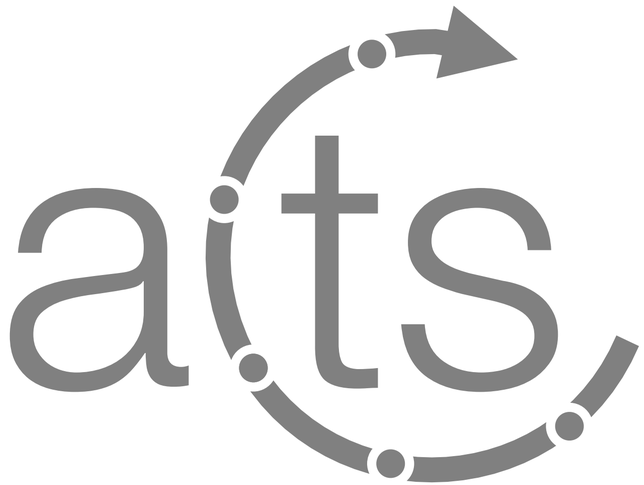
supported by

cooperations



project overview

A. Salzburger (CERN) for the ACTS project

@SaltyBurger

Development and R&D

Core

acts-developers@cern.ch

CPU multi-threaded library of
tracking reconstruction components

R&D1

acts-parallelization@cern.ch

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

R&D2

acts-machinelearning@cern.ch

Machine learning and ML assisted
modules for track reconstruction

Core

acts-developers@cern.ch

CPU multi-threaded library of
tracking reconstruction components

More than **1000** merged PRs, more than **100** forks, ~**75** stars
48 different contributors
289 unique cloners

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 will have to start thinking of C++20 soon)
- minimal core dependencies: CMake, Eigen, BOOST **+ optional Plugins**

Component library structure

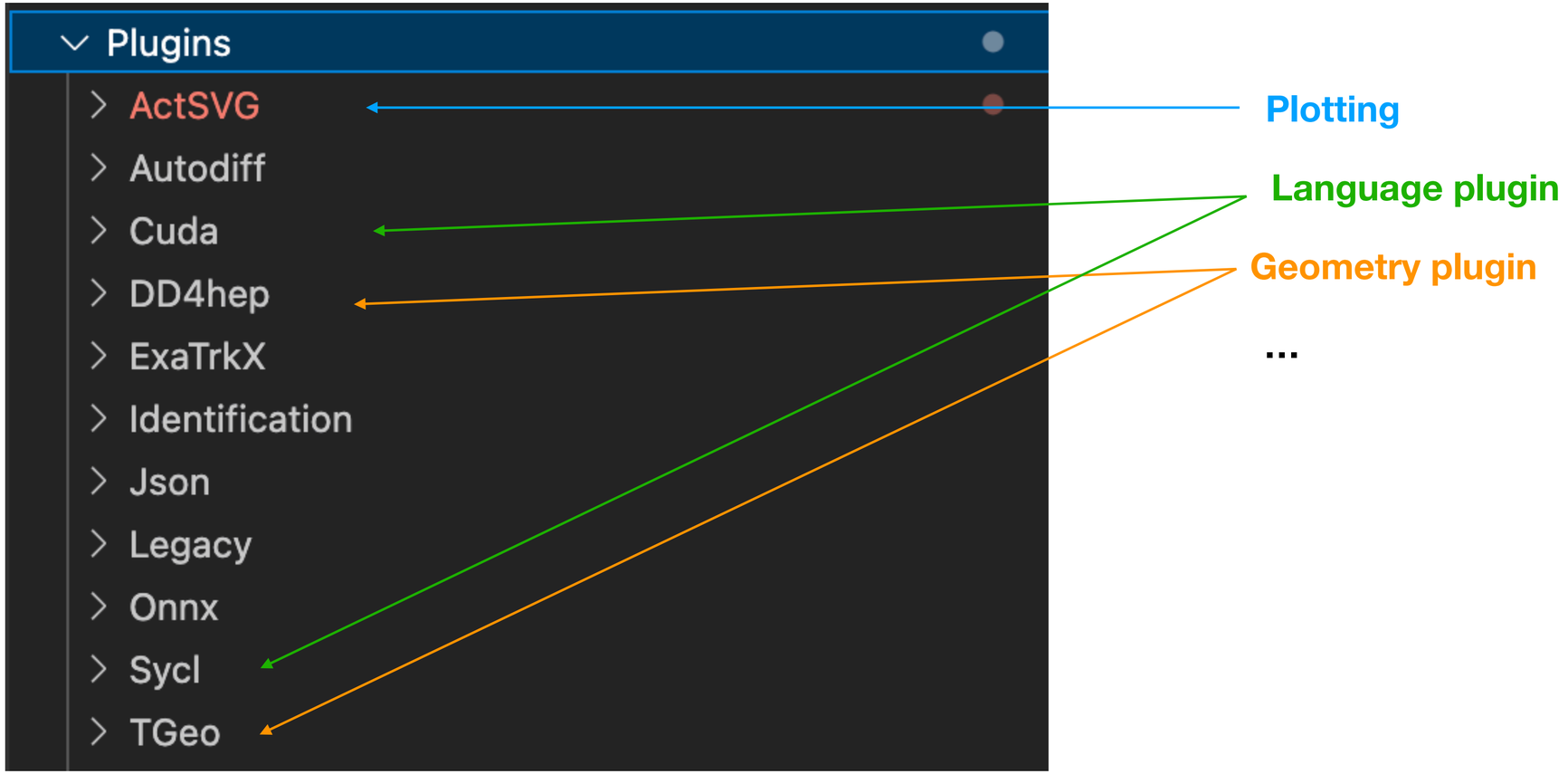
- track & vertex reconstruction components that allow for assembling of a track reconstruction applications for different experimental setups

	Geometry & Event Data Model *31/3-004 - IT Amphitheatre, CERN*	*Andreas Salzburger* 13:30 - 13:45
	Propagation *31/3-004 - IT Amphitheatre, CERN*	*Andreas Salzburger* 13:45 - 14:00
14:00	**Seeding & Pattern** *31/3-004 - IT Amphitheatre, CERN*	*Luis Falda Coelho et al.* 14:00 - 14:30
	Fitting *31/3-004 - IT Amphitheatre, CERN*	*Alexander J Pfleger et al.* 14:30 - 15:00
15:00	**Vertexing** *31/3-004 - IT Amphitheatre, CERN*	*Rocky Bala Garg* 15:00 - 15:20
	Fatras *31/3-004 - IT Amphitheatre, CERN*	*Andreas Salzburger* 15:20 - 15:30

Core: the flagship project

Plugin mechanism

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



Core concepts: multi threading and contextuatlity

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 };
    /// @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 contextuatlity

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; ///< configure this module
   };

   /// @struct Options for this module, changeable on call
    struct Options {
       ActsScalar callParameter; ///< how the horse feels today
   };

   /// @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

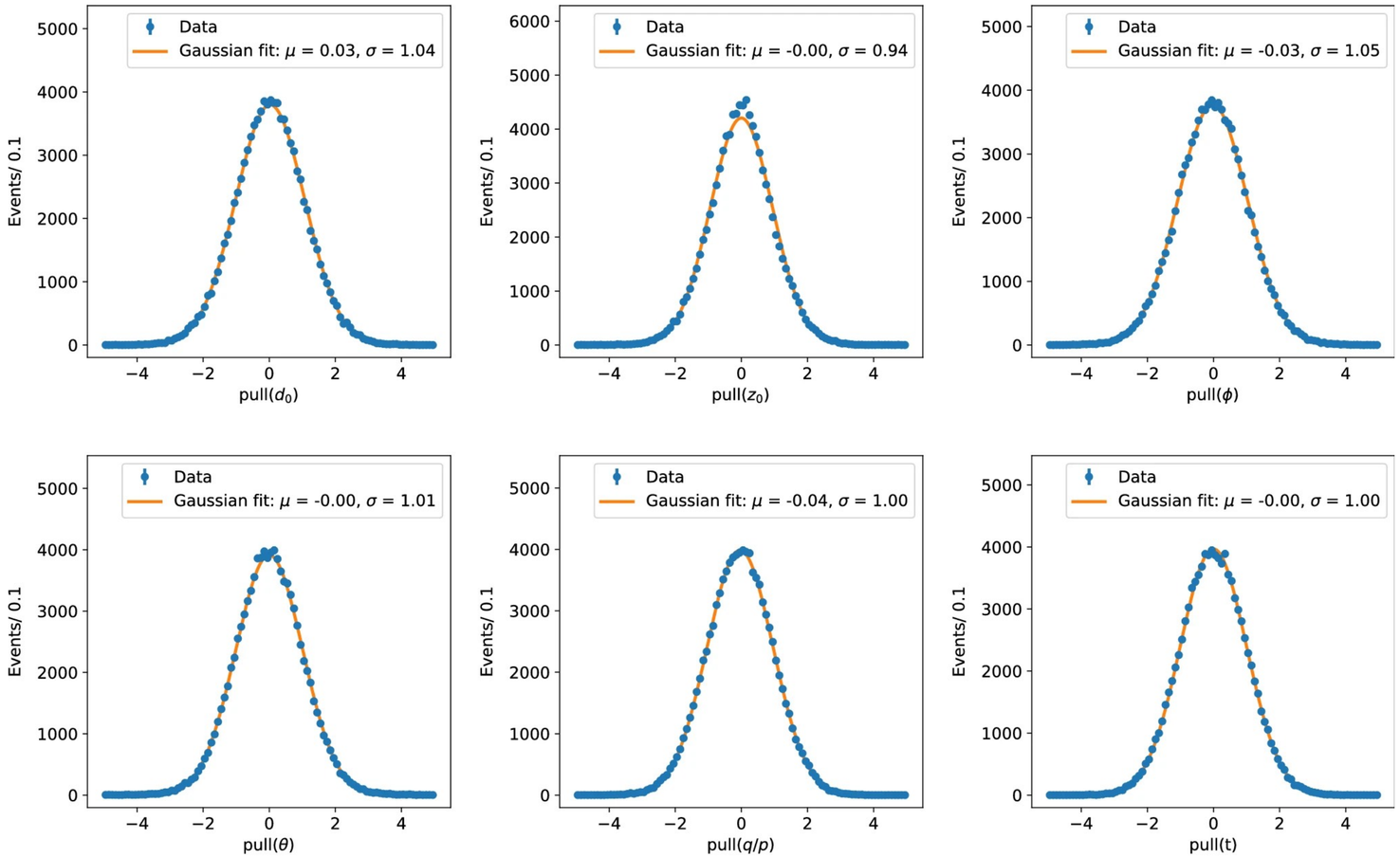
```
namespace Acts {
  /// doxygen documentation
  class WorkHorse {
   /// @struct Config for To
    struct Config {
       ActsScalar coatColor; ///< configure the coat color
       ActsScalar maxPath;    ///< set the max path this horse can run
    };
  };
}
```

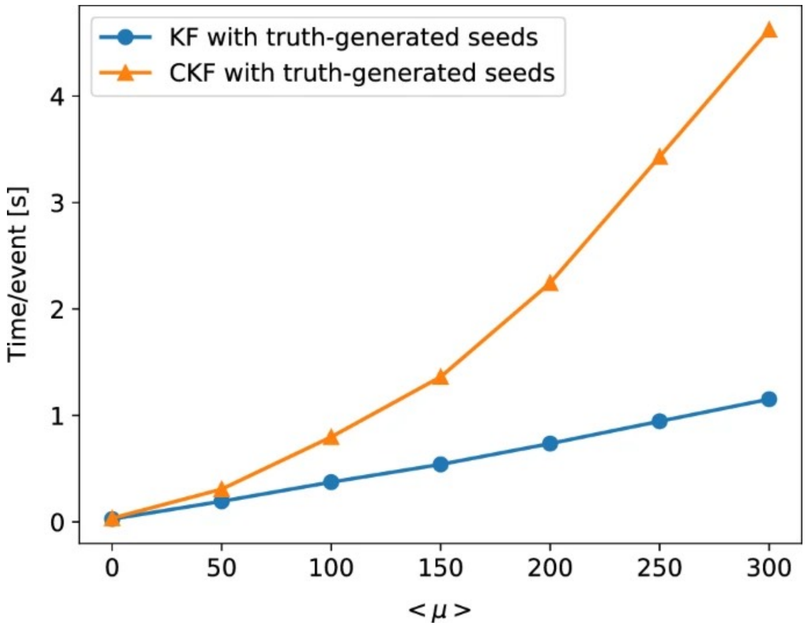
Connection to experiment framework, e.g. Gaudi/Athena

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

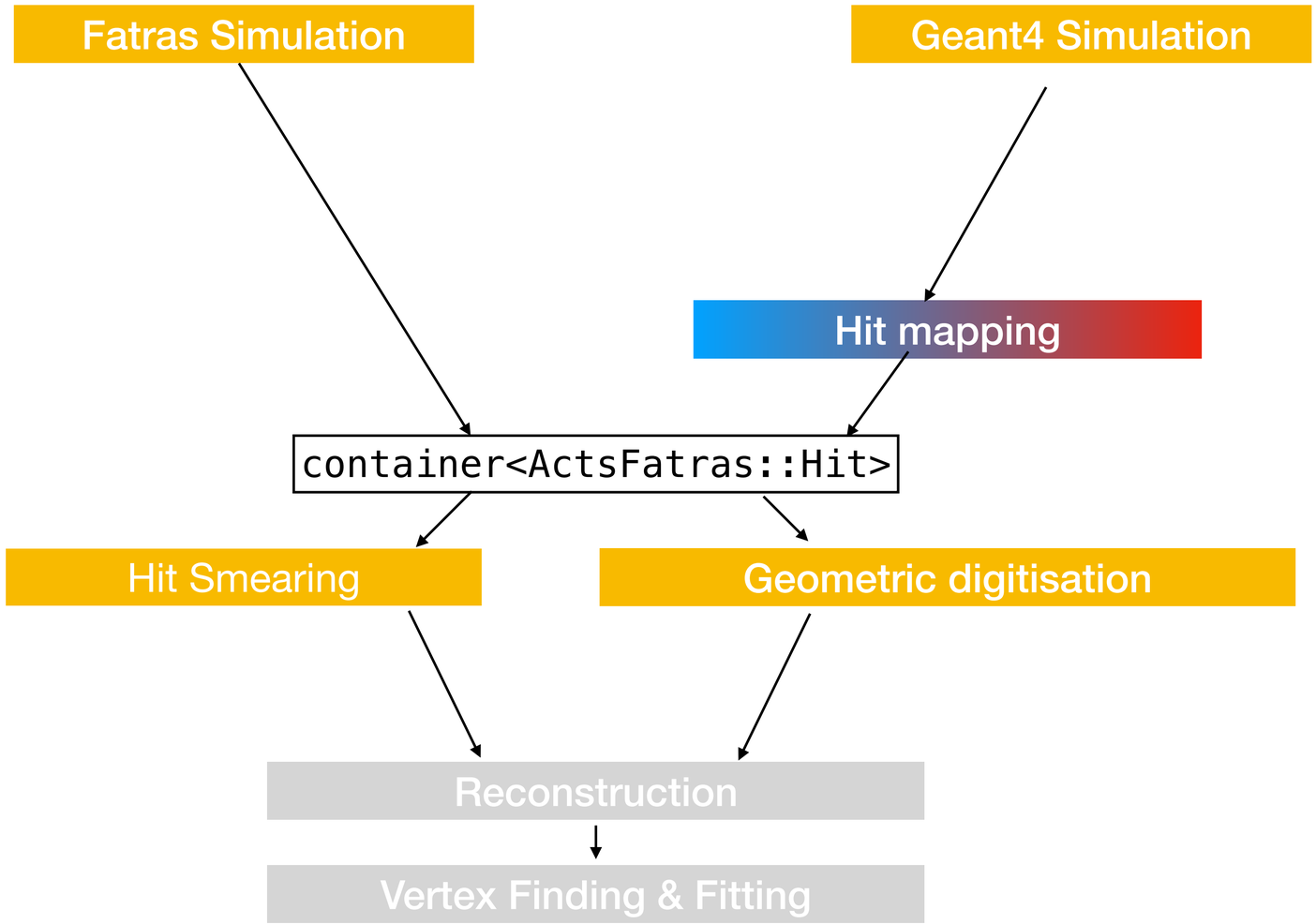
Core functionality:

A first full chain documented using the Open Data Detector / TrackML detector in:
https://link.springer.com/article/10.1007/s41781-021-00078-8

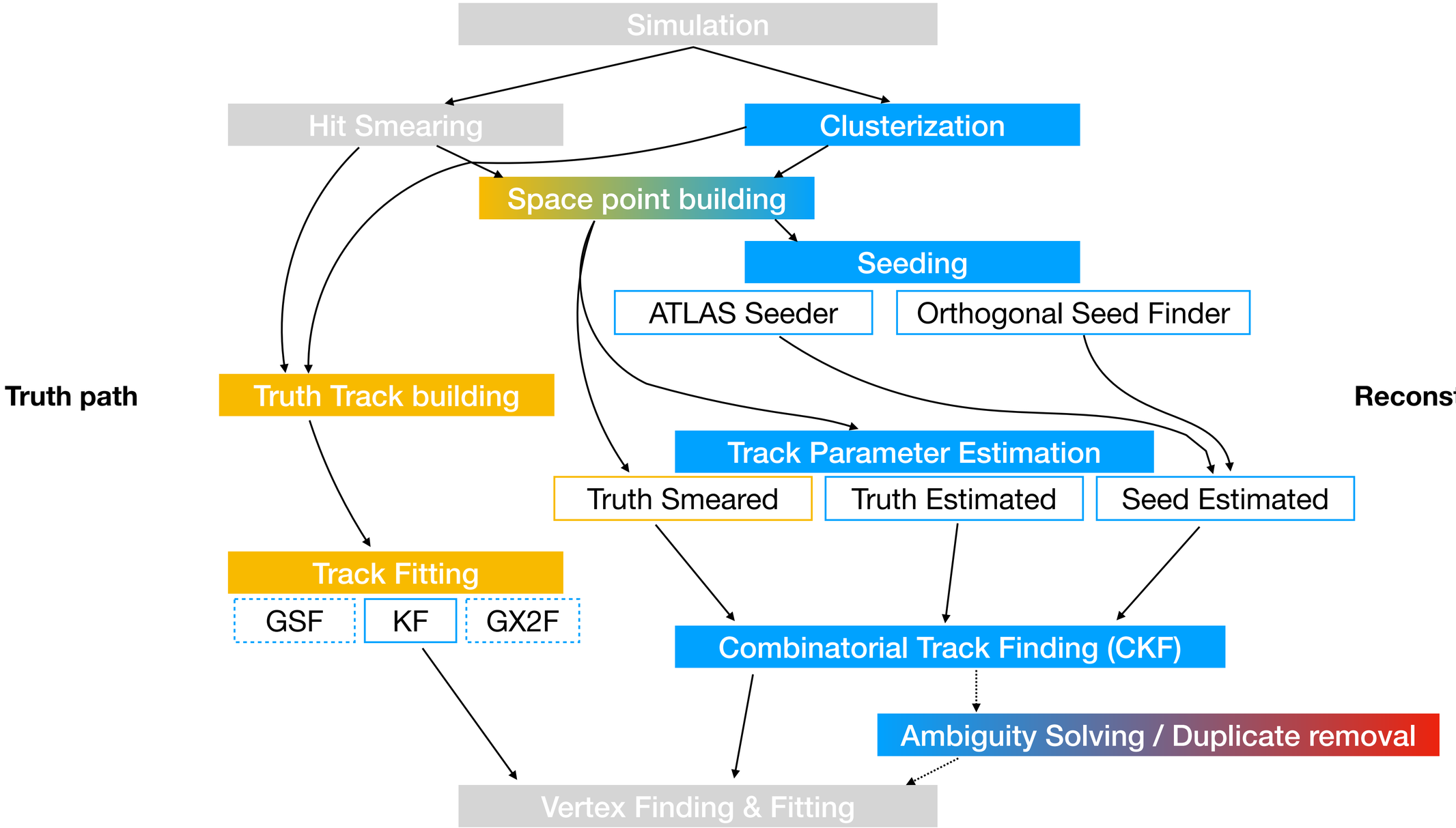




Core functionality: simulation input



Core functionality: a tracking demonstrator chain

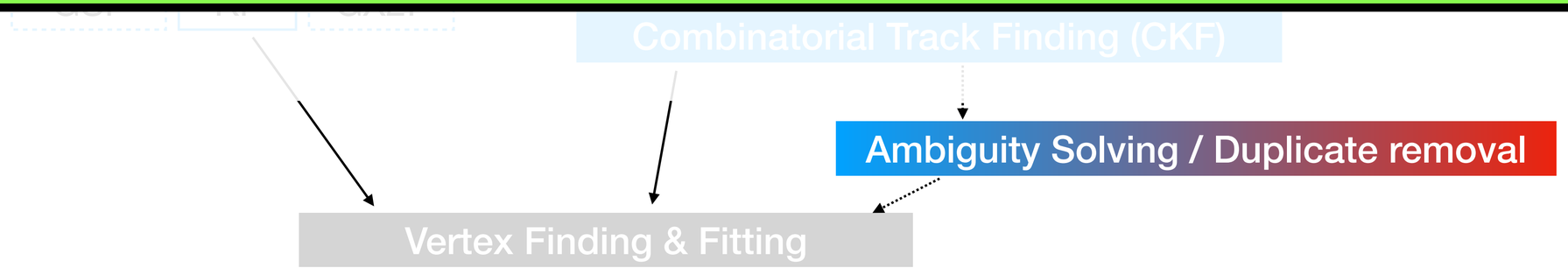


Core functionality: a tracking demonstrator chain

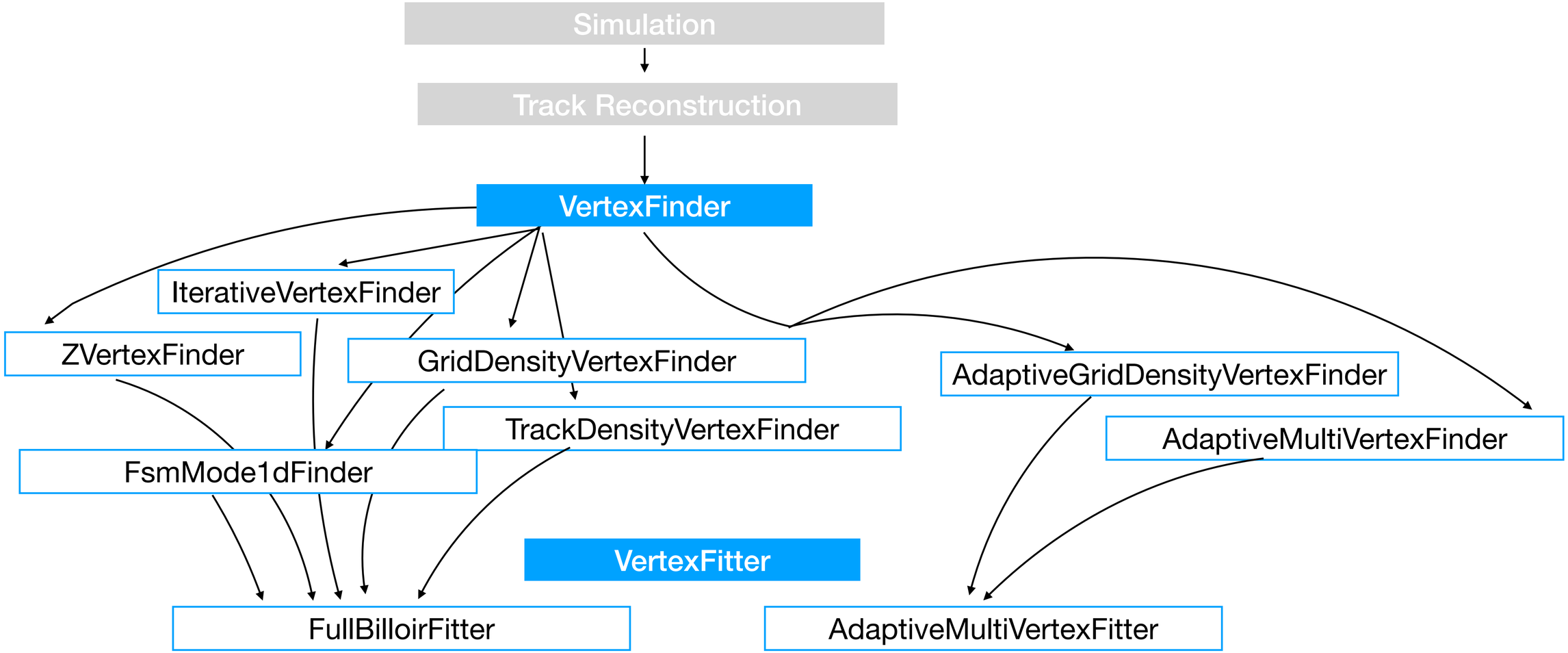
Development Proposal I:

Ambiguity solver and/or duplicate removal is not yet fully covered, is often experiment specific, however, some baseline modules can be done in a generic way

- TrackML scoring based ambiguity solver
- Simple duplicate resolving module
- ML based ambiguity solver exists



Core functionality: vertex reconstruction



Core functionality: vertex reconstruction

Development Proposal:

Track linearisation is currently done using a helical track model

Prototype work from B. Schlag to generatlize this using the propagator

- great development project with immediate client impact (e.g. secondary vertex reconstruction)

Generalization of track linearization using the ACTS::Propagator:

- No assumption of helical track parameters anymore
- Vertex fitter more robust in all detector regions
- Harmonize primary and secondary vertexing with common math kernels
- Fully integrated time propagation in ACTS Vertex fitting with time information possible

$$\vec{q} = \vec{q}(\vec{r}, \vec{p}) = A\vec{r} + B\vec{p} + \vec{c}_0$$

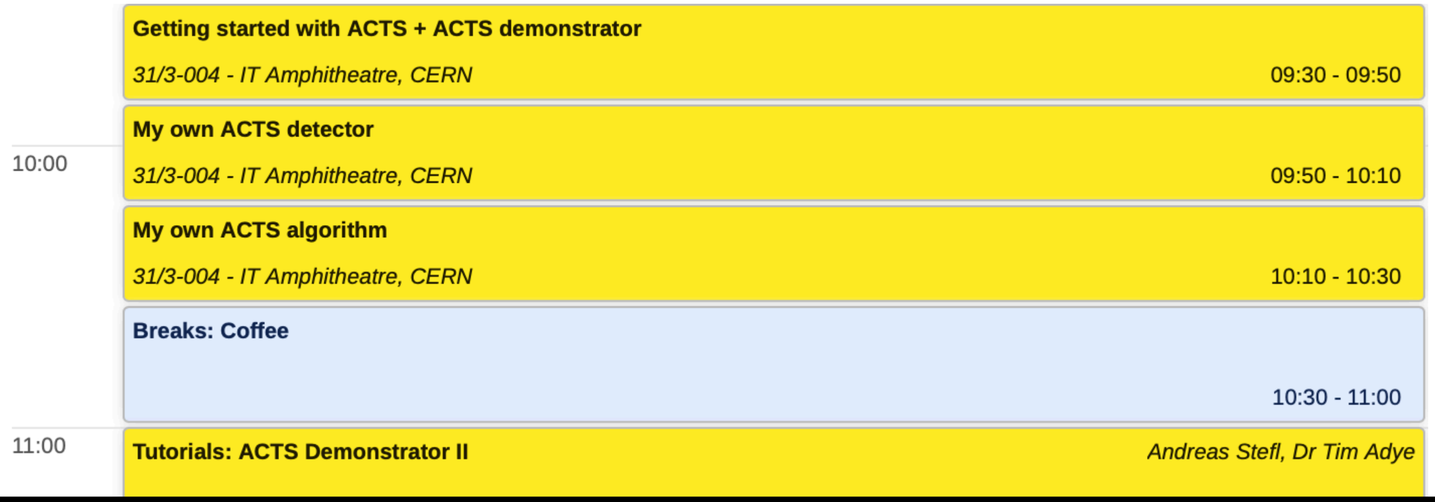
Retrieve dedicated Jacobians from ACTS::Propagator

B. Schlag

Core: example framework & tutorials

Core ships with an **example framework**

- event-parallel framework (based on TBB) with sequencer that holds algorithm chain
- allows to build demonstrator chains, not built as a production framework
- steered with python (recent python bindings, binary examples to be dropped soon)
- recently partly re-built with on top of Gaudi



Wednesday

Development Chain:

Relatively often modules, algorithms are prototyped within the framework and then promoted to the Core library when successful.

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 Diff			@@
##	main	#1551	+/-	##
- Coverage	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		

```cpp
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: testing

Integration testing is another important aspect

- Larger scale tests of code in a quasi realistic environment
- Full chain demonstrator using the ODD as a benchmark

Point of Attention

Some explicit testing of experiment applications would be sometime useful

- Particularly problematic if access to resources are restricted
- Some tests can be abstracted, e.g. by providing generalised input data

Particularly for debugging applications with experts from ACTS we need to find a way to share/give access to the application (or at least problem)

Core: contributing

ACTS is Open Source and invites contributions, corrections, interactions

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

Clone:

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

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

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

Discuss at the open develops meeting

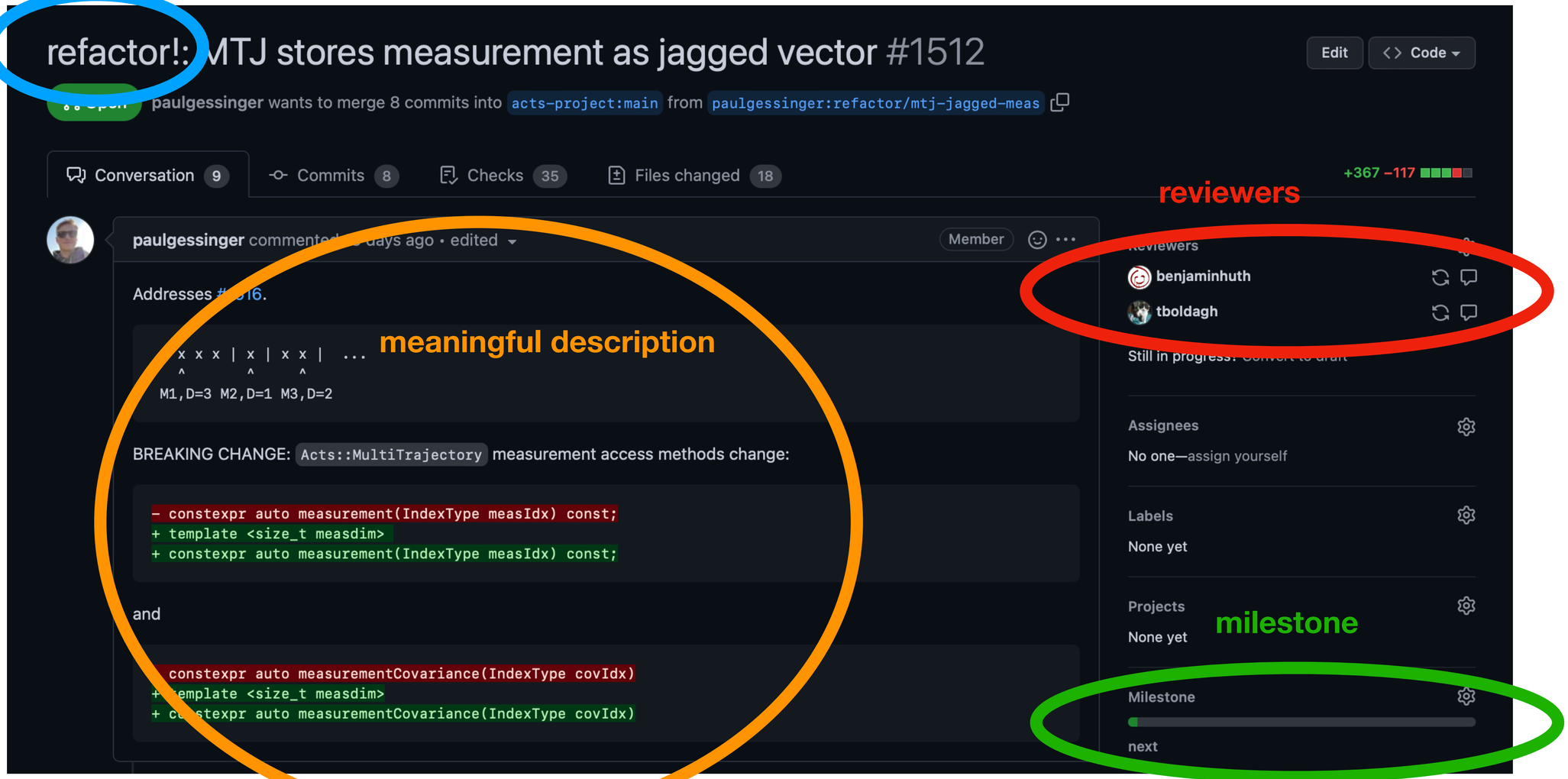
https://indico.cern.ch/category/7968/

Tuesday 17:00, CE(S)T

Core: contributing

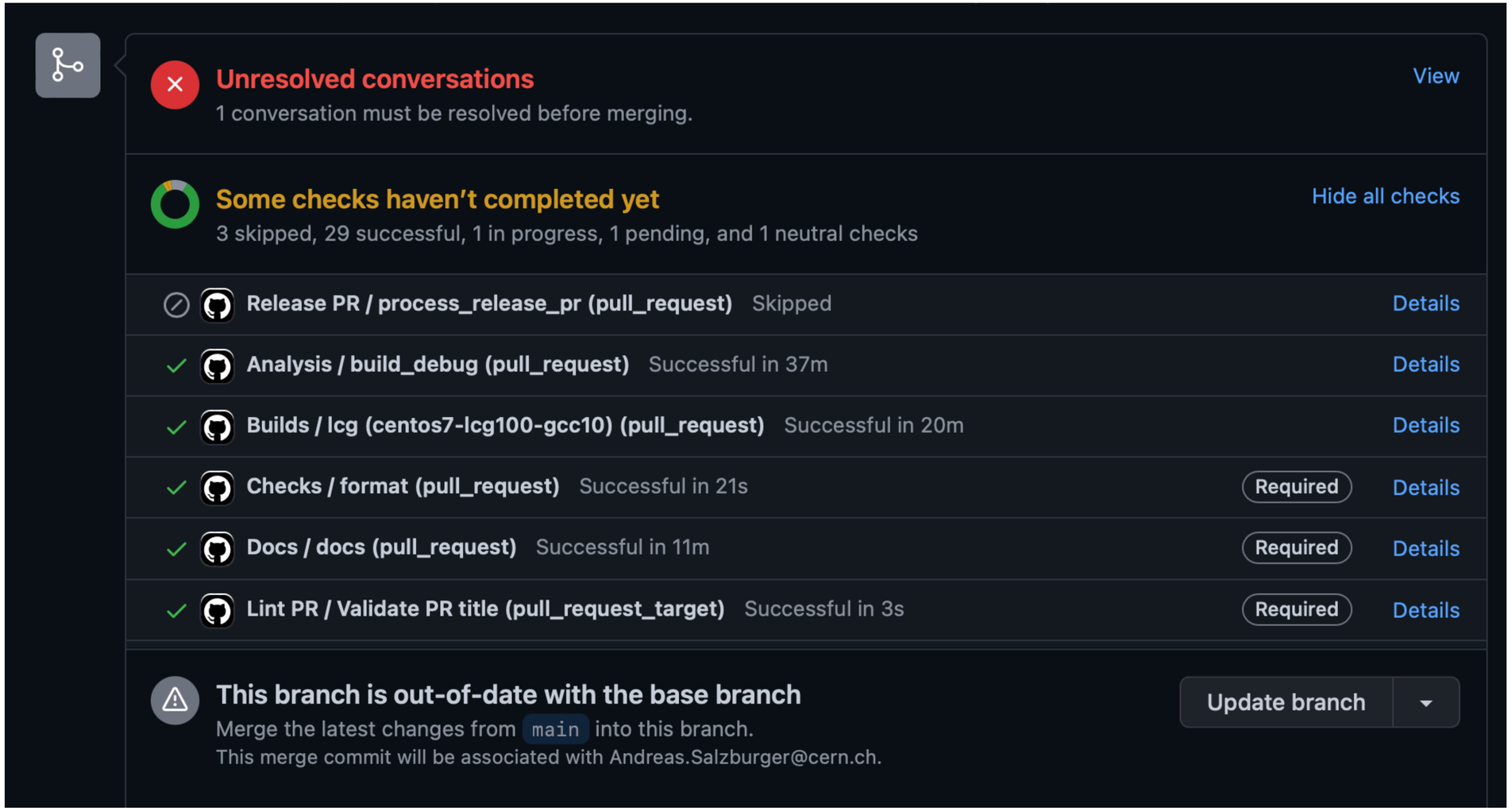
Pull requests come with a template that guides through a proper submission

semantic naming: feat, doc, refactor, fix



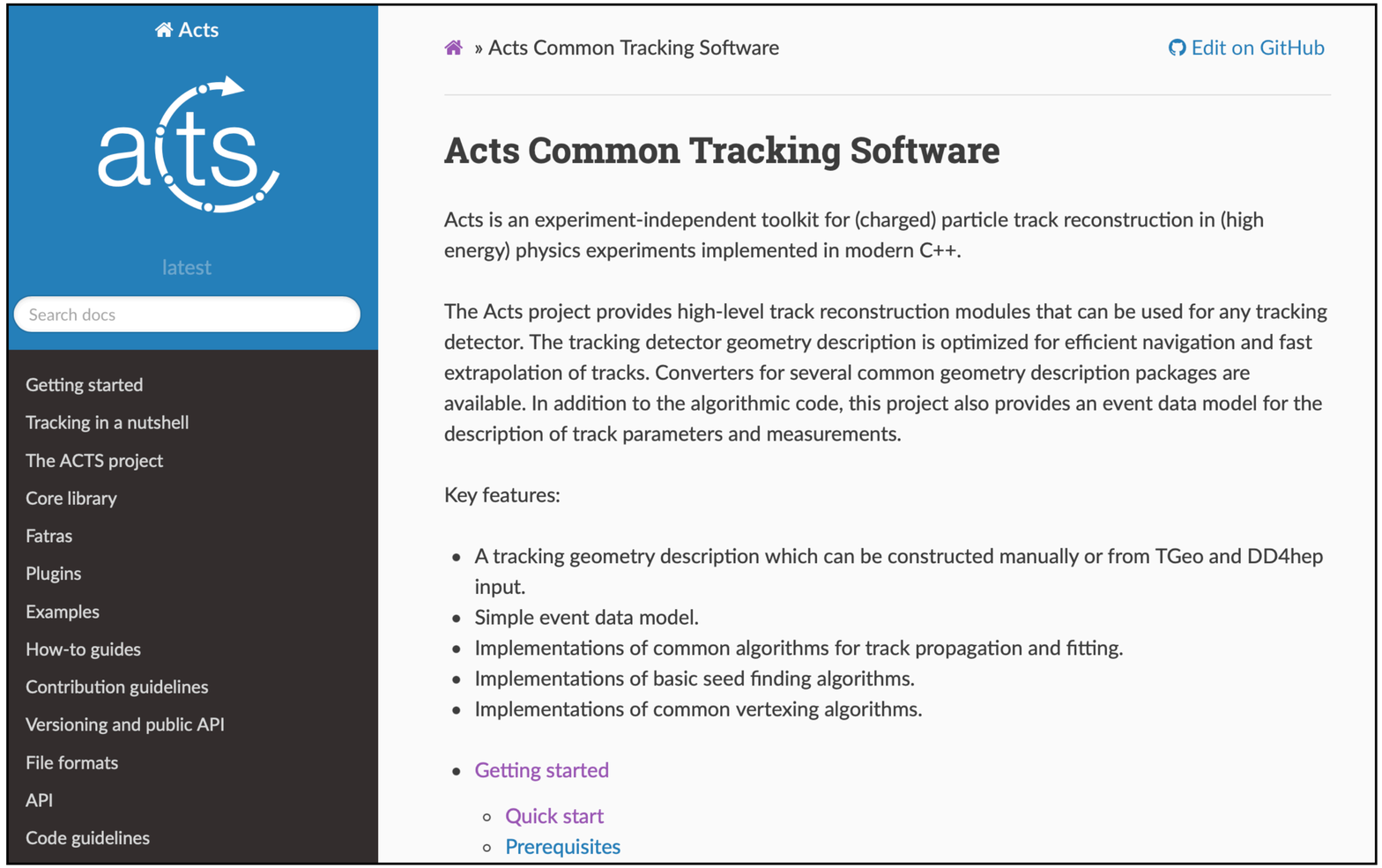
Core: contributing & testing

Pull requests run through a CI pipeline



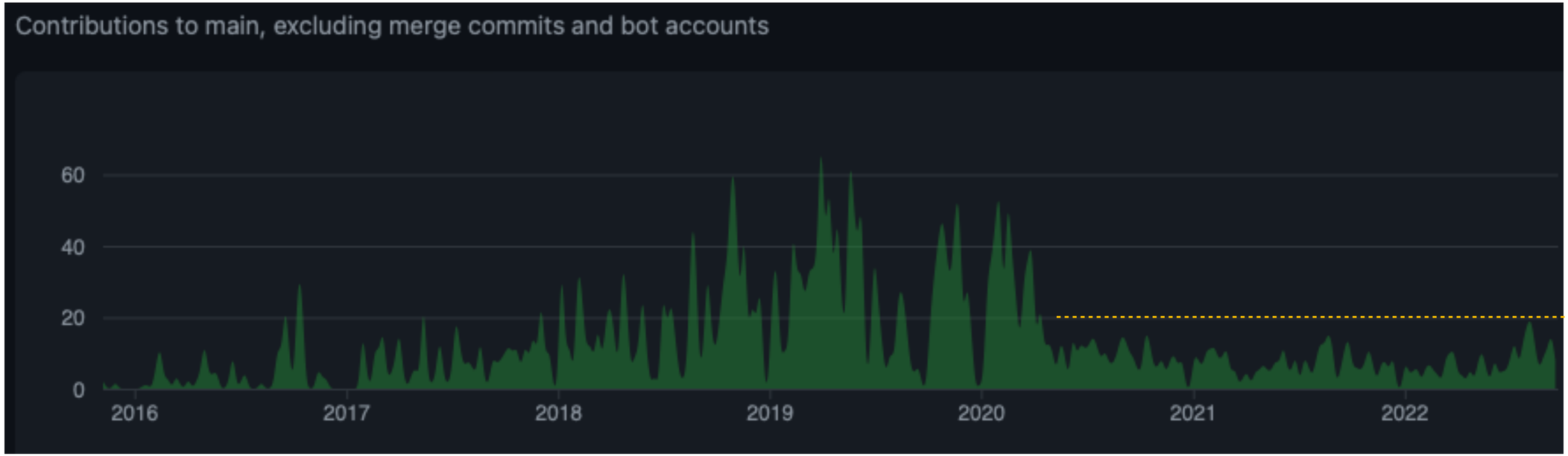
Core: documentation

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



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

Core: person power situation, developers & support



Point of Attention

Peak commit period is passed (a lot of import of core components from 2018-2020)

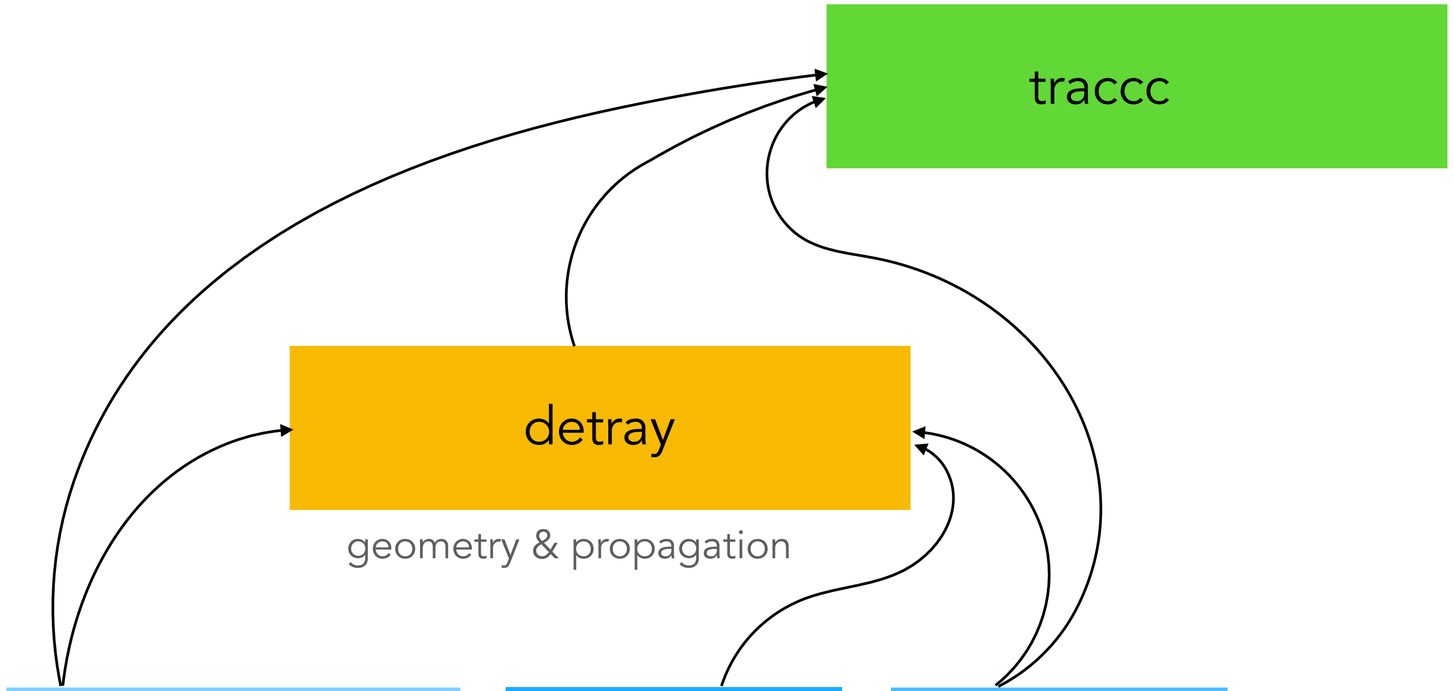
- Library still has missing parts
- Some components miss direct support (e.g. vertex reconstruction)

R&D1

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

encapsulation of algebra operation backend

generic vector field library

stl-style containers with managed memory

EP-R&D
Programme on Technologies for Future Experiments

Goal is to establish a track reconstruction chain without algorithmic compromises. HSF summary talk can be found **[here]**.

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
cmath	✅	✅	✅
Eigen	✅	✅	🟡
SMatrix	✅	⚪	⚪
VC	✅	⚪	⚪

✅: natively supported
🟡: natively supported, but not tested
⚪: no support

vecmem is a generic library which can be of use to a broader community use, **algebra-plugins** is not designed for production.

R&D1: track reconstruction: covfie

A generic vector field library based on composition design

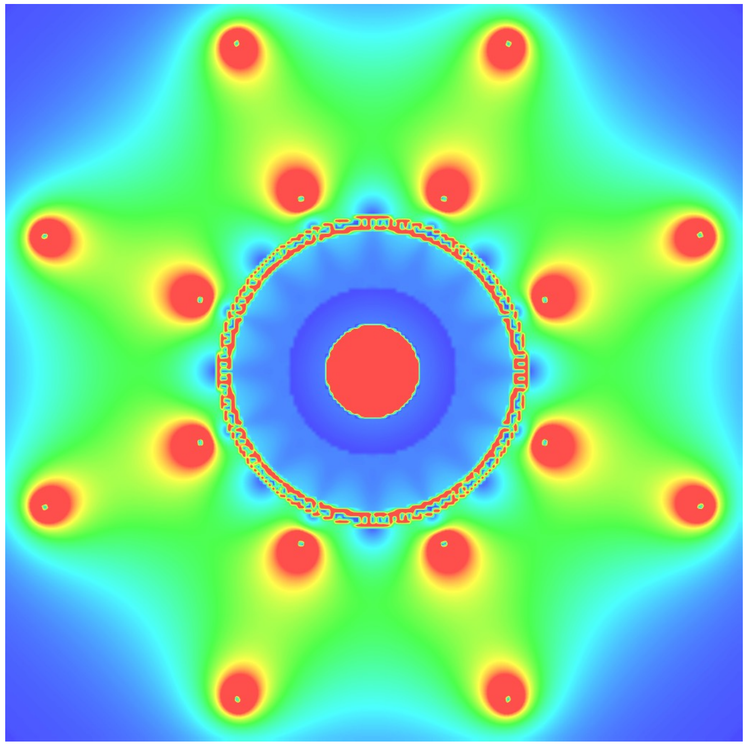
- format, coordinate transform and storage at compile time

```
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

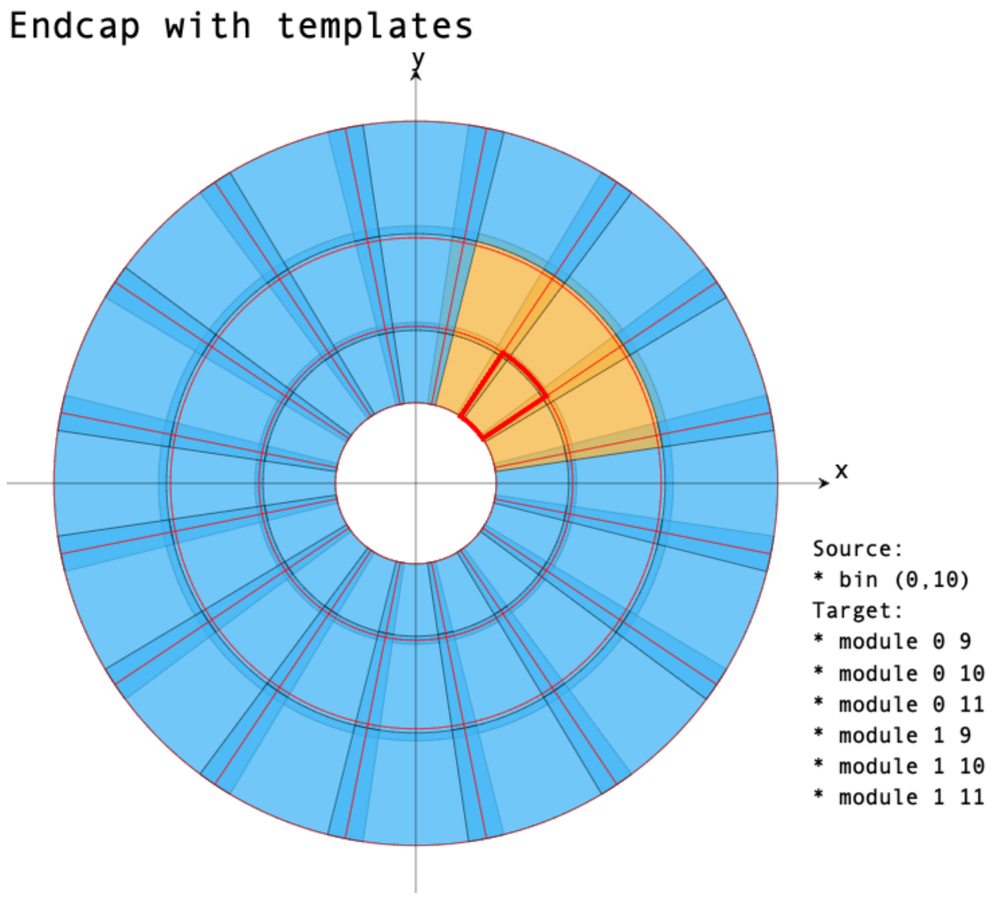
covfie is a generic library which can be of use to a broader community, e.g. full simulation on accelerators.

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 {
```



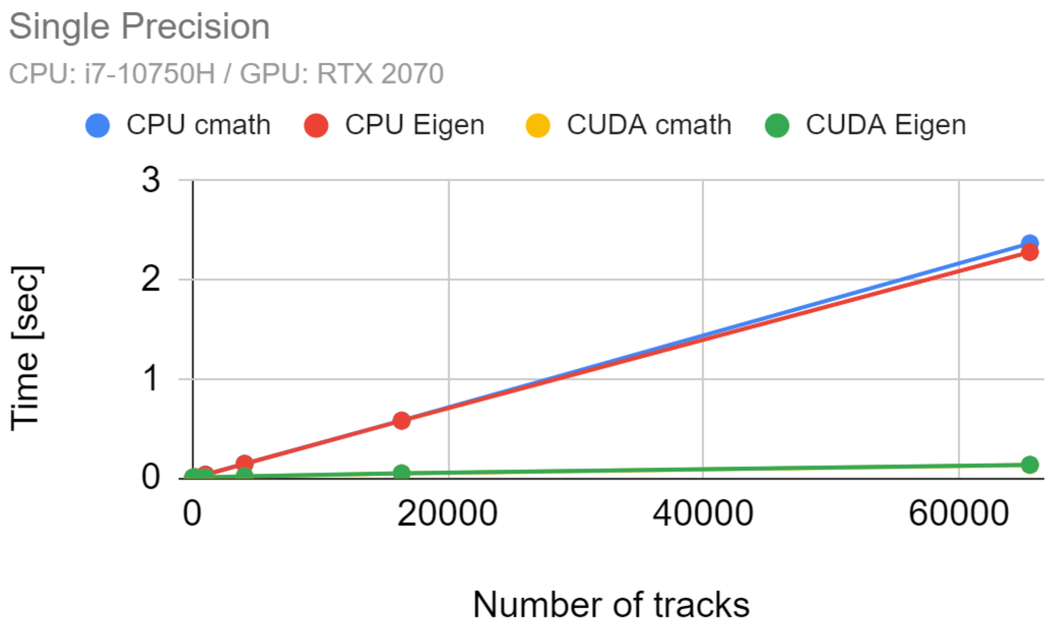
Local navigation resolving by grid binning in detray

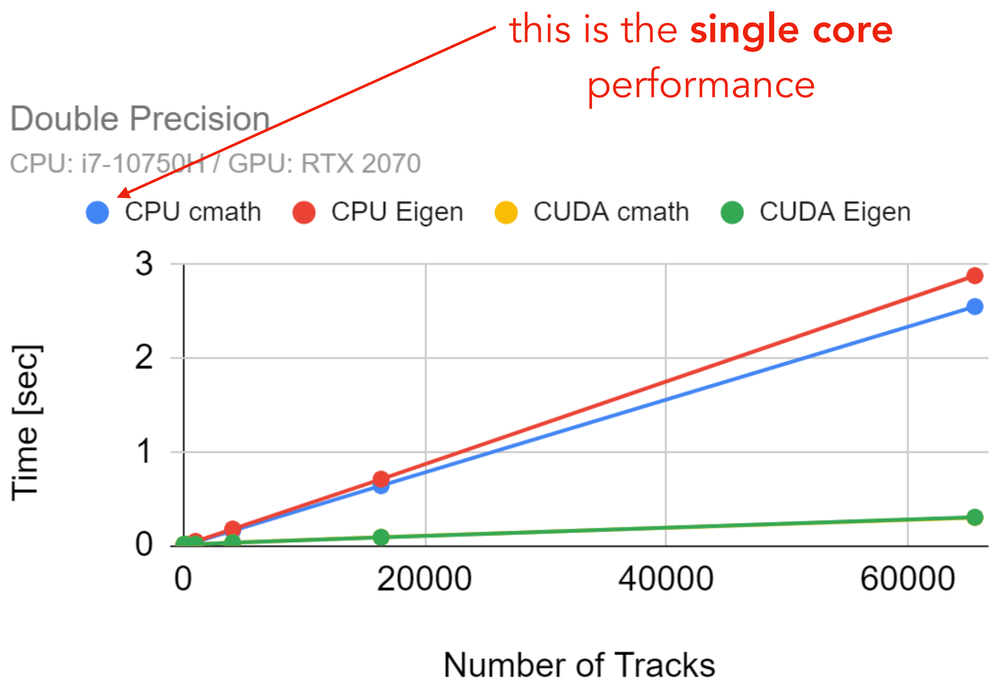
detray & VecGeom developers are already in contact and initial exchange, with plenty of room for more collaboration.

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

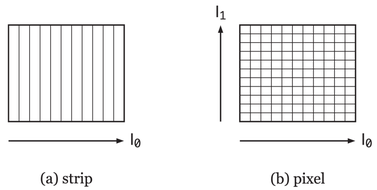




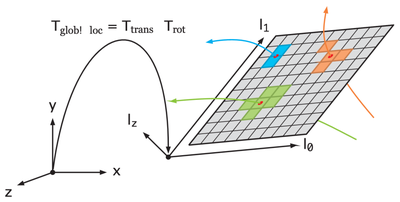
detray adaptive runge-kutta transport code is relatively self-contained and could serve more communities.

R&D1: traccc (1)

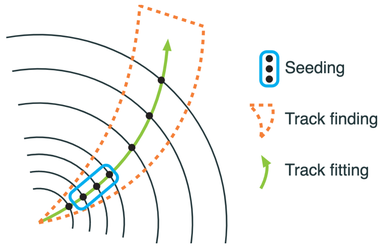
Full chain demonstrator for track reconstruction on CPU/GPU



Clustering



Space Point formation



Track finding & fitting

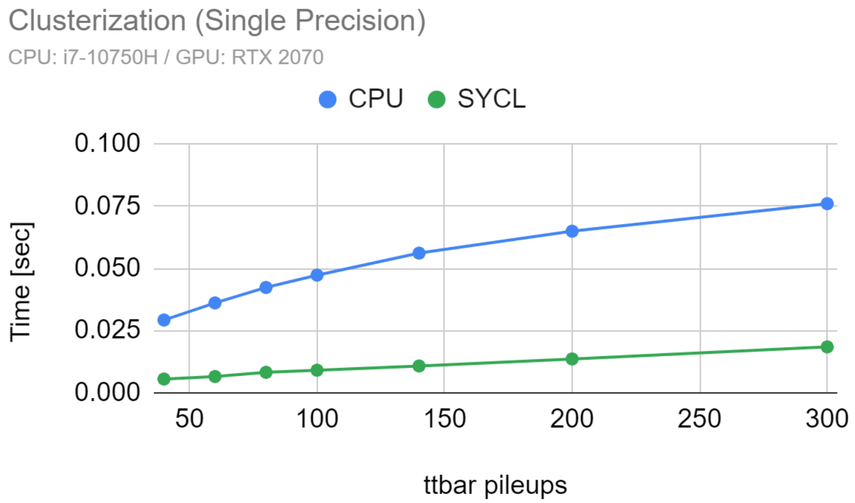
Algorithms	CPU	CUDA	SYCL	std::par
CCL	✅	✅	✅	🟡
Measurement creation	✅	✅	✅	⚪
Spacepoint formation	✅	✅	✅	⚪
Spacepoint binning	✅	✅	✅	⚪
Seed finding	✅	✅	✅	⚪
Track param estimation	✅	✅	✅	⚪
Combinatorial KF	⚪	⚪	⚪	⚪
KF	🟡	🟡	⚪	⚪

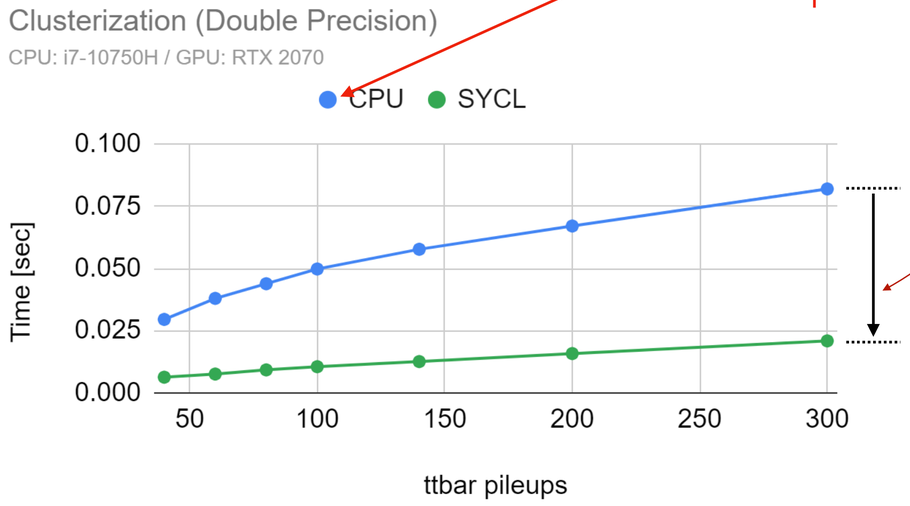
exists 🟡: work started ⚪: work not yet started

traccc is an ideal playground to gather/share/exchange knowledge about code for heterogeneous systems.

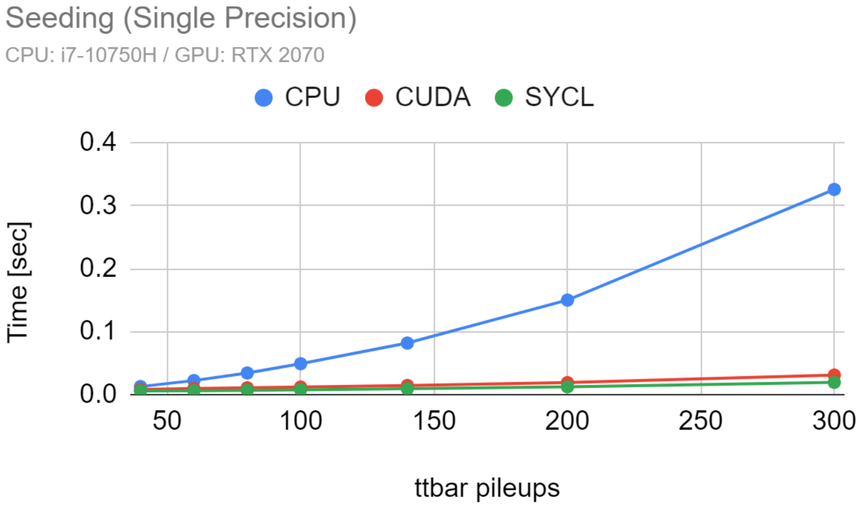
R&D1n: traccc (2)

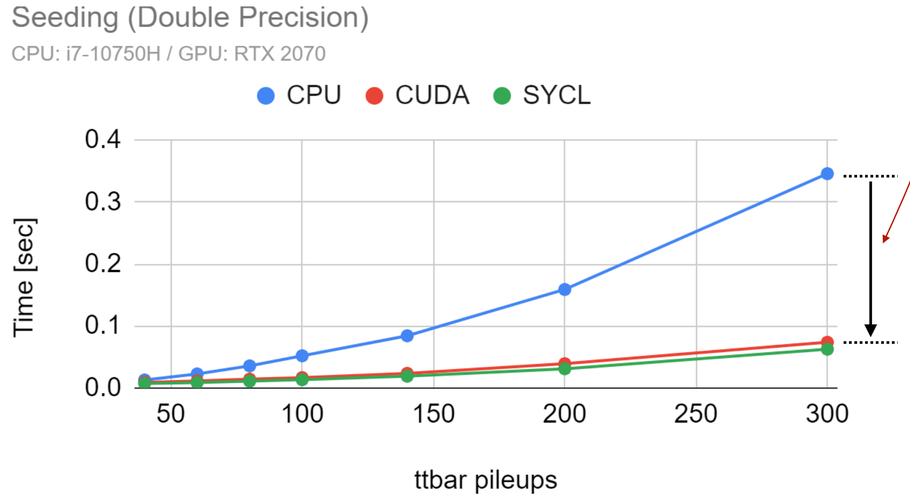
Full chain demonstrator for track reconstruction on CPU/GPU





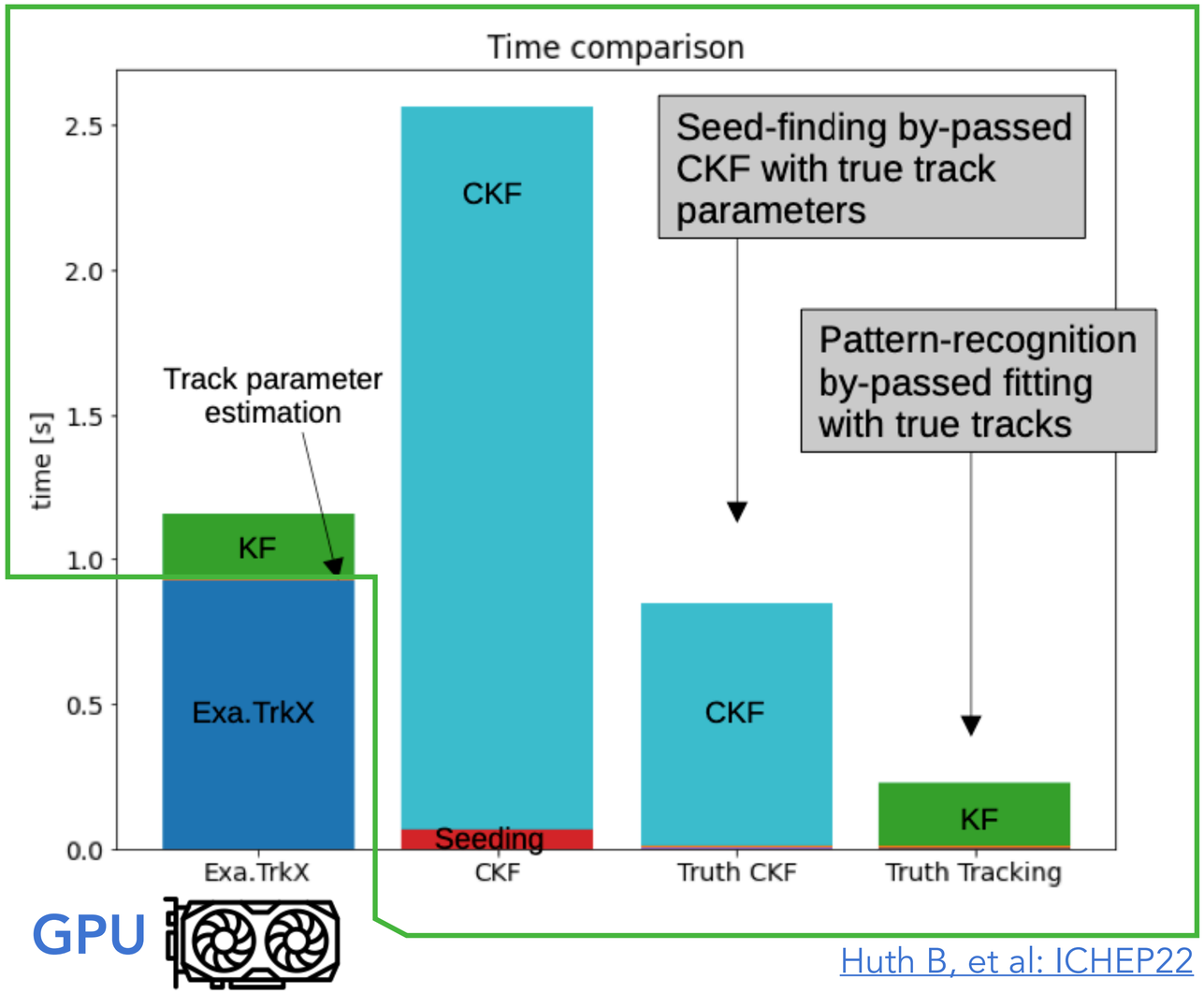
this is the **single core** performance





Q: is this enough gain for re-writing our code for GPUs?

R&D1: technology BINGO

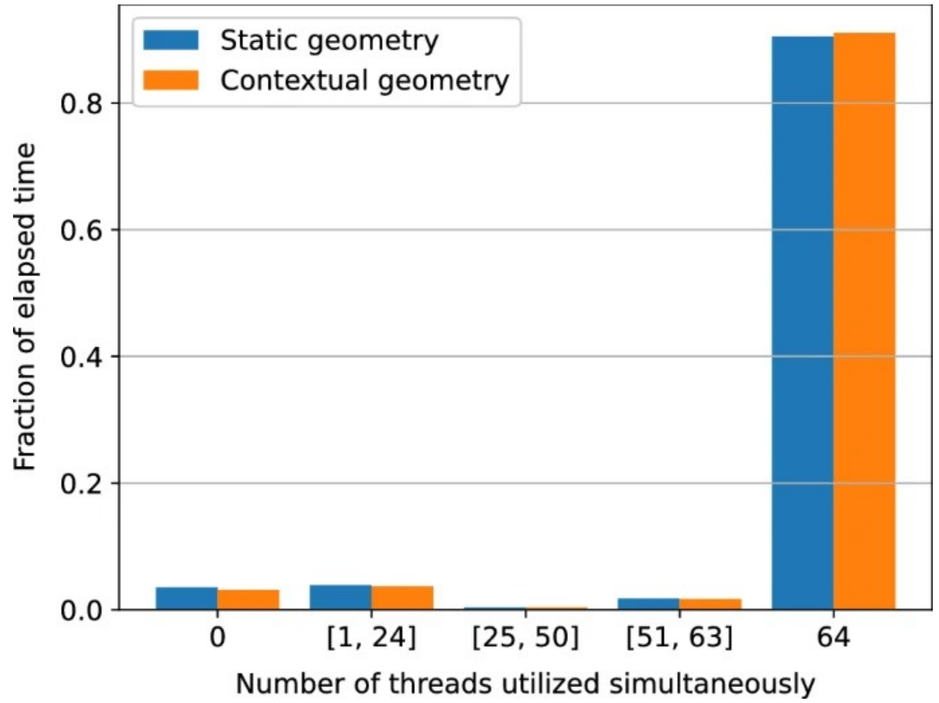


Huth B, et al: ICHEP22

CPU (single threaded)

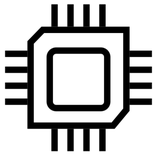
Can we run large-scale multi threaded?

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

CPU  GPU 

acts-project/acts	x86	aarch64	oneAPI/SYCL	CUDA
Core Line	tested	tested	superseded	superseded
	tested (incomplete*)	tested (incomplete*)	tested (incomplete*)	tested (incomplete*)
R&D Line 2 "machine learning"	tested	not tested	not implemented	tested w x86

acts-project/traccc
acts-project/detray
acts-project/covfie
acts-project/vecmem

R&D Line 1 "parallelization"

exatrkx & acts-project/acts

R&D1: presentations & tutorial

Tuesday

Status & Plans: algebra-plugins, detray — *Beom Ki Yeo*
31/3-004 - IT Amphitheatre, CERN — 15:30 - 15:50

16:00

Status & Plans: covfie — *Mr Stephen Nicholas Swatman*
31/3-004 - IT Amphitheatre, CERN — 15:55 - 16:05

Status & Plans: traccc — *Guilherme Metelo Rita De Almeida*
31/3-004 - IT Amphitheatre, CERN — 16:20 - 16:40

Discussion: (Event) Data Model
31/3-004 - IT Amphitheatre, CERN — 16:50 - 17:10

17:00

Discussion: traccc -> Acts(Core)
31/3-004 - IT Amphitheatre, CERN

Wednesday

16:00

Tutorials: traccc, detray & friends — *Beomki Yeo, Joana Niermann, Mr Stephen Nicholas Swatman*
31/3-004 - IT Amphitheatre, CERN — 15:30 - 17:30

17:00

R&D2

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

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

R&D2: presentations & tutorial

Tuesday

Auto-tuning in acts for seeding and vertexing — *Rocky Bala Garg*
31/3-004 - IT Amphitheatre, CERN — 13:30 - 13:50

14:00

Auto-tuning of the Acts material mapping with Orion — *Corentin Allaire*
31/3-004 - IT Amphitheatre, CERN — 13:55 - 14:15

Tracking with Hashing in ACTS — *Dr Jessica Leveque*
31/3-004 - IT Amphitheatre, CERN — 14:20 - 14:40

Exatrkx-ACTS integration & GNN applications — *Daniel Thomas Murnane*
31/3-004 - IT Amphitheatre, CERN — 14:45 - 14:55

15:00

Wednesday

Using Optuna to tune seeding and vertexing parameters — *Rocky Bala Garg*
31/3-004 - IT Amphitheatre, CERN — 13:30 - 14:00

14:00

Using Orion to tune the material mapping — *Corentin Allaire*
31/3-004 - IT Amphitheatre, CERN — 14:00 - 14:30

Using the Exa.TrkX GNN in Acts — *Benjamin Huth*
31/3-004 - IT Amphitheatre, CERN — 14:30 - 15:00

Development and R&D, add-ons:

Core

acts-developers@cern.ch

CPU multi-threaded library of
tracking reconstruction components

Add-ons

OpenDataDetector
ActSVG

R&D1

acts-parallelization@cern.ch

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

R&D2

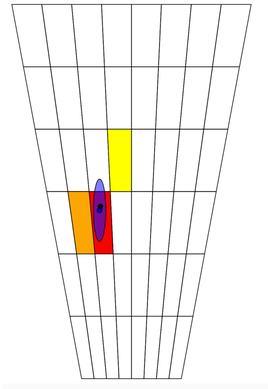
acts-machinelearning@cern.ch

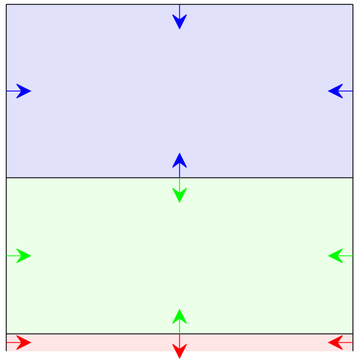
Machine learning and ML assisted
modules for track reconstruction

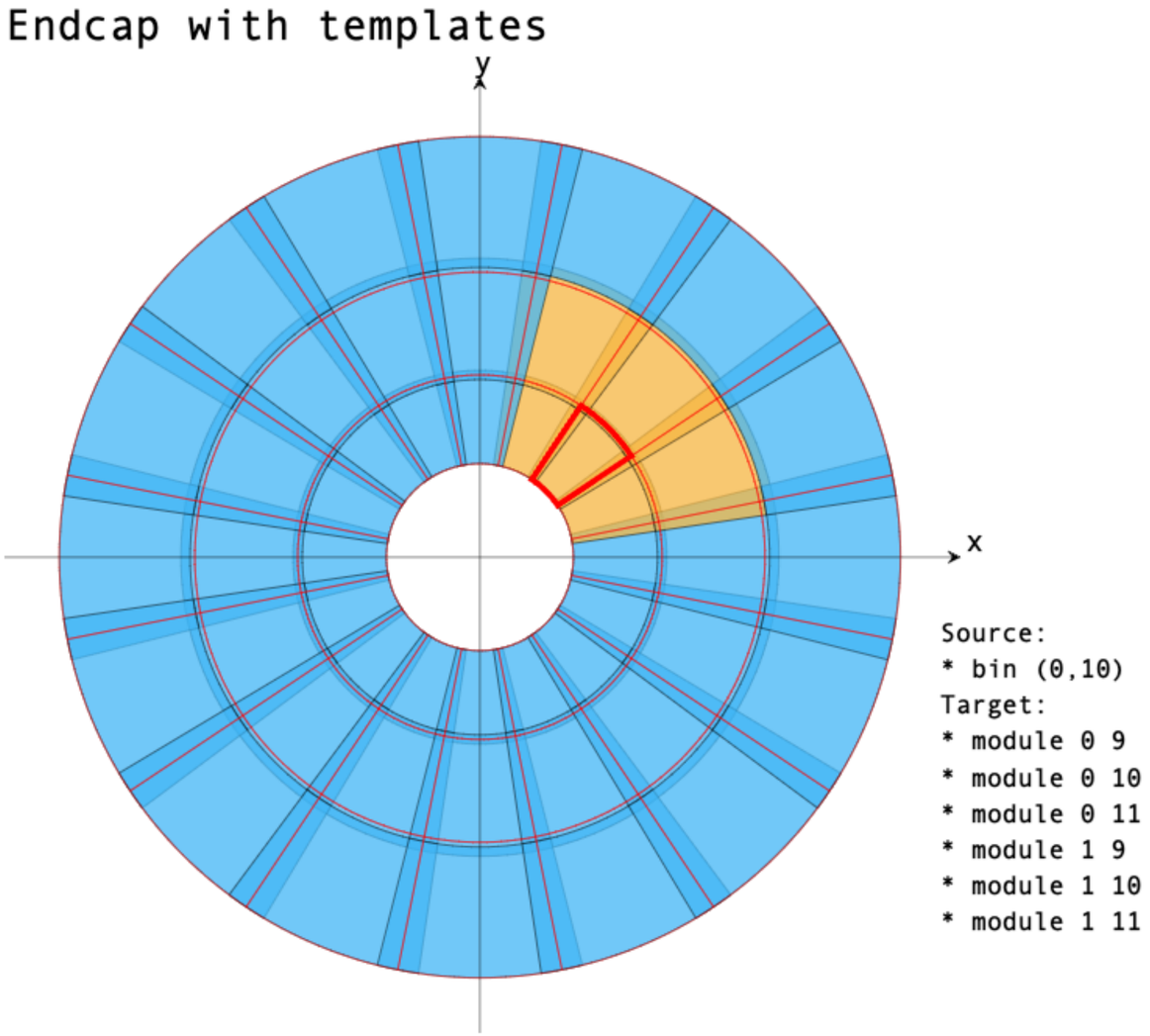
Plotting: actsvg

2D plotting library dedicated for tracking

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







actsvg is a generic plotting library.

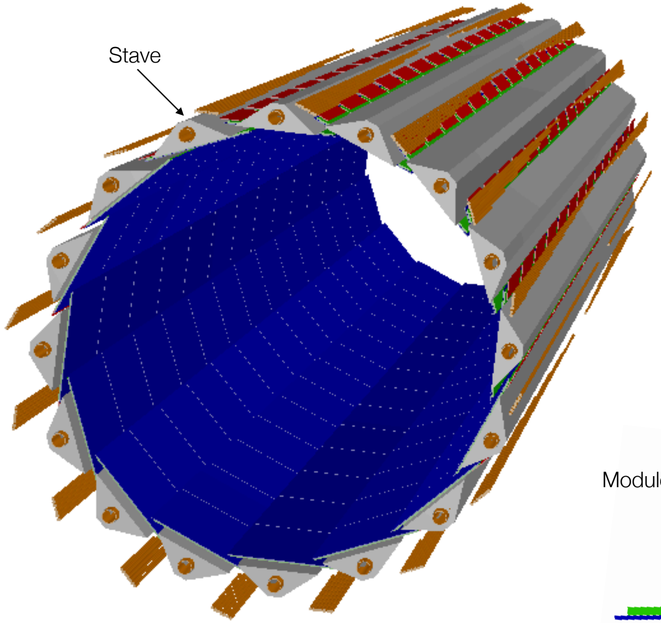
Community: Open Data Detector & key4hep

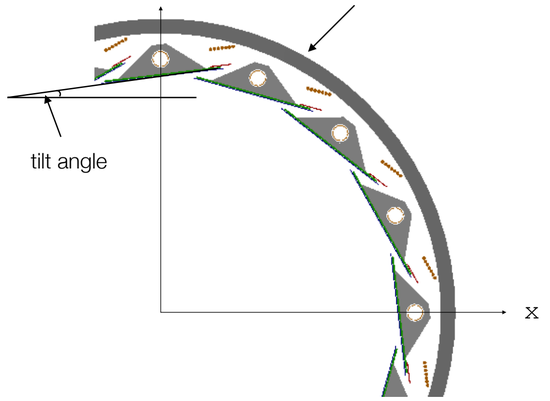
Evolution of TrackML detector

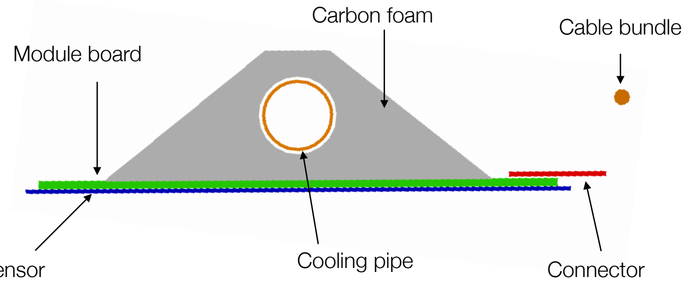
- Re-implemented in DD4Hep to enable full/fast simulation
- Quasi-realistic feedback to allow real-life scenario testing of algorithms
- Supports TrackML output format through ACTS binding (work ongoing to also support edm4hep)

ACTS integration into key4hep SW stack

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







Ongoing activity to include first Calorimeter description, MS to follow.