Track parameter propagation for different detector and magnetic field setups in acts

Fabian Klimpel
CERN, TU Munich
fklimpel@cern.ch
24.07.2019

Technische Universität München

CERN

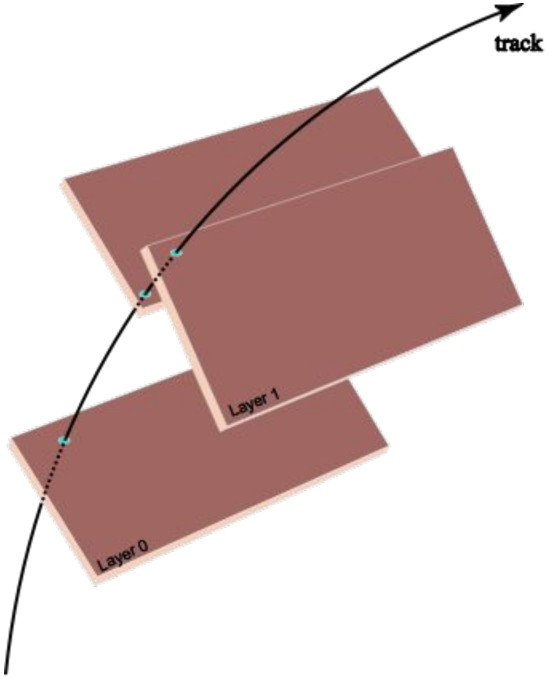
ATLAS EXPERIMENT

Max-Planck-Institut für Physik
(Werner-Heisenberg-Institut)

Overview

- Introduction
  - Tracking
  - Acts
  - Track reconstruction
- Track parametrisation
- Track parameter propagation in the detector
  - StraightLine approximation
  - Runge-Kutta-Nyström integration
  - In matter and time
  - Covariance transport
- Bigger picture and summary

What is Tracking?



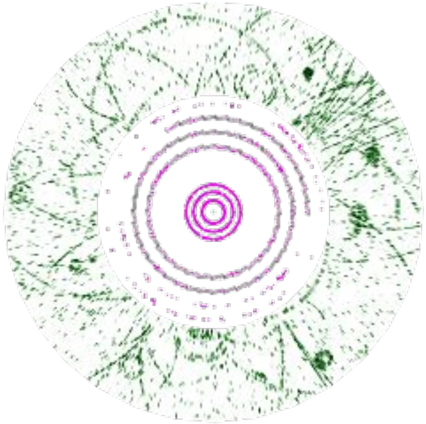
Particle collisions produce particle that traverse the detector(material) and produce signals (measurements)

Cloud of measurements needs to be associated to the particles that produced it

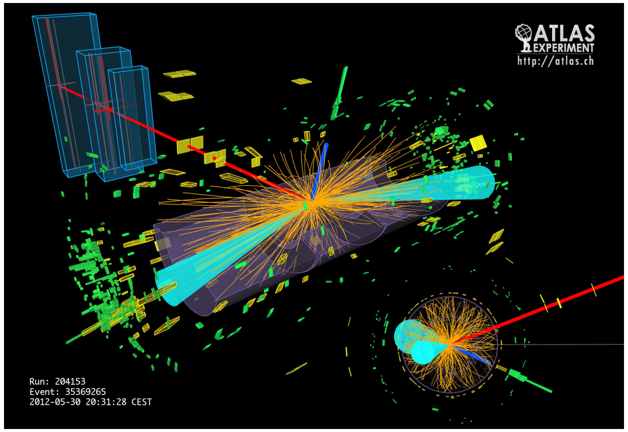
⟶ Knowledge about the event

⟶ Learn about physics

Converts



into



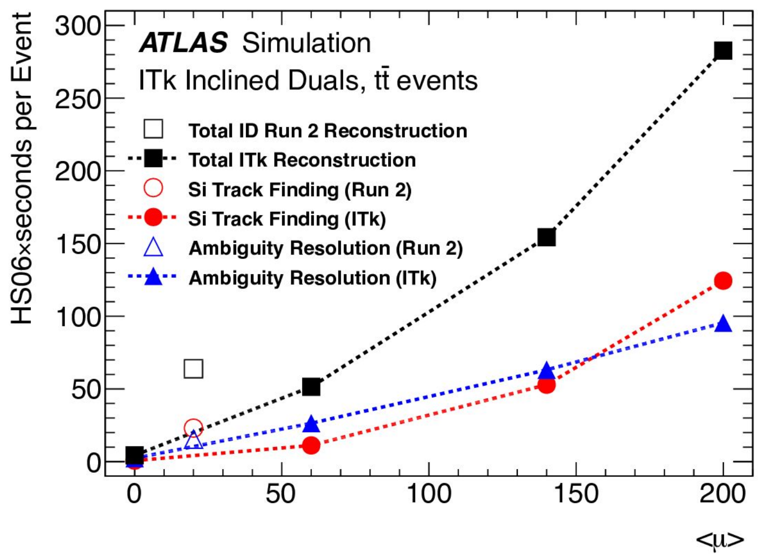
Future of tracking

As time goes on the amount of collisions per bunch crossing (pile-up) will be increased

This leads to more

- measurements per event
- combinations of measurements and tracks
- complex reconstruction

And therewith to a longer reconstruction time



Tracking R&D: ACTS

A project to provide all required tracking components is ACTS
(**A** **C**ommon **T**racking **S**oftware)

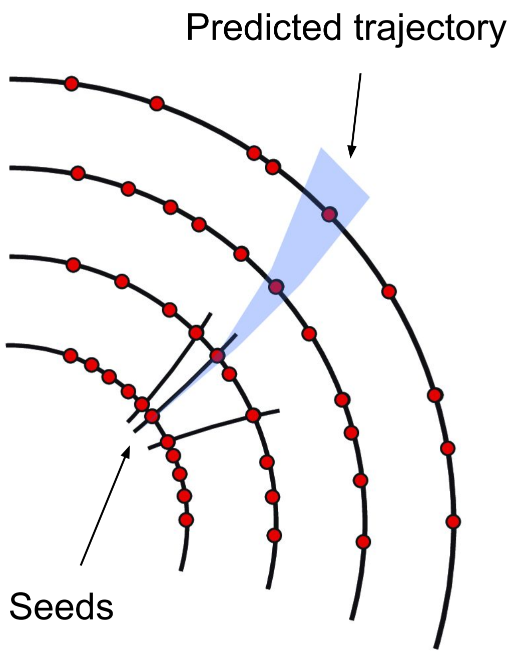
- Detector independent tracking software
- Aims to replace current ATLAS tracking
- Development ongoing since 4 years
- Guidelines:
  - Minimal external dependencies
  - Optimised hardware usage
  - Provide long time maintainability

- Based on rewritten tracking algorithms
- Allows comparing, testing and improving of the code
- Matching at least HL-LHC physics requirements

Progressive tracking - The Kalman filter

Association is performed by

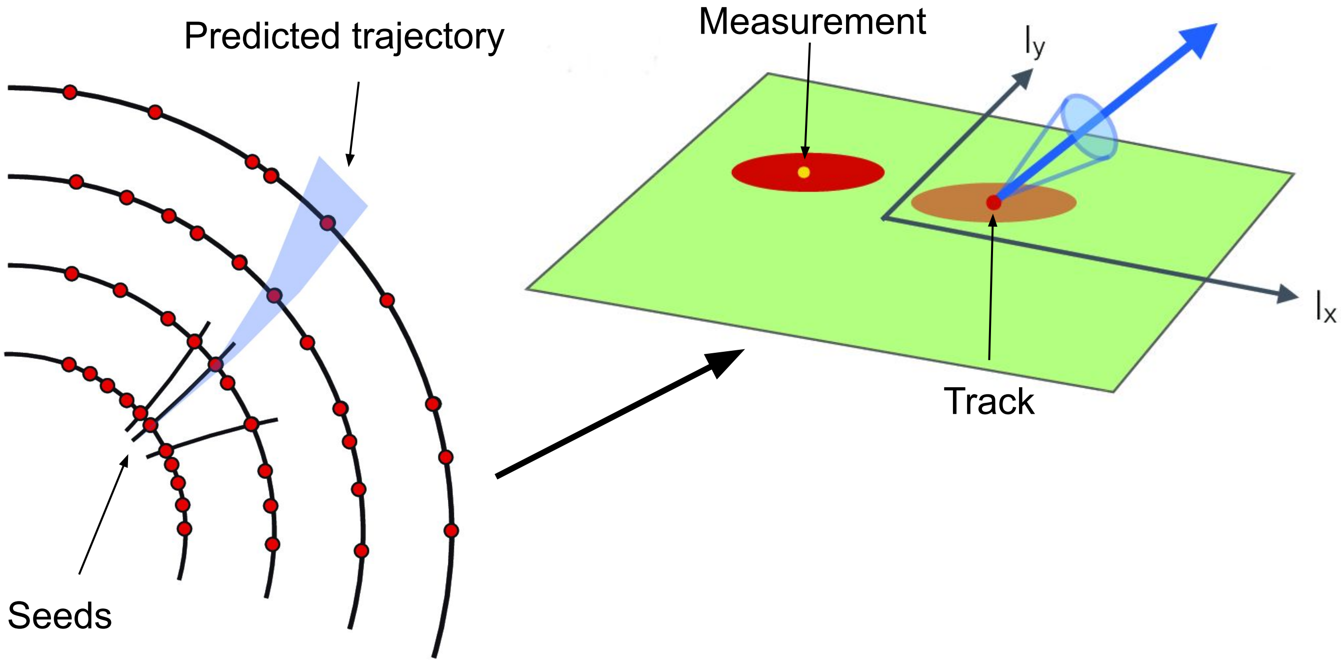
1. using an initial guess of the particle properties (= seed)
2. extending the trajectory



Progressive tracking - The Kalman filter

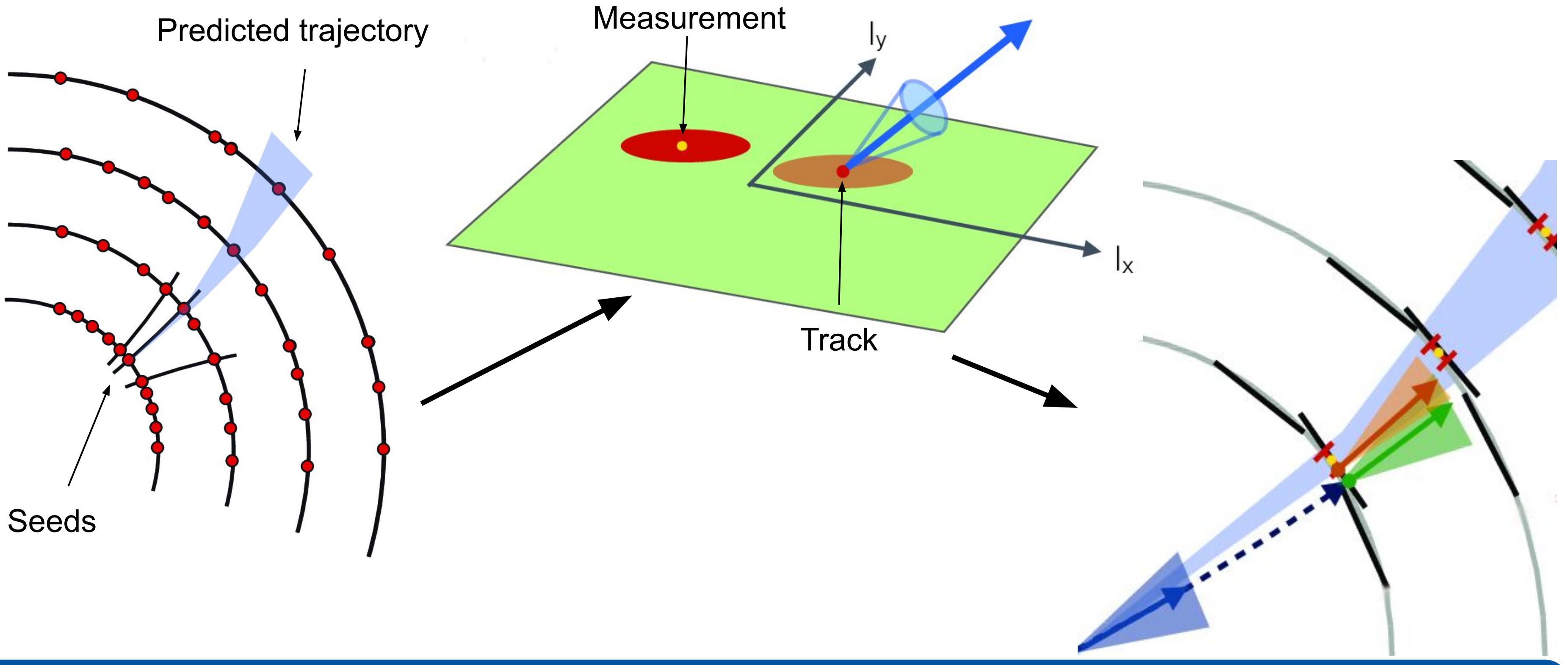
Association is performed by

1. using an initial guess of the particle properties (= seed)
2. extending the trajectory
3. searching for corresponding measurements along the way



Progressive tracking - The Kalman filter

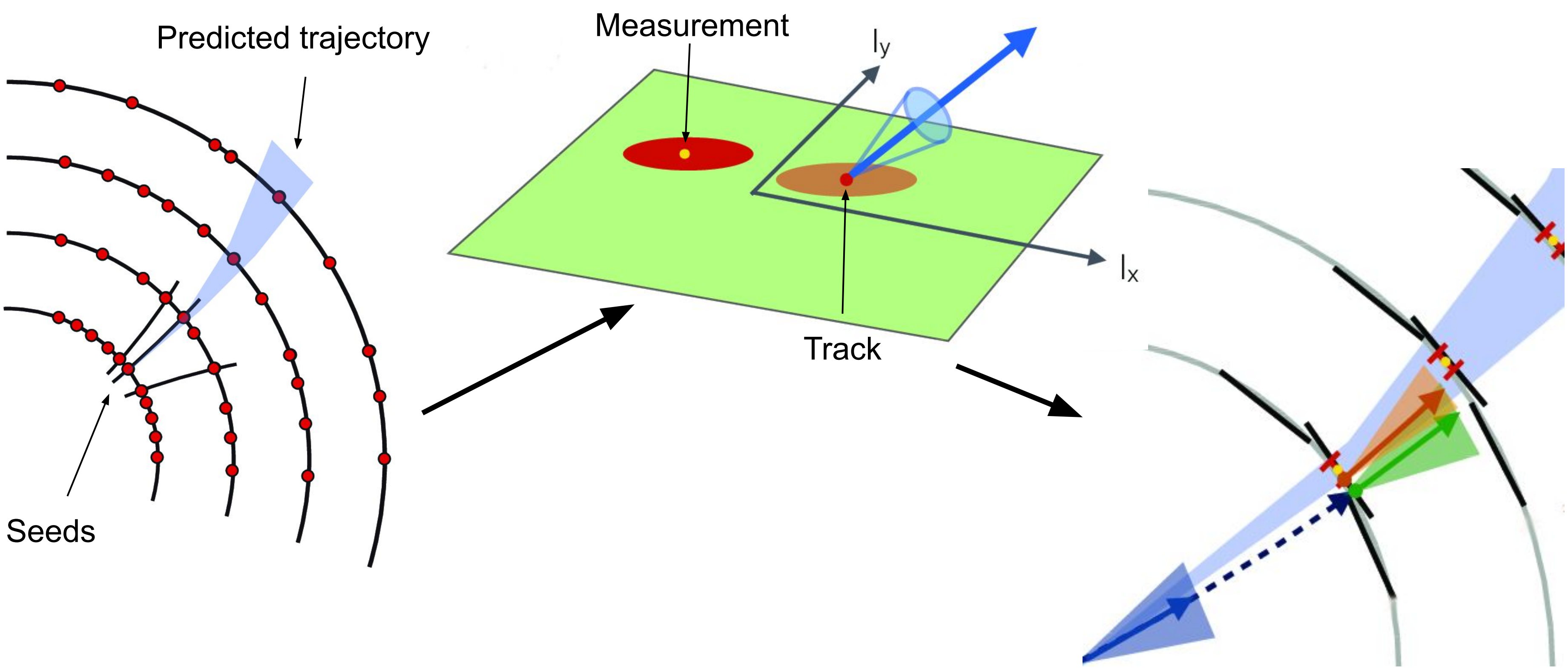
Association is performed by
1. using an initial guess of the particle properties (= seed)
2. extending the trajectory
3. searching for corresponding measurements along the way
4. update the guess with the new data



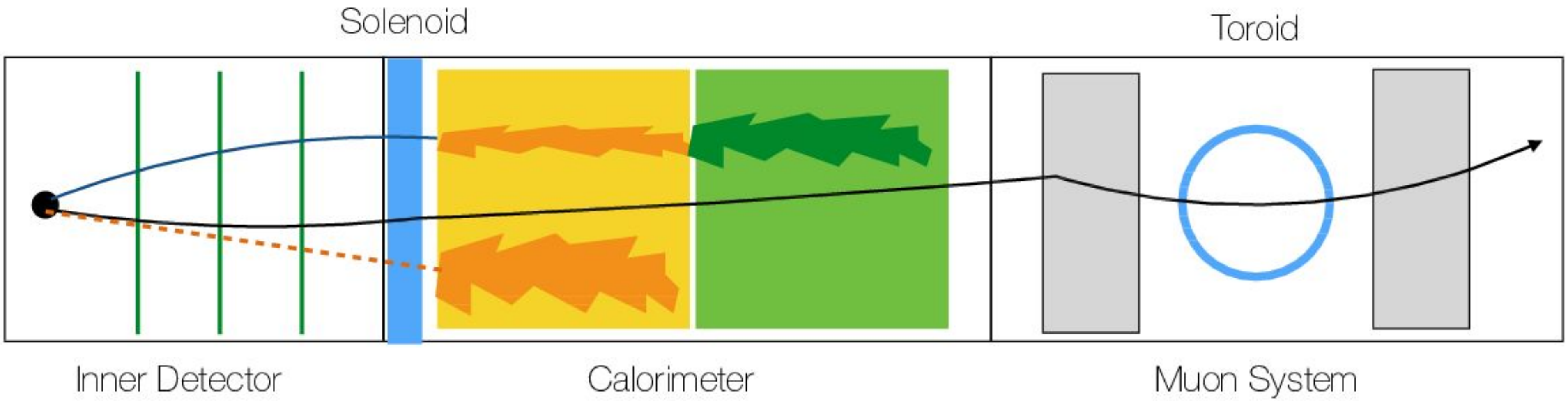
Progressive tracking - The Kalman filter

Association is performed by
1. using an initial guess of the particle properties (= seed)
2. extending the trajectory
3. searching for corresponding measurements along the way
4. update the guess with the new data

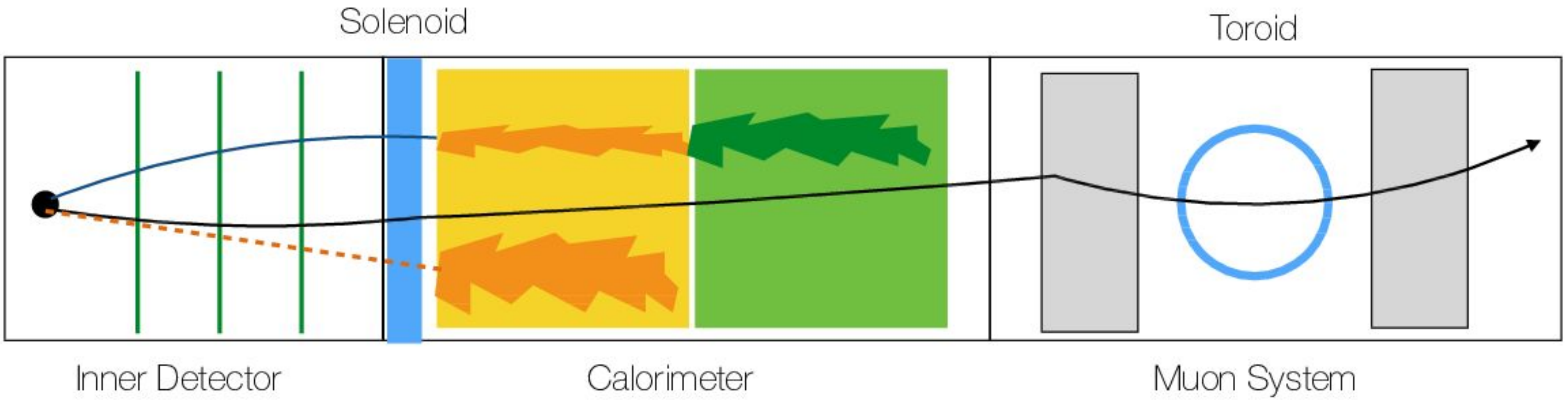
Today's topic



Detector description

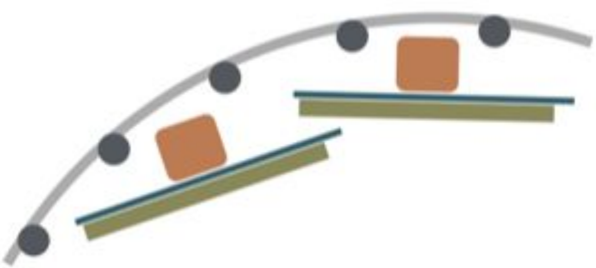


Detector description

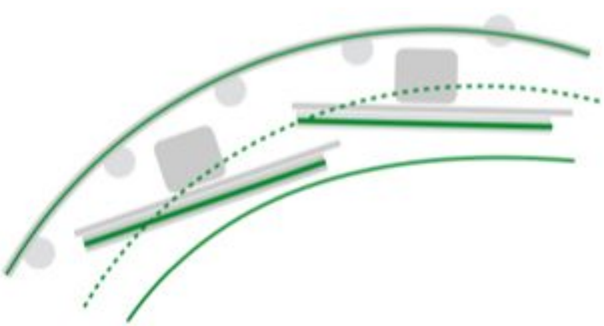


Inner Detector:

- Architecture can be approximated as set of surfaces
- Material is mapped onto the surfaces (discrete interactions)
- Can be either active (= detector module) or passive (= pure material)



Detailed description



Approximated by surfaces

Track parametrisation

In order to parametrise a track, 2 different representations are used:

- Global parametrisation:
  Describes the particle everywhere
  Given (in ATLAS) by $\underbrace{(x, y, z,}_{\text{Position}} \underbrace{T^x, T^y, T^z,}_{\text{(Normalised) Direction}} \underbrace{q/p)}_{\text{Curvature}}$

Track parametrisation

In order to parametrise a track, 2 different representations are used:

- Global parametrisation:
  Describes the particle everywhere
  Given (in ATLAS) by $$(\underbrace{x, y, z}_{\text{Position}}, \underbrace{T^x, T^y, T^z}_{\text{(Normalised) Direction}}, \underbrace{q/p}_{\text{Curvature}})$$

- Local parametrisation:
  Module measures in local frame
  Describes the particle in the context of a surface (e.g. detector module)
  Requires just a reduced set of parameters: $$(\underbrace{x, y}_{\text{Position}}, \underbrace{\phi, \theta}_{\text{Direction angles}}, \underbrace{q/p}_{\text{Curvature}})$$
  Position & shape of surface known

Track parametrisation

In order to parametrise a track, 2 different representations are used:

- Global parametrisation:
  Describes the particle everywhere
  Given (in ATLAS) by $$\left(\underbrace{x, y, z}_{\text{Position}}, \underbrace{T^x, T^y, T^z}_{\text{(Normalised) Direction}}, \underbrace{q/p}_{\text{Curvature}}\right)$$

J_{l2g} J_{g2l}

- Local parametrisation:
  Describes the particle in the context of a surface (e.g. $\overbrace{\text{detector module)}}^{\text{Module measures in local frame}}$
  Requires just a $\underbrace{\text{reduced set}}_{\text{Position \& shape of surface known}}$ of parameters: $$\left(\underbrace{x, y}_{\text{Position}}, \underbrace{\phi, \theta}_{\text{Direction angles}}, \underbrace{q/p}_{\text{Curvature}}\right)$$

Parameter propagation

Goal: Extension of the trajectory to the next surface to search for measurements

To estimate the trajectory we need to consider

1. Deflection by magnetic field
2. (Multiple) Scattering in material
3. Energy loss (e.g. by ionisation)

In the inner detector these effects only occur on the surfaces = <u>discrete & isolated from propatation</u>

Parameter propagation

Goal: Extension of the trajectory to the next surface to search for measurements

To estimate the trajectory we need to consider

1. Deflection by magnetic field
2. (Multiple) Scattering in material
3. Energy loss (e.g. by ionisation)

In the inner detector these effects only occur on the surfaces = discrete & isolated from propatation

Simplest case: No magnetic field in the inner detector = StraightLine approximation

Given our starting position $\vec{y}_0^{global} = \left(x_0, y_0, z_0, T_0^x, T_0^y, T_0^z, q/p_0\right)$

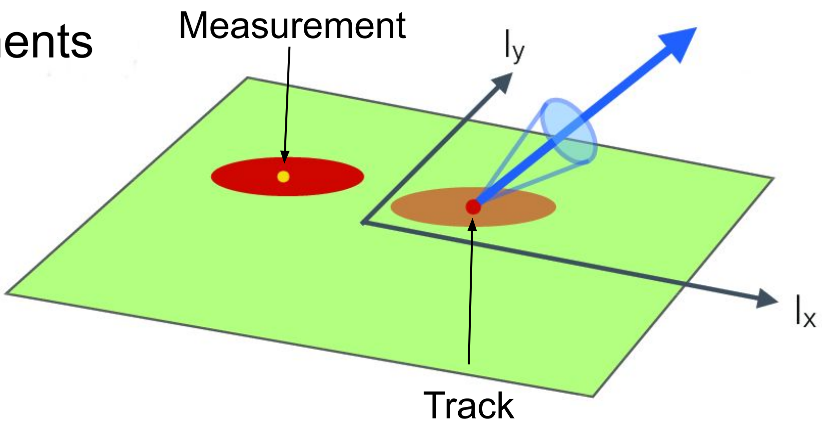
We can move to the next surface in distance h by evaluating

$$\vec{r}_1^{global} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + h \begin{pmatrix} T_0^x \\ T_0^y \\ T_0^z \end{pmatrix}$$

Uncertainty propagation

Beside the propagation one is interested in the propagation of the uncertainties

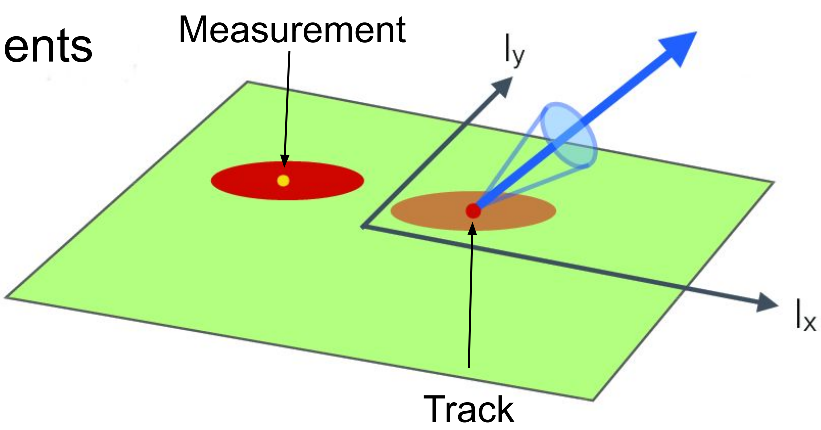
I.e. these define the searching area for measurements



Uncertainty propagation

Beside the propagation one is interested in the propagation of the uncertainties

I.e. these define the searching area for measurements



Starting with the local covariance matrix Σ_0^{local}

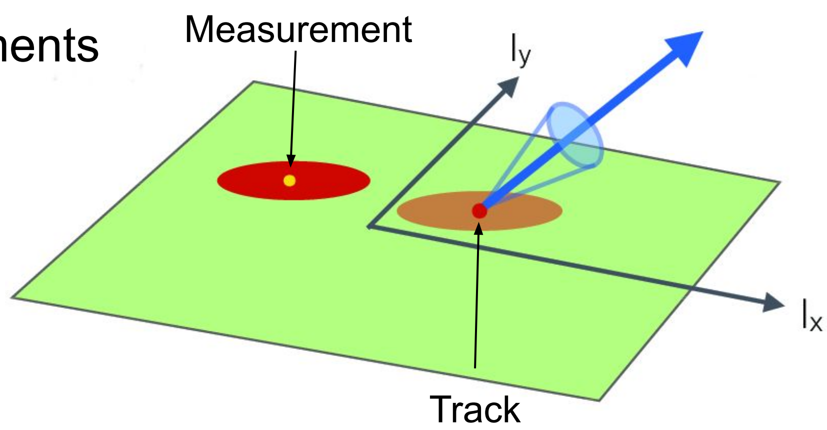
In general the transport is given by:

$$\Sigma_1^{local} = J \cdot \Sigma_0^{local} = J_{g2l} \cdot J_{transport} \cdot J_{l2g} \cdot \Sigma_0^{local}$$

Uncertainty propagation

Beside the propagation one is interested in the propagation of the uncertainties

I.e. these define the searching area for measurements



Starting with the local covariance matrix Σ_0^{local}

In general the transport is given by:

$$\Sigma_1^{local} = J \cdot \Sigma_0^{local} = J_{g2l} \cdot J_{transport} \cdot J_{l2g} \cdot \Sigma_0^{local}$$

In the StraightLine approximation this can be easily done by hand:

No change at all

$$J_{transport} = \begin{pmatrix} 1 & 0 & 0 & h & 0 & 0 \\ 0 & 1 & 0 & 0 & h & 0 \\ 0 & 0 & 1 & 0 & 0 & h \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Effect of direction on position

Propagation in magnetic fields

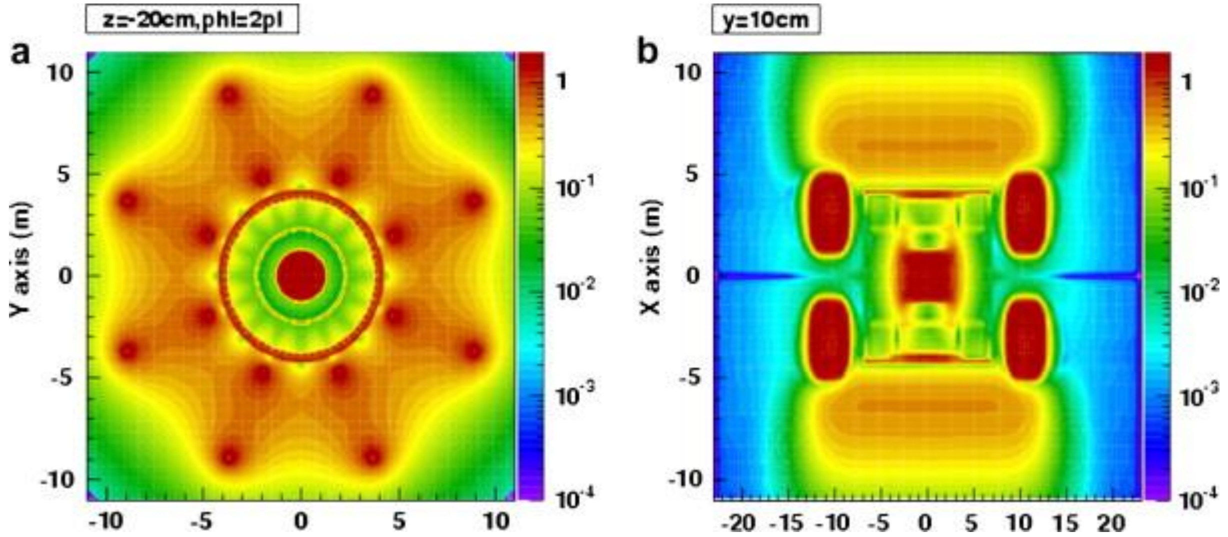
A present B-field makes the propagation more complicated

For each charged particle the StraightLine approach is invalid

The underlying equation of motion is given by the Lorentz force

$$\frac{d^2\vec{r}}{ds^2} = \frac{q}{p}\left(\frac{d\vec{r}}{ds} \times \vec{B}\left(\vec{r}\right)\right)$$

As soon as $\vec{B}\left(\vec{r}\right)$ is not constant (= common real case) there is not analytic solution



→ Numerical approach required

Runge-Kutta-Nyström integration

A numerical approach should satisfy 2 things: Accuracy and Speed

Common choice: Runge-Kutta-Nyström integration of fourth order (RKN4)

Propagates stepwise through detector using iteratively evaluated sub-steps

Runge-Kutta-Nyström integration

A numerical approach should satisfy 2 things: Accuracy and Speed

Common choice: Runge-Kutta-Nyström integration of fourth order (RKN4)

Propagates stepwise through detector using iteratively evaluated sub-steps

1. Translation of the problem into $\frac{d}{ds}\begin{pmatrix} \vec{r} \\ \vec{T} \end{pmatrix} = \begin{pmatrix} \vec{r}\,' \\ \frac{q}{p}\left(\vec{T} \times \vec{B}\left(\vec{r}\right)\right) \end{pmatrix}$

Runge-Kutta-Nyström integration

A numerical approach should satisfy 2 things: <u>Accuracy</u> and <u>Speed</u>

Common choice: Runge-Kutta-Nyström integration of fourth order (RKN4)

Propagates <u>stepwise</u> through detector using iteratively evaluated sub-steps

1. Translation of the problem into $\frac{d}{ds}\begin{pmatrix}\vec{r} \\ \vec{T}\end{pmatrix} = \begin{pmatrix}\vec{r}\,' \\ \frac{q}{p}\left(\vec{T} \times \vec{B}\left(\vec{r}\right)\right)\end{pmatrix}$ $= \vec{T}$

   $= \frac{d\vec{r}}{ds}$

Runge-Kutta-Nyström integration

A numerical approach should satisfy 2 things: Accuracy and Speed

Common choice: Runge-Kutta-Nyström integration of fourth order (RKN4)

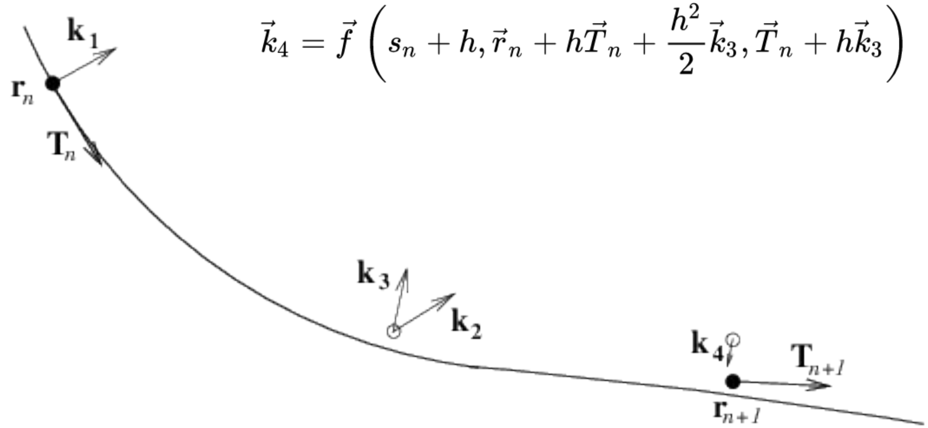
Propagates stepwise through detector using iteratively evaluated sub-steps

1. Translation of the problem into $\frac{d}{ds}\begin{pmatrix}\vec{r}\\ \vec{T}\end{pmatrix} = \begin{pmatrix}\vec{r}' \\ \frac{q}{p}\left(\vec{T}\times\vec{B}(\vec{r})\right)\end{pmatrix}$
2. Evaluating k_1-k_4

$$\vec{r}' = \vec{T}$$

$$\vec{T} = \frac{d\vec{r}}{ds}$$

$$\begin{aligned}
\vec{k}_1 &= \vec{f}\left(s_n, \vec{r}_n, \vec{T}_n\right)\\
\vec{k}_2 &= \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{2}\vec{k}_1, \vec{T}_n + \frac{h}{2}\vec{k}_1\right)\\
\vec{k}_3 &= \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{8}\vec{k}_2, \vec{T}_n + \frac{h}{2}\vec{k}_2\right)\\
\vec{k}_4 &= \vec{f}\left(s_n + h, \vec{r}_n + h\vec{T}_n + \frac{h^2}{2}\vec{k}_3, \vec{T}_n + h\vec{k}_3\right)
\end{aligned}$$



Runge-Kutta-Nyström integration

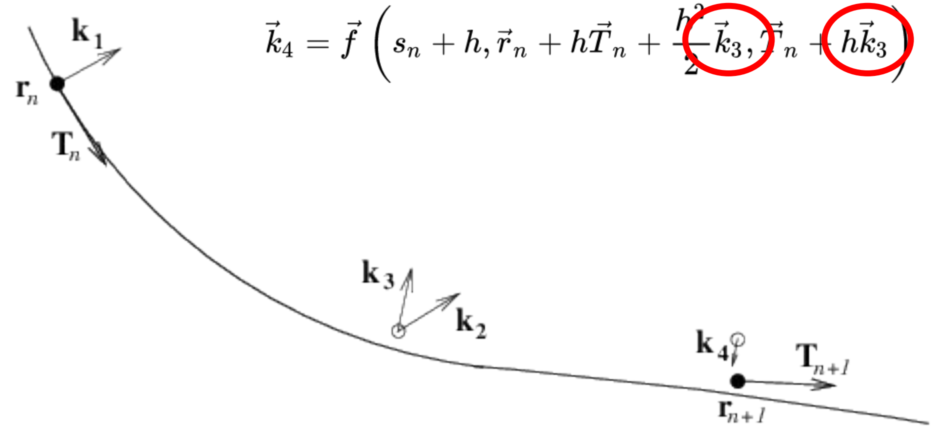
A numerical approach should satisfy 2 things: Accuracy and Speed

Common choice: Runge-Kutta-Nyström integration of fourth order (RKN4)

Propagates stepwise through detector using iteratively evaluated sub-steps

1. Translation of the problem into $\frac{d}{ds}\begin{pmatrix}\vec{r}\\ \vec{T}\end{pmatrix} = \begin{pmatrix}\vec{r}' \\ \frac{q}{p}\left(\vec{T}\times\vec{B}\left(\vec{r}\right)\right)\end{pmatrix}$ $\quad \vec{r}' = \vec{T}$, $\quad \vec{T} = \frac{d\vec{r}}{ds}$
2. Evaluating k_1-k_4

$$\vec{k}_1 = \vec{f}\left(s_n, \vec{r}_n, \vec{T}_n\right)$$
$$\vec{k}_2 = \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{2}\vec{k}_1, \vec{T}_n + \frac{h}{2}\vec{k}_1\right)$$
$$\vec{k}_3 = \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{8}\vec{k}_2, \vec{T}_n + \frac{h}{2}\vec{k}_2\right)$$
$$\vec{k}_4 = \vec{f}\left(s_n + h, \vec{r}_n + h\vec{T}_n + \frac{h^2}{2}\vec{k}_3, \vec{T}_n + h\vec{k}_3\right)$$



Runge-Kutta-Nyström integration

A numerical approach should satisfy 2 things: Accuracy and Speed

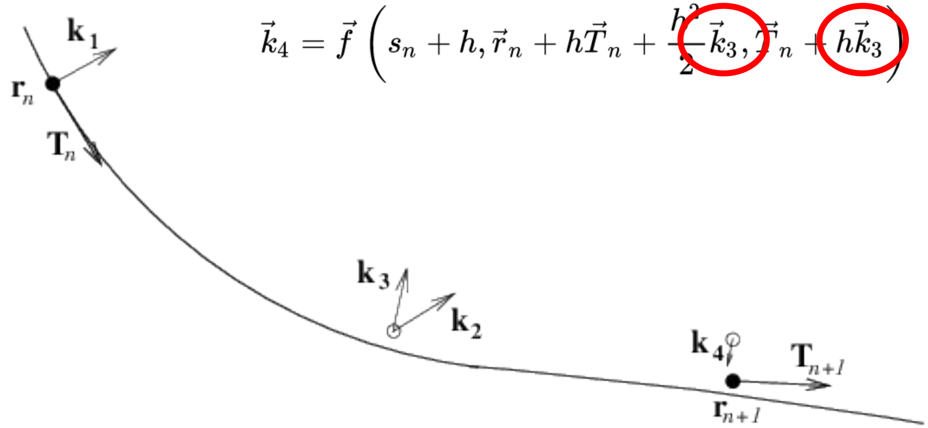
Common choice: Runge-Kutta-Nyström integration of fourth order (RKN4)

Propagates stepwise through detector using iteratively evaluated sub-steps

1. Translation of the problem into $\frac{d}{ds}\begin{pmatrix}\vec{r}\\ \vec{T}\end{pmatrix} = \begin{pmatrix}\vec{r}\,'\\ \frac{q}{p}\left(\vec{T}\times\vec{B}\left(\vec{r}\right)\right)\end{pmatrix}$, $\vec{r}\,' = \vec{T}$, $\vec{T} = \frac{d\vec{r}}{ds}$
2. Evaluating k_1-k_4
3. Updating the parameters

$$\vec{k}_1 = \vec{f}\left(s_n, \vec{r}_n, \vec{T}_n\right)$$
$$\vec{k}_2 = \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{2}\vec{k}_1, \vec{T}_n + \frac{h}{2}\vec{k}_1\right)$$
$$\vec{k}_3 = \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{8}\vec{k}_2, \vec{T}_n + \frac{h}{2}\vec{k}_2\right)$$
$$\vec{k}_4 = \vec{f}\left(s_n + h, \vec{r}_n + h\vec{T}_n + \frac{h^2}{2}\vec{k}_3, \vec{T}_n + h\vec{k}_3\right)$$

$$\vec{T}_{n+1} = \vec{T}_n + \frac{h}{6}\left(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4\right)$$
$$\vec{r}_{n+1} = \vec{r}_n + h\vec{T}_n + \frac{h^2}{6}\left(\vec{k}_1 + \vec{k}_2 + \vec{k}_3\right)$$



Runge-Kutta-Nyström integration

A numerical approach should satisfy 2 things: Accuracy and Speed

Common choice: Runge-Kutta-Nyström integration of fourth order (RKN4)

Propagates stepwise through detector using iteratively evaluated sub-steps

1. Translation of the problem into $\frac{d}{ds}\begin{pmatrix}\vec{r}\\ \vec{T}\end{pmatrix} = \begin{pmatrix}\vec{r}\,' \\ \frac{q}{p}\left(\vec{T}\times\vec{B}\left(\vec{r}\right)\right)\end{pmatrix}$ $= \vec{T}$ $= \frac{d\vec{r}}{ds}$
2. Evaluating k_1-k_4
3. Updating the parameters

$$\vec{T}_{n+1} = \vec{T}_n + \frac{h}{6}\left(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4\right)$$

$$\vec{r}_{n+1} = \vec{r}_n + h\vec{T}_n + \frac{h^2}{6}\left(\vec{k}_1 + \vec{k}_2 + \vec{k}_3\right)$$

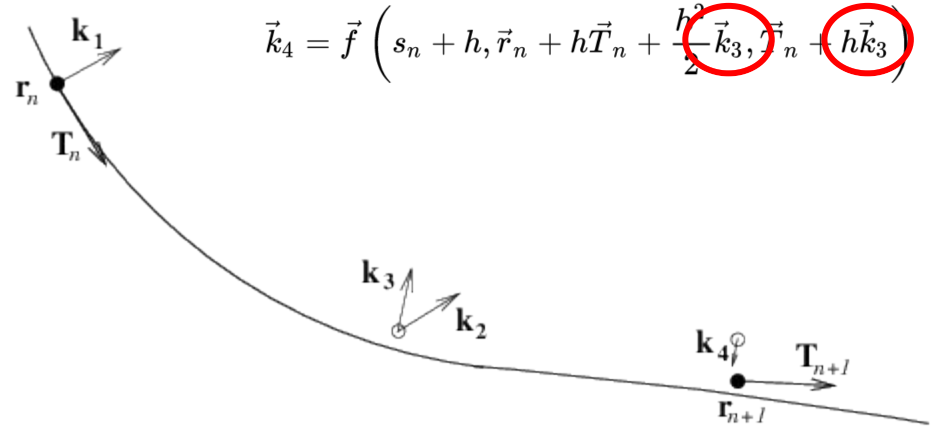
StraightLine solution

$$\vec{k}_1 = \vec{f}\left(s_n, \vec{r}_n, \vec{T}_n\right)$$

$$\vec{k}_2 = \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{2}\vec{k}_1, \vec{T}_n + \frac{h}{2}\vec{k}_1\right)$$

$$\vec{k}_3 = \vec{f}\left(s_n + \frac{h}{2}, \vec{r}_n + \frac{h}{2}\vec{T}_n + \frac{h^2}{8}\vec{k}_2, \vec{T}_n + \frac{h}{2}\vec{k}_2\right)$$

$$\vec{k}_4 = \vec{f}\left(s_n + h, \vec{r}_n + h\vec{T}_n + \frac{h^2}{2}\vec{k}_3, \vec{T}_n + h\vec{k}_3\right)$$



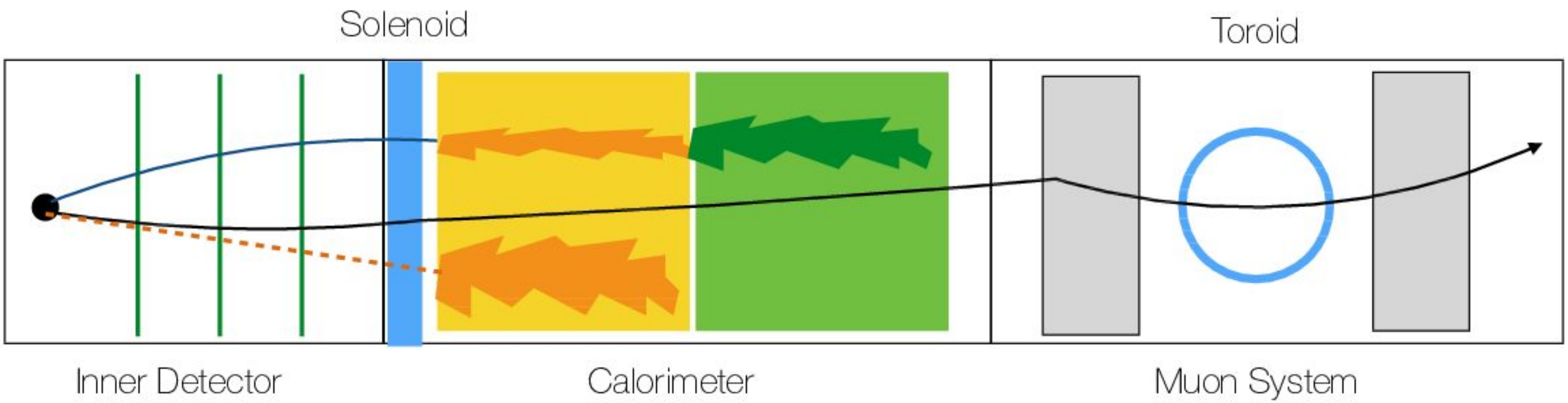
Propagation in matter

Beside the data from the inner detector we are also interested in data from Muon system

- This requires to extrapolate <u>through the calorimeter</u>

In the inner detector the momentum is constant along the propagation between surfaces

- Now we need to consider the energy loss along the propagation as well



Propagation in matter

Beside the data from the inner detector we are also interested in data from Muon system

This requires to extrapolate through the calorimeter

In the inner detector the momentum is constant along the propagation between surfaces

Now we need to consider the energy loss along the propagation as well

The energy loss is a composition of multiple effects:

$$g = \underbrace{\langle \frac{dE}{ds} \rangle_{Bethe-Bloch}}_{\text{Excitation \& ionisation}} + \underbrace{\langle \frac{dE}{ds} \rangle_{Bethe-Heitler}}_{\text{Bremsstrahlung}} + \underbrace{\langle \frac{dE}{ds} \rangle_{\text{Direct pair production + Photonuclear interaction}}}_{\text{Muon specific effects}}$$

Each effect depends on the current

- energy or velocity (directly)
- and the position (indirectly in material look-up)

$$\longrightarrow \frac{dq/p}{ds} = -\frac{qE}{p^3} g(q, p, \vec{r})$$

Propagation in matter

Beside the data from the inner detector we are also interested in data from Muon system

This requires to extrapolate through the calorimeter

In the inner detector the momentum is constant along the propagation between surfaces

Now we need to consider the energy loss along the propagation as well

The energy loss is a composition of multiple effects:

$$g = \underbrace{\langle \frac{dE}{ds} \rangle_{Bethe-Bloch}}_{\text{Excitation \& ionisation}} + \underbrace{\langle \frac{dE}{ds} \rangle_{Bethe-Heitler}}_{\text{Bremsstrahlung}} + \underbrace{\langle \frac{dE}{ds} \rangle_{\text{Direct pair production + Photonuclear interaction}}}_{\text{Muon specific effects}}$$

Each effect depends on the current

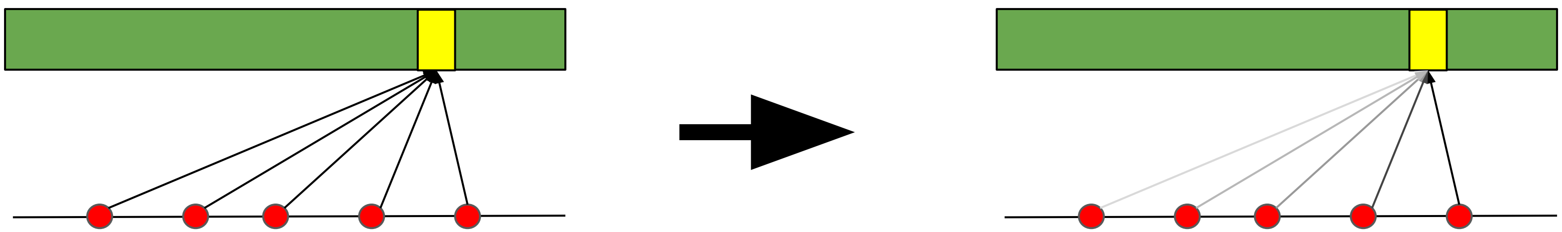
- energy or velocity (directly)
- and the position (indirectly in material look-up)

$$\longrightarrow \frac{dq/p}{ds} = -\frac{qE}{p^3} g(q, p, \vec{r})$$

This cannot be treated discretised and is therefore treated by the RKN4

Propagation in time

Using a timestamp on each measurement allows to filter measurements and therewith reduce the complexity of the event



Equal likelihood for each track — Improved likelihood by new dimension

How does the time t change along the path s? $\frac{dt}{ds} = \frac{1}{\beta} = \frac{E}{p}$

Discrete material: This can be evaluated immediately (dt/ds = const)

Continuous material: p in RKN4 formalism $\longrightarrow$ t propagated in the same way

$\longrightarrow$ The total parameter set then will be: $\vec{r} = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} \quad \vec{T} = \begin{pmatrix} T^x \\ T^y \\ T^z \\ \frac{E}{p} \end{pmatrix}$

@Acts: It's currently q/p

Covariance matrix in RKN4

Evaluating $J_{transport}$ in the StraightLine approach for the inner detector was simple

It was also assumed that a single step is enough to reach a surface

RKN4 requires usually $\overbrace{\text{multiple steps}}^{\text{Numerical integration has step-size dependent error}}$ - the total transport Jacobian will then become

$$J_{transport} = \Pi_i J^i_{transport}$$

Covariance matrix in RKN4

Evaluating $J_{transport}$ in the StraightLine approach for the inner detector was simple

It was also assumed that a single step is enough to reach a surface

Numerical integration has step-size dependent error

RKN4 requires usually multiple steps - the total transport Jacobian will then become

$$J_{transport} = \Pi_i J^i_{transport}$$

The individual components of $J^n_{transport}$ are build by deriving

$$\vec{T}_{n+1} = \vec{T}_n + \frac{h}{6}\left(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4\right)$$

$$\vec{r}_{n+1} = \vec{r}_n + h\vec{T}_n + \frac{h^2}{6}\left(\vec{k}_1 + \vec{k}_2 + \vec{k}_3\right)$$

and the equations of motions $\frac{d^2\vec{r}}{ds^2} = \frac{q}{p}\left(\frac{d\vec{r}}{ds} \times \vec{B}\left(\vec{r}\right)\right)$ $\qquad \frac{dq/p}{ds} = -\frac{qE}{p^3}g(q,p,\vec{r})$

$$\frac{dt}{ds} = \frac{1}{\beta} = \frac{E}{p}$$

Covariance matrix in RKN4

Evaluating $J_{transport}$ in the StraightLine approach for the inner detector was simple

It was also assumed that a single step is enough to reach a surface

RKN4 requires usually $\overbrace{\text{multiple steps}}^{\text{Numerical integration has step-size dependent error}}$ - the total transport Jacobian will then become

$$J_{transport} = \Pi_i J^i_{transport}$$

The individual components of $J^n_{transport}$ are build by deriving

$$\vec{T}_{n+1} = \vec{T}_n + \frac{h}{6}\left(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4\right)$$

$$\vec{r}_{n+1} = \vec{r}_n + h\vec{T}_n + \frac{h^2}{6}\left(\vec{k}_1 + \vec{k}_2 + \vec{k}_3\right)$$

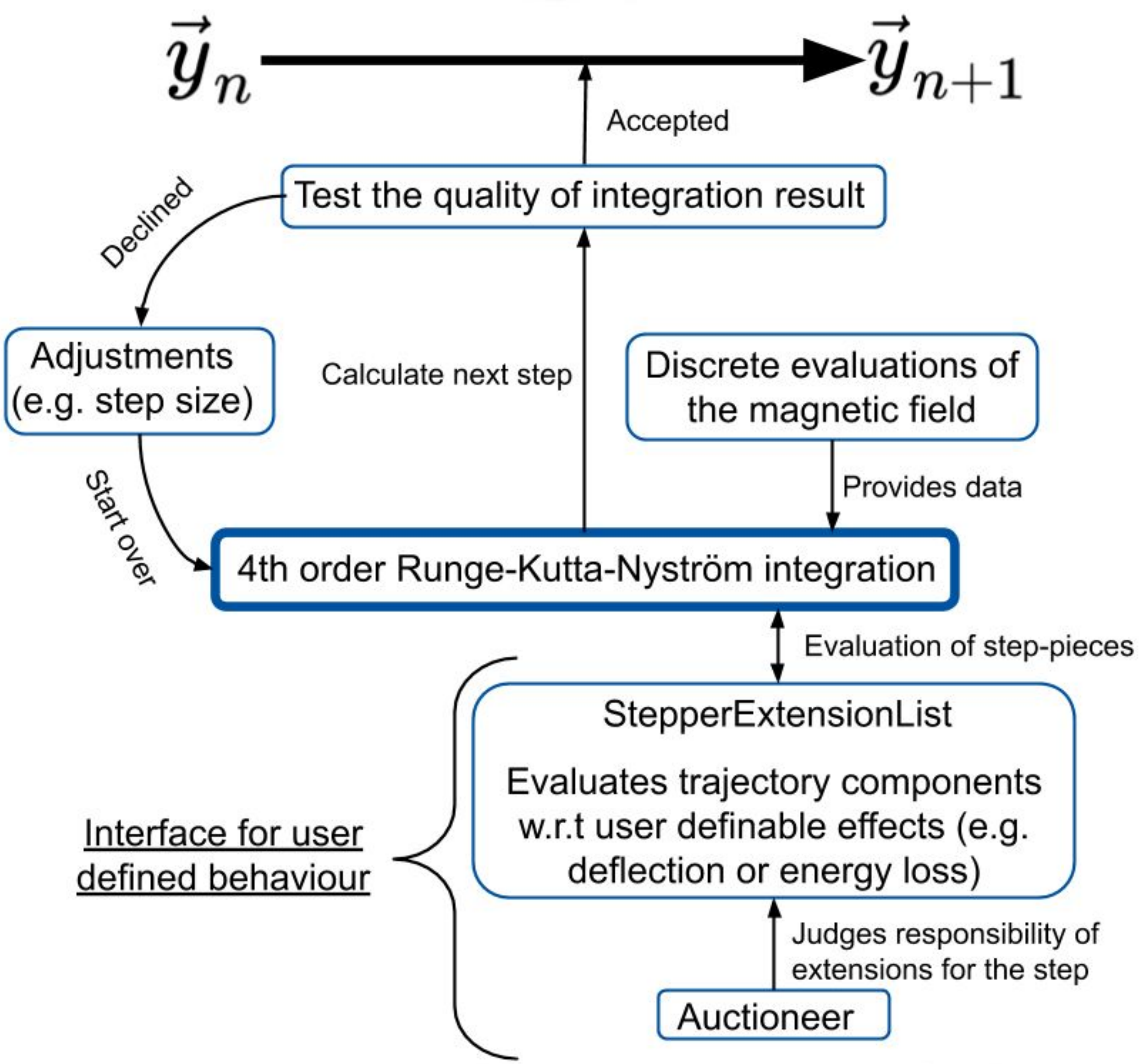
and the equations of motions $\frac{d^2\vec{r}}{ds^2} = \frac{q}{p}\left(\frac{d\vec{r}}{ds} \times \vec{B}(\vec{r})\right)$ $\quad \frac{dq/p}{ds} = -\frac{qE}{p^3} g(q, p, \vec{r})$

$$\frac{dt}{ds} = \frac{1}{\beta} = \frac{E}{p}$$

In short:

$$\boxed{\text{This is a \underline{huge matrix} if written-out completely and \underline{computationally expensive!}}}$$

Stepping in a bigger picture



Summary

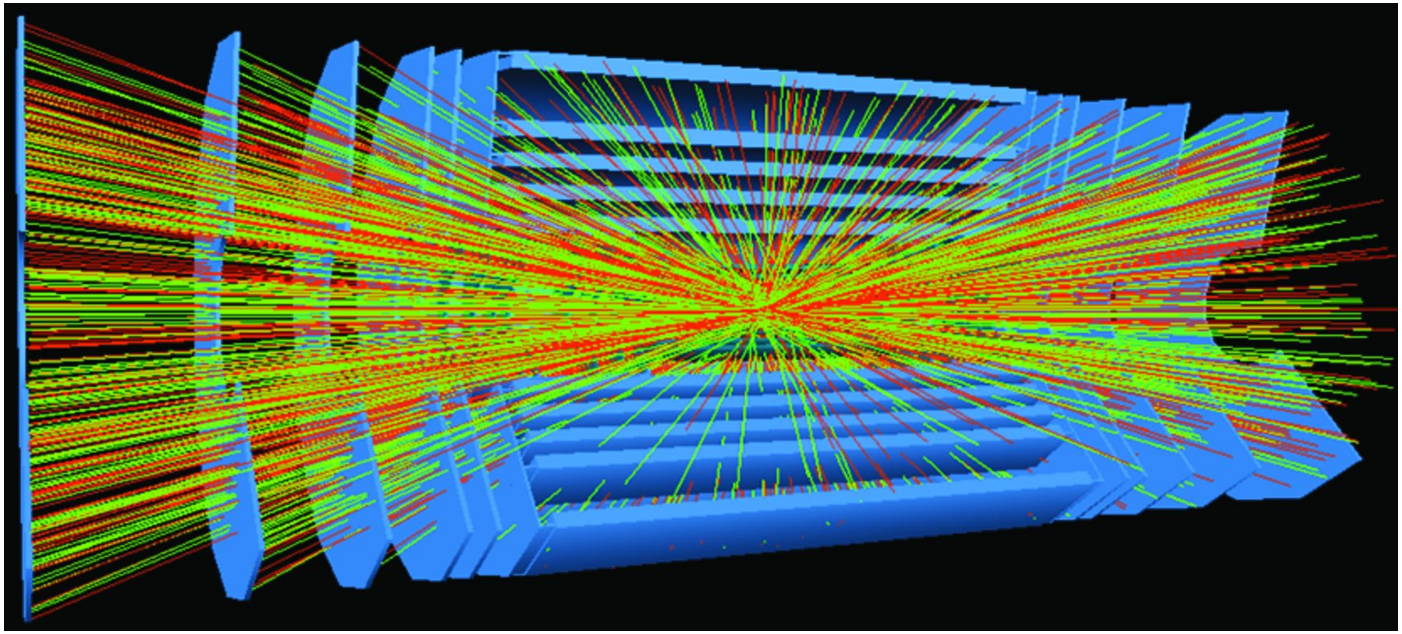
- Tracking is crucial to understand a HEP event
- The goal of Acts is to become here a “HEP standard”

- Measurement-particle association requires knowledge of particle properties at detector modules
- In simple cases can be approximated by StraightLine
- Requires numerical approaches in general ⟶ Computationally heavy
- In Acts a single setup adapts its stepping based on its environment

Backup

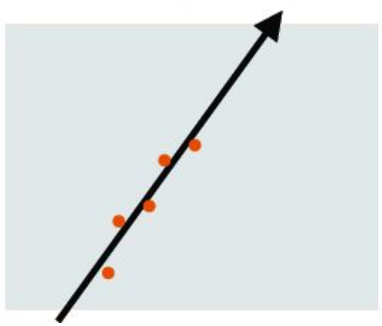
Terminology

- In a particle collision like it is performed by the LHC many particles are created
- Event: The combination of in- and outgoing particles
- Measurement: Signals/interactions of certain particles in certain detector parts
- ATLAS: 10^5-10^6 measurements per event in the tracker called innermost part
- Track reconstruction/Tracking: Multiple interactions of particles during their propagation through the detector allow the reconstruction of their trajectory

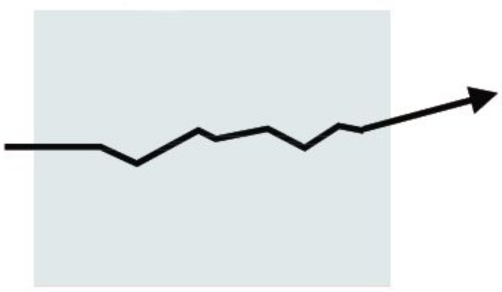


Interaction of particles and detectors

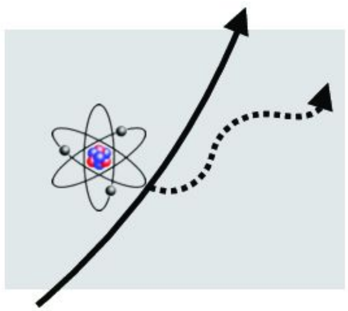
- Particles need to interact with the detector → measurement
- otherwise they are not reconstructed in the event (e.g. neutrinos)
- Interactions categorised in



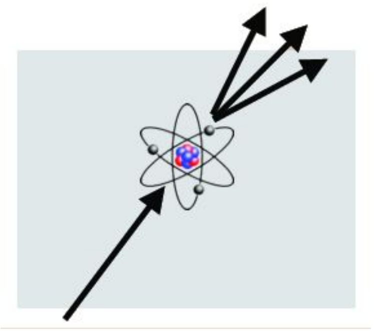
Ionisation



Scattering



Bremsstrahlung



Hadronic interaction

HEP-SPEC06 Results for SL7 x86_64 (gcc 4.8)

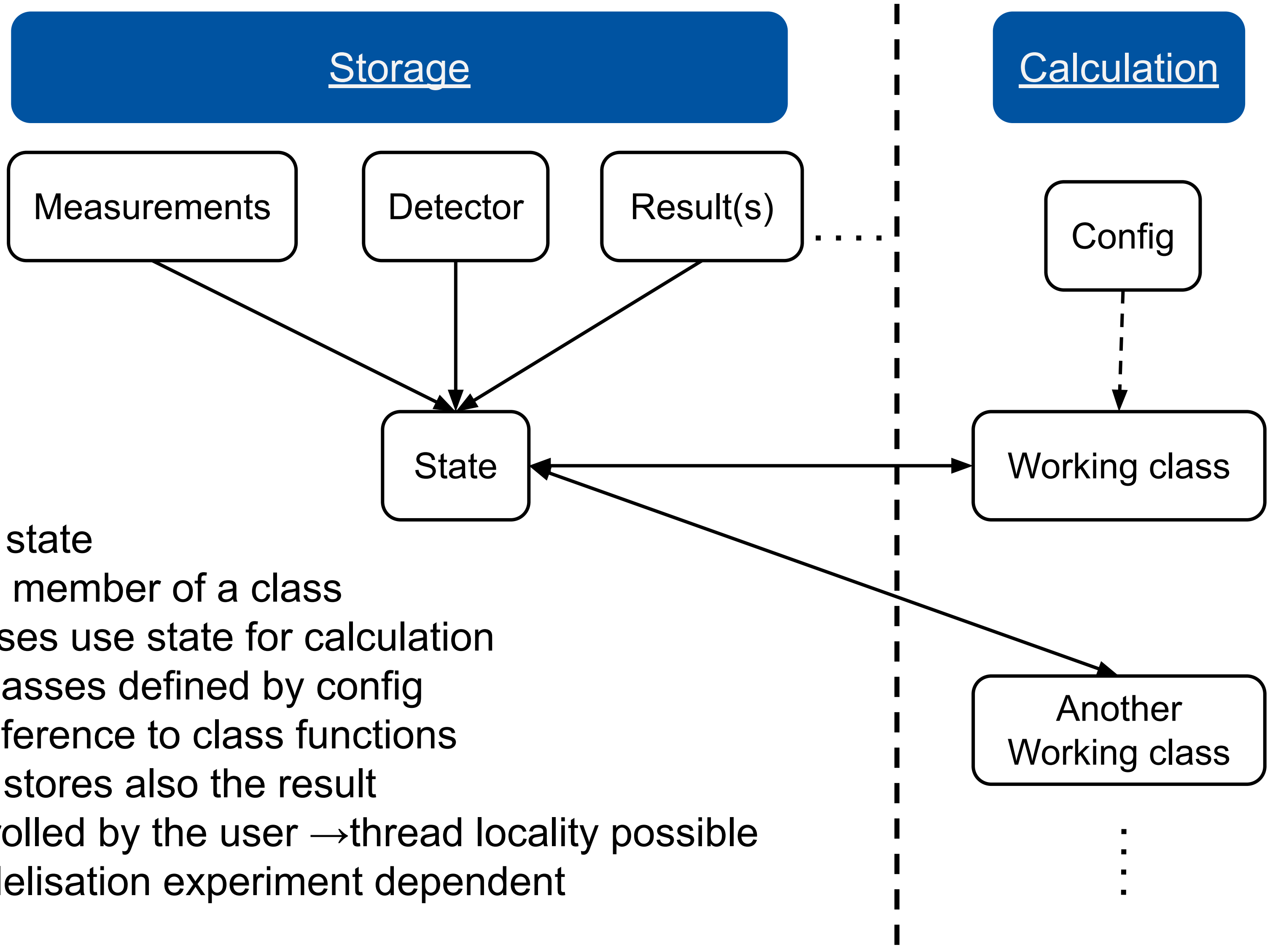
Benchmark Environment

Operating system:	Scientific Linux 7 / CentOS 7 x86_64
Compiler package:	gcc-4.8.x (default compiler)
Compiler flags:	-O2 -pthread -fPIC -m32

Benchmark Results

CPU	HS06	Clock speed (MHz)	L2+L3 cache size (grand total, KB)	Cores (runs)	Memory (GB)	Mainboard type	Site
Intel Xeon E5-2660v3	488	2600	5120+51200	40 HT on	256 (16x16 PC4-2133)	Huawei CH121 V3	(GridKa)
Intel Xeon E5-4669v4	1836	2200	22528+225280	176 (HT on)	512 (16x32 PC4-2400)	Dell FC830	(GridKa)
Intel Xeon E5-2699v4	987	2200	11264+112640	88 (HT on)	512 (16x32 PC4-2400)	Dell R730	(GridKa)
Intel Xeon E5-2620v4	305	2100	4096+40960	32	64 (8 modules)	Dell 082F9M	UKI-NORTHGRID-MAN-HEP
Intel Xeon Gold 6130	577	2100	32768+45056	32 (HT off)	192 (12 modules)	Dell 0K2TT6	UKI-NORTHGRID-MAN-HEP
Intel Xeon Gold 6130	717	2100	32768+45056	64 (HT on)	192 (12 modules)	Dell 0K2TT6	UKI-NORTHGRID-MAN-HEP

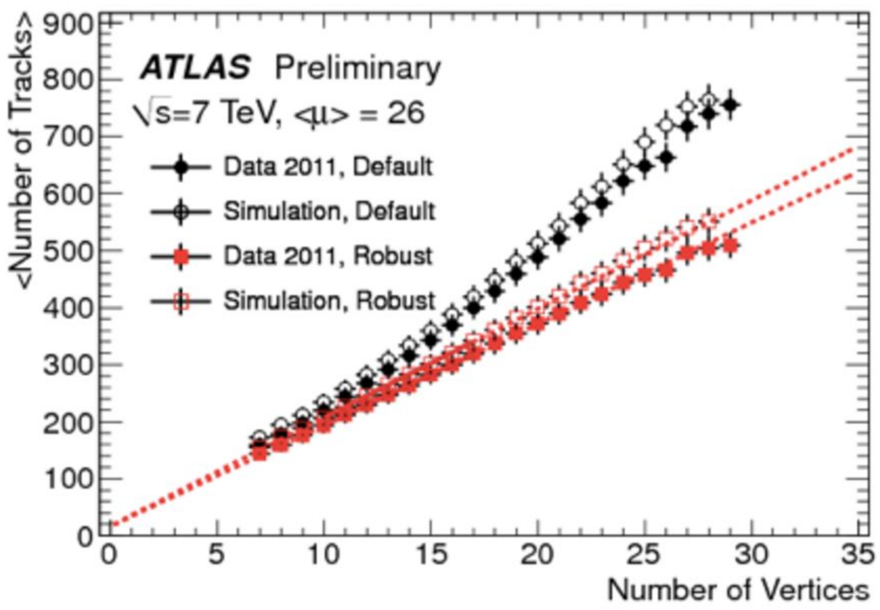
Software structure

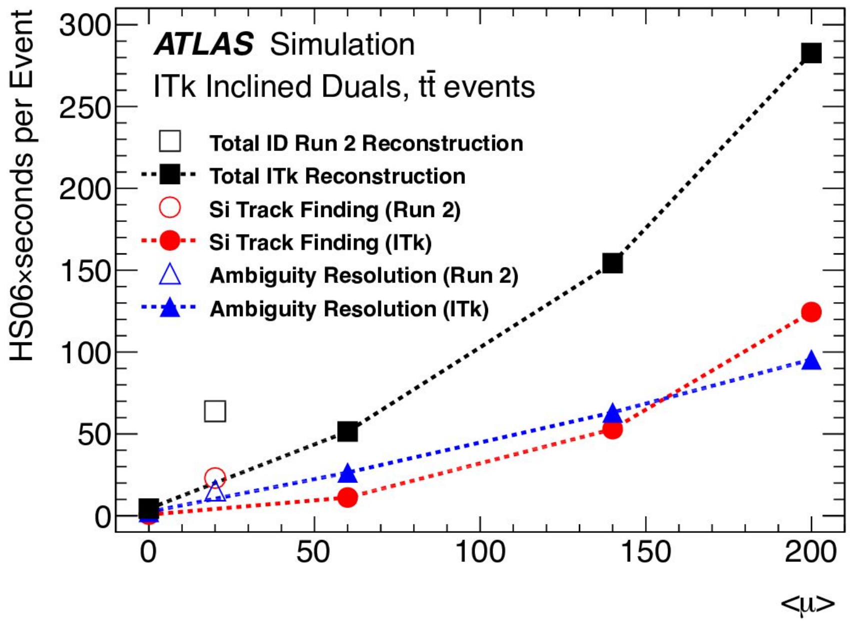


- User defined state
- State is not a member of a class
- Working classes use state for calculation
- Steering of classes defined by config
- Passed as reference to class functions
  → State stores also the result
- State is controlled by the user →thread locality possible
- Explicit parallelisation experiment dependent

Mission statement 2: Complexity

1. ATLAS uses combinatorial complexity algorithms to reconstruct tracks
2. In HL-LHC roughly 200 vertices per collision





→ Increasing rate of fake-tracks due to combinatorics

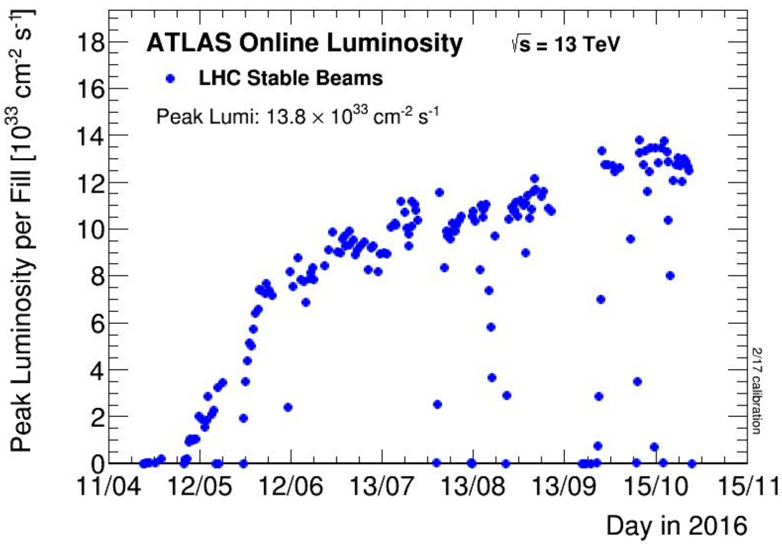
⇒More efficient fake candidate rejection

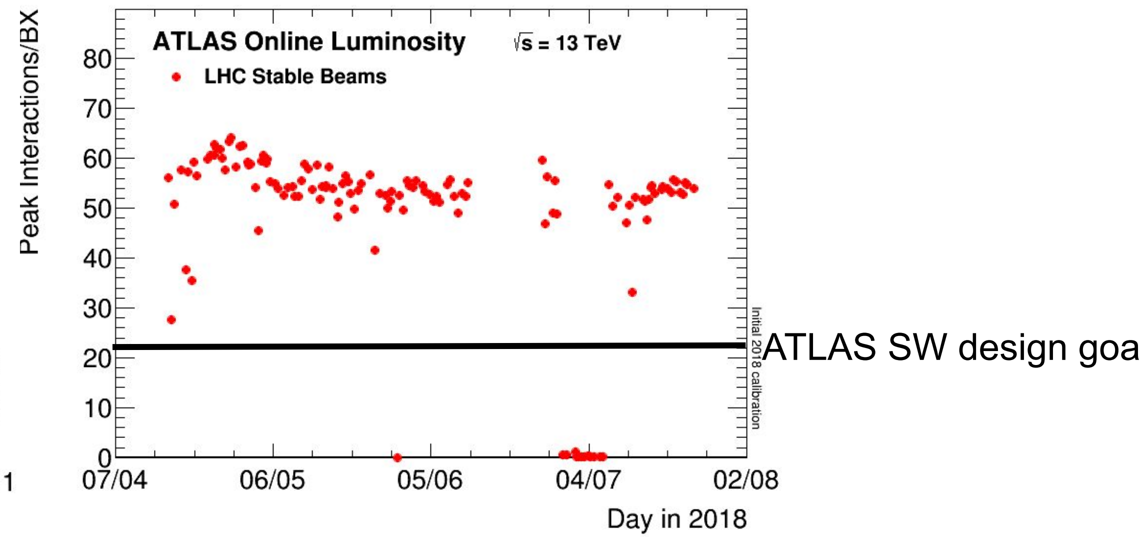
→ Reconstruction time is going to explode

⇒Better usage of computing resources to become faster

Mission statement 2: Complexity

To collect decent statistics for the physics programme at reasonable time-scales, LHC needs to run with high instantaneous luminosity ($\mathscr{L} \approx 10^{34}$ cm^{-2}s^{-1})





Increasing the luminosity leads to more vertices & tracks in the same events

We're already running beyond the design specifications of the ATLAS tracking software

Athena

- Modular software framework
- Used in ATLAS for offline event reconstruction
- $\approx\mathcal{O}(6\text{M})$ lines of code (≡ 200k DIN A4 pages)
- ≈ 20 years old (≡ 7 generations of PhD Students)

→Gigantic project grown over many generations of developers

= few have a total overview = many (undocumented) code fragments

➢ Optimising legacy code provides problems, too:

  - Lost of knowledge
  - Outdated programming patterns
  - No flexibility in the application and the order of execution
    →Integral structures cannot be changed

➢ Most beneficial in the long run is rewriting the tracking code from scratch

Components of track reconstruction

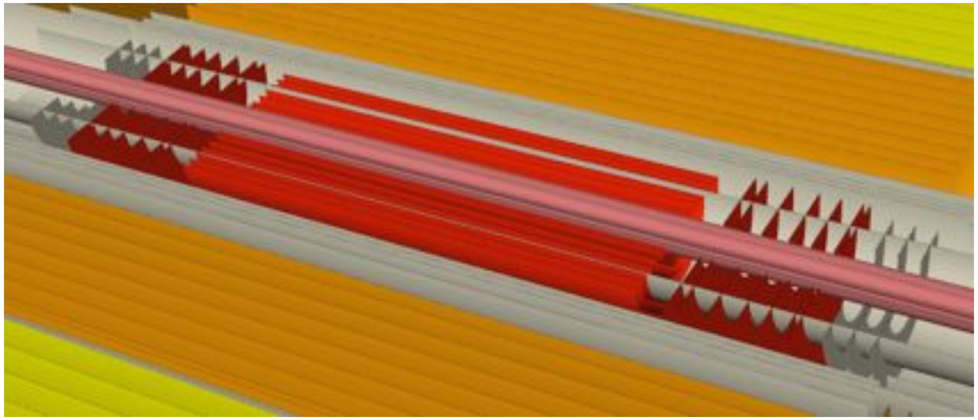
Tracking software

Geometry & magnetic field	Event data model	Algorithms
Material	Track data	Track fitting
Detector components	Measurements	Pattern recognition
Magn. field	Vertex	Seeding
Detector conditions*	Truth data*	Truth association*

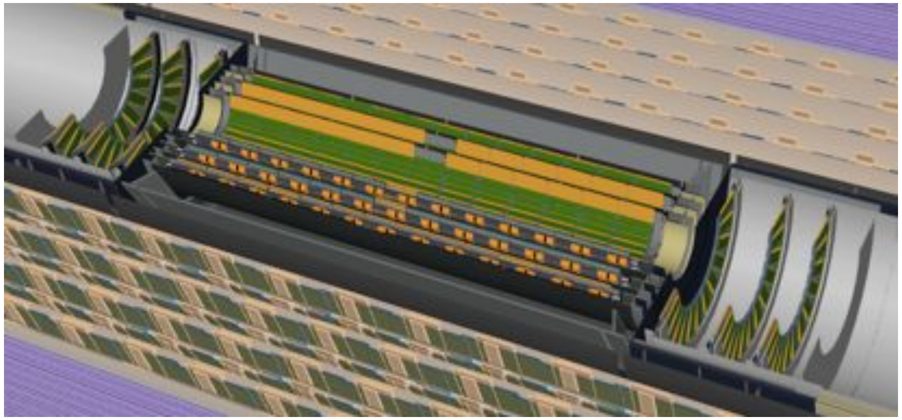
Detector representations

- Track reconstruction consists of 2 different parts:
  - Collecting measurements that belong to a track (fast)
  - Fitting the track (precise, full geometry)
- These techniques rely on 2 different representations of a detector:

FastSim: Coarse granularity



Geant4: Fine granularity

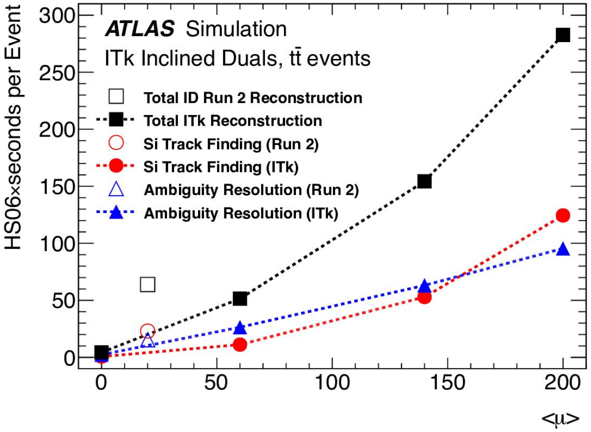


ATLAS ttbar event in kSI2k sec:

FastSim: 7.4

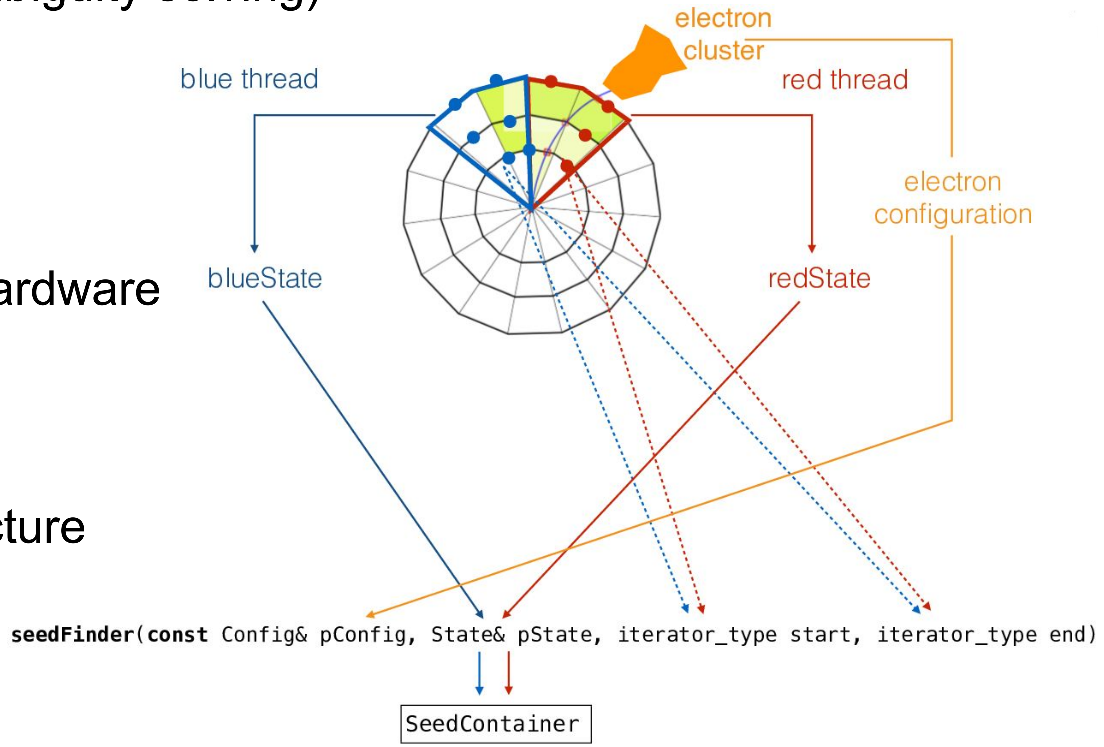
Geant4: 1990

Parallelised track reconstruction



- Runtime improvement by parallelisation
- Sharing detector & splitting event into segments
  - →Data locality!
  - →Parallel processing of events
- Results need to be merged (ambiguity solving)

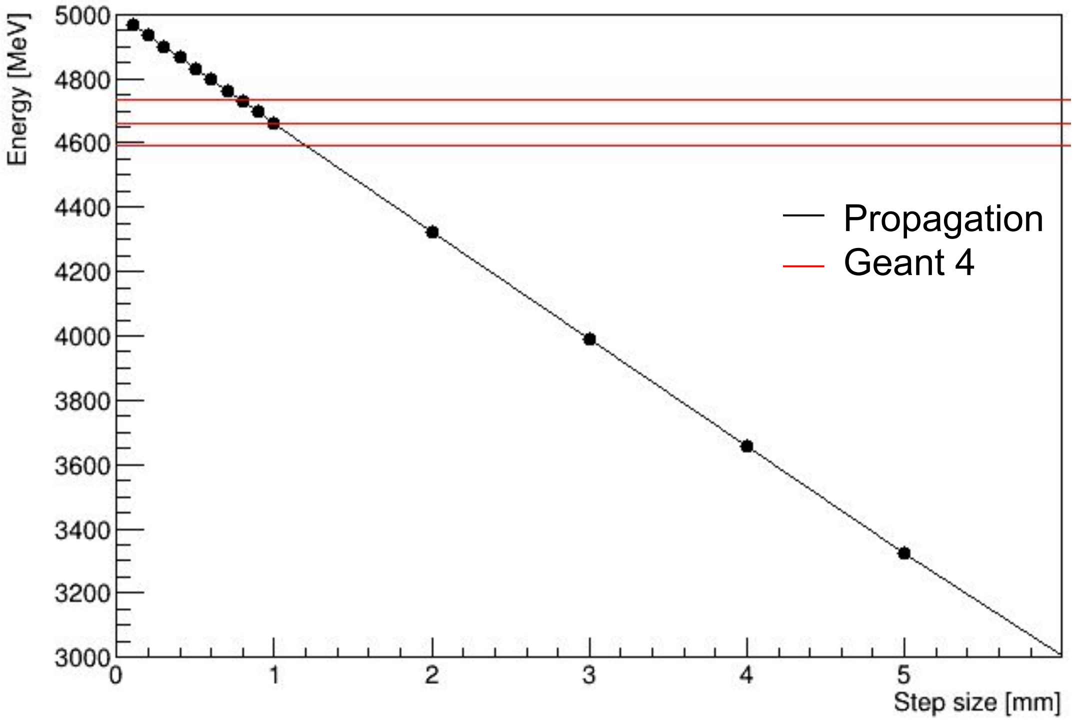
- Best splitting strategy?
  - Minimal communication
  - Optimised for computing hardware
  - Detector dependent
  - → Depends on the experiment
  - → ACTS provides the infrastructure



Propagation in matter

$$g = \underbrace{\langle \frac{dE}{ds} \rangle_{Bethe-Bloch}}_{\text{Excitation \& ionisation}} + \underbrace{\langle \frac{dE}{ds} \rangle_{Bethe-Heitler}}_{\text{Bremsstrahlung}} + \underbrace{\langle \frac{dE}{ds} \rangle_{\text{Direct pair production + Photonuclear interaction}}}_{\text{Muon specific effects}}$$

$$\frac{dq/p}{ds} = -\frac{qE}{p^3} g(q, p, \vec{r})$$



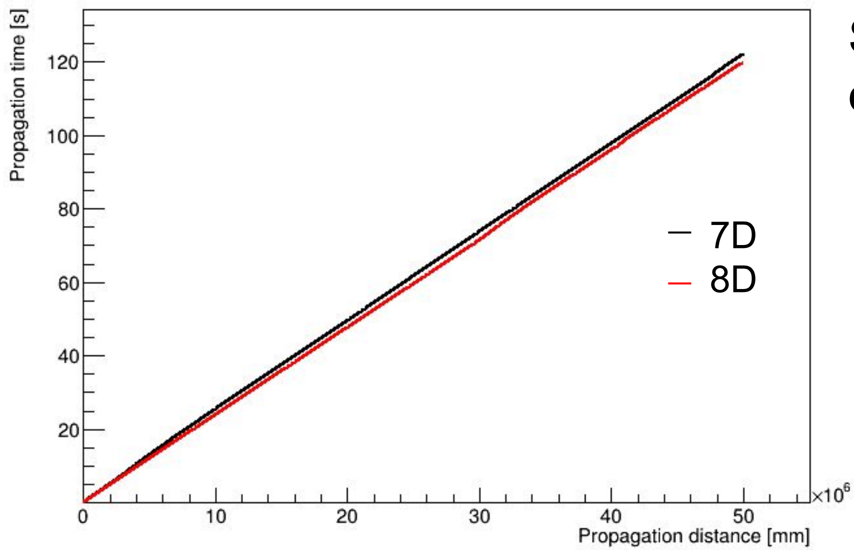
Time needed for the time

Propagating the time as parameter requires additional calculations:

This increases the

1. parameter vector by 1 component (from 7 to 8)
2. transport Jacobian by 15 components (from 7x7 to 8x8)

but the resulting time for the propagation improves



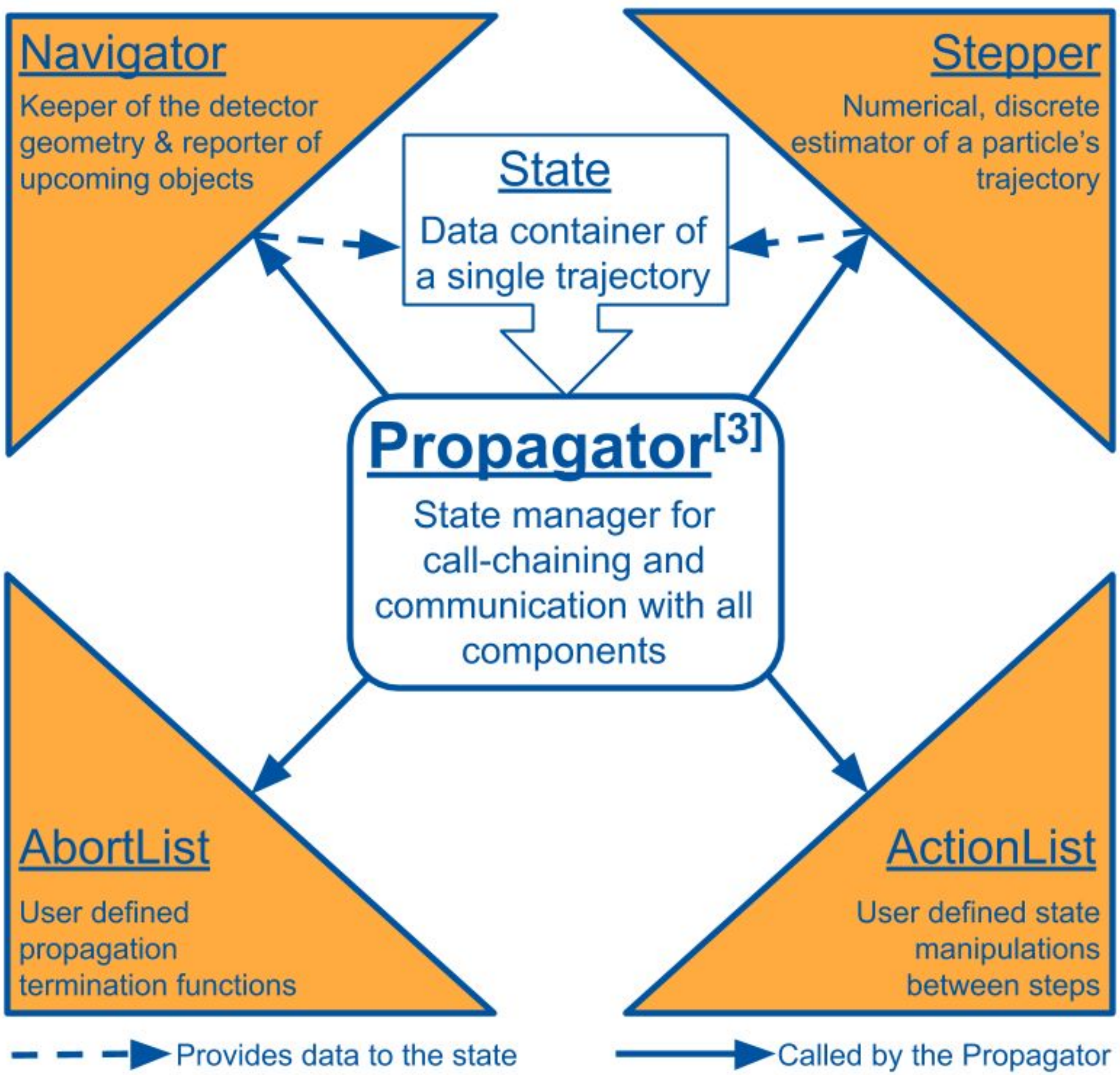
Sources should be <u>caching effects</u> or <u>compiler optimisations</u>

This difference is just a result of the covariance transport

Could be even further improved if $\vec{T} = \begin{pmatrix} T^x \\ T^y \\ T^z \\ \frac{q}{p} \end{pmatrix} \rightarrow \begin{pmatrix} T^x \\ T^y \\ T^z \\ \frac{E}{p} \end{pmatrix}$

Better vectorisation

Propagation in a bigger picture



Status of ACTS

Tracking					
Geometry (TrkDetDescr)	Event Data Model (TrkEvent)	Extrapolation (TrkExtrapolation)	Fitting (TrkFitter)	Calibration, general (TrkTools)	Alignment (TrkAlignment)
95 %	60 %	60 %	35 %	0 %	0 %

InnerDetector			Calorimeter	MuonSpectrometer		
EDM (InDetRecEvent)	Geometry (InDetTrackingGeometry)	Tools (InDetRecTools)	Geometry (CaloTrackingGeometry)	EDM (MuonRecEvent)	Geometry (MuonTrackingGeometry)	Tools (MuonRecTools)
35 %	95 %	35 %	0 %	0 %	0 %	0 %

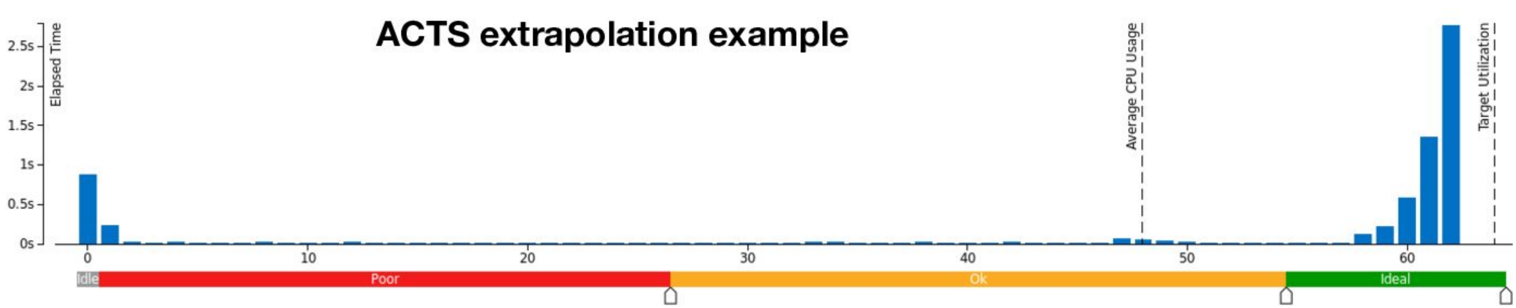
Status from march 2018

CERN openlab

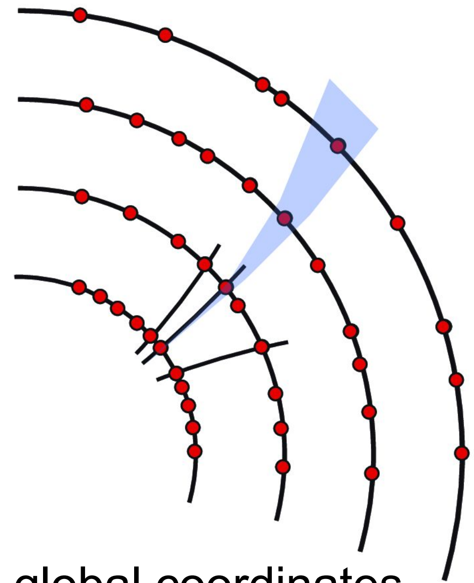
Intel Xeon e5-2698 v3, 2 sockets
32 Cores, 2 threads per core
64 Processors

CPU Usage Histogram

This histogram displays a percentage of the wall time the specific number of CPUs were running simultaneously. Spin and Overhead time adds to the Idle CPU usage value.



Parameter transformation



Starting point is a collection of measurements

$\longrightarrow$ Parameters in local coordinates $\vec{y}_0^{local}$

TODO: mit folie davor kombinieren

Parametrisation bound to surface

- > Extension of trajectory in local coordinates easier in global coordinates

Requires transformation $\vec{y}_0^{global} = \underbrace{J_{l2g}}_{\text{Projection Jacobian from local to global coordinates}} \cdot \vec{y}_0^{local}$

Projection Jacobian from local to global coordinates $J_{l2g} = \frac{\partial \vec{y}_0^{global}}{\partial \vec{y}_0^{local}}$

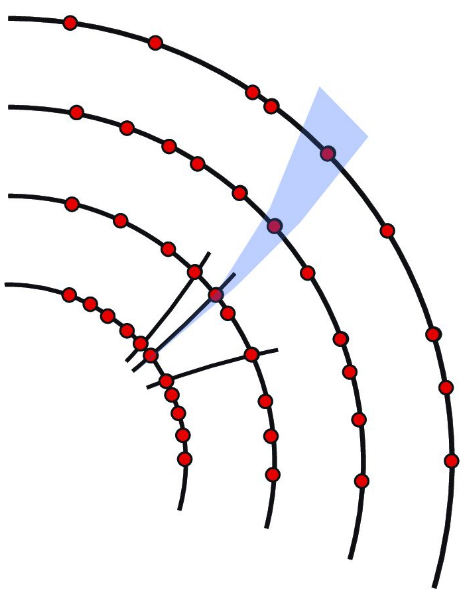
The transformation depends on the shape of the surface

but exists also in the other direction: $J_{g2l} \cdot J_{l2g} = 1$

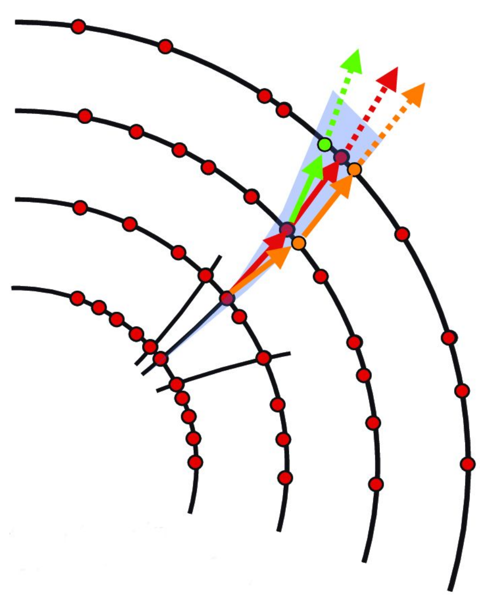
Progressive tracking - The Kalman filter

Association is performed by

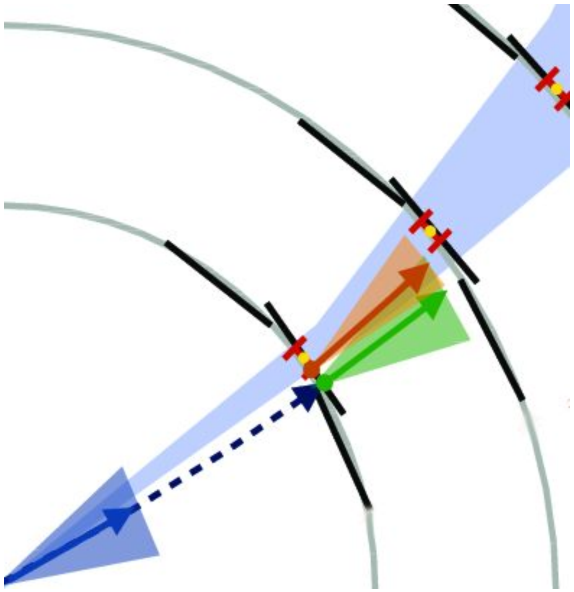
1. using an initial guess of the particle properties (= seed)
2. extending the trajectory
3. searching for corresponding measurements along the way
4. update the guess with the new data



Seeding

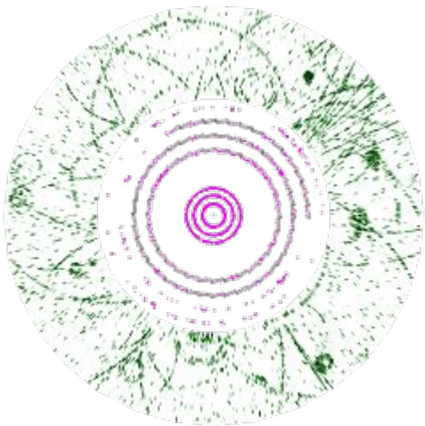
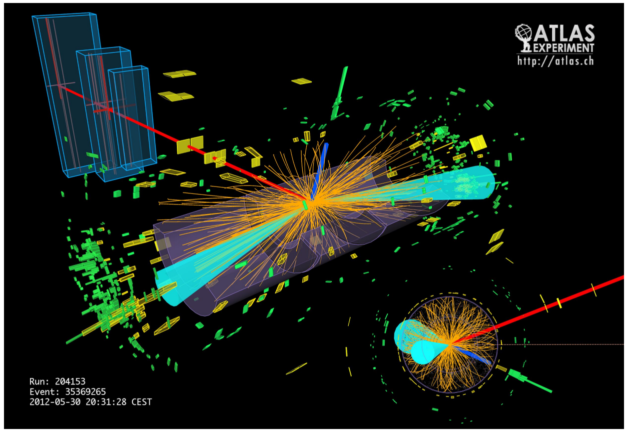


Track finding

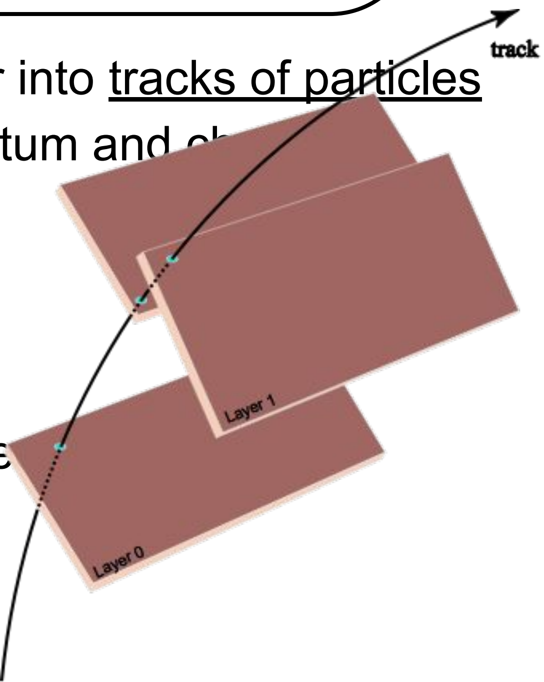


Track fitting

Mission statement: Tracking

Converts  into 

- Tracking converts measurements from the tracking detector into tracks of particles
- Tracks allow to estimate particle properties such as momentum and ch

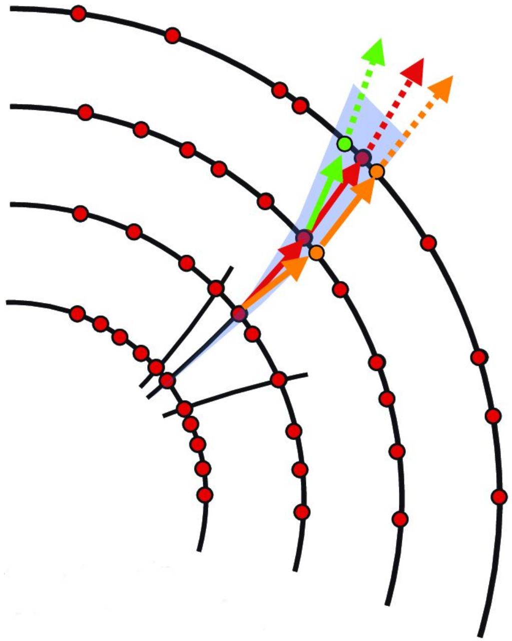


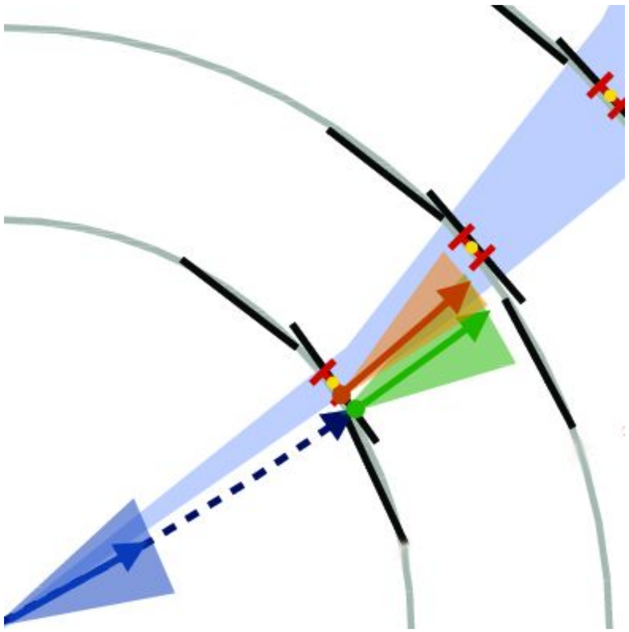
→ Tracking is an inevitable step of event reconstruction

… but also the computationally most expensive (se

Detector representations

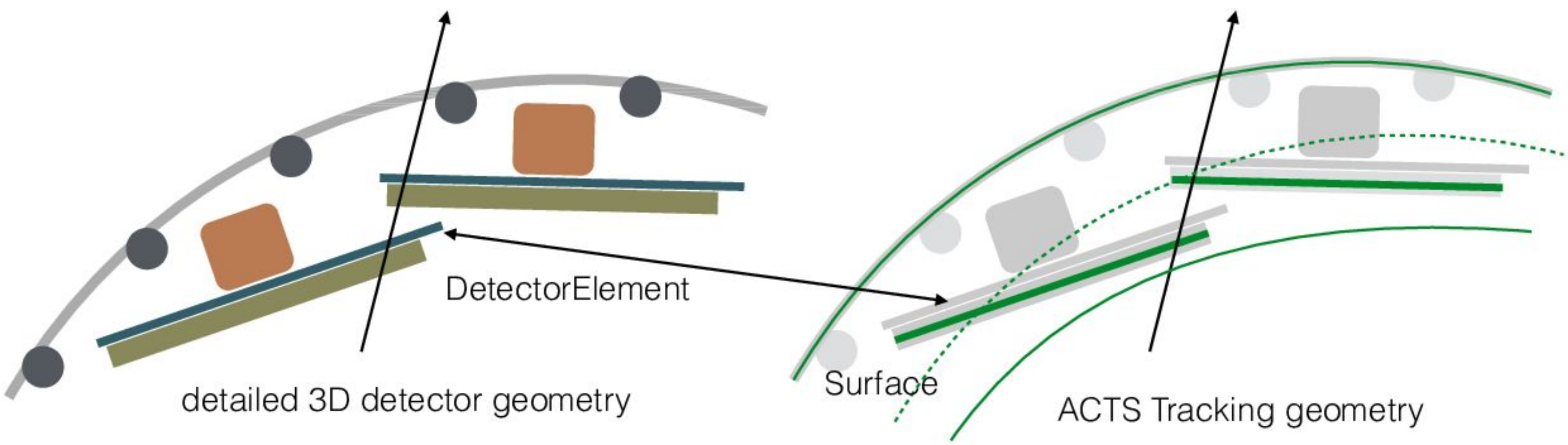
- Track reconstruction consists of 2 different parts:
  - Collecting measurements that belong to a track (fast)
  - Fitting the track (precise, full geometry)





Detector description in FastSim

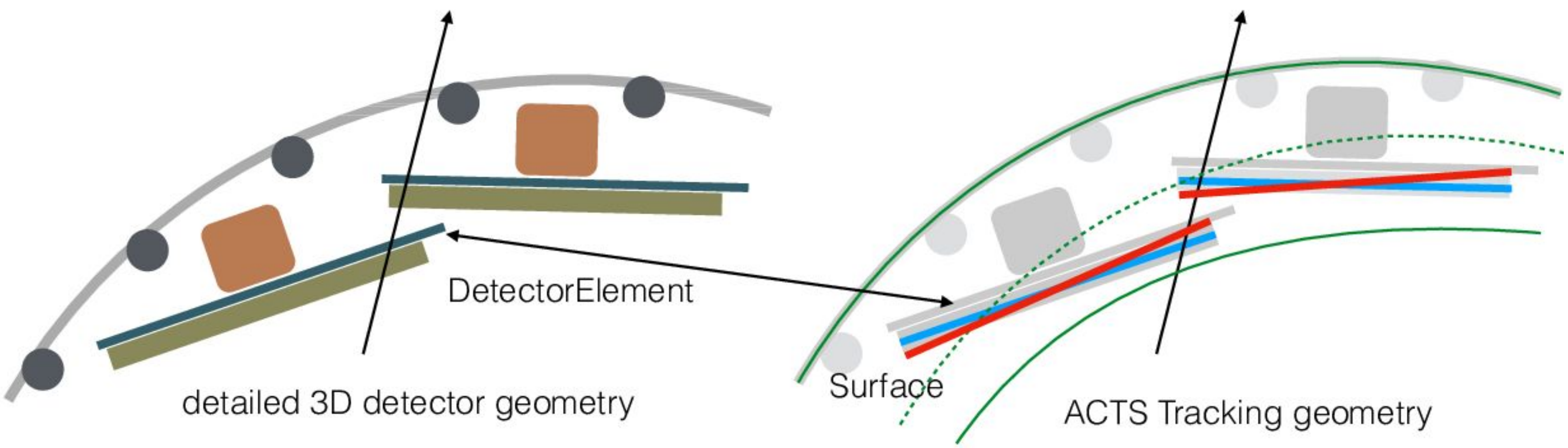
- In Simulation the detector is represented by simple surfaces
- Each geometric element is assumed to be a homogenic body
- Interactions of the traversing particles only happens at the surfaces



- Drawback of the simplified representation:
  - Particles could be propagated beyond the a surface and miss the interaction
  - Actual particle's path length through the detector element is totally neglected
  - Surface alignment can be varied and tuned

Detector description in FastSim

- In Simulation the detector is represented by simple surfaces
- Each geometric element is assumed to be a homogenic body
- Interactions of the traversing particles only happens at the surfaces



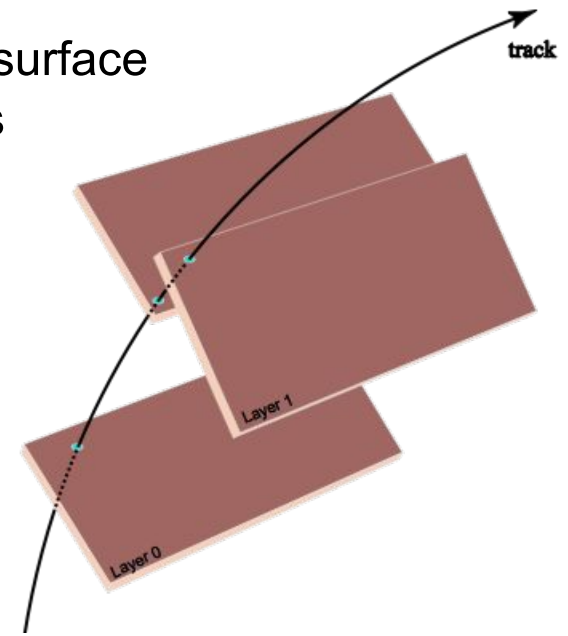
- Drawback of the simplified representation:
  - Particles could be propagated beyond the a surface and miss the interaction
  - Actual particle's path length through the detector element is totally neglected
  - Surface alignment can be varied and tuned

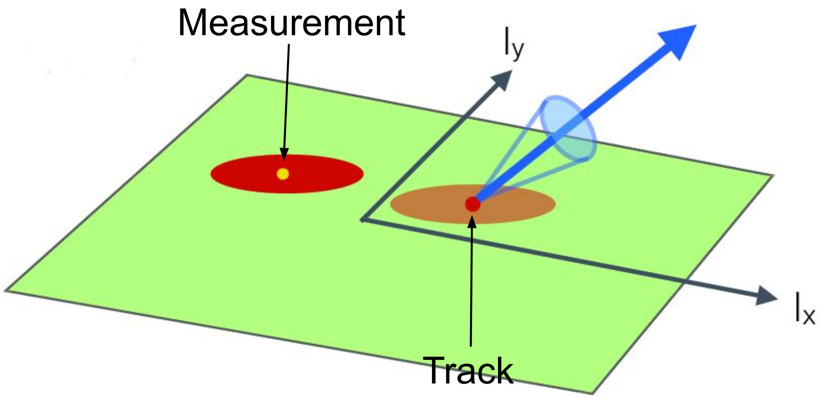
Event representation and dense environments

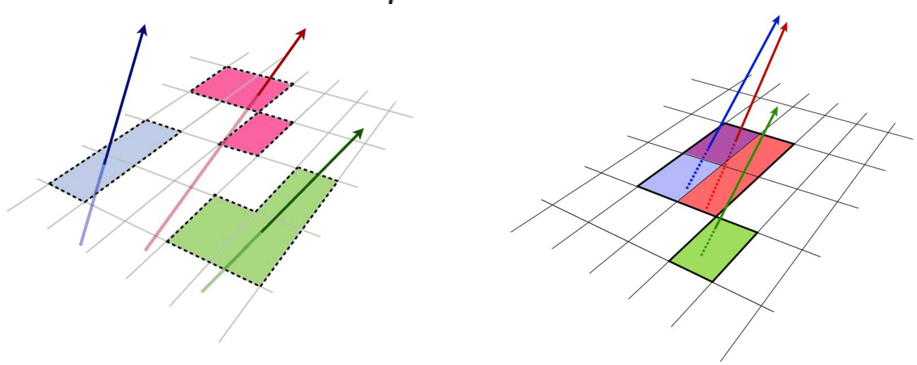
- Event = many measurements
- A measurements is expressed in local coordinates of the surface
- Track reconstrunction requires coordinate transformations

  →Lot’s of linear algebra
- Additional level of complexity in dense environments:
- Cluster formation
  - Multiple tracks per cluster

    → Ambiguity solving
  - Complexity grows with the number of tracks







(a) Single-particle pixel clusters

(b) Merged pixel cluster