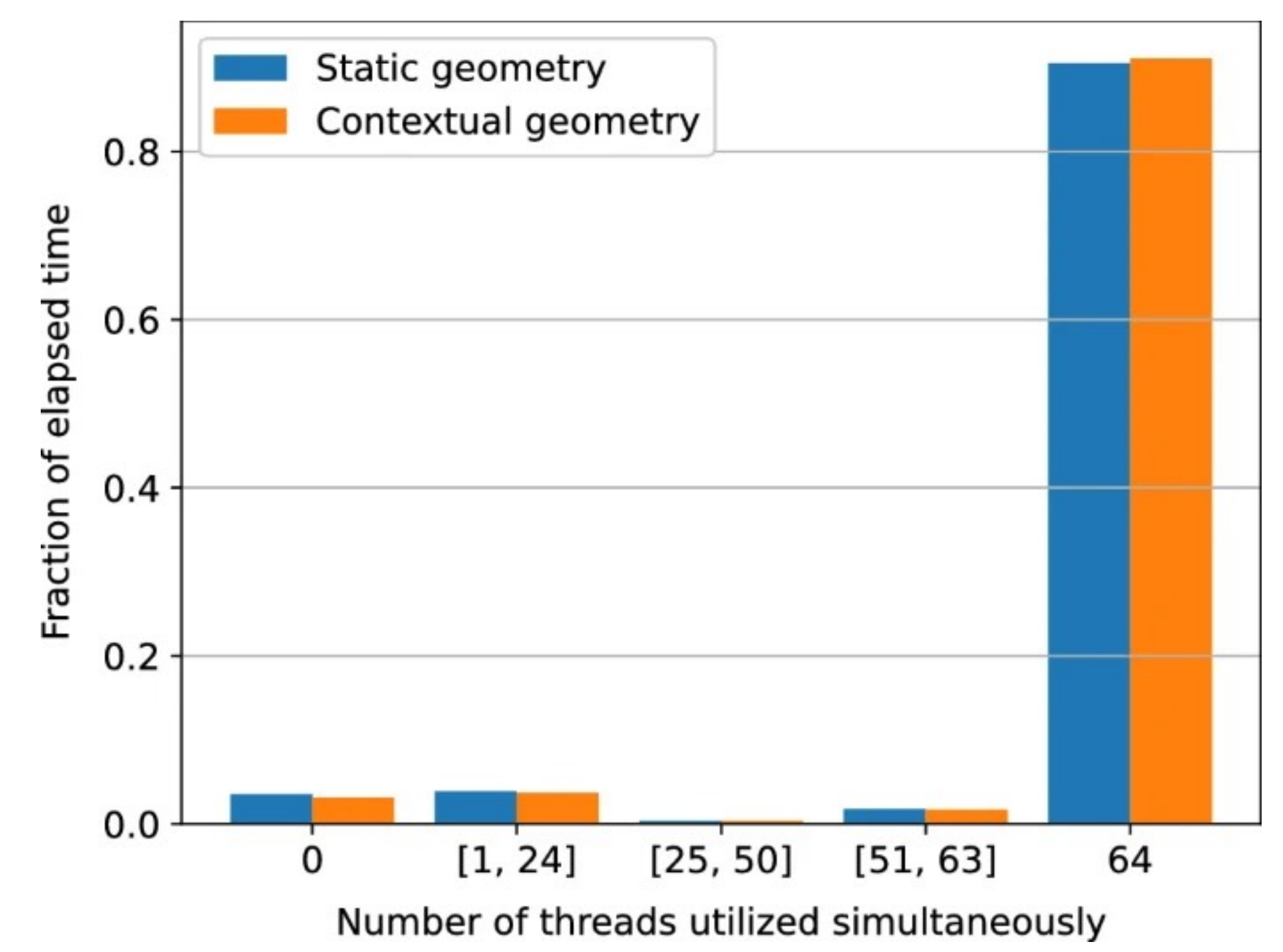


# R&D1: technology BINGO



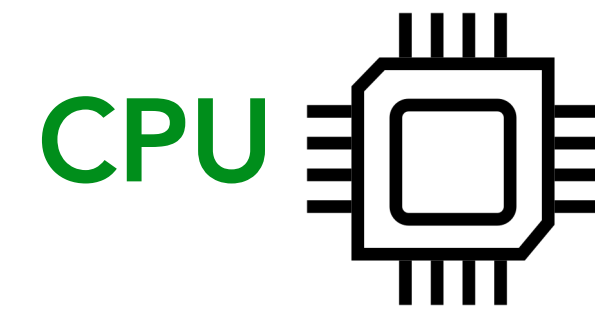
**CPU**  (single threaded)

Can we run large-scale multi threaded?  
[acts-project/acts](#) designed specifically for MT application



Exa.TrkX project is the state of the art end-to-end Graph Neural Network based track finding library.

# R&D1: technology BINGO



	x86	aarch64	oneAPI/SYCL	CUDA	
<a href="#">acts-project/acts</a>					<a href="#">acts-project/tracc</a> <a href="#">acts-project/detray</a>
<b>Core Line</b>	tested	tested	superseded	superseded	<a href="#">acts-project/covfie</a> <a href="#">acts-project/vecmem</a>
	tested (incomplete*)	tested (incomplete*)	tested (incomplete*)	tested (incomplete*)	<b>R&amp;D Line 1</b> “parallelization”
<b>R&amp;D Line 2</b> “machine learning”	tested	not tested	not implemented	tested w x86	

[exatrnx](#) & [acts-project/acts](#)



**R&D2**

[acts-machinelearning@cern.ch](mailto:acts-machinelearning@cern.ch)

Machine learning and ML assisted  
modules for track reconstruction

# R&D2: machine learning application and assistance

Diverse ML (assisted) applications

- ML module research:

**ML Ambiguity Solver**

[\[C. Allaire, CHEP2023, Parallel Talk\]](#)

**ML Navigator**

- integration of ML partial or end-to-end pipelines

**Exa.TrkX + ACTS**

**Hashing + ACTS**

- ML technology enhanced

**Parameter Tuning**

**Auto-diff covariance transport**

# Development and R&D, add-ons:

## Core

[acts-developers@cern.ch](mailto:acts-developers@cern.ch)

CPU multi-threaded library of  
tracking reconstruction components

## Add-ons

OpenDataDetector  
ActSVG

## R&D1

[acts-parallelization@cern.ch](mailto:acts-parallelization@cern.ch)

CPU/GPU “single source” demonstrator  
re-implementing the main Core chain

## R&D2

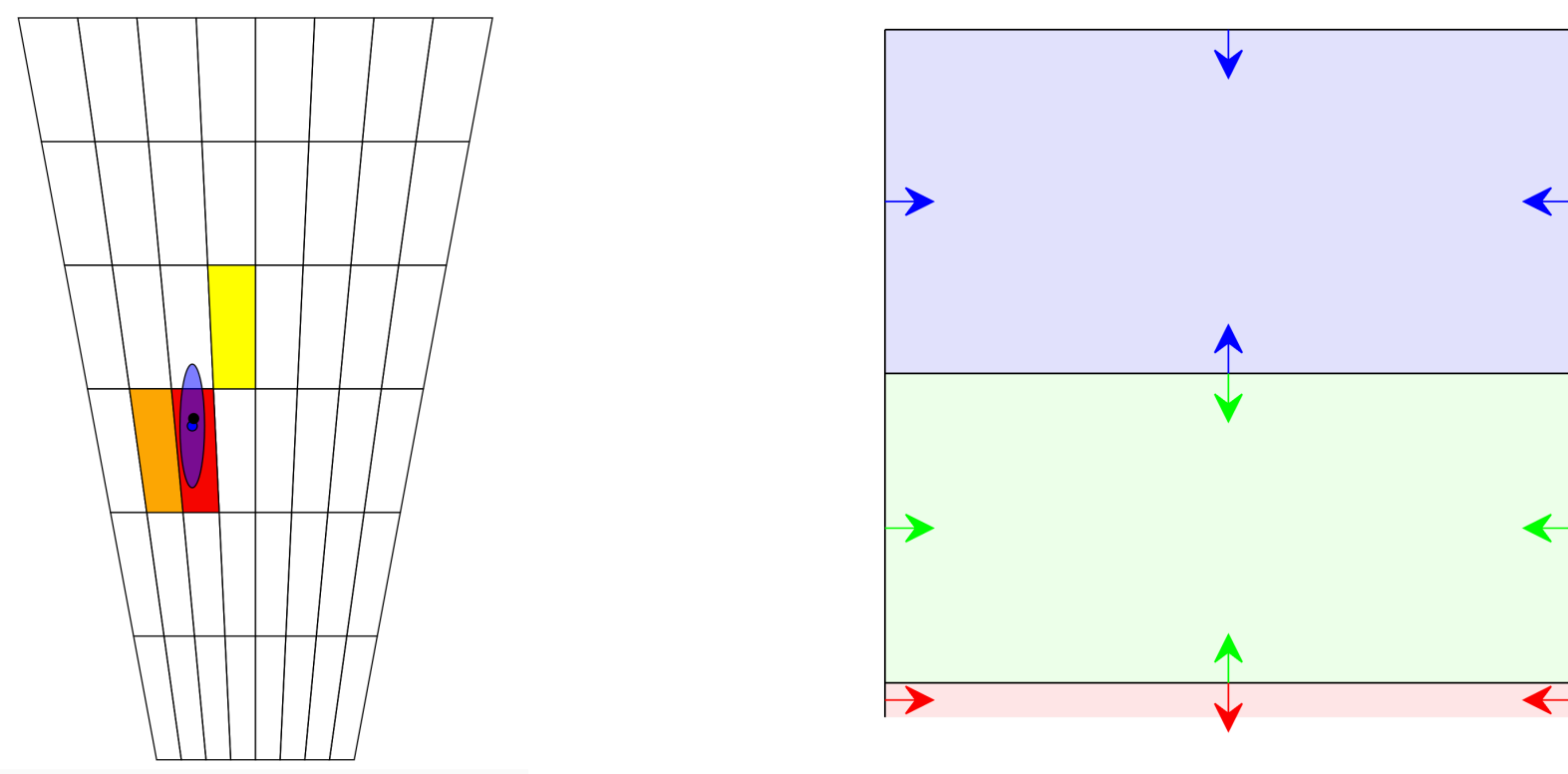
[acts-machinelearning@cern.ch](mailto:acts-machinelearning@cern.ch)

Machine learning and ML assisted  
modules for track reconstruction

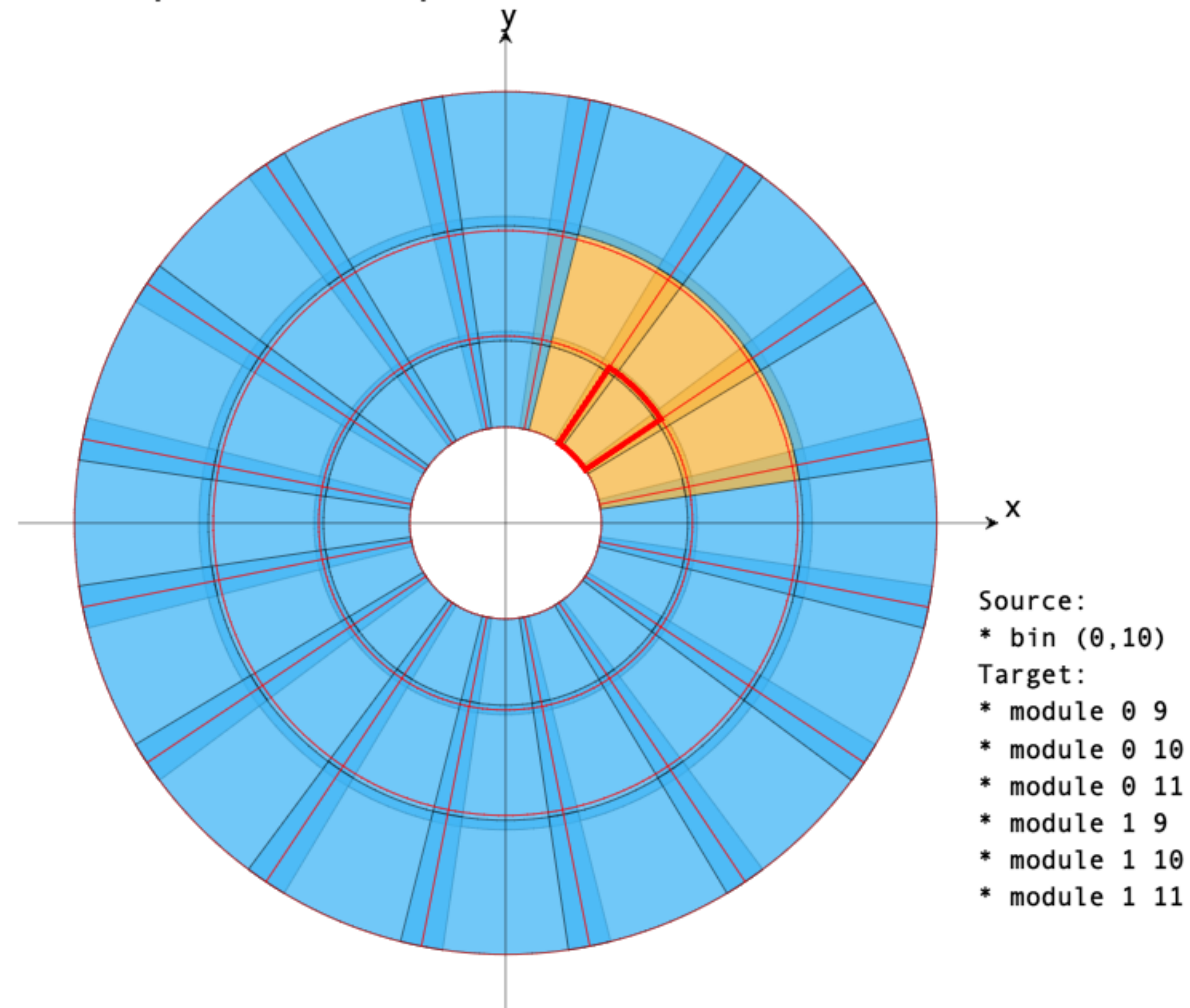
# Plotting: actsvg

2D plotting library dedicated for tracking

- No dependencies, C++ header only, no ACTS dependency
  - ACTS and detrays translate into `actsvg::meta` objects
- Plot geometry & geometric relations (on mouse over effects for debugging)
- Plot clusters & cluster information



Endcap with templates



44



# Community: Open Data Detector & key4hep

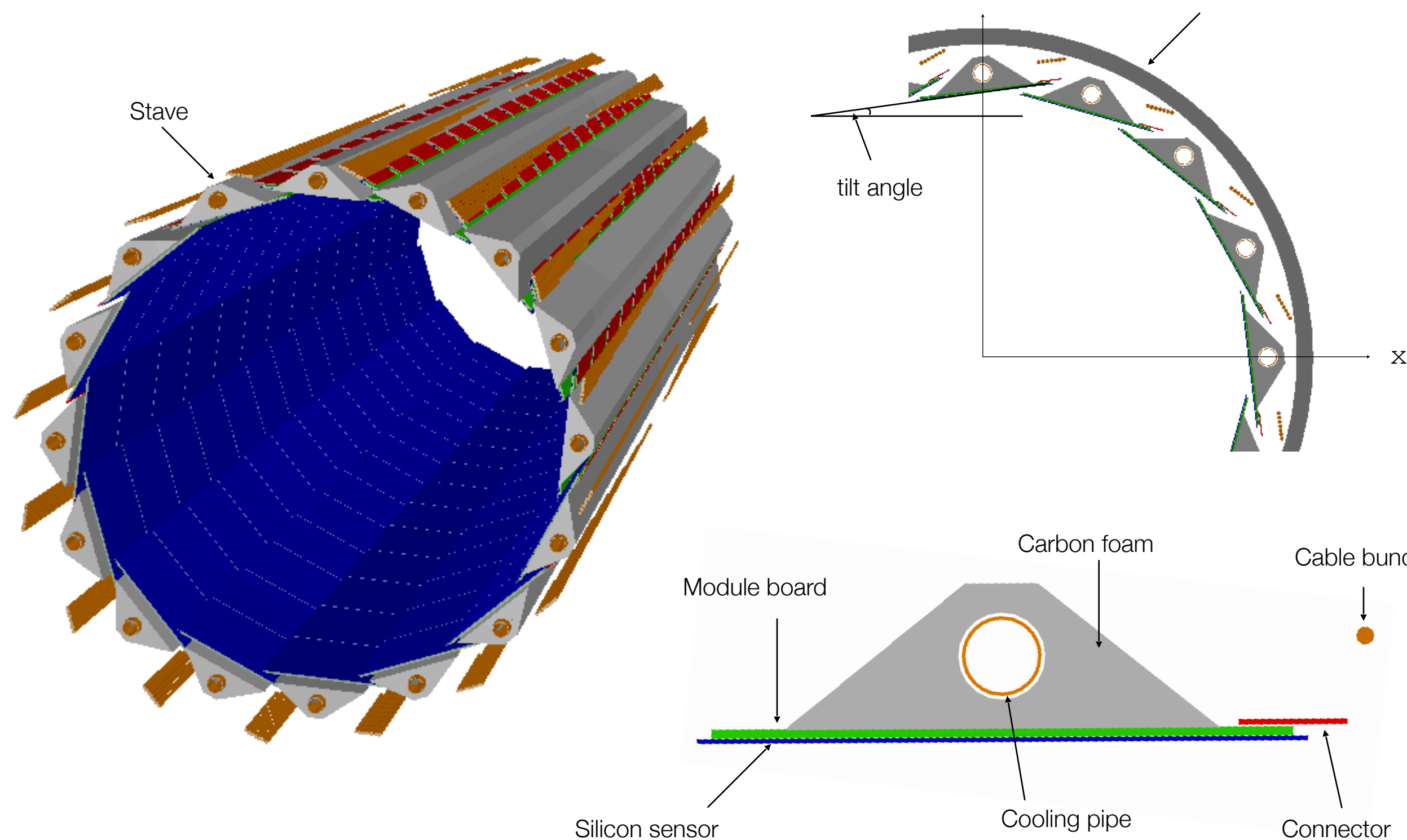
## Evolution of TrackML detector

- Re-implemented in DD4Hep to enable full/fast simulation
- Quasi-realistic feedback to allow real-life scenario testing of algorithms
- Supports TrackML output format through ACTS binding (work ongoing to also support edm4hep)

## ACTS integration into key4hep SW stack

- Codename: acts4hep
- Summer student project to make a ACTS Gaudi based demonstrator

[[AS, CHEP2023 Parallel talk](#)]



## Concluding remarks

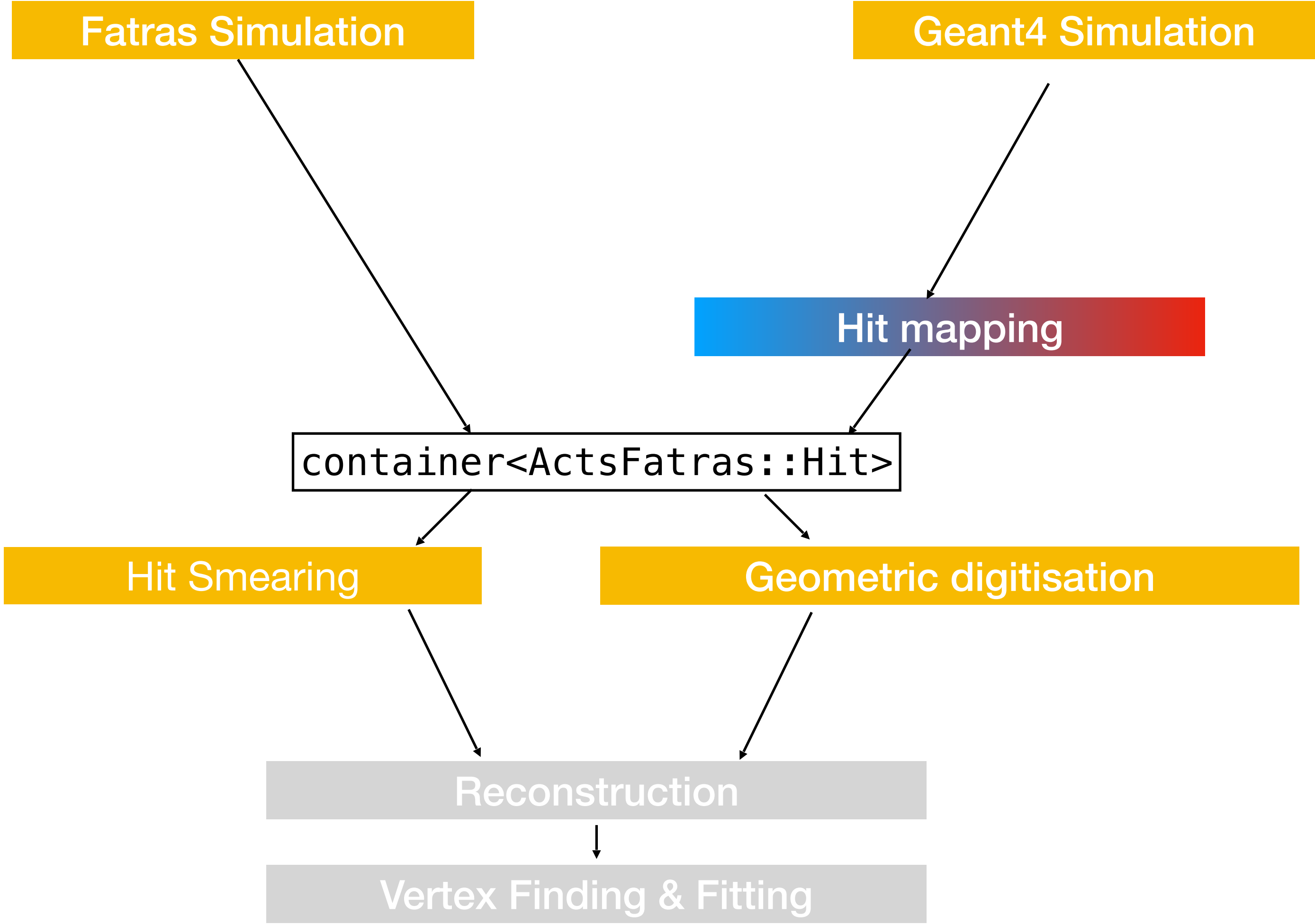
### **ACTS is a community driven software project**

- idea is to develop a broadly carried, supported and sustainable component library

**Join. Enhance. Profit. Give.**

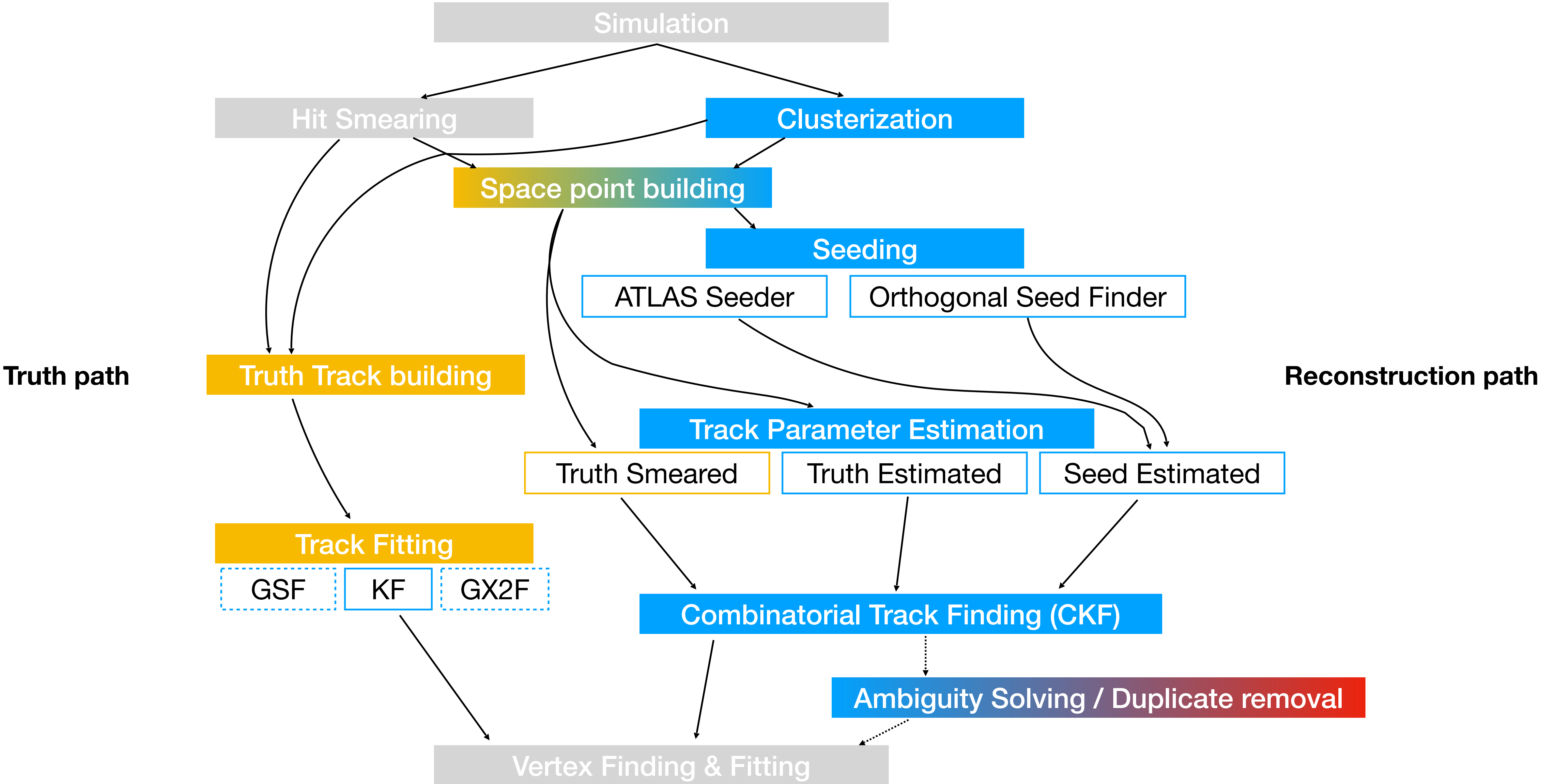


# Core functionality: simulation input





# Core functionality: a tracking demonstrator chain



# Core functionality: vertex reconstruction

