Introduction to ACTS: status, development, and integration

Jin Zhang, Yubo Han, Hongbo Zhu, Gang li

2019/10/24, Dalian

Outline

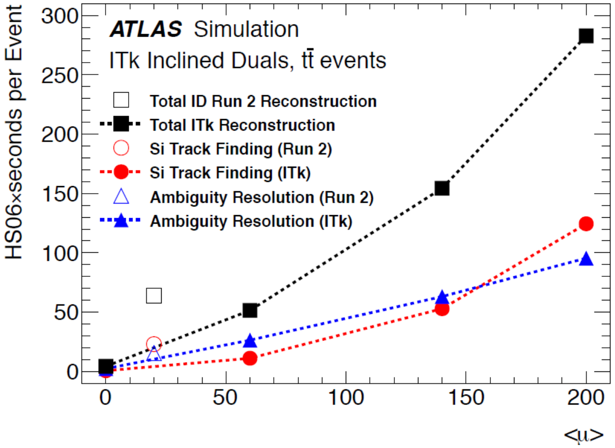
- ACTS concept and status
- Participation in ACTS
- ACTS integration to ATLAS/CEPC detector

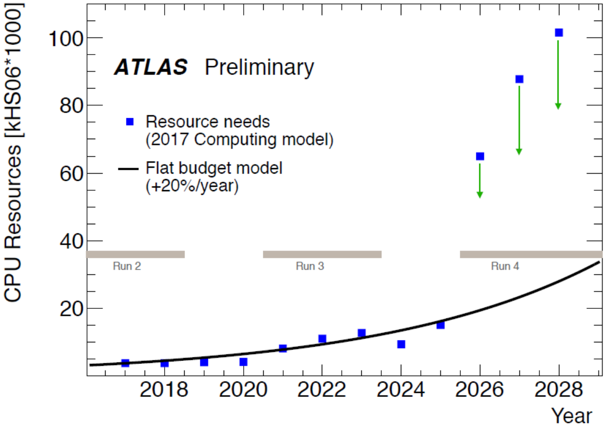
Motivations

- LHC Run-1/2 exceeded all expectations in terms of provided data
  - Design pile-up ~21 for Run-1 and ~40 for Run-2
  - Track reconstruction worked extremely well

- HL-LHC will bring great challenges to computing in track reconstruction

	LHC Run-1	LHC Run-2	LHC Run-4
muon	21	40	150-200
Tracks	~280	~600	~7-10k





Keep physics performance && Tackle computing resource problem for future LHC era

ACTS: A Common Tracking Software

ATLAS (Athena)

- Well tested code
- High tracking performance
- Not thread safe – tough task from AthenaMT
- Grown structure, not long-time maintenance

ACTS

- Encapsulate the existed code from ATLAS(currently), or other experiment
- Thread safe/long vectorization
- Modern C++ 17
- Independent from experiment and framework

Contributions come mostly from ATLAS people, but there is increasing involvement from CEPC, Belle-2, FCC-hh and more

Project Picture and Basic Design

Plugins: DD4HEP, TGEO, Geant4…

Core

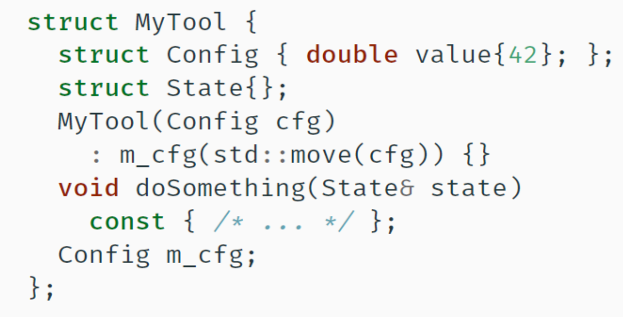
- Geometry
- Surface
- EDM
- Material
- MagneticField
- Utilities
- Propagator
  - ✓ RKN Stepper
  - ✓ StraightLine Stepper
  - ✓ Navigator
- Fitter
  - ✓ Kalman Fitter
  - Gaussian Sum Fitter
- Seeding
- Vertexing
- □ Track Finding
  - CKF
- □ Calibration – no need
- □ Alignment – no need

A light-weight Gaudi framework for integration and concurrency test

FATRAS: ATLAS fast simulation

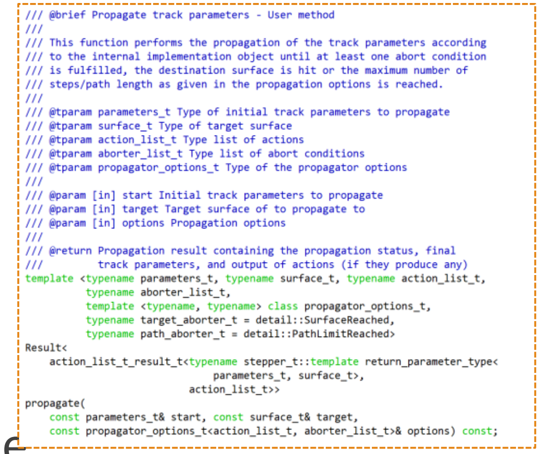
Thread-safe code and configurations

- Const-correctness and visitor pattern
- The caller create the cache which stays local to the thread
- Constant configuration as config struct at construction



```
struct MyTool {
  struct Config { double value{42}; };
  struct State{};
  MyTool(Config cfg)
    : m_cfg(std::move(cfg)) {}
  void doSomething(State& state)
    const { /* ... */ };
  Config m_cfg;
};
```

Development in ACTS

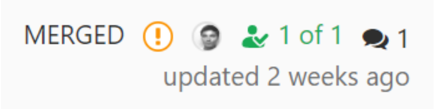


```
/// @brief Propagate track parameters - User method
///
/// This function performs the propagation of the track parameters according
/// to the internal implementation object until at least one abort condition
/// is fulfilled, the destination surface is hit or the maximum number of
/// steps/path length as given in the propagation options is reached.
///
/// @tparam parameters_t Type of initial track parameters to propagate
/// @tparam surface_t Type of target surface
/// @tparam action_list_t Type list of actions
/// @tparam aborter_list_t Type list of abort conditions
/// @tparam propagator_options_t Type of the propagator options
///
/// @param [in] start Initial track parameters to propagate
/// @param [in] target Target surface of to propagate to
/// @param [in] options Propagation options
///
/// @return Propagation result containing the propagation status, final
///         track parameters, and output of actions (if they produce any)
template <typename parameters_t, typename surface_t, typename action_list_t,
          typename aborter_list_t,
          template <typename, typename> class propagator_options_t,
          typename target_aborter_t = detail::SurfaceReached,
          typename path_aborter_t = detail::PathLimitReached>
Result<
    action_list_t_result_t<typename stepper_t::template return_parameter_type<
                               parameters_t, surface_t>,
                           action_list_t>>
propagate(
    const parameters_t& start, const surface_t& target,
    const propagator_options_t<action_list_t, aborter_list_t>& options) const;
```

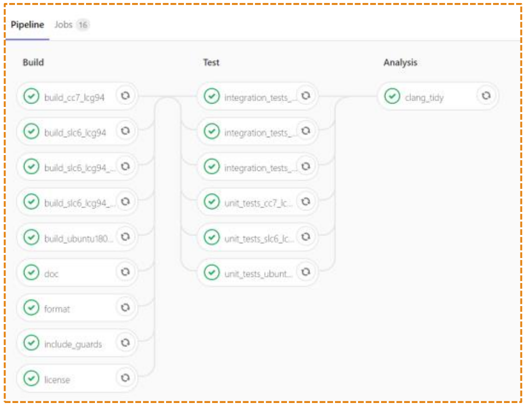
- Most of the development in gitlab

https://gitlab.cern.ch/acts

- The explanations are as clear as possible
- Boost unit tests and integration test to check the code
- CI (Continuous integration) for every merge request
  - CI success is one required approver for accepting a merge request
  - Others reviewing is the second required approval



MERGED ! 1 of 1 1
updated 2 weeks ago



  - Make examples in acts-framework for the whole workflow check

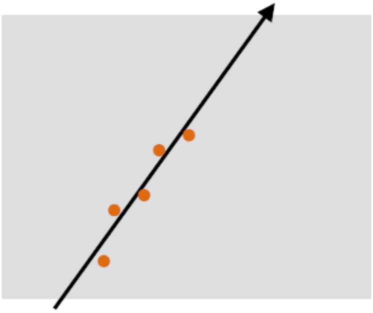
Join in ACTS

- Develop Gaussian Sum Filter fitting
- Code reviewing and validations
- Integration to CEPC

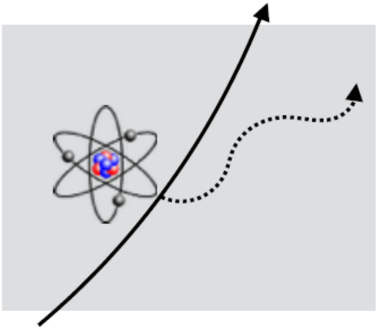
Gaussian Sum Filter

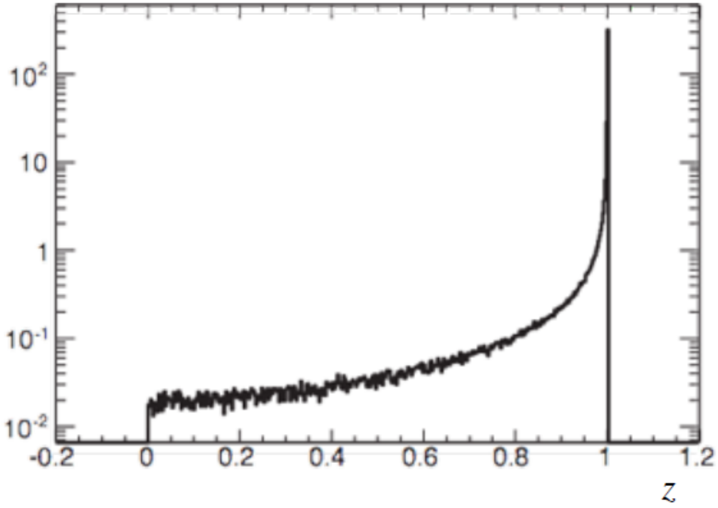
Kalman Filter : linearized filter allows all experiment noise is gaussian distributed

- Measurement errors – usually can be controlled
- Multiple scattering – a small gaussian tails
- Ionization loss – Landau distributed, fortunately dE<<E



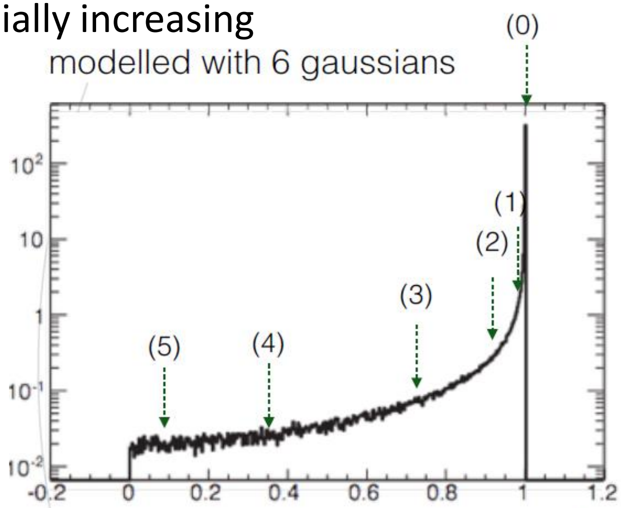
The electron reconstruction is significant and difficult
Energy loss is a bremsstrahlung effect -> strongly non gaussian

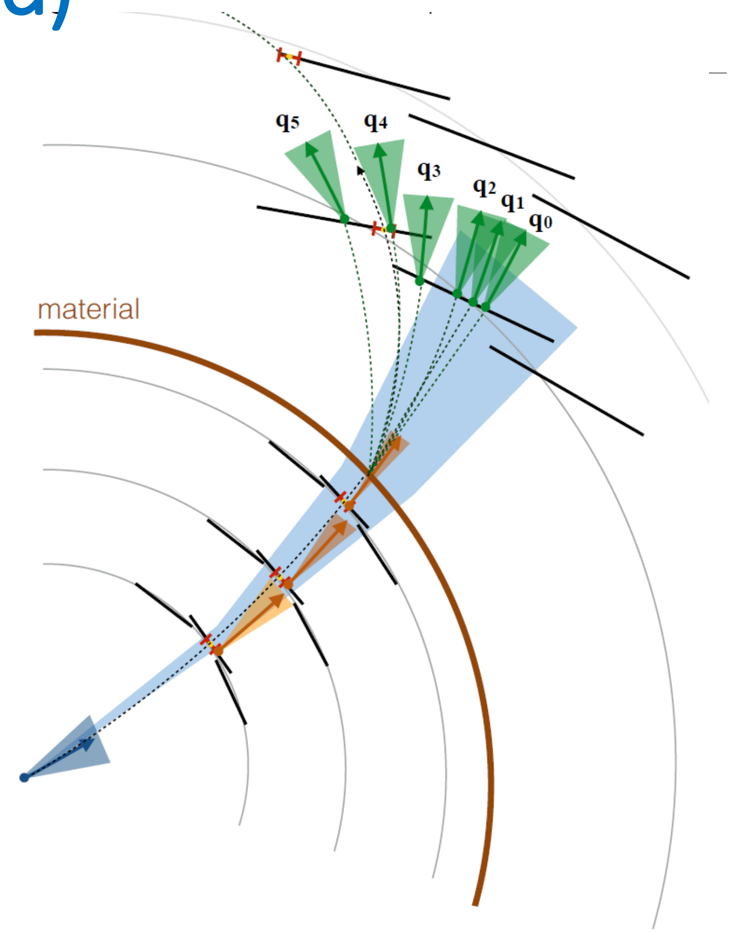




Gaussian Sum Filter(cont'd)

- Electron reconstruction are well handled with Gaussian Sum Filter, which is a parallel sets of Kalman Filter
- The bremsstrahlung energy loss distribution can be approximated as a weighted sum of gaussian components
- Each component behaves like a Kalman component, **propagate** individually
- Components should be merged to avoid the exponentially increasing





One component splits into 6 components

The (mean/cov/weight) of each component taken from ATLAS at the first step

Propagation : A general issue dealing with particle motion

In the magnetic field

Stepper

Reliable accuracy

integrated to tracking geometry

Navigation

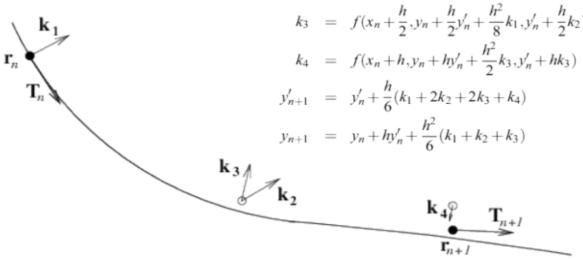
On surface

material effect, measurement update

Actors

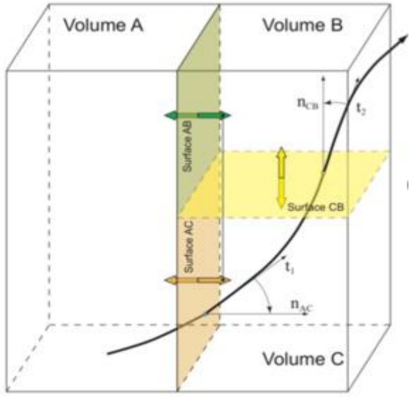
Runge-Kutta-Nystroem method

$$
\begin{aligned}
k_1 &= f(x_n, y_n, y'_n) \\
k_2 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}y'_n + \frac{h^2}{8}k_1, y'_n + \frac{h}{2}k_1) \\
k_3 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}y'_n + \frac{h^2}{8}k_1, y'_n + \frac{h}{2}k_2) \\
k_4 &= f(x_n + h, y_n + hy'_n + \frac{h^2}{2}k_3, y'_n + hk_3) \\
y'_{n+1} &= y'_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
y_{n+1} &= y_n + hy'_n + \frac{h^2}{6}(k_1 + k_2 + k_3)
\end{aligned}
$$

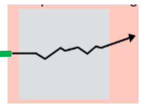


Track calculates and returns the error
Adaptive of step size according to error

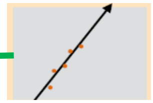
Vacuum/dense Environment(Calo) is chosen with a voting strategy



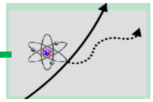
our fully connected tracking geometry
Come in convenient! ☺



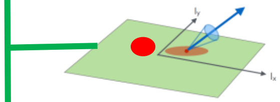
Multiple scattering



Ionization loss



Bremsstrahlung energy loss



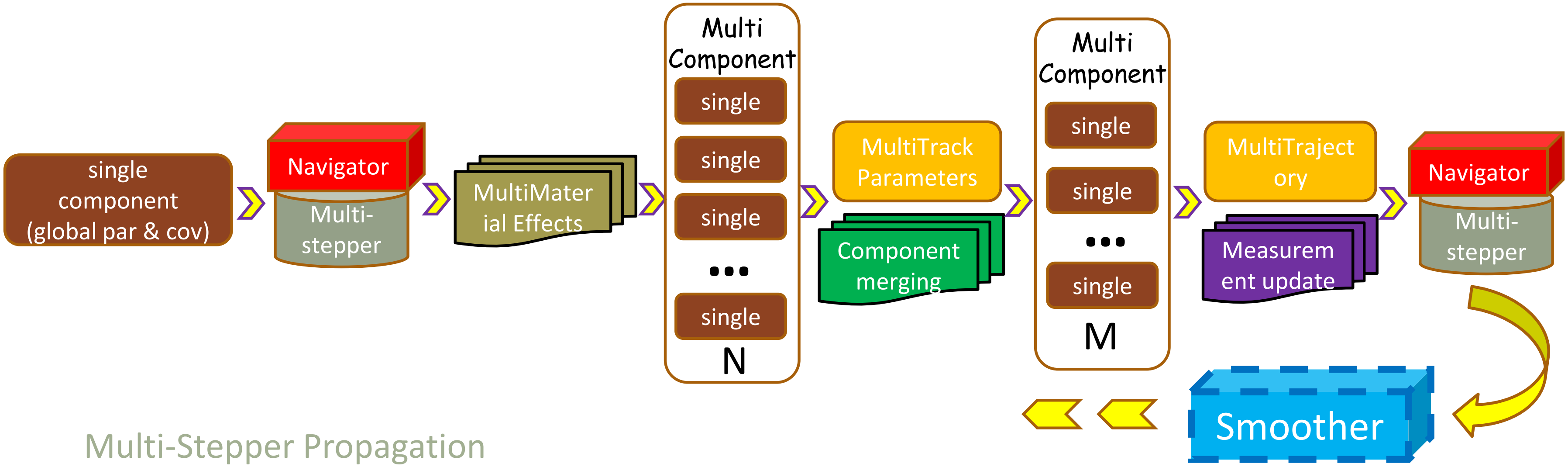
Measurement update

……

Aborters

- End of the world
- Path limit
- Target surface
- ……

Gaussian Sum Filter: implementation



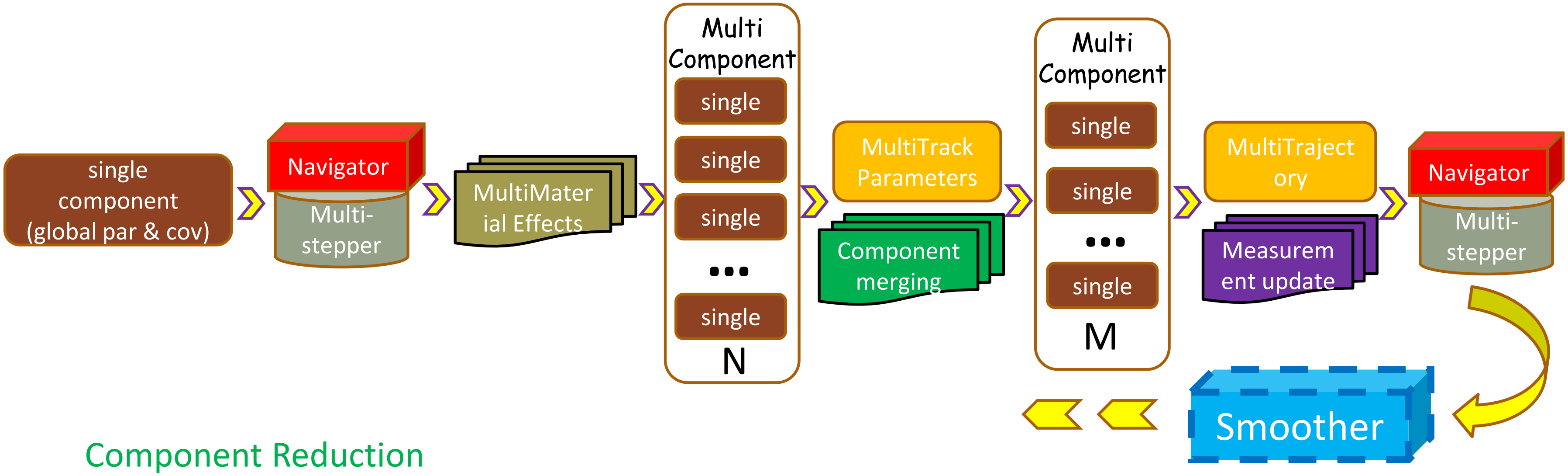
Multi-Stepper Propagation

- Take the combination of components – behave like single component in Navigation
- Each component owns its path
- Status(Free, Lock, Dead) of components decide if/when step forward

MultiMaterial Effect - Energy loss + Multiple scattering

- Bethe-Heitler – Currently take the ATLAS parameters to construct 6 components in each material effect

GSF: implementation (Cont'd)



Component Reduction

- Iteration to combine closet components to a maximum number

EDM for Gaussian sum filter

- Multitrajectory: TrackState store minimize heap allocation
- MultiTrackParameters : used for calculations of different components, e.g. component reduction

Measurement update with each component but modify the weight

Smoother: similar backward propagation, not prepared

Integration tests and validations for performance check will be done in the next step

ACTS Integration

- ACTS is derived from ATLAS tracking, but open to the high energy physics community and algorithm R&D (e.g. TrackingML challenge)
- Detector prototype from other experiments for tests/validations are expected

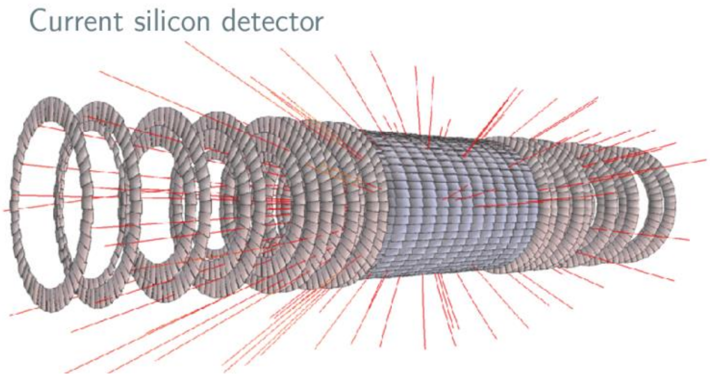
➢Integration to ATLAS

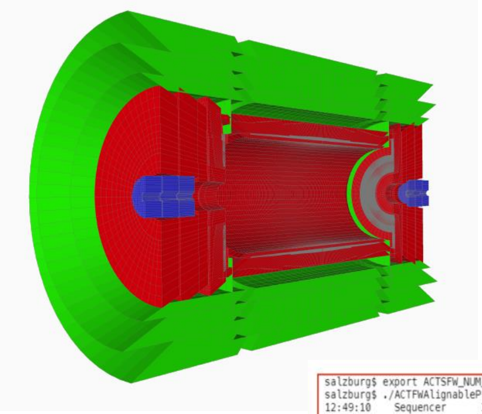
➢Integration to CEPC

➢Integration to other experiments: FCC-hh, Belle-2, FASER …

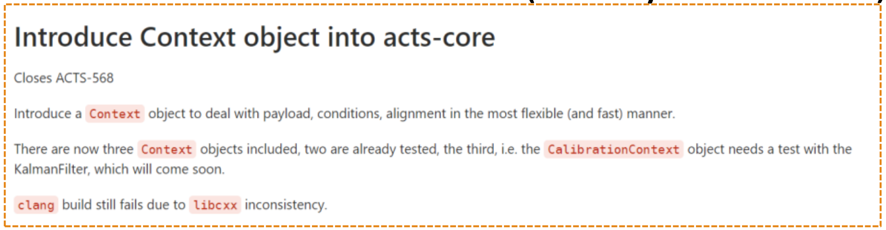
ACTS-ATLAS integration

- Tests of ATLAS geometry (ID + first tests for calorimeter modeling and navigation)
- Tool for propagation already exits


Current silicon detector



- ACTS apply a more clean solutions for the Context data in a MT environment (ideally re-entrant)



Introduce Context object into acts-core

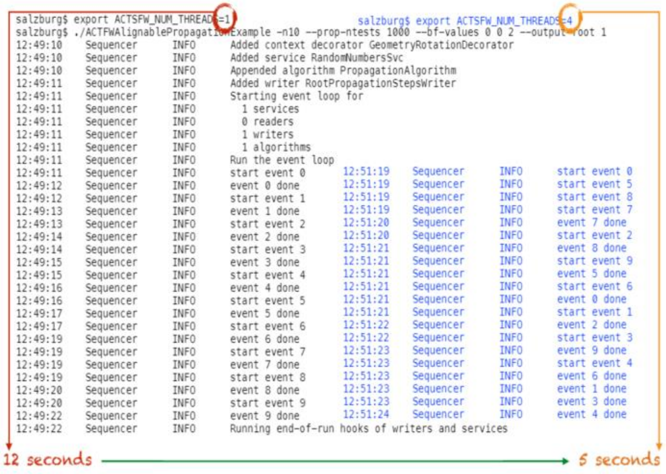
Closes ACTS-568

Introduce a `Context` object to deal with payload, conditions, alignment in the most flexible (and fast) manner.

There are now three `Context` objects included, two are already tested, the third, i.e. the `CalibrationContext` object needs a test with the KalmanFilter, which will come soon.

`clang` build still fails due to `libcxx` inconsistency.

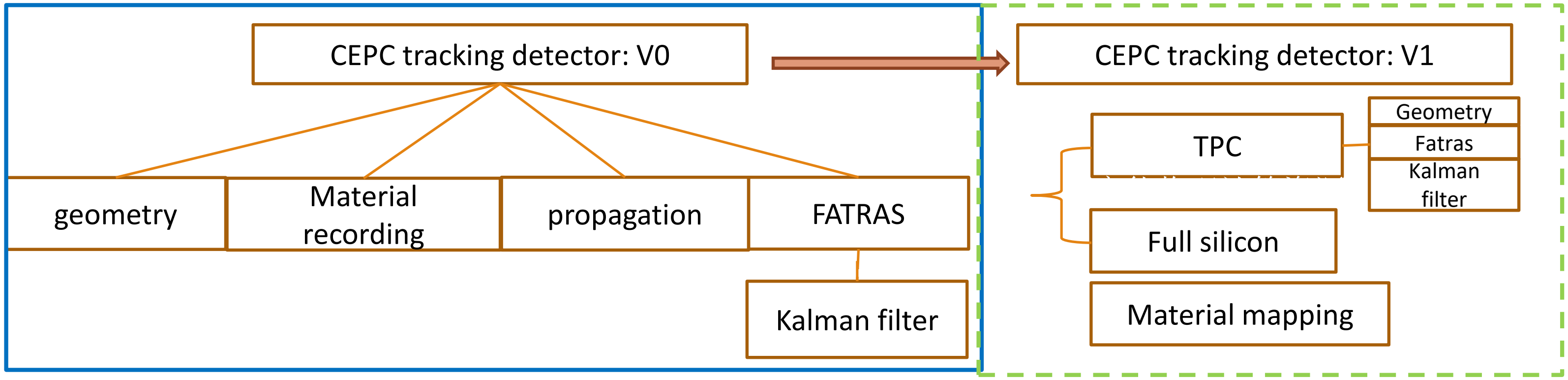
https://gitlab.cern.ch/acts/acts-core/merge_requests/520



```
salzburg$ export ACTSFW_NUM_THREADS=1                            salzburg$ export ACTSFW_NUM_THREADS=4
salzburg$ ./ACTFWAlignablePropagationExample -n10 --prop-ntests 1000 --bf-values 0 0 2 --output-root 1
12:49:10    Sequencer     INFO      Added context decorator GeometryRotationDecorator
12:49:10    Sequencer     INFO      Added service RandomNumbersSvc
12:49:10    Sequencer     INFO      Appended algorithm PropagationAlgorithm
12:49:11    Sequencer     INFO      Added writer RootPropagationStepsWriter
12:49:11    Sequencer     INFO      Starting event loop for
12:49:11    Sequencer     INFO        1 services
12:49:11    Sequencer     INFO        0 readers
12:49:11    Sequencer     INFO        1 writers
12:49:11    Sequencer     INFO        1 algorithms
12:49:11    Sequencer     INFO      Run the event loop
12:49:11    Sequencer     INFO      start event 0    12:51:19    Sequencer     INFO      start event 0
12:49:12    Sequencer     INFO      event 0 done     12:51:19    Sequencer     INFO      start event 5
12:49:12    Sequencer     INFO      start event 1    12:51:19    Sequencer     INFO      start event 8
12:49:13    Sequencer     INFO      event 1 done     12:51:19    Sequencer     INFO      start event 7
12:49:13    Sequencer     INFO      start event 2    12:51:20    Sequencer     INFO      event 7 done
12:49:14    Sequencer     INFO      event 2 done     12:51:20    Sequencer     INFO      start event 2
12:49:14    Sequencer     INFO      start event 3    12:51:21    Sequencer     INFO      event 8 done
12:49:15    Sequencer     INFO      event 3 done     12:51:21    Sequencer     INFO      start event 9
12:49:15    Sequencer     INFO      start event 4    12:51:21    Sequencer     INFO      event 5 done
12:49:16    Sequencer     INFO      event 4 done     12:51:21    Sequencer     INFO      start event 6
12:49:16    Sequencer     INFO      start event 5    12:51:21    Sequencer     INFO      event 0 done
12:49:17    Sequencer     INFO      event 5 done     12:51:21    Sequencer     INFO      start event 1
12:49:17    Sequencer     INFO      start event 6    12:51:22    Sequencer     INFO      event 2 done
12:49:19    Sequencer     INFO      event 6 done     12:51:22    Sequencer     INFO      start event 3
12:49:19    Sequencer     INFO      start event 7    12:51:23    Sequencer     INFO      event 9 done
12:49:19    Sequencer     INFO      event 7 done     12:51:23    Sequencer     INFO      start event 4
12:49:19    Sequencer     INFO      start event 8    12:51:23    Sequencer     INFO      event 6 done
12:49:20    Sequencer     INFO      event 8 done     12:51:23    Sequencer     INFO      event 1 done
12:49:20    Sequencer     INFO      start event 9    12:51:23    Sequencer     INFO      event 3 done
12:49:22    Sequencer     INFO      event 9 done     12:51:24    Sequencer     INFO      event 4 done
12:49:22    Sequencer     INFO      Running end-of-run hooks of writers and services
```

12 seconds → 5 seconds

ACTS-CEPC integration – CEPC-ACTS group


CEPC tracking detector: V0

geometry

Material recording

propagation

FATRAS

Kalman filter

CEPC tracking detector: V1

TPC

Geometry

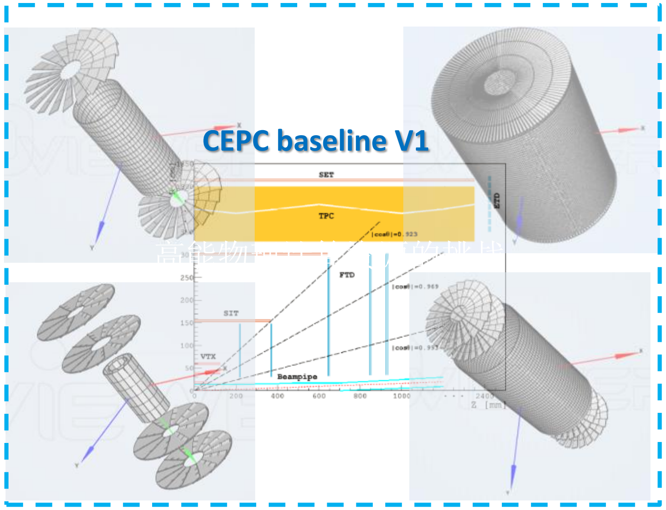
Fatras

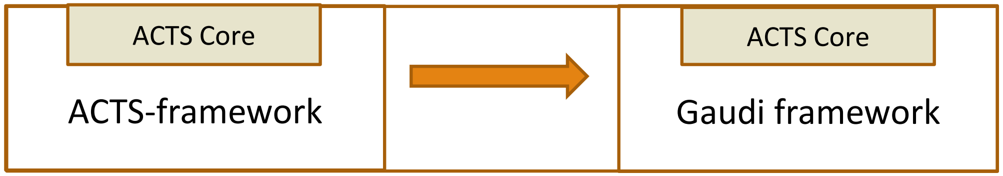
Kalman filter

Full silicon

Material mapping

- The basic version of V0 detector(without TPC) is implemented
- V1 detector are prepared for the geometry
  - Silicon detectors
  - TPC implemented with 220 layers currently
  - Material mapping is on going
  - Needs some tests and validations




ACTS Core

ACTS-framework

ACTS Core

Gaudi framework

- Migrate ACTS into Gaudi for CEPC at proper time

Current study in https://gitlab.cern.ch/jinz/acts-framework-cepc, all participations are welcome

Summary

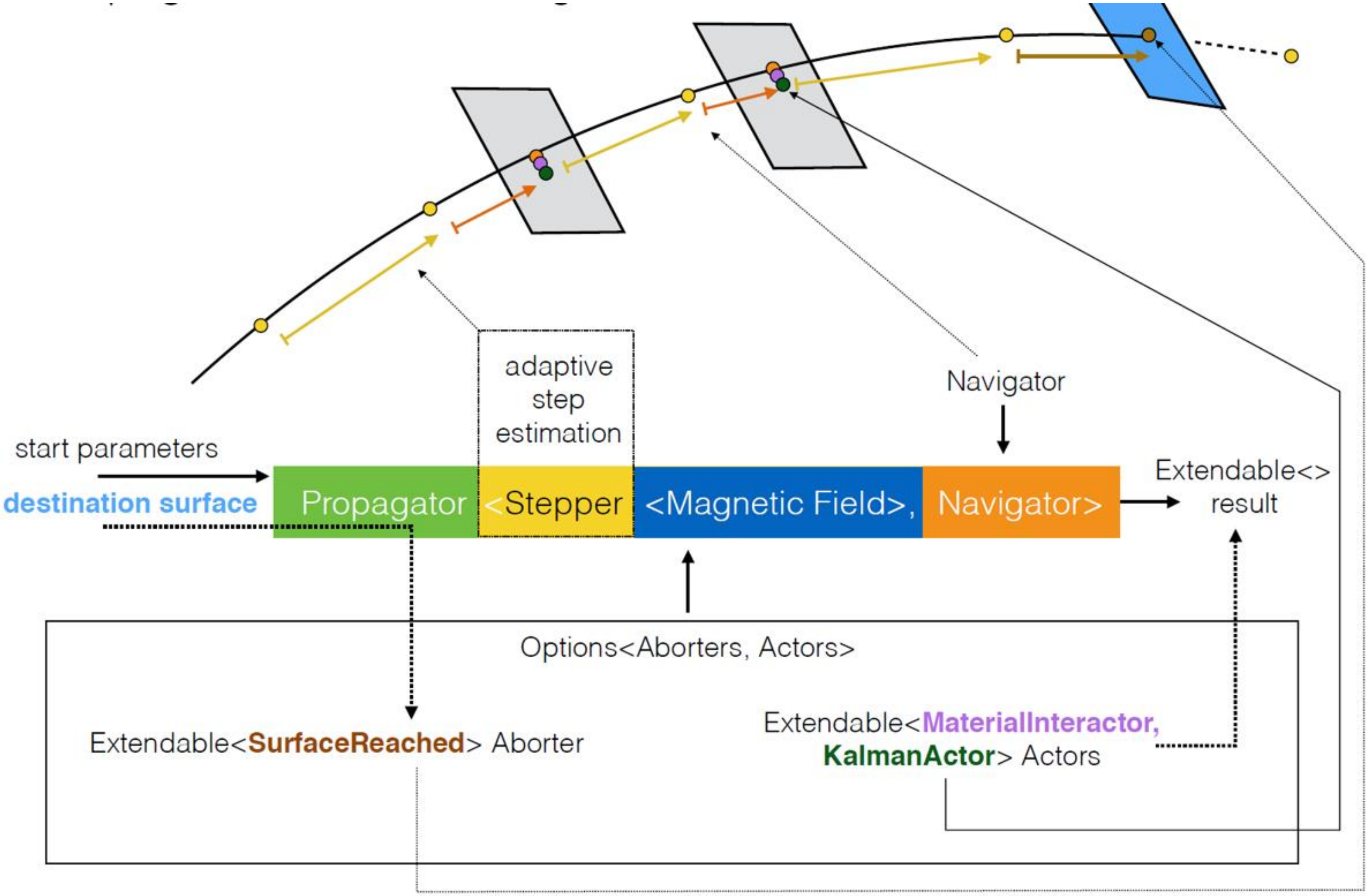
- ACTS is a future-oriented tracking project derived from ATLAS tracking, but open to tracking community
  - Geometry, Propagation, Kalman filter, Seeding …
  - Track finding (CKF) are working in progress

- We are taking part in ACTS project
  - Task of GSF development
  - Reviewing and validations
  - Integration ACTS to CEPC detector

- Validations and performance studies begin at the end of this year

BACKUP

Single stepper Propagation



adaptive step estimation

Navigator

start parameters

destination surface

Propagator <Stepper <Magnetic Field>, Navigator>

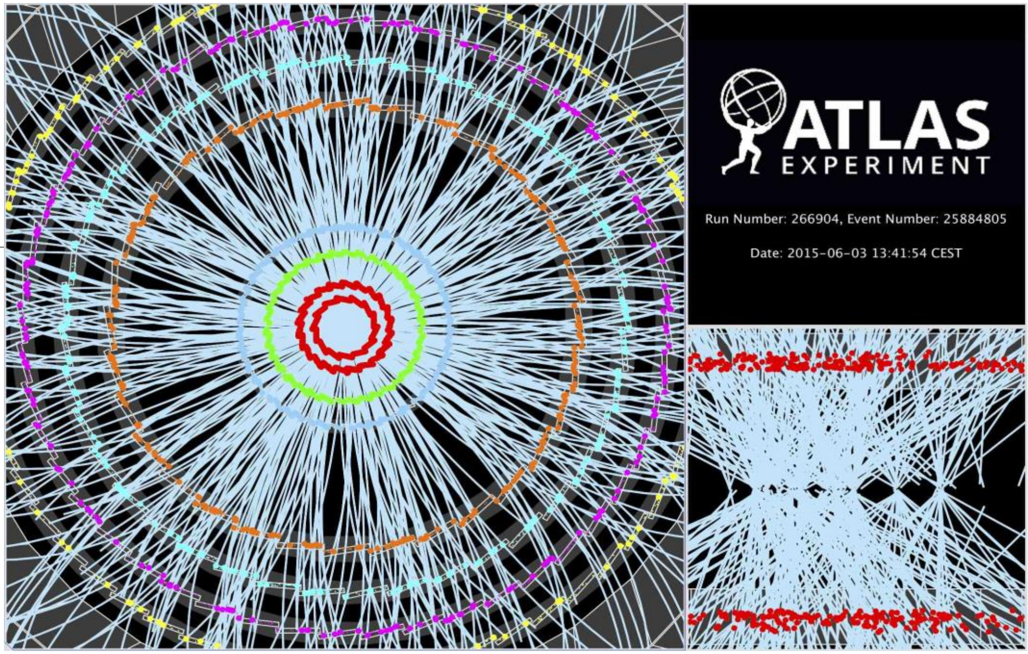
Extendable<> result

Options<Aborters, Actors>

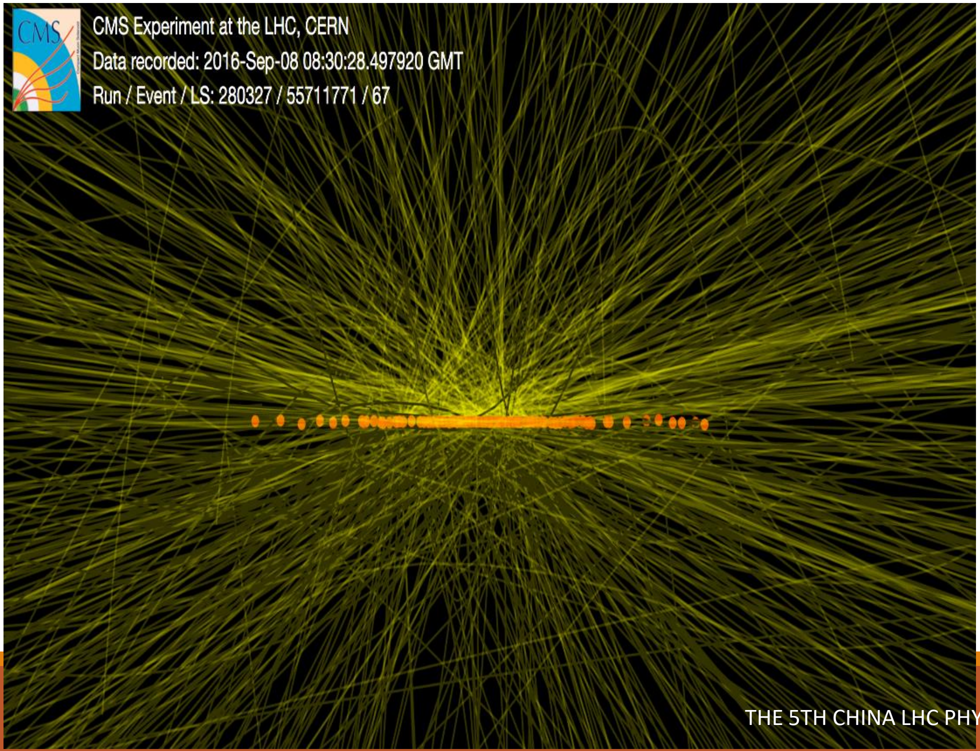
Extendable<**SurfaceReached**> Aborter

Extendable<**MaterialInteractor,**
KalmanActor> Actors

The challenge of computing resource in the future


ATLAS EXPERIMENT
Run Number: 266904, Event Number: 25884805
Date: 2015-06-03 13:41:54 CEST

A high pile up 13TeV LHC collision (86 vertices reconstructed)

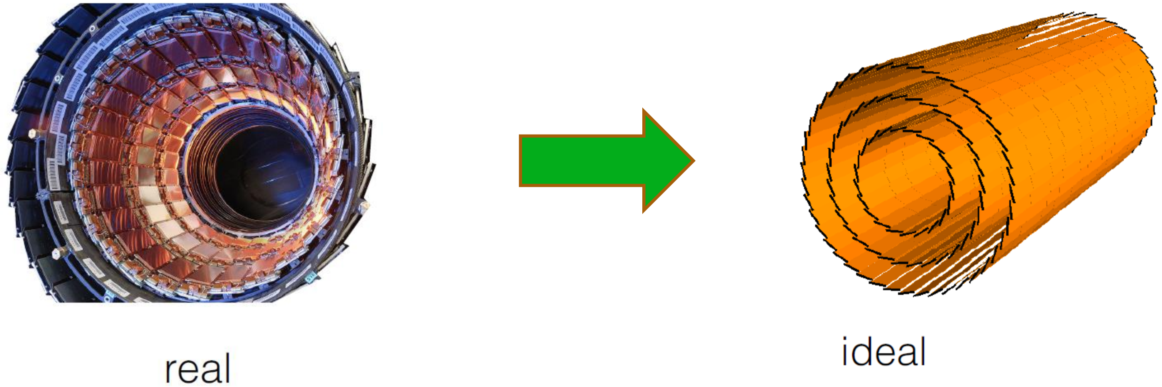

CMS Experiment at the LHC, CERN
Data recorded: 2016-Sep-08 08:30:28.497920 GMT
Run / Event / LS: 280327 / 55711771 / 67

A 13 TeV LHC collision (17 vertices)

	LHC Run-1	**LHC Run-2**	**LHC Run-4**	**FCC ~?**
muon	21	40	150-200	1000
Tracks	~280	~600	~7-10k	~100k

Building Tracking Geometry

First, we should have some tools for the tracking geometry that you can



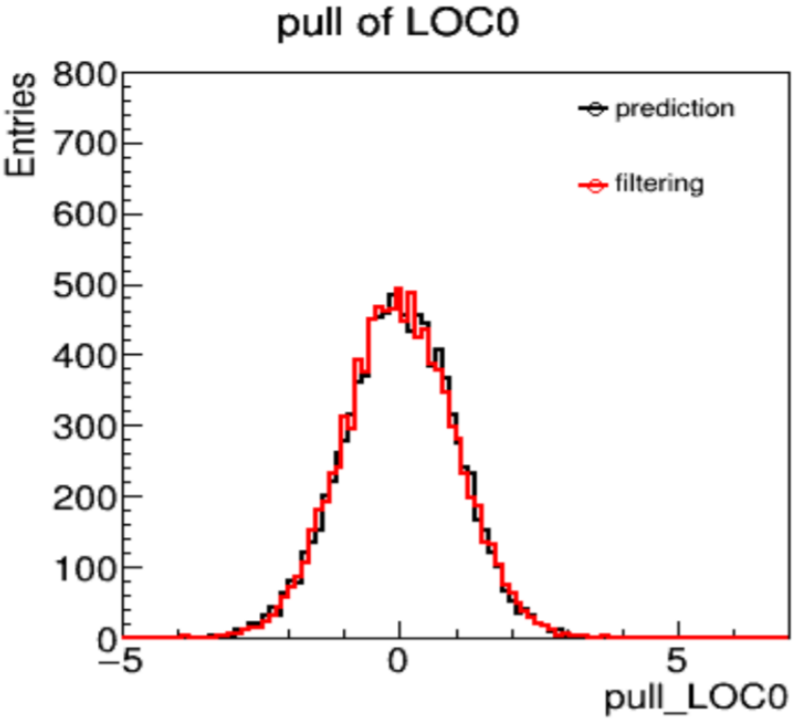
General idea to construct a detector for tracking is to

- Build them as light as possible <-> keep the measurement reliable
- The detection module should be kept full detailed
- The material should be simplified

Tracking Geometry = Simplified geometry + approximated material setup

KalmanFitter Status in Acts

- KalmanFilter implemented as Actor
- Called automatically during propagation
- Updates direction, uncertainties after filtering step



- Aims to minimize heap allocation
- Runtime performance: So far no direct comparison, comparable test setup not trivial
- Study of numerical performance by Xiaocong Ai [1] [2]