Community Driven Common (Tracking) Software
Track Reconstruction

Hits from one particle
Track Reconstruction

- Hits from one particle

- Seeding
  - triplet, quadruplet, n-plet finding
Track Reconstruction

- Hits from one particle

- Seeding:
  - triplet, quadruplet, n-plet finding

- Track finding:
  - combinatorial Kalman filtering
Track Reconstruction

- Hits from one particle
- seeding
  - triplet, quadruplet, n-plet finding
- track finding
  - combinatorial Kalman filtering
- track fitting & classification
  - Kalman filtering, smoothing
  - $\chi^2$ minimization
  - track classification, etc.
Track Reconstruction

Hits from one particle

Hits from a fraction of particles in HL-LHC environment
ATLAS Simulation Preliminary
RDO to ESD

\[ \sqrt{s} = 14 \text{ TeV} \]
\[ \langle \mu \rangle = 40 \]
25 ns bunch spacing
Run 1 Geometry
pp \rightarrow t\bar{t}
HS06 = 13.08

- Full reconstruction
- Inner Detector only

> 1 year w/o working head release

\(<\mu> \approx 40\)
From CHEP2015 to CHEP2018

 CMS Simulation, √s = 13 TeV, tt + PU, BX=25ns

- Full Reco Current
- Track Reco Current
- Full Reco Run1
- Track Reco Run1

Time/Event [b.u.]

Luminosity [10^{34} cm^{-2} s^{-1}]

<μ> ~ 40

LHC Run-1

optimisation A

rewrite/redesign B

LHC Run-2

LS 2

LHC Run-3
ATLAS Physics stream

~ $3 \times 10^{10}$ events
~ $10^{11}$ p-p collisions
~ $10^{13}$ tracks
~ $10^{14}$ track candidates
~ $10^{15}$ track seeds
Reconstruction in rel. 21.0.37:
high-mu run 335302 (2 051 jobs)
produced only single (AOD) output

ATLAS ID
designed for

<µ> ~23

ATLAS Preliminary

Resource needs
(2017 Computing model)
Flat budget model
(+20%/year)
From CHEP2018 to CHEP2026

**ATLAS** Preliminary

- Resource needs (2017 Computing model)
- Flat budget model (+20%/year)

**CPU Resources [kHS06*1000]**

- Run 2
- Run 3
- Run 4

**Year**

- 2018
- 2020
- 2022
- 2024
- 2026
- 2028

**ATLAS Simulation**

- ITk Inclined Duals, tt events

- Total ID Run 2 Reconstruction
- Total ITk Reconstruction
- Si Track Finding (Run 2)
- Si Track Finding (ITk)
- Ambiguity Resolution (Run 2)
- Ambiguity Resolution (ITk)

**Resource needs** (2017 Computing model)

- Flat budget model (+20%/year)

**LS 2**

LHC Run-3

**LS 3**

HL-LHC Run

FCC-hh Run 1

HE-LHC Run 1

Upgraded detector & tighter tracking cuts
From CHEP2018 to CHEP2026

see talk by G. Raven
The story of a Runge-Kutta propagator

Numerical integration algorithm,
HEP usage for:
solving equation of motion through complex magnetic field
transporting the track covariance matrix
The story of a Runge-Kutta propagator

First implementations for ATLAS Track reconstruction
atrecon (FORTRAN based) based propagation function
The story of a Runge-Kutta propagator

Solenoid

Inner Detector

Calorimeter

Toroid

Muon System

Move to C(++)

Athena-Gaudi framework

xKalman monolithic C(++) program
The story of a Runge-Kutta propagator
The story of a Runge-Kutta propagator

One possible future...

NN based Kalman filtering
AI pattern recognition techniques

see TrackML talk by M. Kiehn
(and the entire Thursday session)
The story of a Runge-Kutta propagator

trend to concurrent algorithmic approaches e.g. **AthenaMT** framework for ATLAS

- 10-15(+) years old HEP code in general not thread safe
- vectorization is only little used in **offline** code
- caching is/was often used for performance

Another path ... or better **many other paths**
Future detector technologies

ATLAS and CMS will insert first timing detectors for HL-LH
- currently time tagging
- correct treatment: 5D tracking becomes 6D
  (math & code development)

see CTD2018 talk by L. Gray
Future detector technologies

ATLAS and CMS will insert first timing detectors for HL-LHC
- currently time tagging
- correct treatment: 5D tracking becomes 6D
  (math & code development)

see CTD2018 talk by L. Gray
**R&D testbed**
- establish toolkit with thread safe tracking tools
- build up concurrent pattern recognition chains
- test physics driven recognition configuration

**Turn-key software**

- **seedFinder** (const Config& pConfig, State& pState, iterator_type start, iterator_type end)

- **Electron cluster**
- **blue thread**
- **red thread**
- **blueState**
- **redState**
- **SeedContainer**
Experiment SW ecosystem

Configuration

Framework

Algorithmic Component

Event Data Model

Algorithmic Component

Common SW

Generic algorithmic code

Bare Event Data Model

D&T&D Development, Testing, Deployment

Turn-key software

Fast development & testing ecosystem
Common structure motivation

Solenoid

Inner Detector

Calorimeter

Muon System

Toroid

Geometry

Event Data Model

Extrapolation

Fitting

Calibration, general

Alignment

InnerDetector

EDM

Geometry

Tools

Generic solutions for general tasks

Keep detector expertise where it belongs

Digitization, Calibration, Error Parameterization, …
We present the Minimal requirements for core functionality:

- First test deployment in ATLAS was extremely simple.
- Extended with plugins for ROOT, DD4Hep, Geant4, etc.

Minimal requirements for **core** functionality:

- A C++14 compatible compiler, e.g. `gcc (≥ 6.2)` or `clang (≥ 4.0)`
- `cmake (≥ 3.7)`
- `boost (≥ 1.62, with `program_options` and `unit_test_framework`)`
- `Eigen (≥ 3.2.9)`

Visit [https://acts.web.cern.ch](https://acts.web.cern.ch) for more information.
Repository & Testing

Repository structure

https://gitlab.cern.ch/acts/

- acts-core
- acts-framework
- acts-data
- acts-fatras

Fast development & testing ecosystem

- algorithms & tools are tested for thread safety and performance

Unit Test coverage

<table>
<thead>
<tr>
<th>Exec</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date: 2018-06-13</td>
<td>Lines: 2951</td>
</tr>
<tr>
<td>Branches: 3091</td>
<td>22905</td>
</tr>
</tbody>
</table>

Legend:
- low: < 75.0%
- medium: >= 75.0%
- high: >= 90.0%

see also talk by H. Graslan
Repository & Testing

Repository structure

https://gitlab.cern.ch/acts/

- acts-core
- GaudiHive
- acts-data
- acts-fatras

Fast track simulation extension

Generic algorithmic code

Bare Event Data Model

Turn-key software

Fast development & testing ecosystem

Branches:
- 3091
- 22905
- 13.5%

Lines:
- 2951
- 4708
- 62.7%

Legend:
- low: < 75.0 %
- medium: >= 75.0 %
- high: >= 90.0 %

Unit Test coverage

https://acts.web.cern.ch/ACTS/coverage/

Exec

Date: 2018-06-13

Coverage

Coverage

https://acts.web.cern.ch/ACTS/coverage/

Lines:
- 2951
- 4708
- 62.7%

Branches:
- 3091
- 22905
- 13.5%

see also talk by H. Graslan
namespace Acts {
    /// doxygen documentation
    class DetectorElementBase {
        /// the according represented surface
    virtual const Surface& associatedSurface() const = 0;
    }
}

class MyDetectorElement {
    /// @copydoc DetectorElementBase::associatedSurface
    const PlaneSurface& associatedSurface() const;
};

Geometry binding with via a simple/single detector element class on detector software side

Problematic:
- units
- coordinate system conventions
Geometry

C++

TrackML Detector

CMS Pixel & SVX

sPhenix

TGeo (Root)

FCC-hh

CLIC Vertex

DD4Hep

TGeo (Root)
namespace Acts {
    /// doxygen documentation
    class DetectorElementBase {
        /// the according represented surface
        virtual const Surface& associatedSurface() const = 0;
    };
}  

class MyDetectorElement {
    /// @copydoc DetectorElementBase::associatedSurface
    const PlaneSurface& associatedSurface() const;
};

Experiments concurrent SW ecosystem

Conclusion

Concurrent Geometry Service

Configuration

Framework

validity intervall 0

validity intervall 1

detected 3D detector geometry

ACTS Tracking geometry
Event Data

Flat event data model design based on Eigen
- leave calibration/details with detector SW
- use fixed size data structures

Problematic:
- different local coordinate systems
- heterogenous containers for measurements

CDF \( q'' = (l_1, l_2, \phi, \cot(\theta), C') \)
CMS \( q' = (l_1, l_2, \phi, \lambda, q/p) \)
ATLAS \( q = (l_1, l_2, \phi, \theta, q/p) \)

Achieved speed-up w.r.t. CLHEP in 5x5 matrix multiplication testbed

CheP2015 talk by AS

http://eigen.tuxfamily.org/
Event Data

Flat event data model design based on Eigen
- leave calibration/details with detector SW
- use fixed size data structures

Problematic:
- different local coordinate systems
- heterogenous containers for measurements

CDF \( q'' = (l_1, l_2, \phi, \cot(\theta), C') \)
CMS \( q' = (l_1, l_2, \phi, \lambda, q/p) \)
ATLAS \( q = (l_1, l_2, \phi, \theta, q/p) \)

http://eigen.tuxfamily.org/
Propagation & Magnetic Field

ATLAS Tracking

Magnetic field map in memory as 3D grid

Field look up in Runge-Kutta integration

Propagator
Navigator
MaterialEffectsUpdater

start parameters
destination surface

Extrapolator

next surface, layer, volume
material effects integration

Field look up in Runge-Kutta integration

Magnetic Field

Runge-Kutta integration with expensive getField() calls, adaptive step estimation to next surface

MaterialEffectsUpdater

start parameters
destination surface

(2)

(1)

(3)

ATLAS Tracking

ACTS

start parameters
destination surface

Propagator
<Stepper<br>Magnetic Field>, Navigator>

Extendable<> result

Options<Aborters, Actors>

Magnetic field map in memory as 3D grid

Field look up in Runge-Kutta integration

Magnetic Field

Runge-Kutta integration with expensive getField() calls, adaptive step estimation to next surface

MaterialEffectsUpdater

start parameters
destination surface

(2)

(1)

(3)

ATLAS Tracking

ACTS

start parameters
destination surface

Propagator
<Stepper<br>Magnetic Field>, Navigator>

Extendable<> result

Options<Aborters, Actors>
Propagation & Extrapolation

start parameters

Propagator <Stepper <Magnetic Field>, Navigator>

Extendable<> result

Options<Aborters, Actors>

AtlasStepper

EigenStepper

Output of StepLengthLogger (“Actor”)
**Propagation & Extrapolation**

Propagator < Stepper < Magnetic Field >, Navigator >

Start parameters → Result → Extendable []

Options < Aborters, Actors >

---

**STEP 3**

**PROFIT!**

- Tested on a Sandy Bridge-EP CPU
- SSE version (assembly & intrinsics): 2.4x faster
- AVX version: 1.5x faster
  - slower than SSE because of costly cross lane permutations

Vectorized versions

**Output of StepLengthLogger (“Actor”)**
Propagation & Extrapolation

start parameters

Propagator <Stepper <Magnetic Field>, Navigator>

Options <Aborters, Actors>

Extendable <> result

Vectorized versions

I ate your Web page.
Forgive me; it was tasty
And tart on my tongue.

The link you requested:
http://anwww.lbl.gov/~vitillo/simd.tar.gz

is not available.

ROBERTO AGOSTINO VITILLO

Output of StepLengthLogger ("Actor")
Propagation & Extrapolation

---

Start parameters → Propagator → Stepper → Magnetic Field, Navigator → Extendable

Options < Aborters, Actors > → Vectorized versions

```c
for(int j = 0; j < N; j++) {
    for(int i = 0; i < 4; i++) {
        n256d da = mm256_loadu_pd(&P[i]);
        n256d da_201 = CROSS_SHUFFLE_201(da);
        n256d da_120 = CROSS_SHUFFLE_120(da);
        n256d da = mm256_sub_pd(mm256_mul_pd(ha_201, da_201), da_120);
        if(i==35) {
            da = mm256_add_pd(da, V0, 0.012);
        }
        n256d d2 = mm256_add_pd(d0, da);
        n256d d2_201 = CROSS_SHUFFLE_201(d0);
        n256d d2_120 = CROSS_SHUFFLE_120(d0);
        n256d d3 = mm256_sub_pd(mm256_mul_pd(ha_201, d2_201), d2_120);
        n256d d3_120 = CROSS_SHUFFLE_120(d2);
        if(i==35) {
            d3 = mm256_add_pd(d3, mm256_sub_pd(V3, 0.012, A, 0.012);
        }
        n256d d4 = mm256_sub_pd(mm256_add_pd(d2, mm256_add_pd(d3, da_201), da_120), da_120);
        if(i==35) {
            d4 = mm256_add_pd(d4, mm256_sub_pd(V4, 0.012, A, 0.012);
        }
    }
}
```

Output of StepLengthLogger ("Actor")

---

SSE  | AVX

A. Salzburger - CHEP2018
namespace Acts {
  /// doxygen documentation
  class WorkHorse {
    /// @struct Config for To
    struct State {
      float hoursWorked; ///< state parameter
    };

    /// @brief work method
    Result doWork(State& hState, const Input& workInput) const;
  };
} // end of namespace Acts

Concurrency

Caller must provide a by construction thread local state.

All modules are tested for consistency between single-threaded or multi-threaded mode.
Configuration

Configuration struct (nested)

```cpp
namespace Acts {
    /// doxygen documentation
    class WorkHorse {
        /// @struct Config for To
        struct Config {
            float coatColor; ///< configure the coat color
            float maxHours; ///< set the max hours this horse can work
        };
    } // end of namespace Acts
}
```

Binding to detector SW ecosystem

```cpp
/// feed from Framework into ACTS configuration
declareProperty("CoatColor", m_cfg.coatColor);
declareProperty("MaxHours", m_cfg.maxHours);
```
Logging

Screen output logging is an important debugging tool

```cpp
/** @brief macro for verbose debug output */
/** @ingroup Logging */
/** */
/** @param x debug message */
/** */
/** @pre @c logger() must be a valid expression in the scope where this */
/** macro is used and it must return a Acts::Logger object. */
/** */
/** The debug message is printed if the current Acts::Logging::Level <= */
/** Acts::Logging::VERBOSE. */
#define ACTS_VERBOSE(x)
  if (logger().doPrint(Acts::Logging::VERBOSE))
    logger().log(Acts::Logging::VERBOSE) << x;
```

Problematic:
- binding different logging technologies is nasty
- currently SW needs to provide ACT_MSG(X) macros

Tested successfully with Athena (ATLAS) & Gaudi (FCC)

Different approach prototyped:

```
using ul = DebugOutputLogger;
```

Options<Aborters, Actors> -> Propagator <Stepper <Magnetic Field>, Navigator> -> Extendable< ul::result_type > result
contains std::string dump it in your framework
Friends & Family: **TrickTrack**

- hits on barrel layer
- hit doublet
- CA Cell

**TrickTrack** is the encapsulated Cellular-Automaton library from CMSSW
- header-only library templated on input data type, easy to integrate into your SW framework
- builds n-dimensional track candidates
- minimal dependency: **Eigen**

https://github.com/HSF/TrickTrack

see [CTD2018 talk](https://www.ctd2018.org/) by V. Voelkl
**Friends & Family: mkFit**

**mkFit** is a vectorized track fitting and track finding approach developed with CMSSW:
- embedded in CMSSW
- extended to realistic detector geometry
- core of vectorization is a code generator for matrix multiplications

---

*See CTD2018 talk by M. Tadel Wednesday in T2*
Reference dataset

Reference datasets used very successfully in many fields - particularly machine learning

e.g. MNIST ("Modified National Institute of Standards and Technology")

Stating the obvious
- allows for traceability of algorithm development and progress
- allows apple-to-apple comparisons
- allows to perform mid-scale migrations (e.g. Eigen vs. SMatrix comparison)

Stating the less obvious
- eases publication of algorithmic research
Common dataset for development within the community

- detector used for TrackML

- dataset produced with ACTS fast simulation

- proposal: iron out the few features we discovered & dataset (LHC/HL-LHC), publish on opendata CERN
Quasi-realistic full silicon detector
- non-Gaussian measurements,
  with realistic cluster shapes
- realistic material budget
- main particle-material interactions

Pixel residuals for TrackML detector
(50 µm x 50 µm pixel size)
Reference detector & dataset

Enlarge community
- there are a lot of experts & enthusiasts

Experiment SW ecosystem

development & testing ecosystem

publish
Summary

HL-LHC and FCC-hh environment will need re-thinking of current track reconstruction
- time of low-hanging code optimization has passed
- need real R&D for data structures, algorithms, steering
  *and it is happening already*
- encourage to develop and share across experiment boarders
  *and it is happening already*
- open software and public license is key to this
- with ACTS we try to do achieve this regarding ATLAS tracking code
- we attempt to establish a **turn-key** software for development
  with a **feedback path** into the experiment code

Reference dataset
- a **very useful** concept that is common to several fields
- should help to **share, interact & publish**
Backup
Problematic:

- pure parsing the geometry is always difficult
- not recommended, direct integration is certainly better
Units are difficult to harmonize
- try to use units consistently throughout the ACTS code (an we write a static code parser for this?)
- can be interleaved with experiment SW with Units conversions constant
Timing

Extrapolator

start parameters

destination surface

parameters on next surface

next surface, layer, volume

material effects integration

destination parameters

MaterialEffectsUpdator

Extrapolator

Navigator

Propagator

Options<Aborters, Actors>

AtlasStepper

EigenStepper

start parameters

destination surface

parameters on next surface

next surface, layer, volume

material effects integration

destination parameters

ExtrapolationEngine

AtlasStepper

EigenStepper

ExtrapolationEngine

Infrastructure

Atlas Stepper

Eigen Stepper

Extrapolation Engine

Constant field
Reference

Results shown are done with acts branches

acts-core: 1bd13177..d30185b5 ACTS-431_Extrapolator
acts-fatras: 15ca56d..4d6e766 ACTSFW-112_
acts-framework: 6f0f1e2..344500b ACTSFW-112_Fatras_New_Propagator