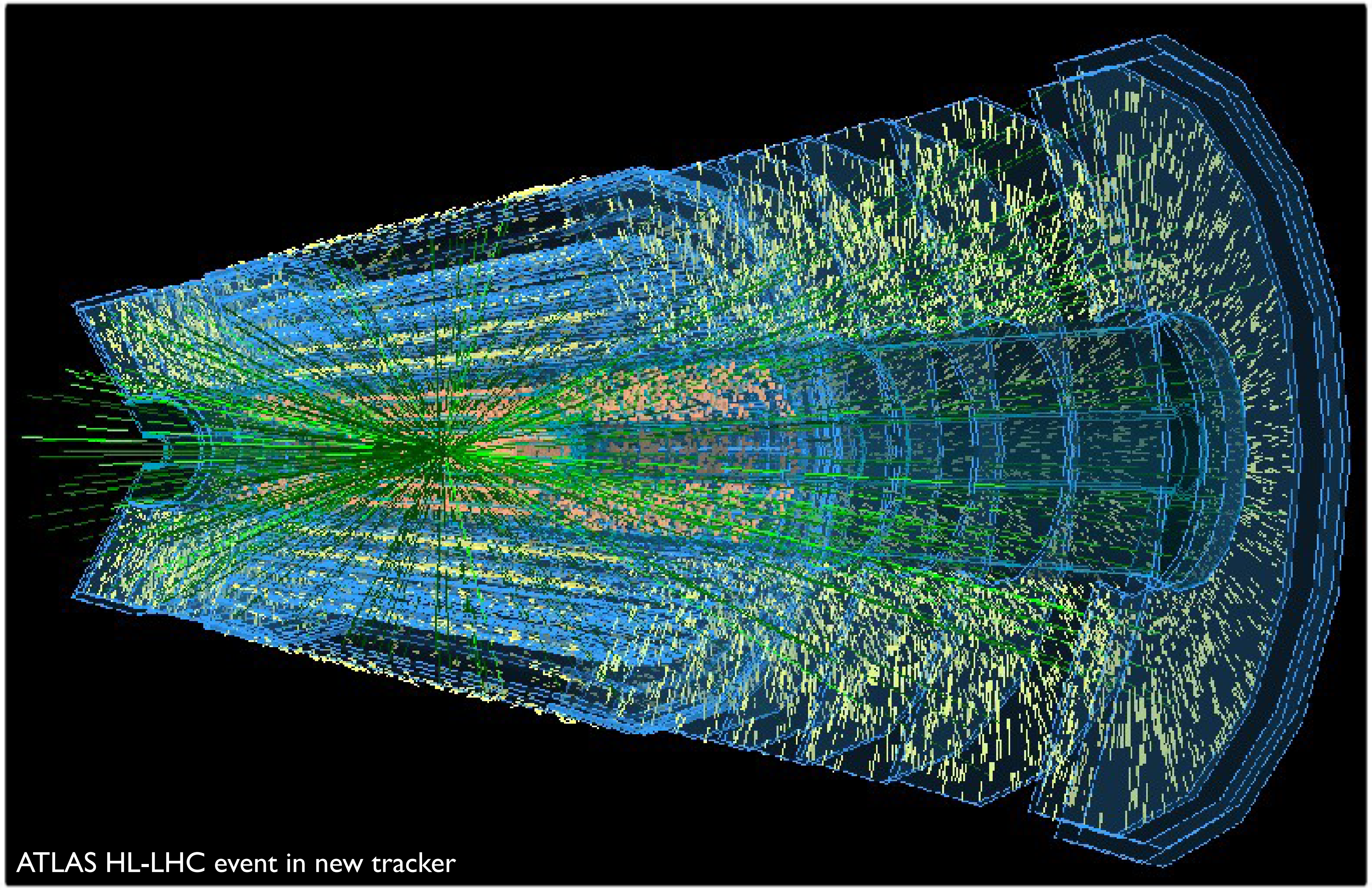
Reconstructing Events, from Electronic Signals to Tracks

CERN Openlab Summer Student Lecture

Markus Elsing, 28 July 2015



ATLAS HL-LHC event in new tracker

About this Lecture

- this lecture was originally written for physics students
  - ➡ but it is not required to be a physicist to follow this lecture (I think)
  - ➡ I will speak more about concepts and techniques, so don't get lost in details which I will flag as such
  - ➡ some (basic) knowledge on statistics helps for the mathematical details

- don't be afraid to stop me and ask
  - ➡ it is probably me not explaining things well enough
    - I may take too many things for granted or may use slang
  - ➡ we want to make this as useful as possible for YOU

➡ further reading: http://elsing.web.cern.ch/elsing/teaching.html

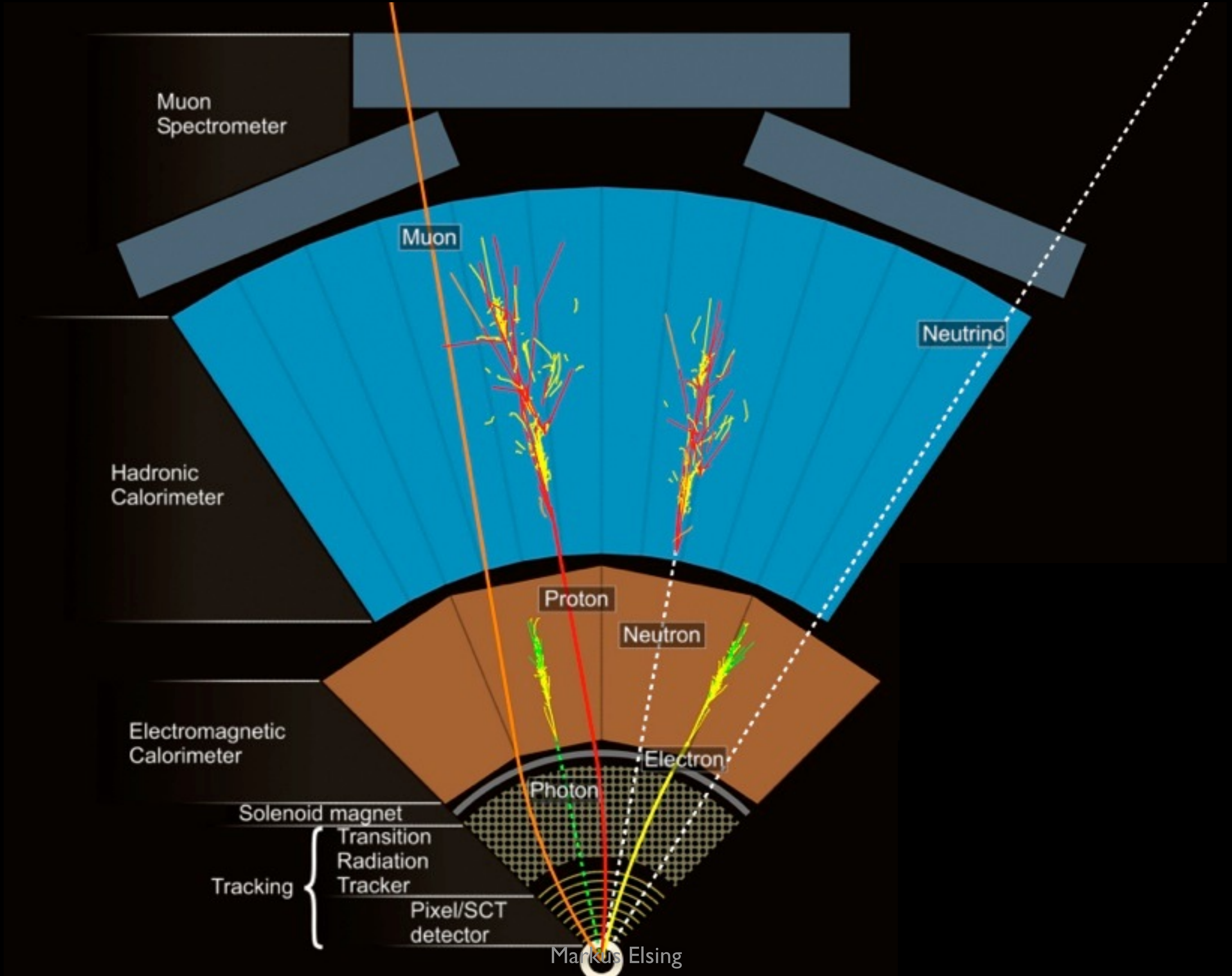
Event Reconstruction

- ➡ LHC experiments are giant "cameras" to take "pictures" of p-p collisions
  - taking a picture every 25 nsec (40 MHz) with 100 million channels
- ➡ task of the reconstruction is the interpretation of the picture !
  - answer the question: which particles were produced ?

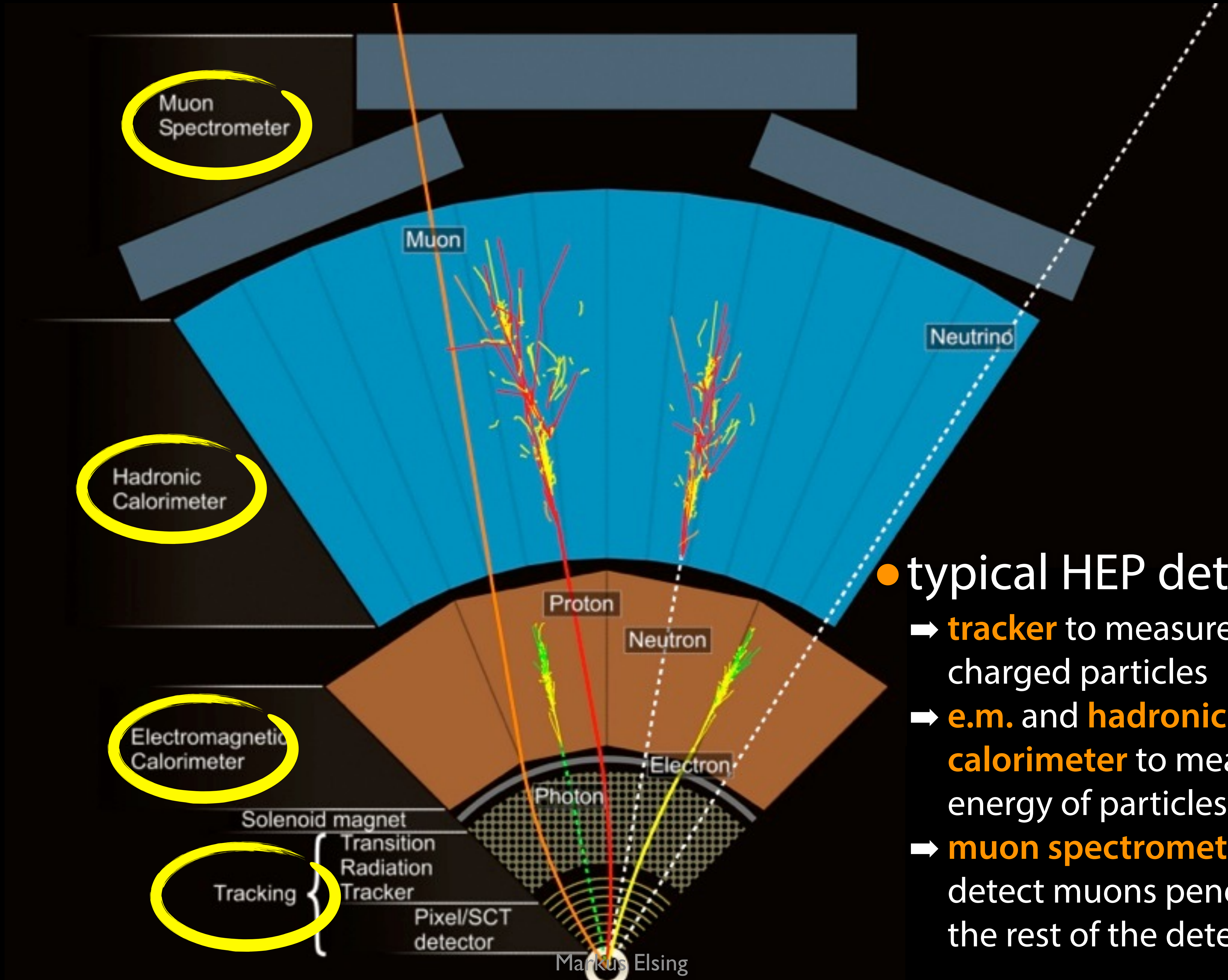
Event Reconstruction

- ➡ LHC experiments are giant "cameras" to take "pictures" of p-p collisions
  - taking a picture every 25 nsec (40 MHz) with 100 million channels
- ➡ task of the reconstruction is the interpretation of the picture !
  - answer the question: which particles were produced ?

Event Reconstruction "in a Nutshell"

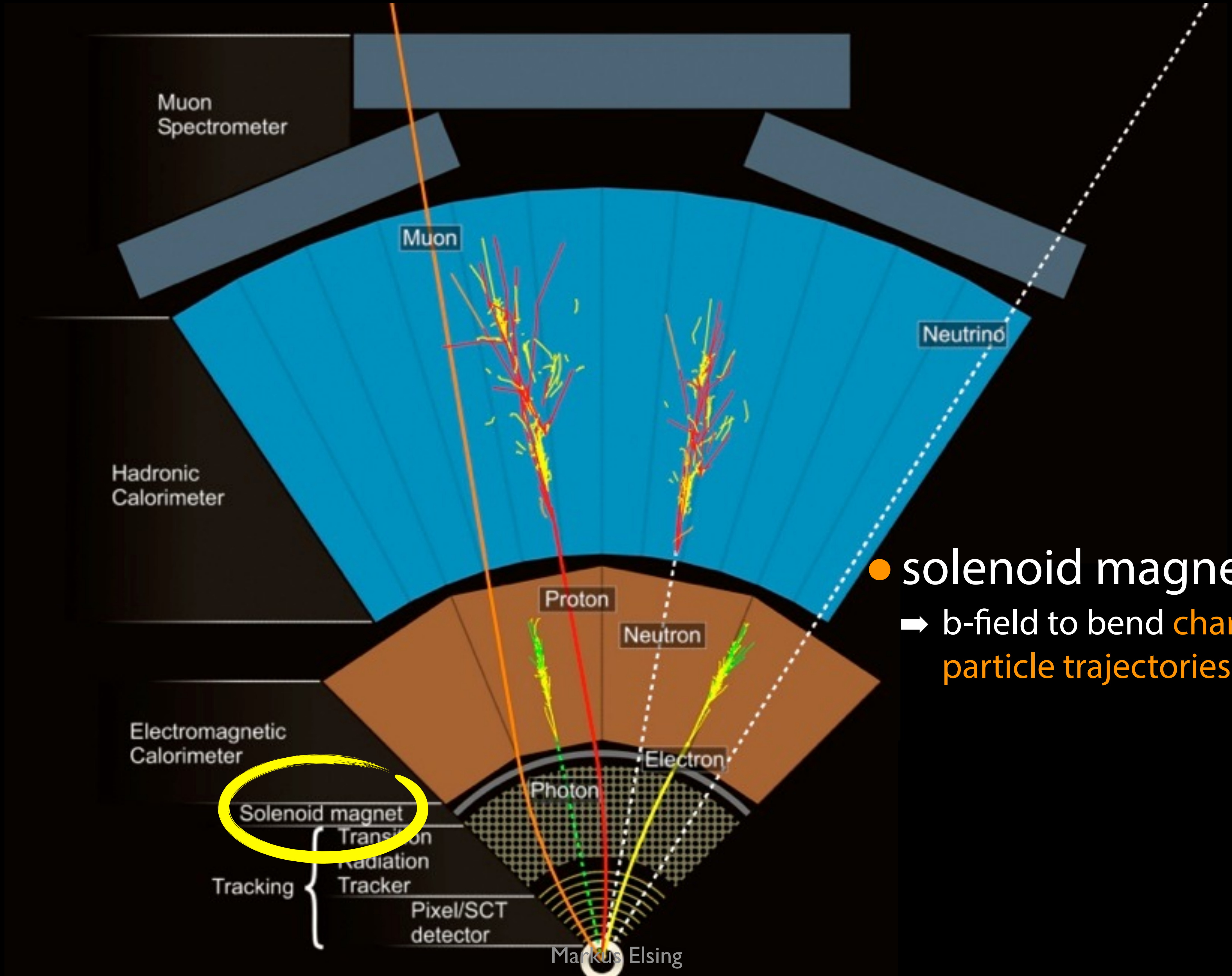


Event Reconstruction "in a Nutshell"



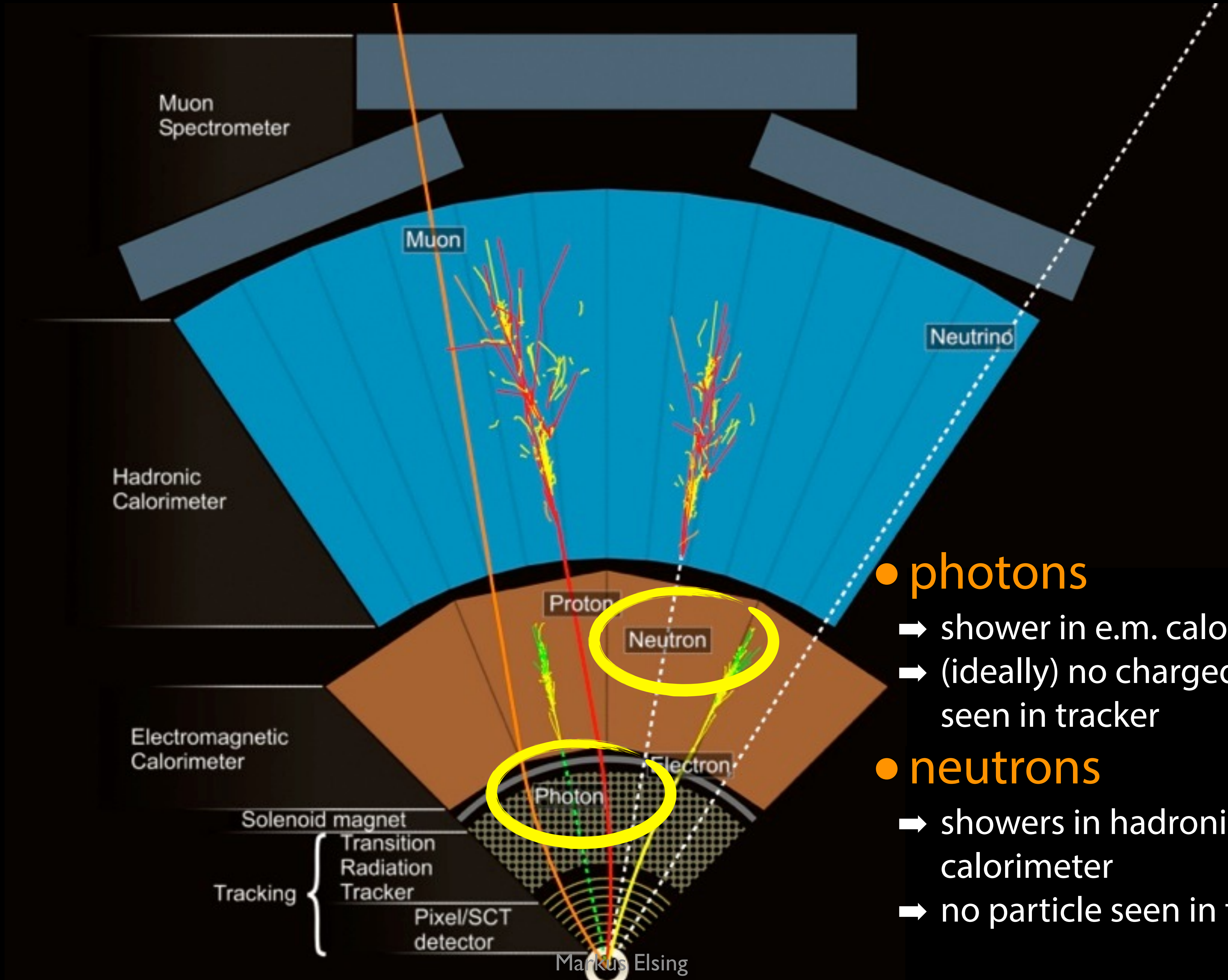
- typical HEP detector
  - ➡ **tracker** to measure charged particles
  - ➡ **e.m.** and **hadronic calorimeter** to measure energy of particles (jets)
  - ➡ **muon spectrometer** to detect muons penetrating the rest of the detector

Event Reconstruction “in a Nutshell”



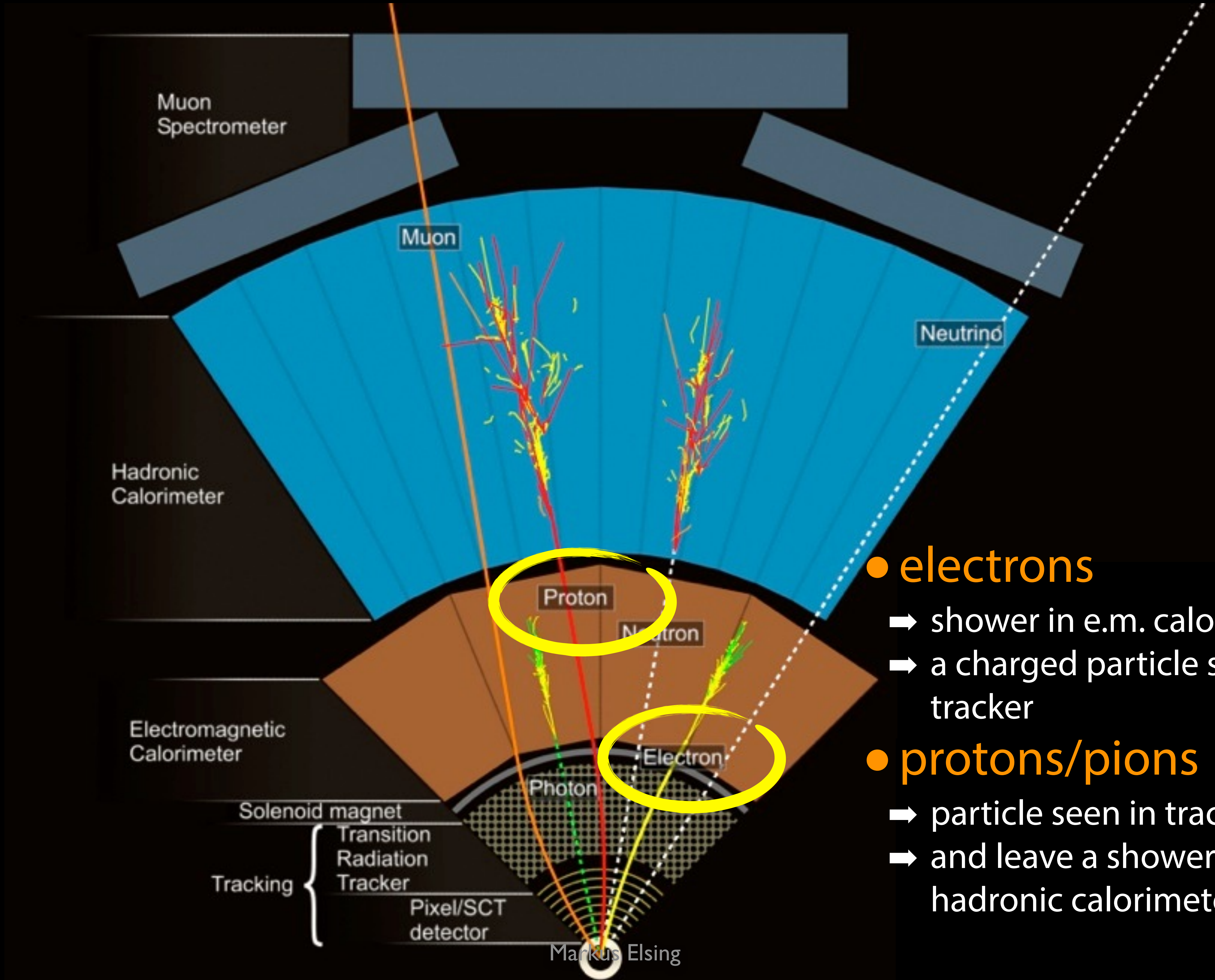
- solenoid magnet
  - ➡ b-field to bend charge particle trajectories

Event Reconstruction "in a Nutshell"



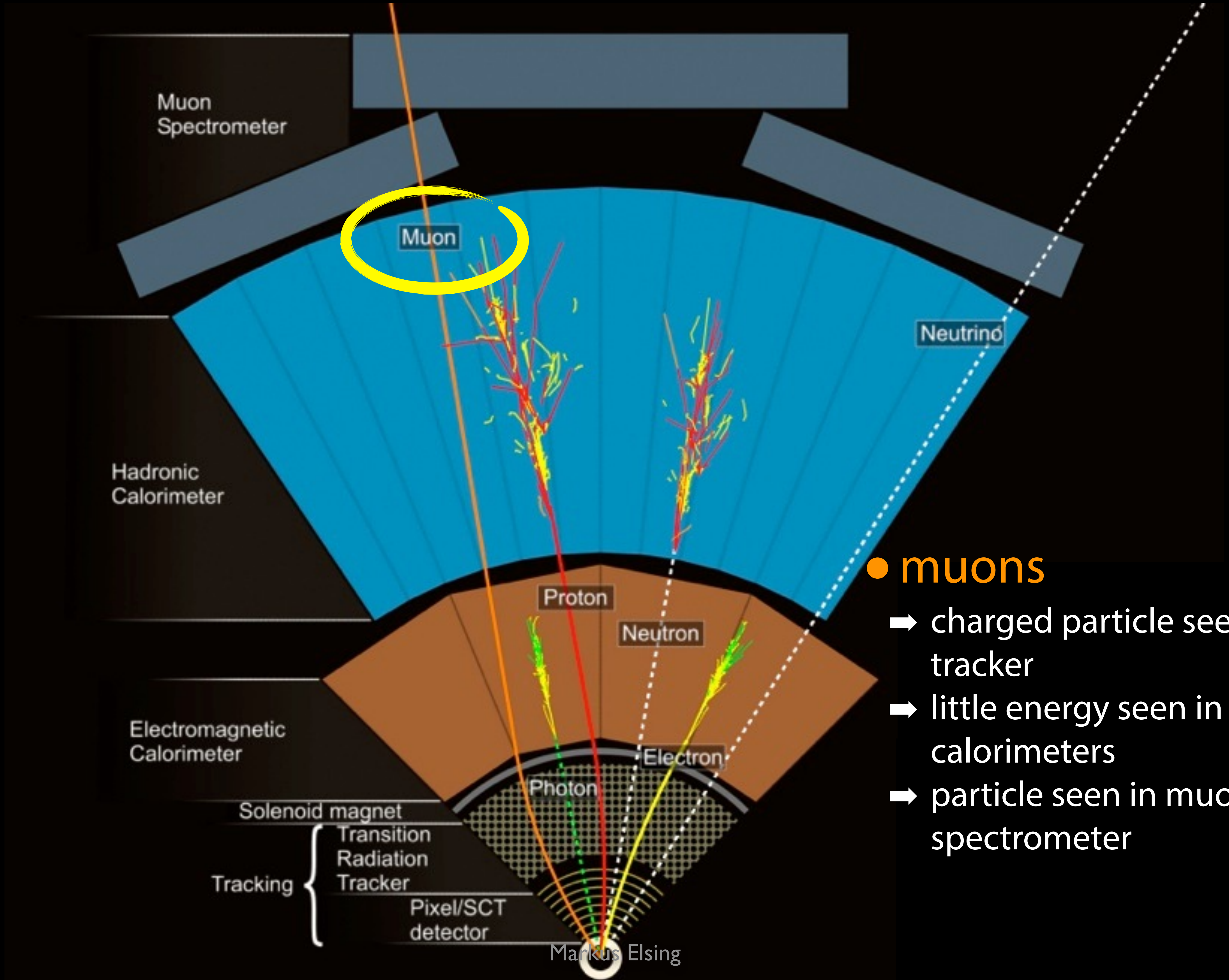
- photons
  - ➡ shower in e.m. calorimeter
  - ➡ (ideally) no charged particle seen in tracker
- neutrons
  - ➡ showers in hadronic calorimeter
  - ➡ no particle seen in tracker

Event Reconstruction "in a Nutshell"



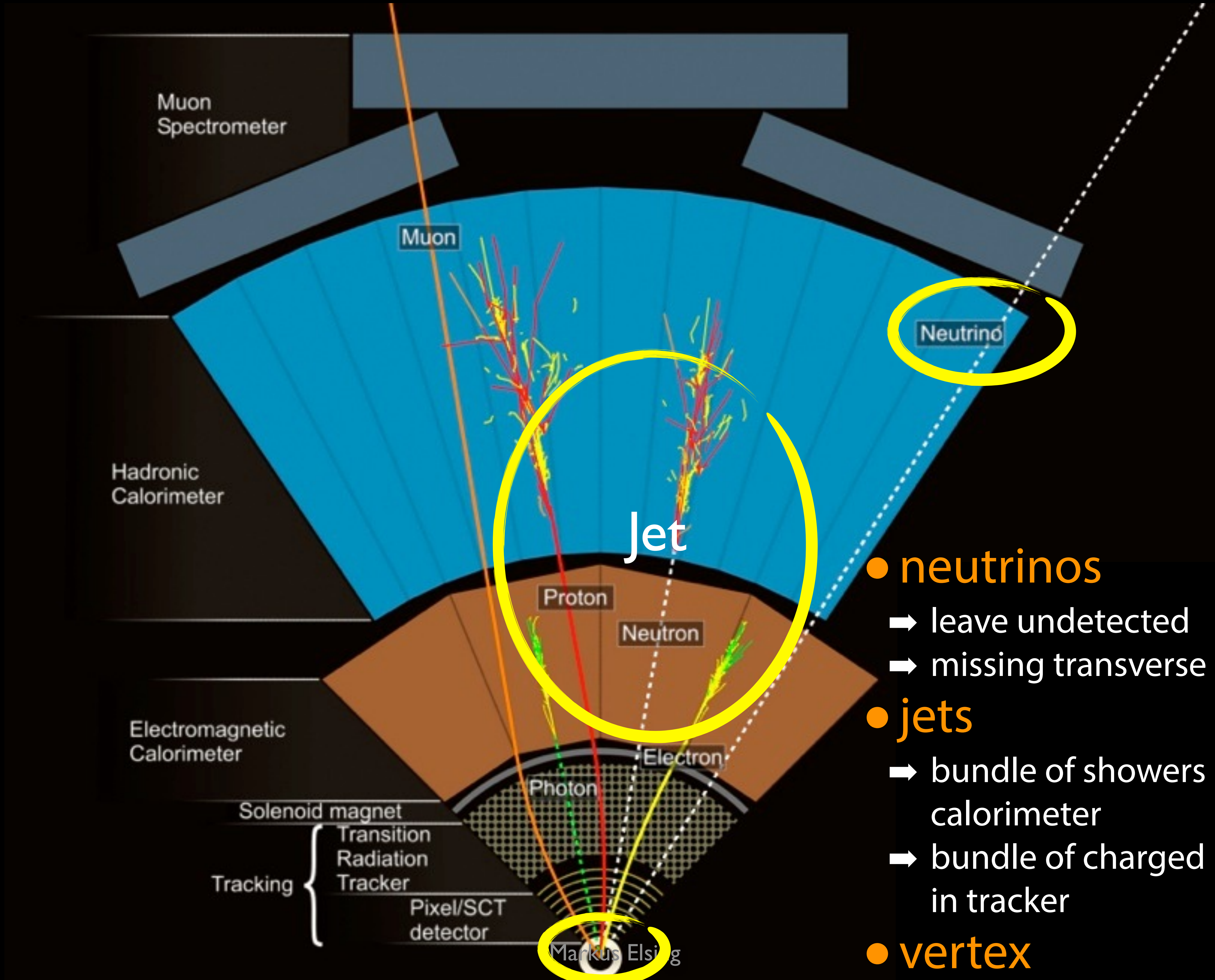
- electrons
  - ➡ shower in e.m. calorimeter
  - ➡ a charged particle seen in tracker
- protons/pions
  - ➡ particle seen in tracker
  - ➡ and leave a showers in hadronic calorimeter

Event Reconstruction "in a Nutshell"



- muons
  - ➡ charged particle seen in tracker
  - ➡ little energy seen in calorimeters
  - ➡ particle seen in muon spectrometer

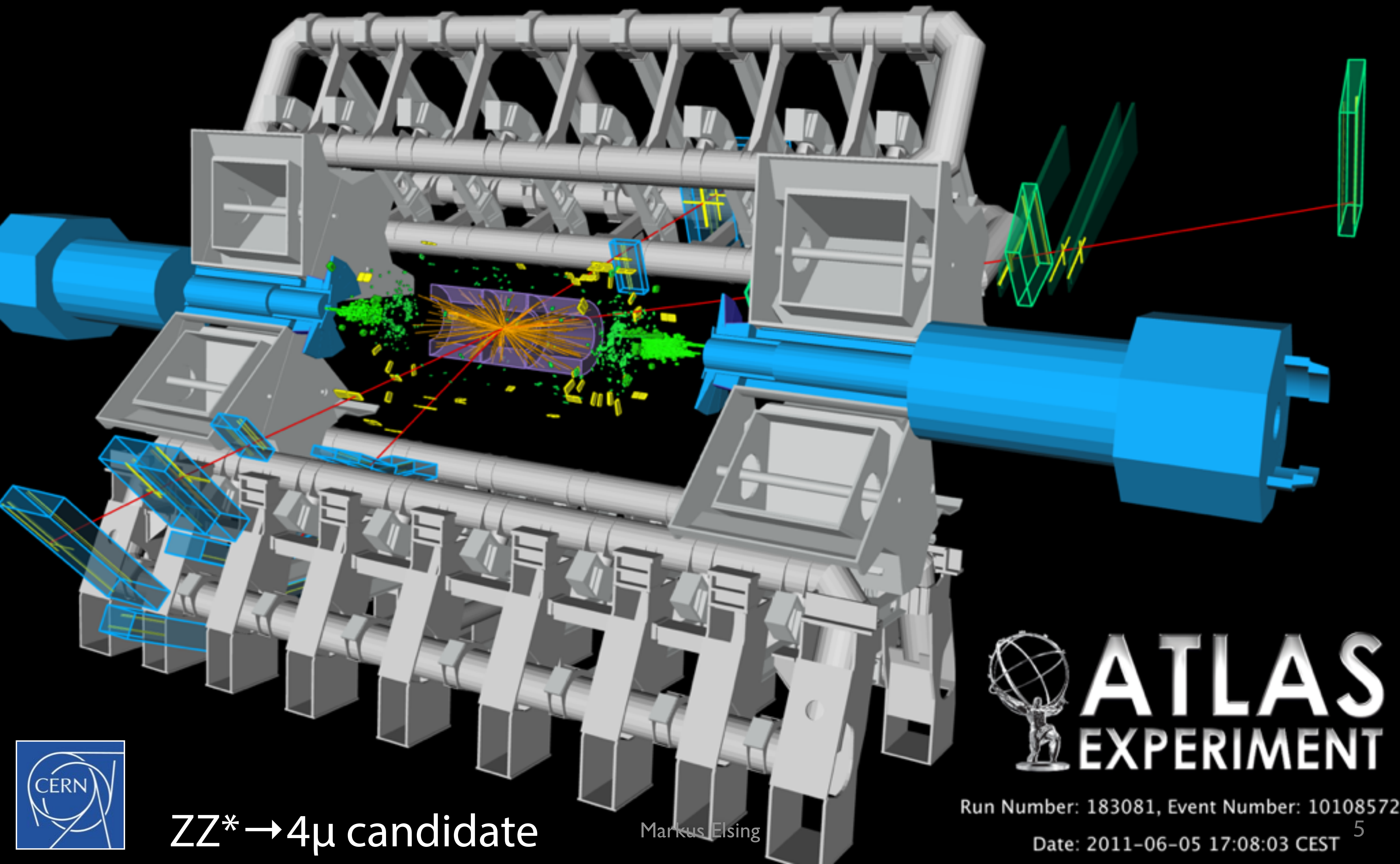
Event Reconstruction "in a Nutshell"



- neutrinos
  - ➡ leave undetected
  - ➡ missing transverse energy
- jets
  - ➡ bundle of showers in calorimeter
  - ➡ bundle of charged particles in tracker
- vertex

In Reality ?

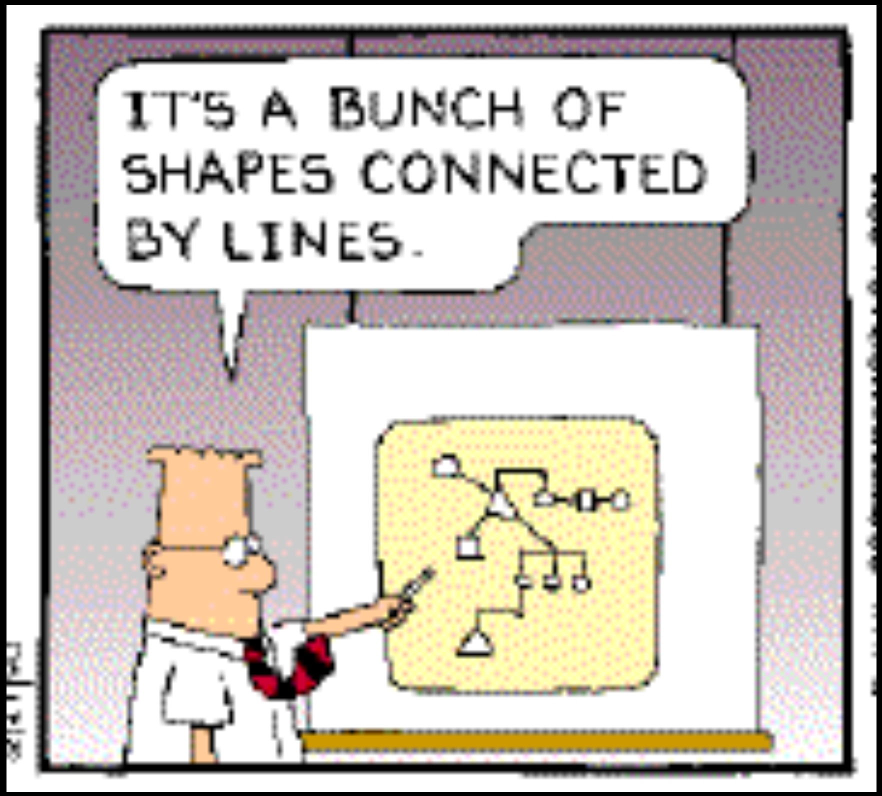
... a bit more complicated



ZZ*→4μ candidate

Introduction

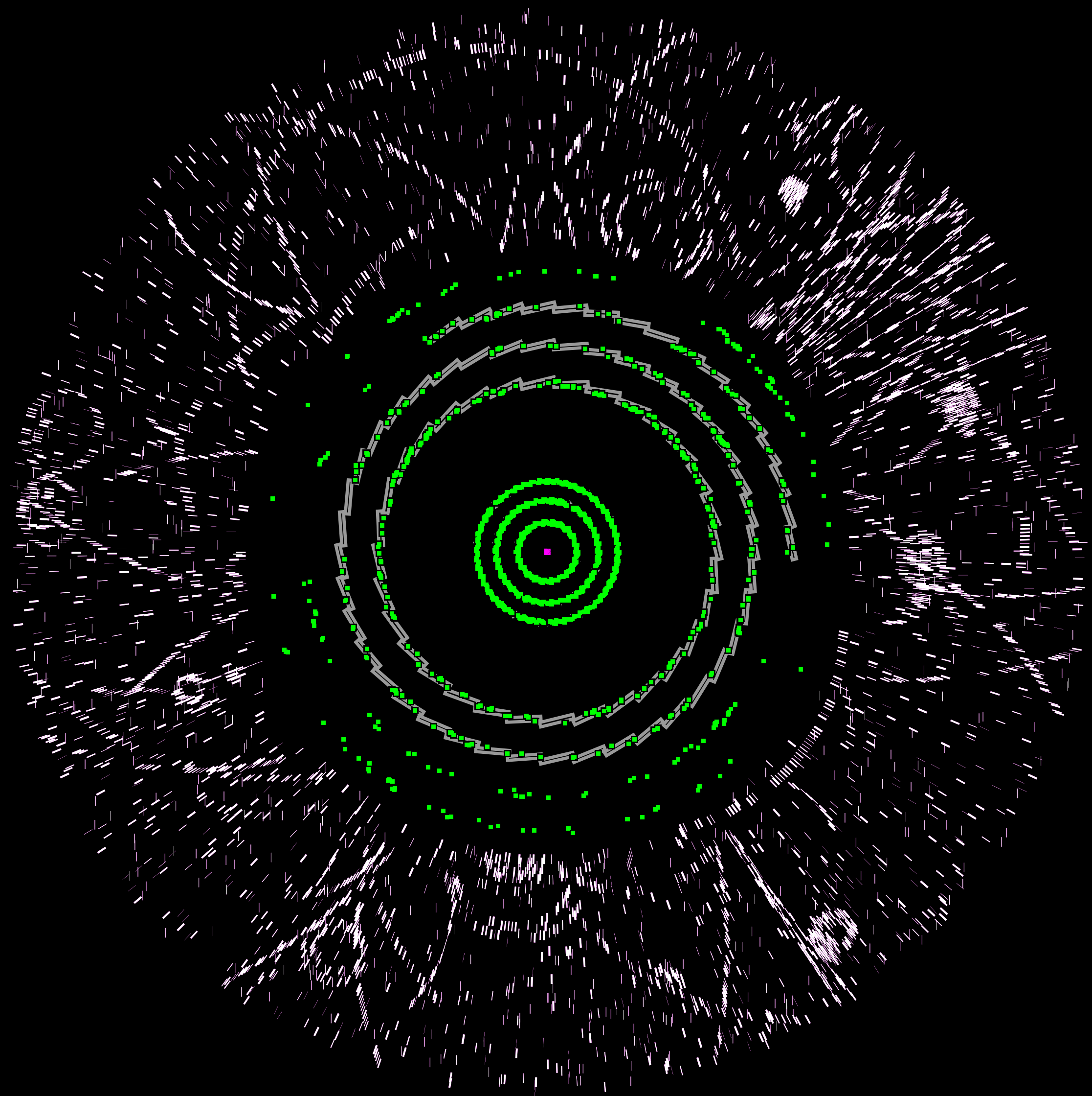
- in this lecture I will discuss the most complex and CPU consuming aspect of event reconstruction at the LHC
  - ➡ finding trajectories (tracks) of charged particles produced in p-p collisions
- will have to introduce various techniques for
  - ➡ pattern recognition, detector geometry, track fitting, extrapolation ...
  - ➡ including mathematical concepts and aspects of software design



... so why does it matter ?

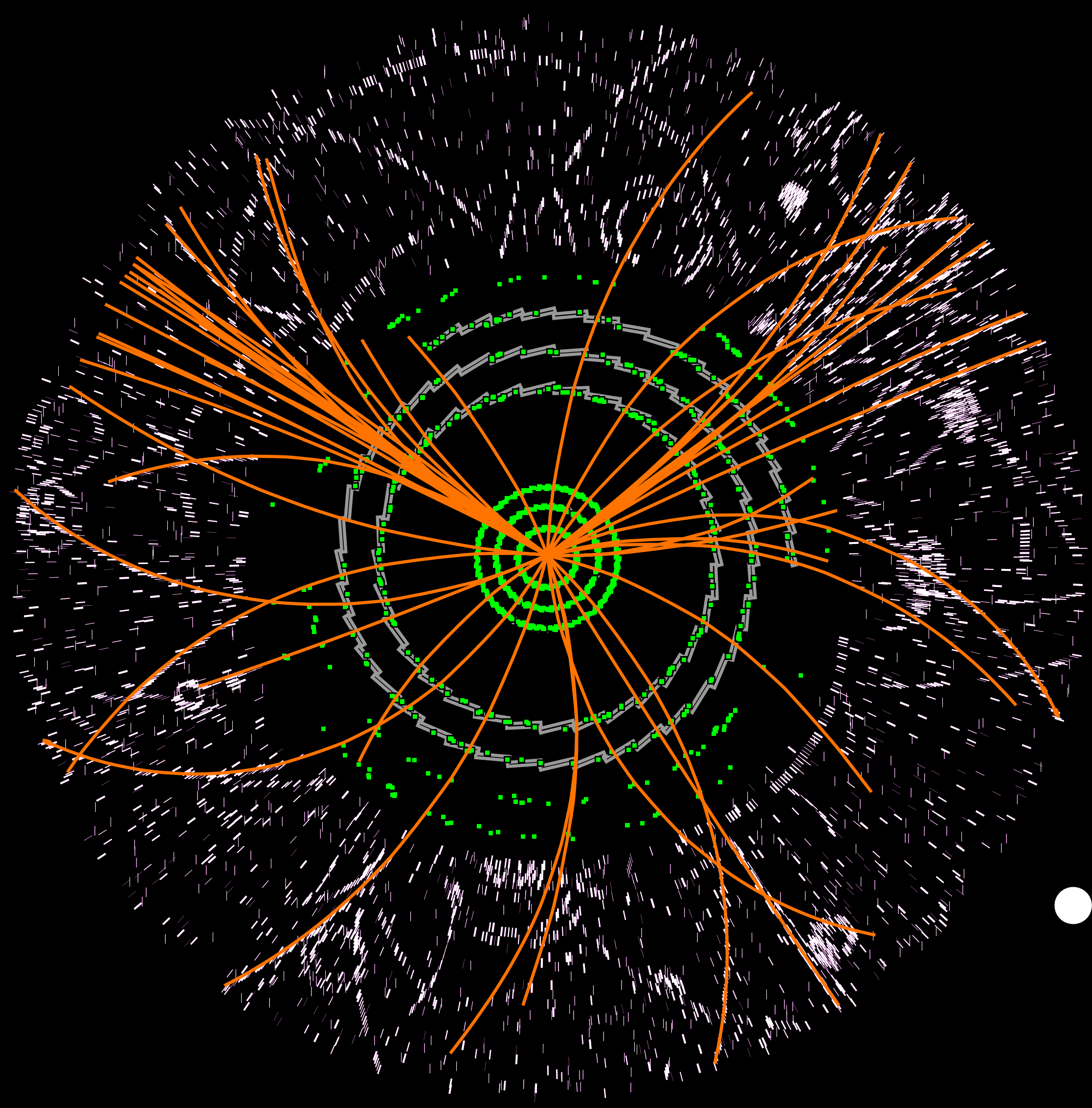
The Tracking Problem

- particles produce in a p-p interaction leave a cloud of hits in the detector



The Tracking Problem

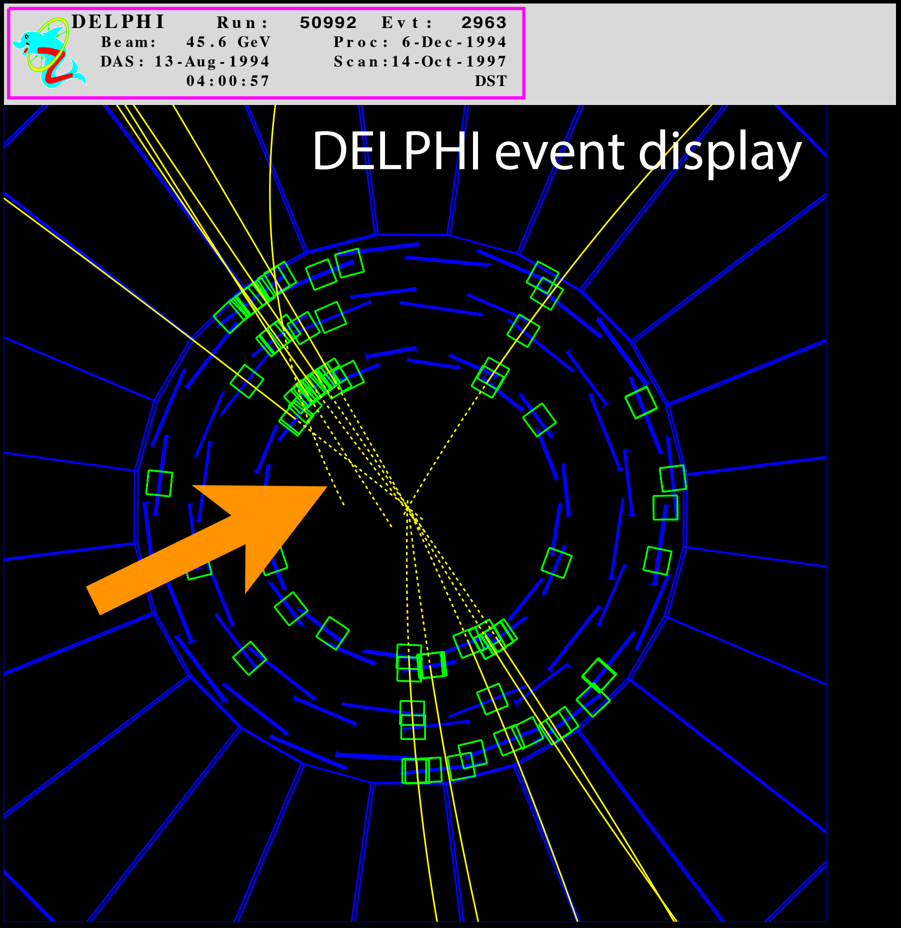
- particles produce in a p-p interaction leave a cloud of hits in the detector

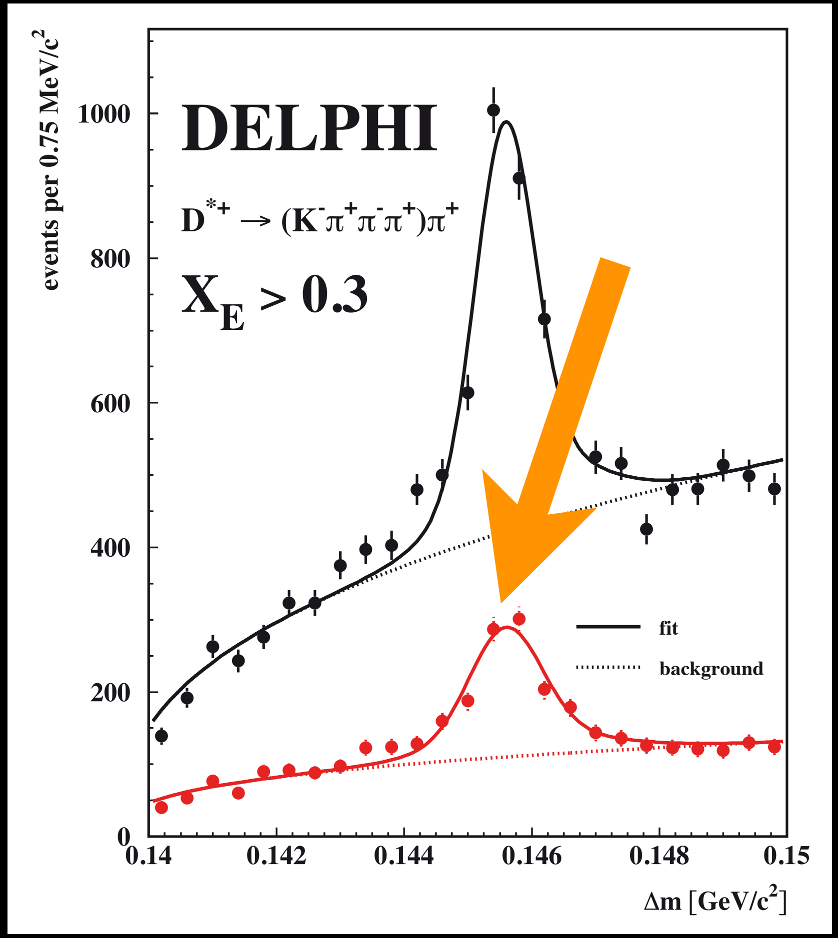


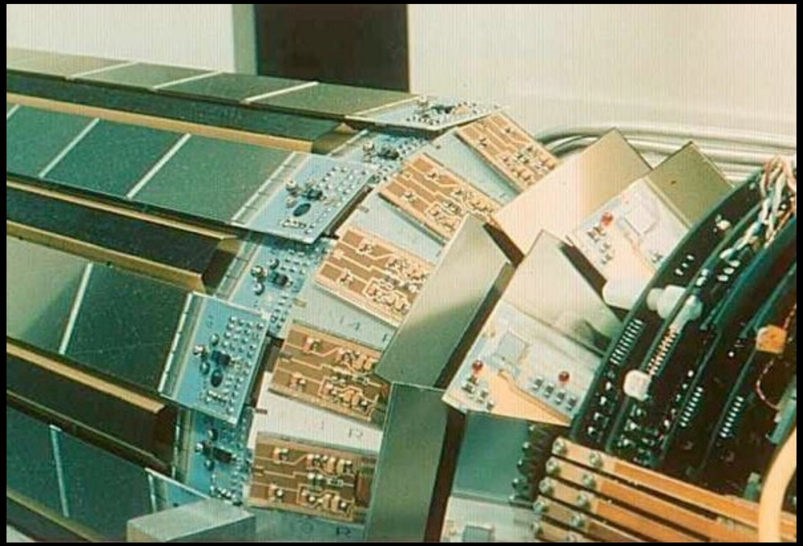
- tracking software is used to reconstruct their trajectories

Role of Tracking Software

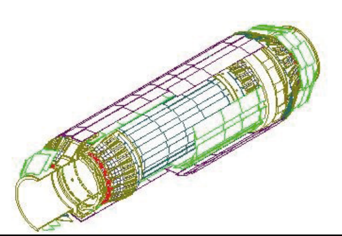
- **optimal** tracking software
  - ➡ required to fully explore performance of detector
- **example**: DELPHI Experiment at LEP
  - ➡ silicon vertex detector upgrade
    - initially not used in tracking to resolve dense jets
    - pattern mistakes in jet-chamber limit performance







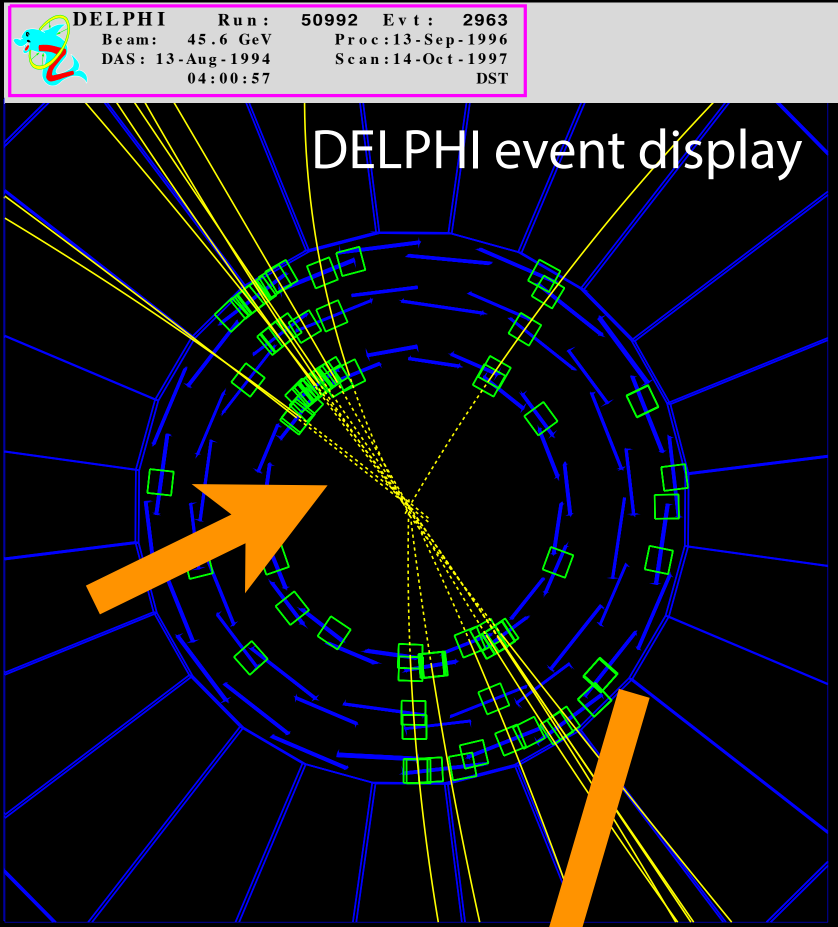
DELPHI vertex detector

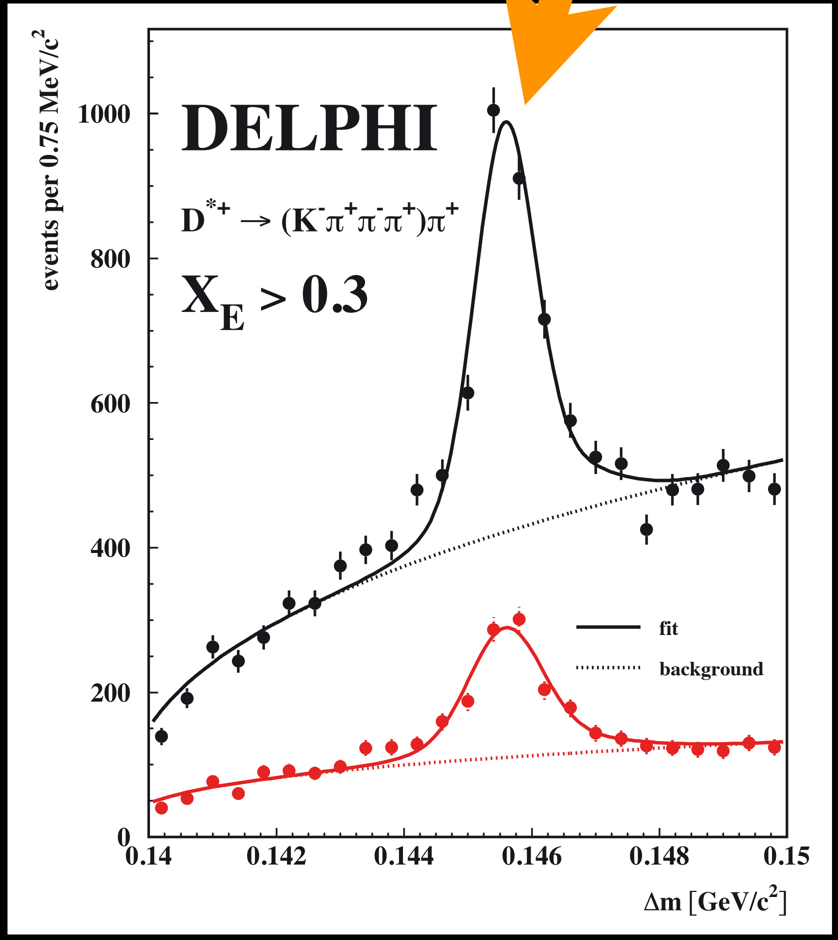


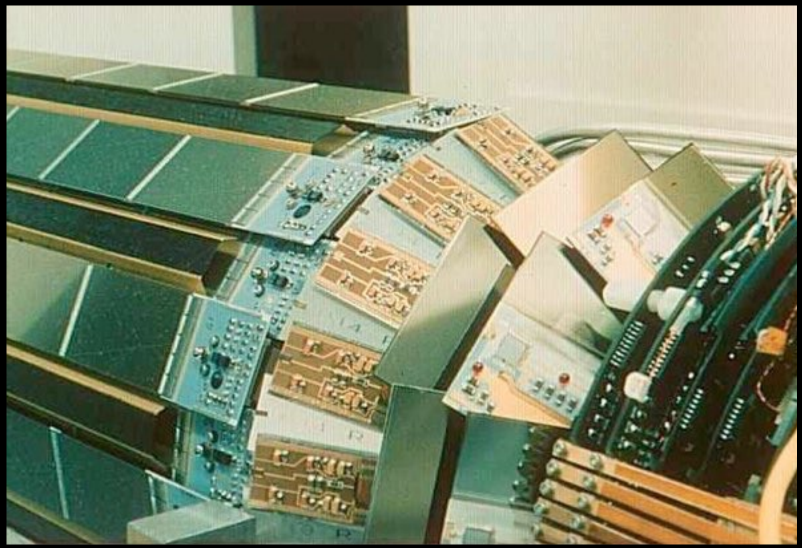
Role of Tracking Software

- **optimal** tracking software
  - ➡ required to fully explore performance of detector
- **example**: DELPHI Experiment at LEP
  - ➡ silicon vertex detector upgrade
    - initially not used in tracking to resolve dense jets
    - pattern mistakes in jet-chamber limit performance
  - ➡ 1994: redesign of tracking software
    - start track finding in vertex detector
  - ➡ **factor ~ 2.5 more D* signal** after reprocessing

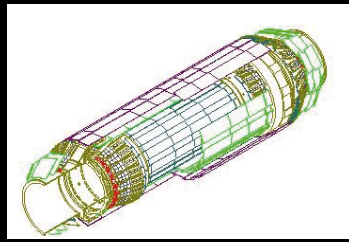
(M.Feindt, M.E. et al)







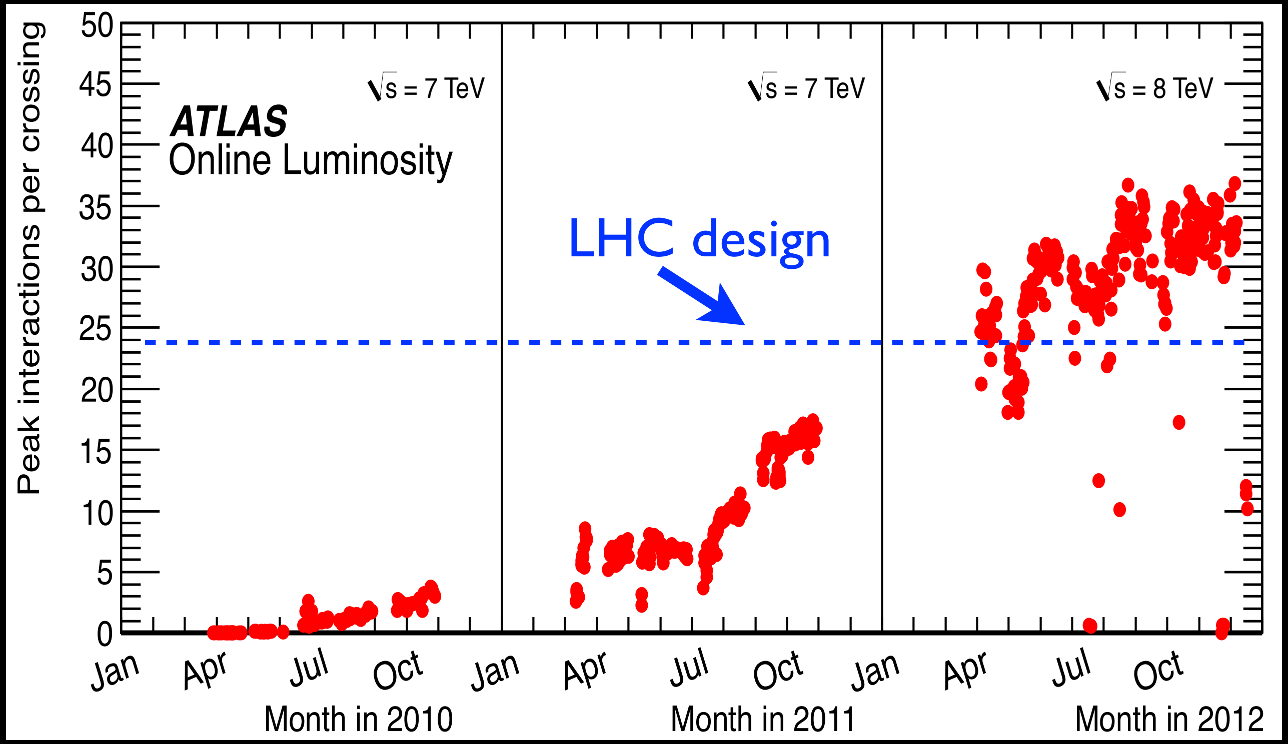
DELPHI vertex detector



Tracking at the LHC ?

- reminder: (first lecture by Helge Meinhard)
  - ➡ LHC is a high luminosity machine
    - proton bunches collide every 25 (50) nsec in experiments
    - each time > 20 p-p interactions are observed ! (event pileup)
  - ➡ our detectors see hits from particles produced by all > 20 p-p interactions
    - ~100 particles per p-p interaction
    - each charged particle leaves ~50 hits

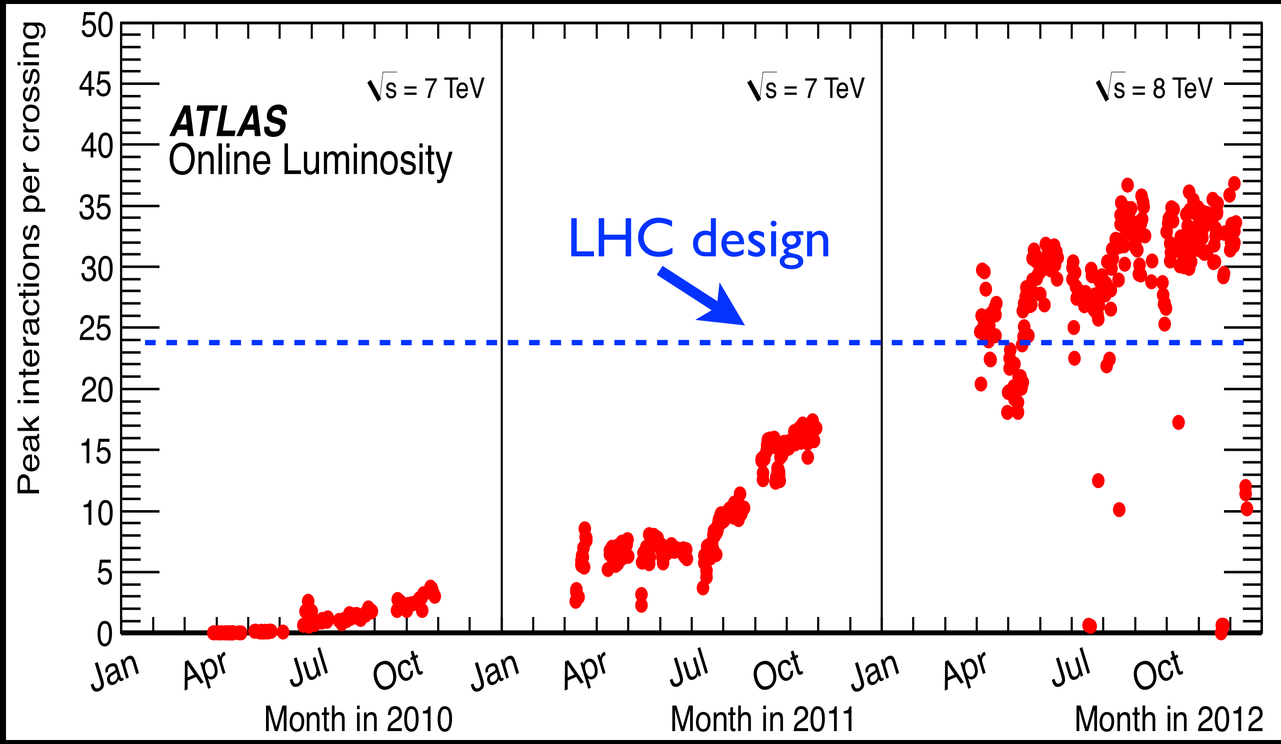
pileup display shown by Helge

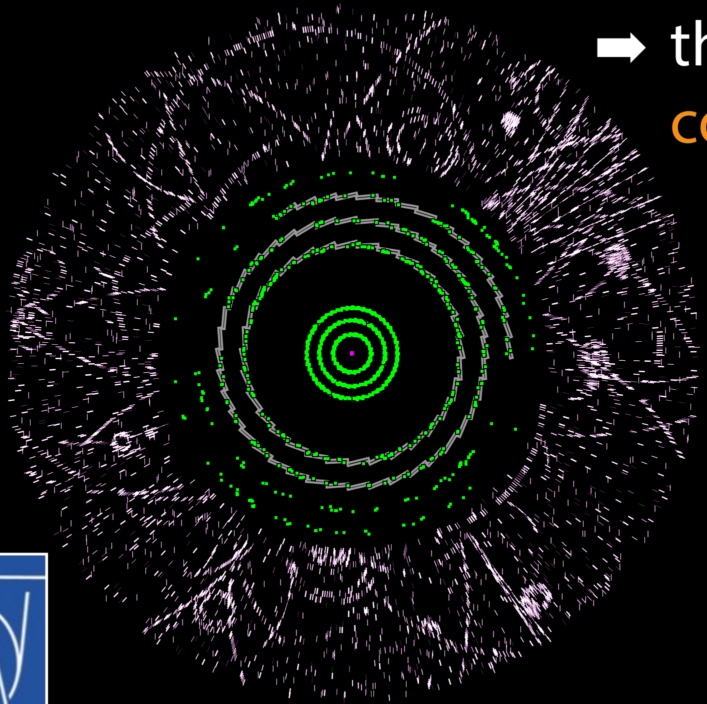


Tracking at the LHC ?

pileup display shown by Helge

- reminder: (first lecture by Helge Meinhard)
  - ➡ LHC is a high luminosity machine
    - proton bunches collide every 25 (50) nsec in experiments
    - each time > 20 p-p interactions are observed ! (event pileup)
  - ➡ our detectors see hits from particles produced by all > 20 p-p interactions
    - ~100 particles per p-p interaction
    - each charged particle leaves ~50 hits



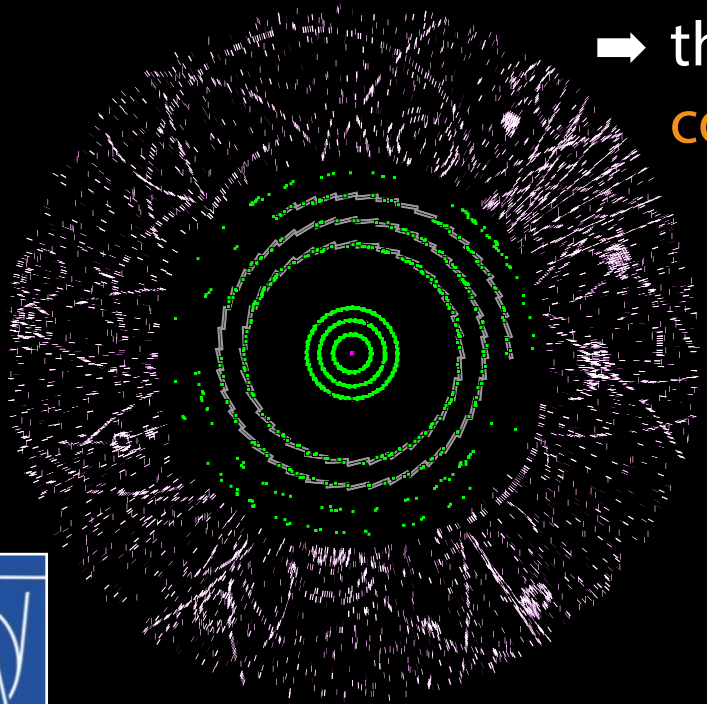


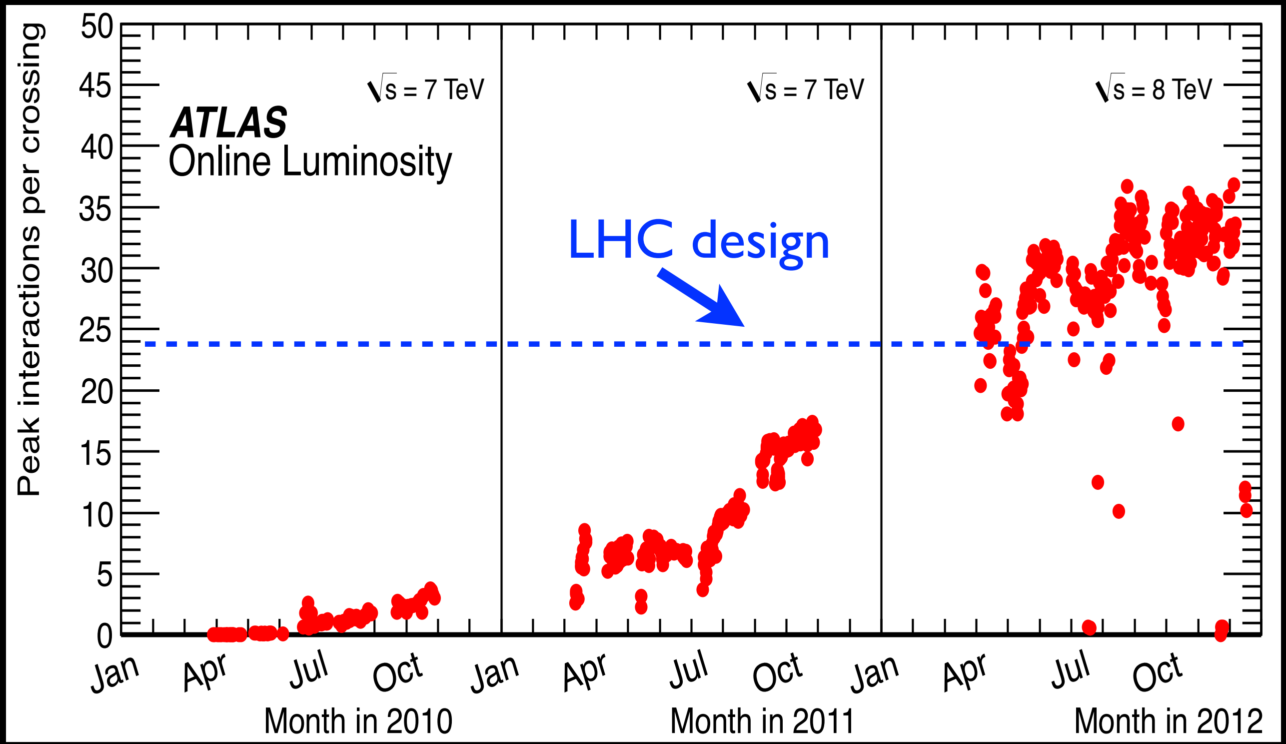
➡ this is how 1 pp collisions looks like

Tracking at the LHC ?

pileup display shown by Helge

- reminder: (first lecture by Helge Meinhard)
  - ➡ LHC is a high luminosity machine
    - proton bunches collide every 25 (50) nsec in experiments
    - each time > 20 p-p interactions are observed ! (event pileup)
  - ➡ our detectors see hits from particles produced by all > 20 p-p interactions
    - ~100 particles per p-p interaction
    - each charged particle leaves ~50 hits

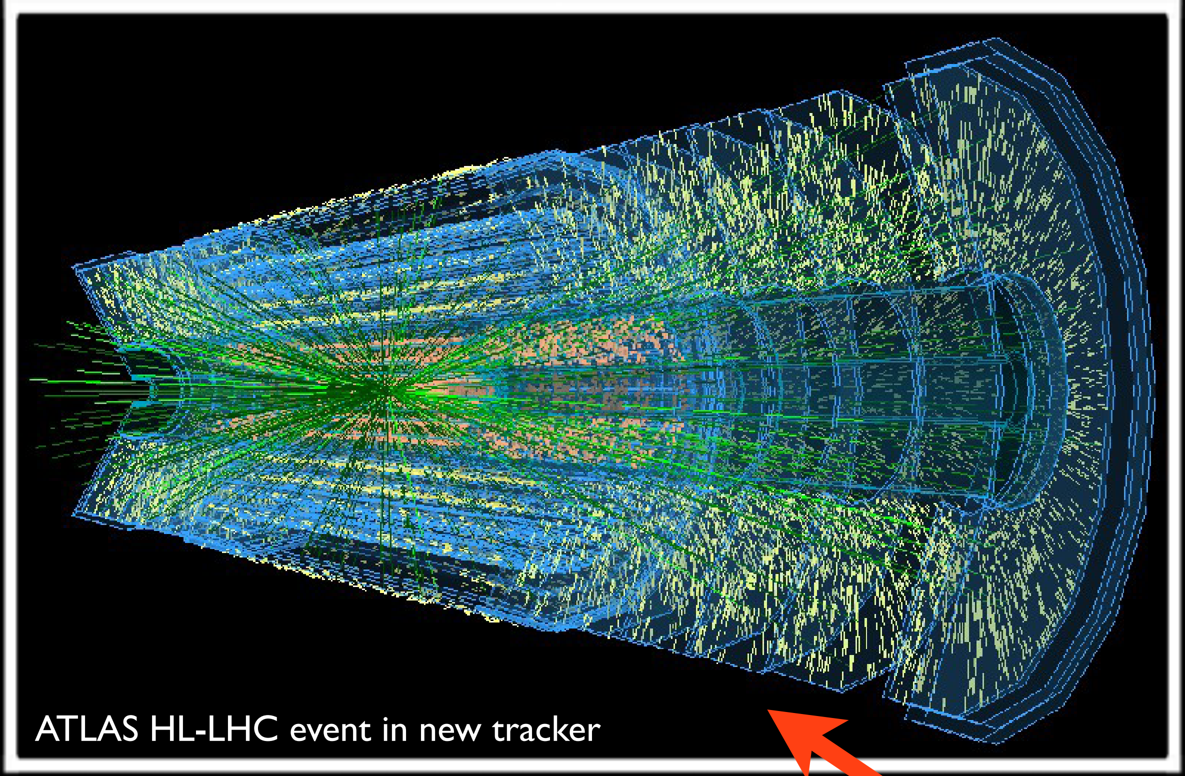




- ➡ this is how 1 pp collisions looks like
  - now imagine 30 of them overlapping
  - task of tracking software is to resolve the mess ...

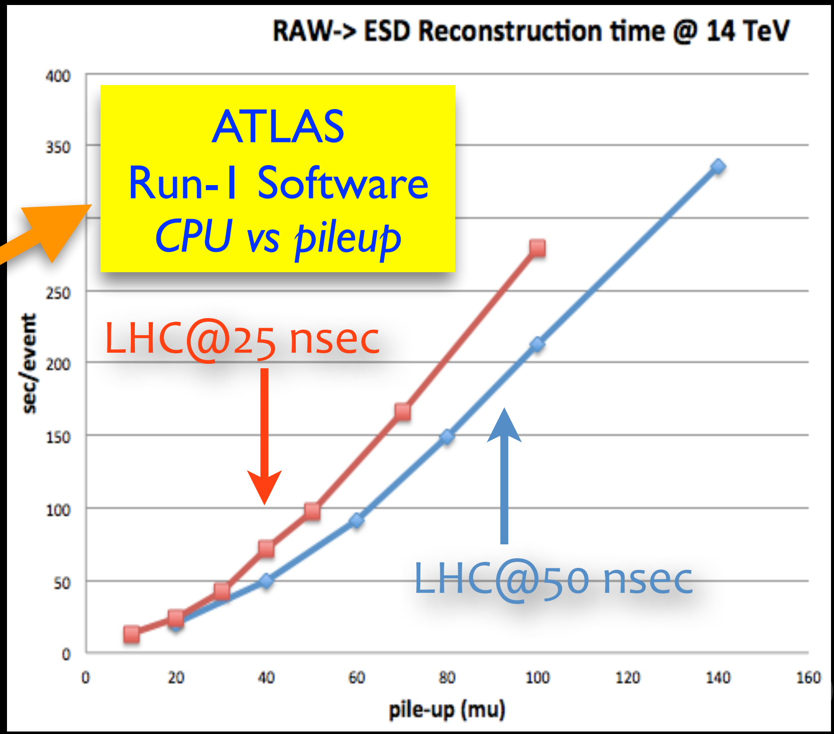
Tracking at the LHC ?

- track reconstruction
  - ➡ combinatorial problem grows with pileup
  - ➡ naturally resource driver (CPU/memory)
- the million dollar question:
  - ➡ how to reconstruct LH-LHC events within resources ? (pileup ~ 140-200)
- more than 10 years of R&D on LHC tracking software
  - ➡ we knew that tracking at the LHC is going to be challenging
    - building on techniques developed for previous experiments
  - ➡ processor technologies will change in the future
    - need to rethink some of the design decisions we did
    - adapt software to explore modern CPUs: threading, data locality...



event display from title page

...see bonus slides



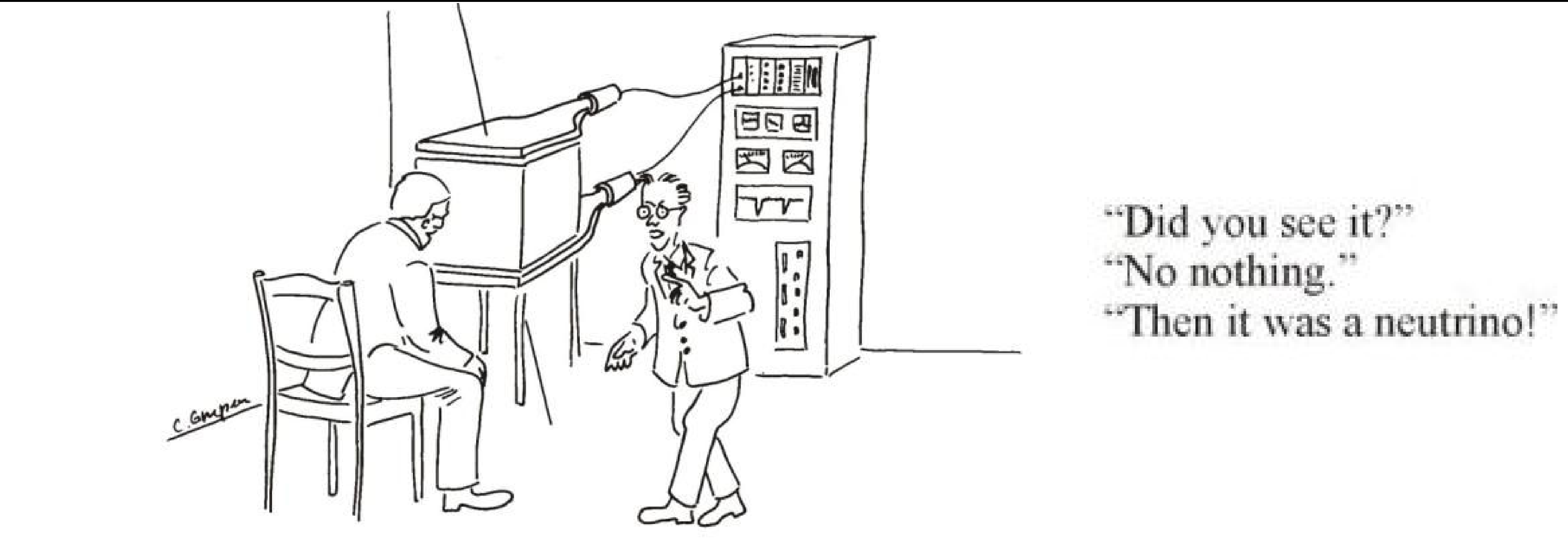
Outline of this Lecture

- Tracking Detectors
  - ➡ semiconductor tracker
  - ➡ drift tubes
- Charged Particle Trajectories and Extrapolation
  - ➡ trajectory representations and trajectory following in a realistic detector
  - ➡ detector description, navigation and simulation toolkits
- Track Fitting
  - ➡ classical least square track fit and a Kalman filter track fit
  - ➡ examples for advanced techniques
- Track Finding
  - ➡ search strategies, Hough transforms, progressive track finding, ambiguity solution
- ATLAS Track Reconstruction

Tracking Detectors

Passage of Particles through Matter

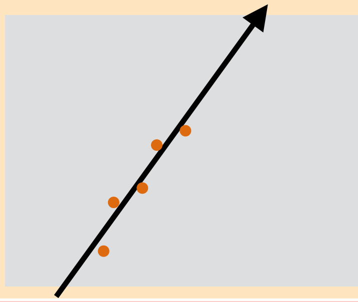
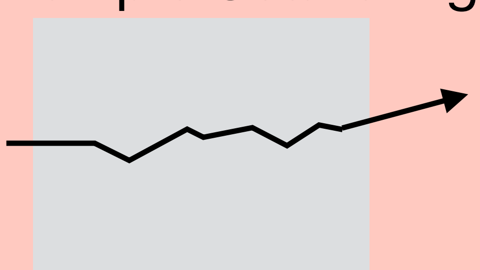
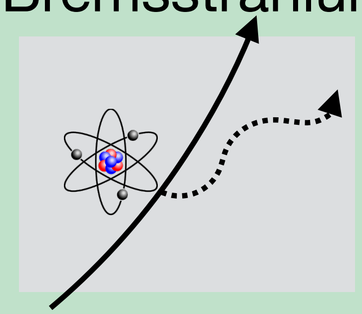
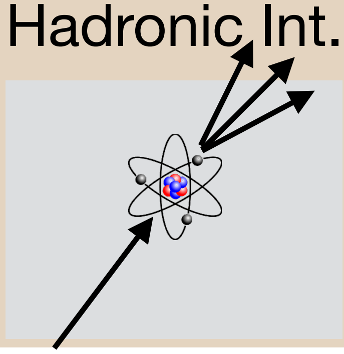
- any device that is to detect a particle must interact with it in some way
  - ➡ well, almost...
  - ➡ in many experiments neutrinos are measured by missing transverse momentum



Claus Grupen, Particle Detectors, Cambridge University Press, Cambridge 1996 (455 pp. ISBN 0-521-55216-8)

Interactions most relevant to Tracking

for completeness

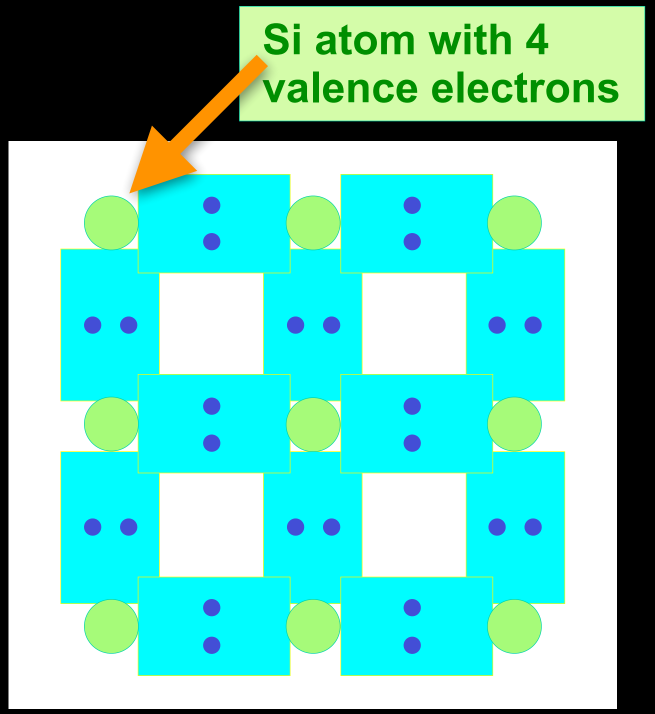
Type	particles	parameter	characteristics	effect
Ionisation loss 	all charged particle	effective density $A/Z^{*}\ \rho$	small effect in tracker, small dependence on p	increases momentum uncertainty
Multiple Scattering 	all charged particle	radiation length X_0	almost gaussian average effect 0, depends $\sim 1/p$	deflects particles, increases measurement uncertainty
Bremsstrahlung 	all charged particle, dominant for e	radiation length X_0	energy loss proportional $\sim E$, highly non-gaussian, depends $\sim 1/m^2$	introduces measurement bias and inefficiency
Hadronic Int. 	all hadronic particles	nuclear interaction length Λ_0	incoming particle lost, rather constant effect in p	main source of track reconstruction inefficiency

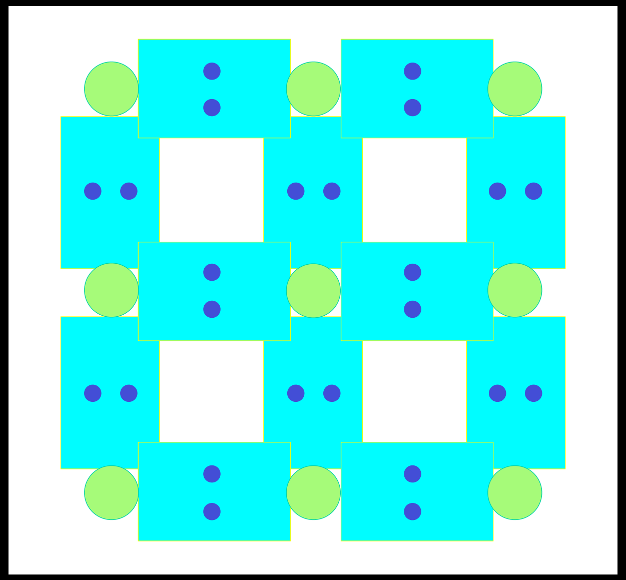
➡ tracking detectors explore effects like **ionisation** to measure charged particles
- let's discuss the basic principles of **semiconductor trackers** and **drift tubes**

Semiconductor Trackers

Semiconductors as Particle Detectors

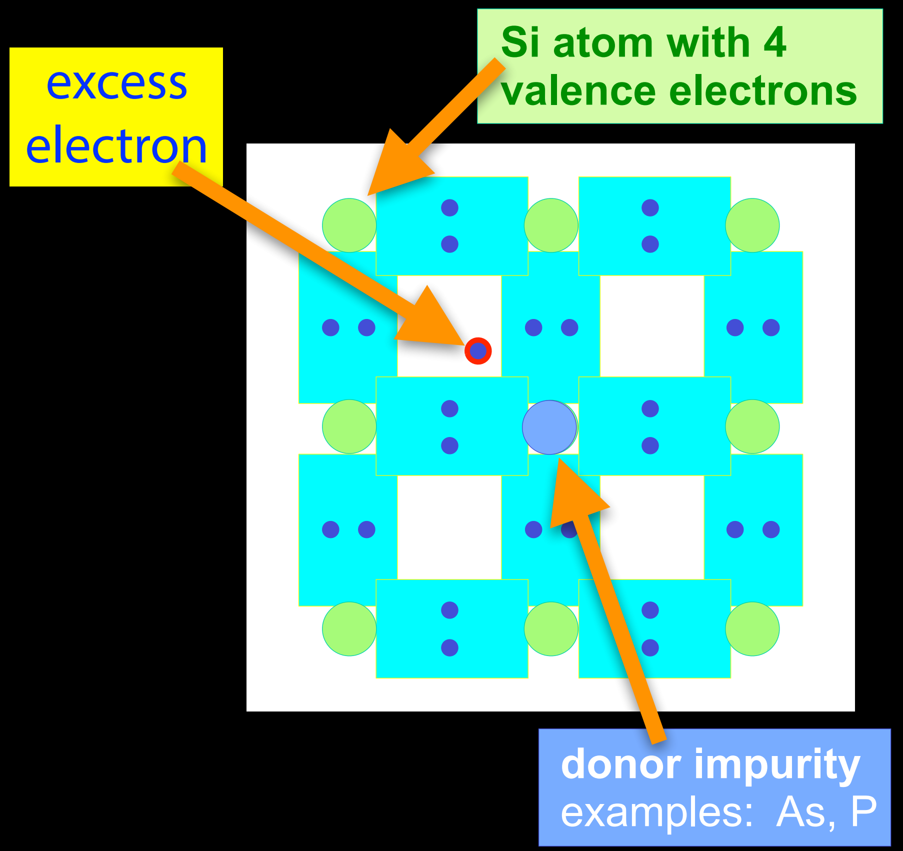
- schema of a silicon diode (p-n junction)
  - ➡ doping silicon cristal semiconductor to implant excess electrons or "holes"

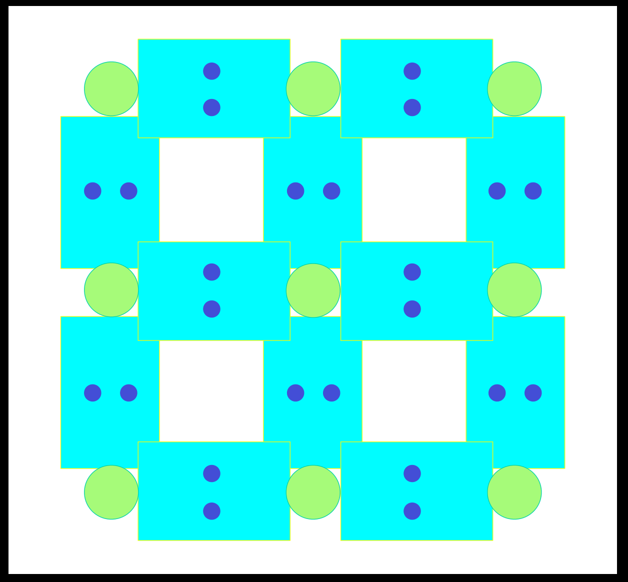




Semiconductors as Particle Detectors

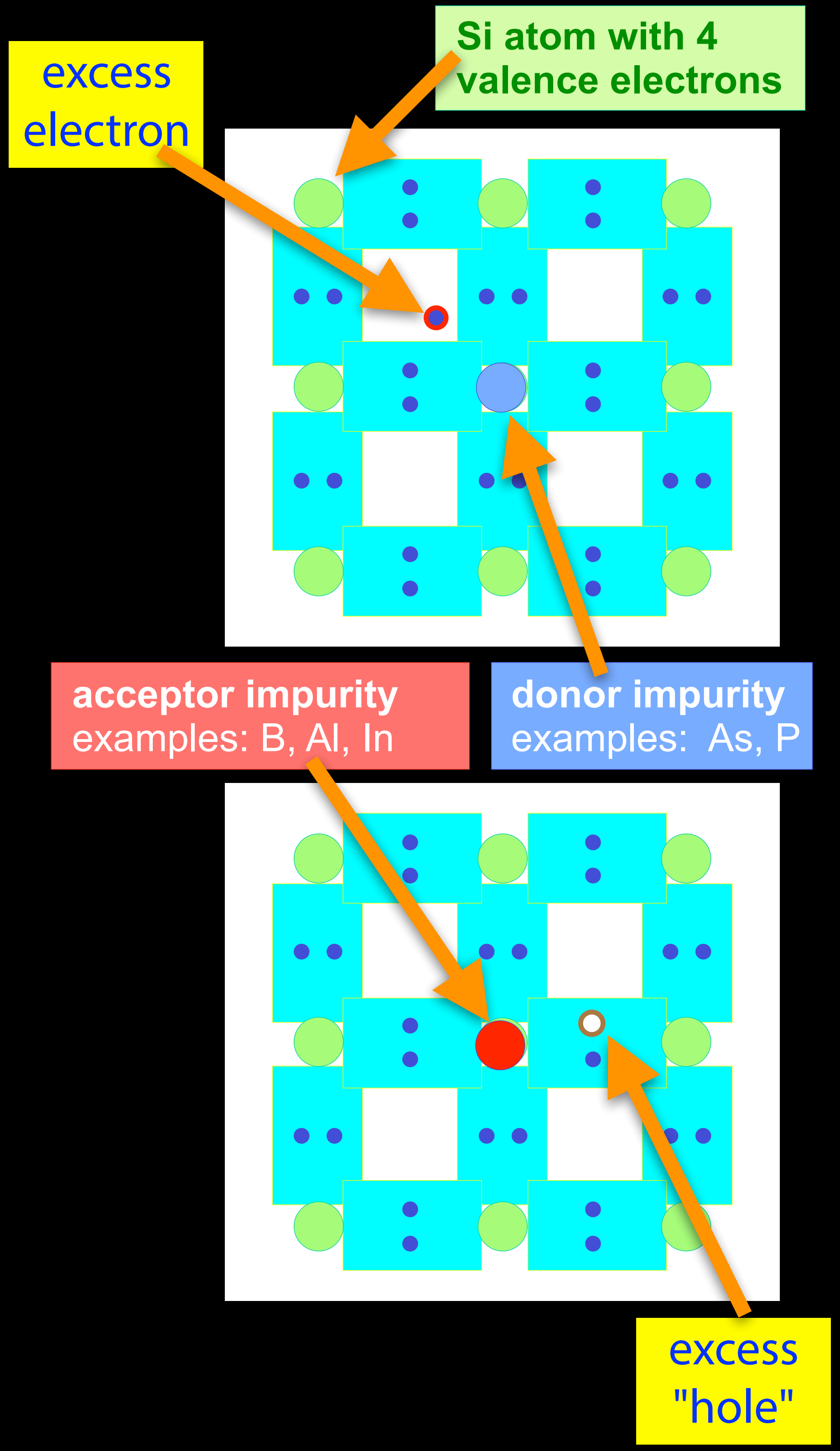
- schema of a silicon diode (p-n junction)
  - ➡ doping silicon cristal semiconductor to implant excess electrons or "holes"
    - n doping adds electro-phile atoms





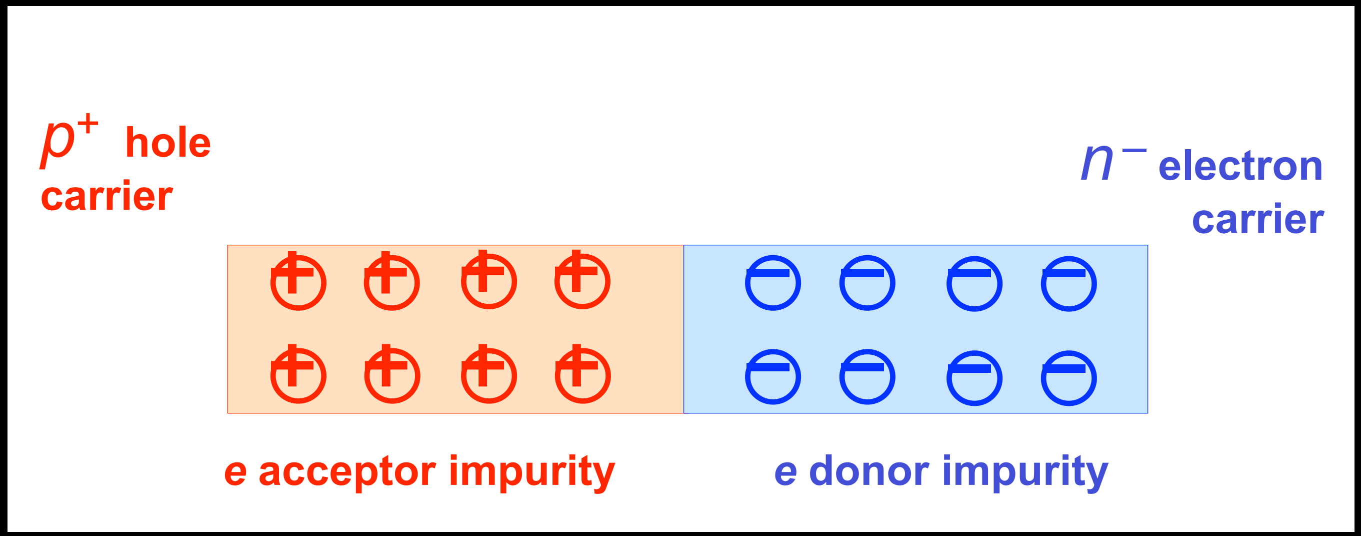
Semiconductors as Particle Detectors

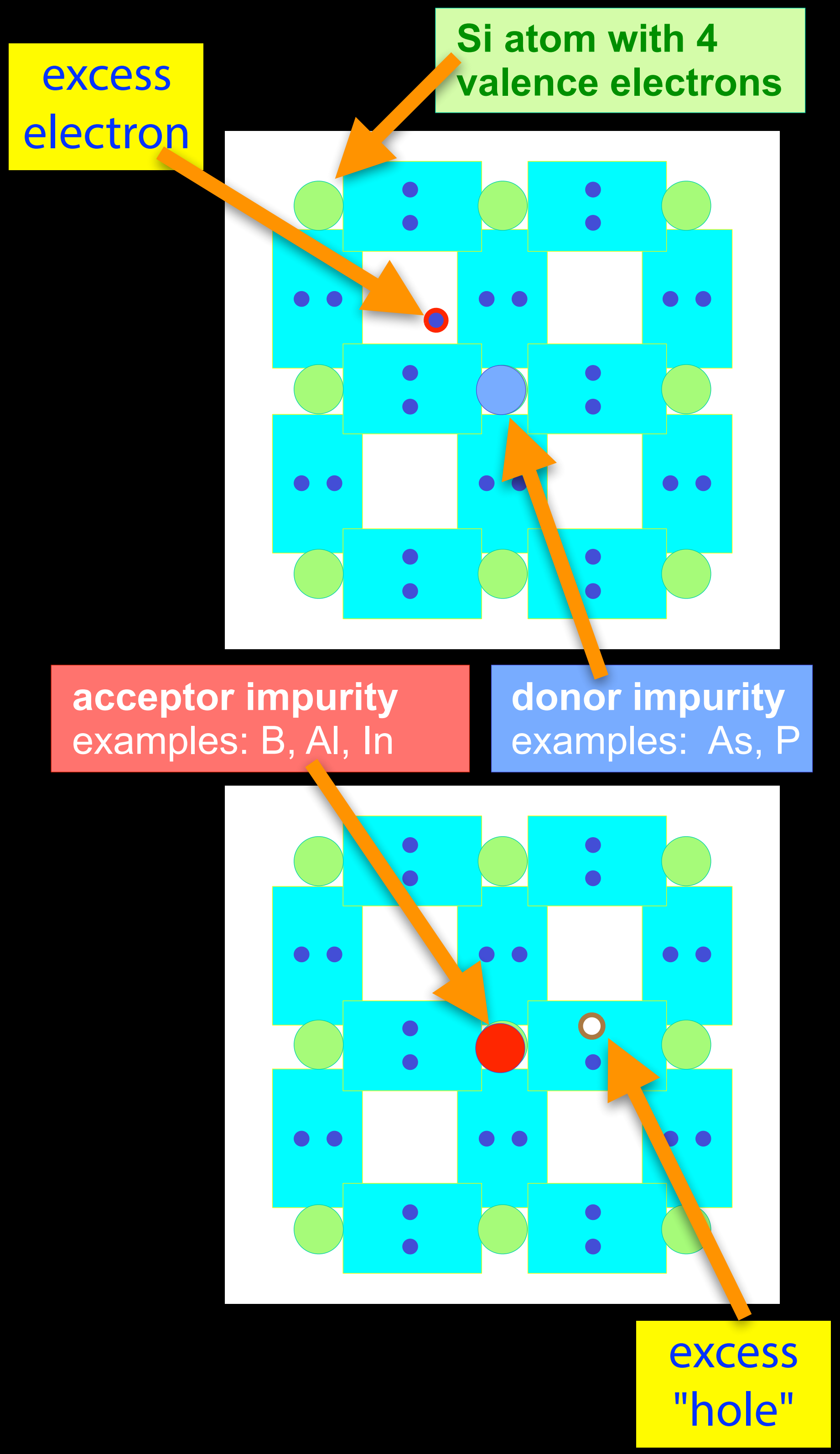
- schema of a silicon diode (p-n junction)
  - ➡ doping silicon cristal semiconductor to implant excess electrons or "holes"
    - n doping adds electro-phile atoms
    - p doping adds electro-phobe atoms



Semiconductors as Particle Detectors

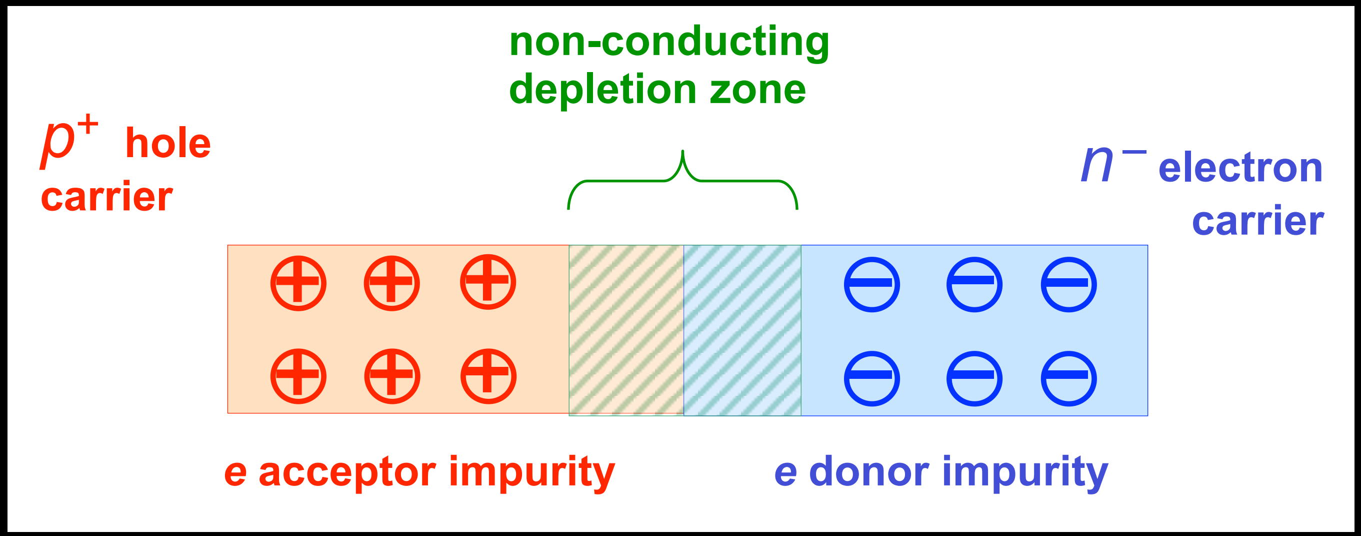
- schema of a silicon diode (p-n junction)
  - ➡ doping silicon cristal semiconductor to implant excess electrons or "holes"
    - n doping adds electro-phile atoms
    - p doping adds electro-phobe atoms
  - ➡ both materials together form a diode



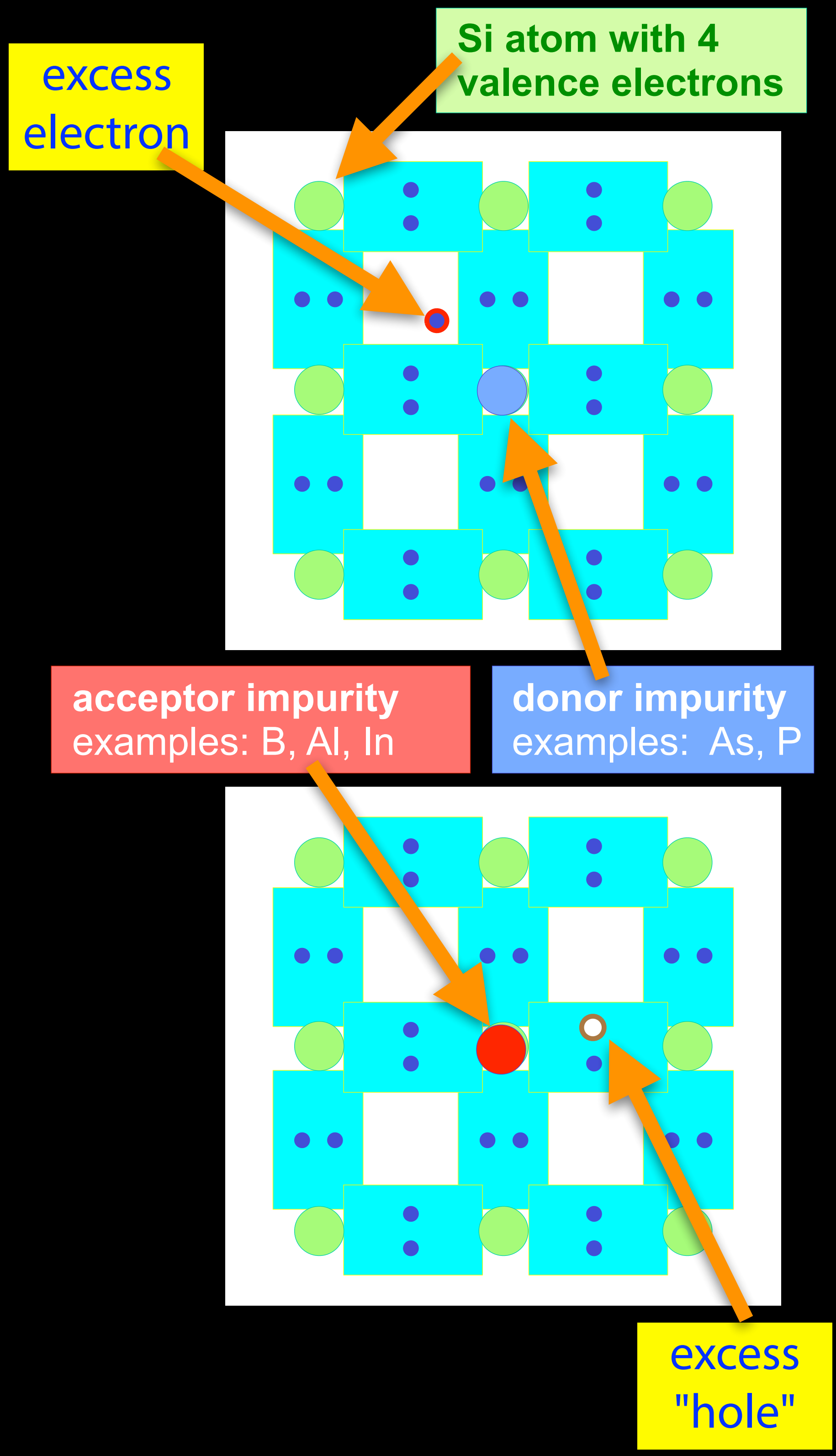


Semiconductors as Particle Detectors

- schema of a silicon diode (p-n junction)
  - ➡ doping silicon cristal semiconductor to implant excess electrons or "holes"
    - n doping adds electro-phile atoms
    - p doping adds electro-phobe atoms
  - ➡ both materials together form a diode

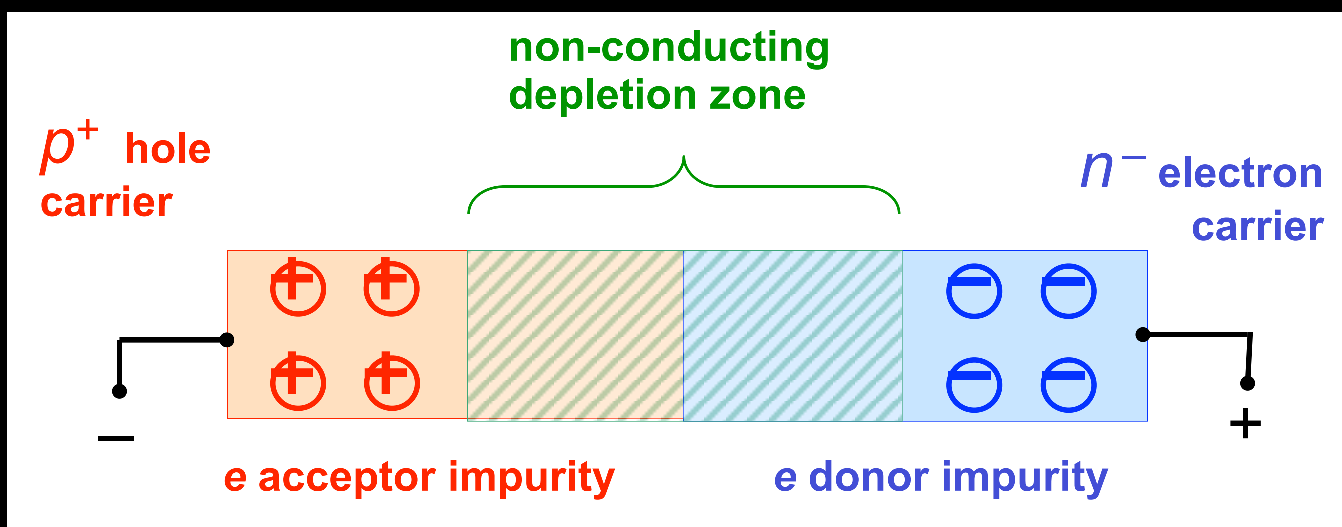


- recombination in junction creates depletion zone, acts as potential barrier against doping potential

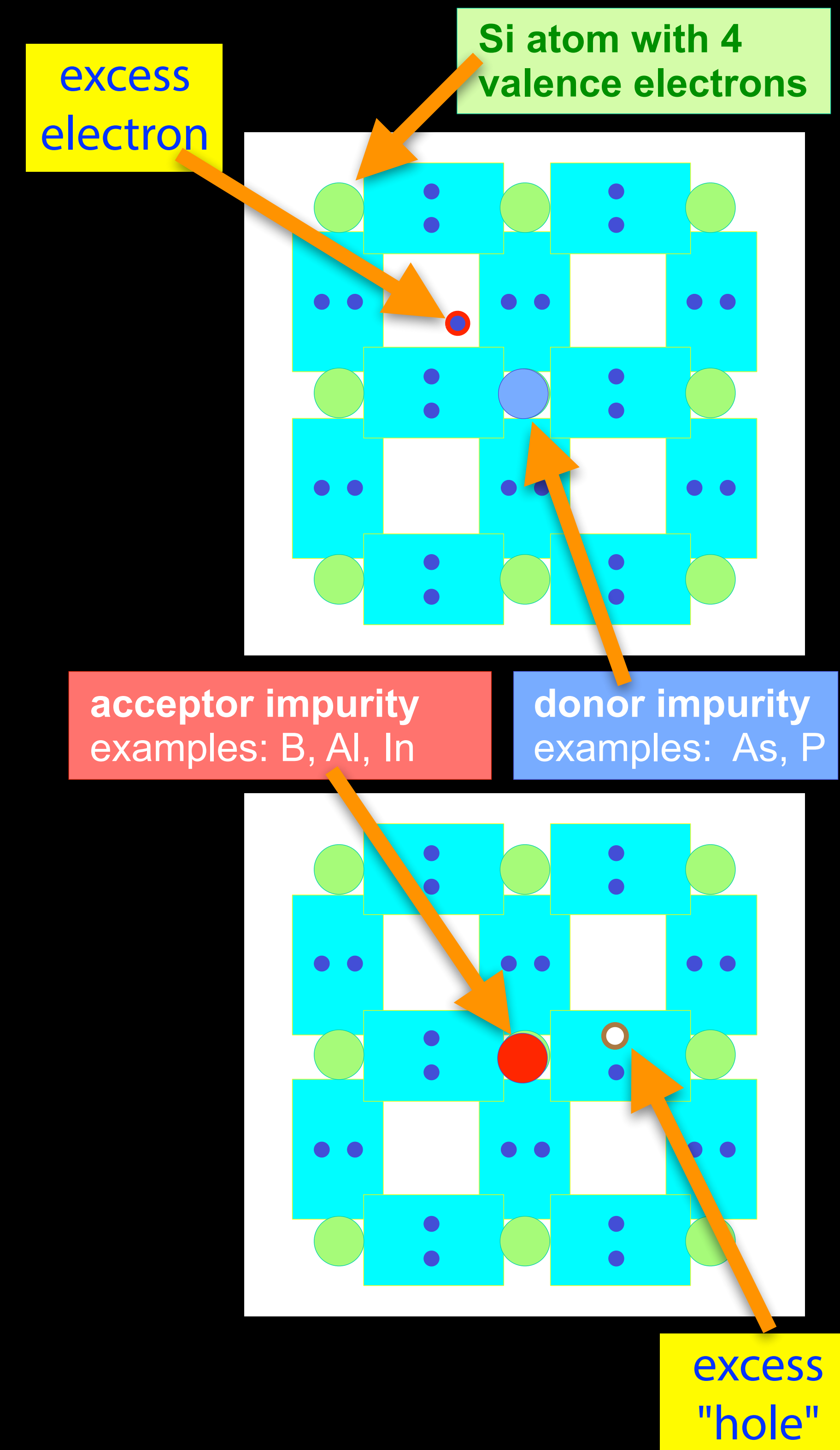


Semiconductors as Particle Detectors

- schema of a silicon diode (p-n junction)
  - ➡ doping silicon cristal semiconductor to implant excess electrons or "holes"
    - n doping adds electro-phile atoms
    - p doping adds electro-phobe atoms
  - ➡ both materials together form a diode

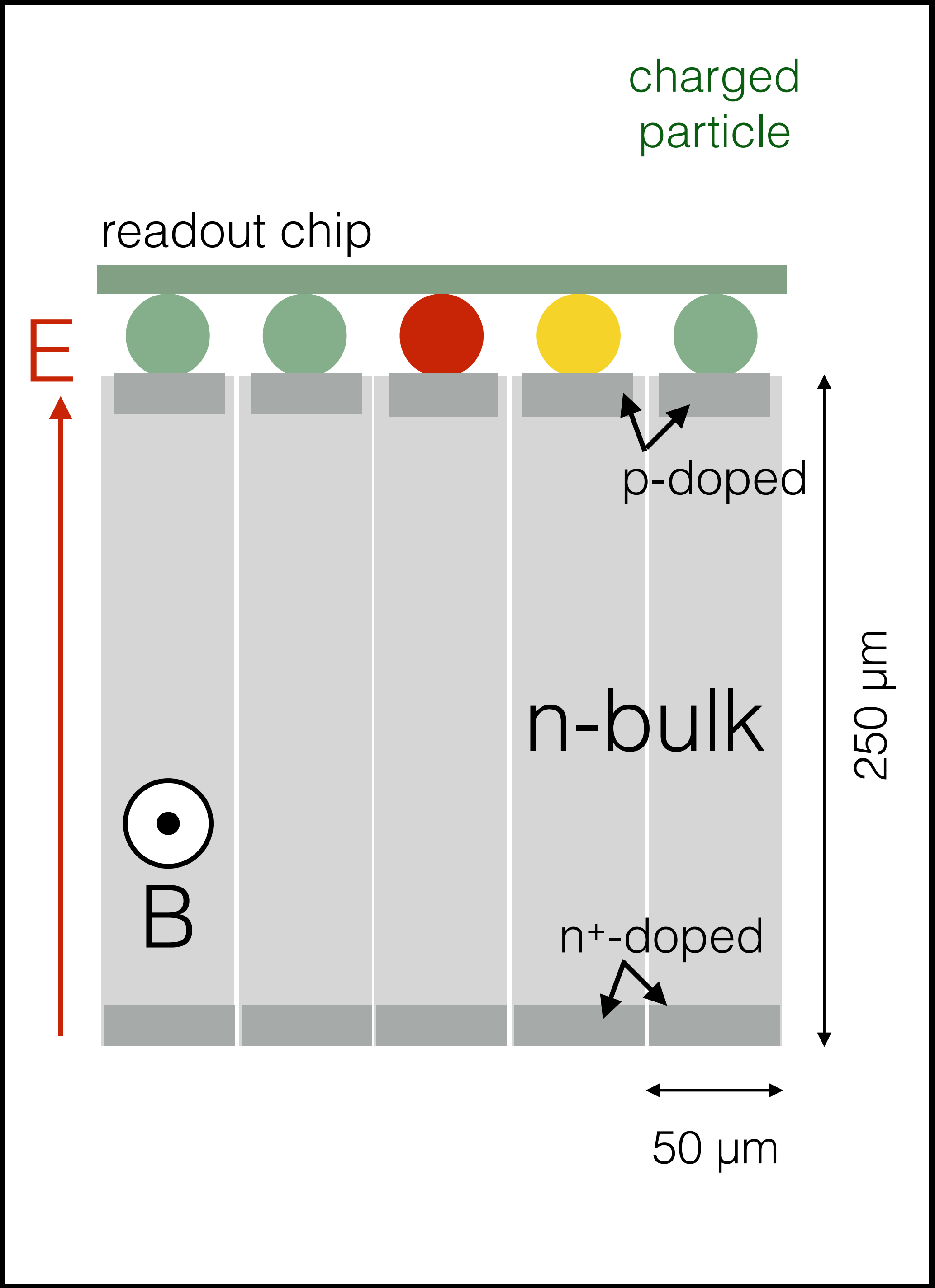


  - recombination in junction creates depletion zone, acts as potential barrier against doping potential
  - apply reverse bias voltage to enlarge potential barrier in depletion zone, increases its resistance further



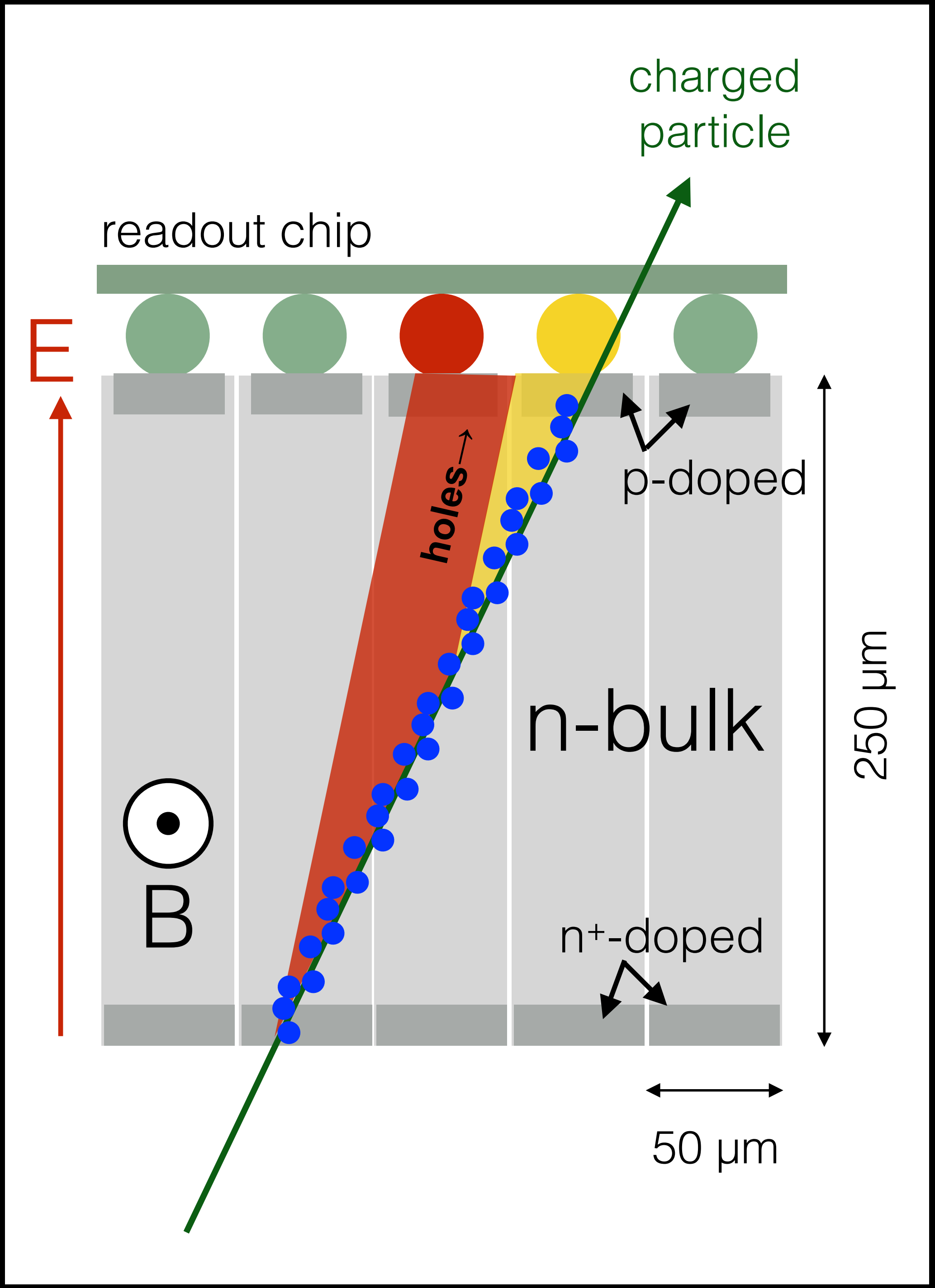
Semiconductors as Particle Detectors

- basic schema of a silicon detector
  - ➡ many reverse biased large diodes on a silicon wafer
    - allows for small structures, typical pitch is 50 µm



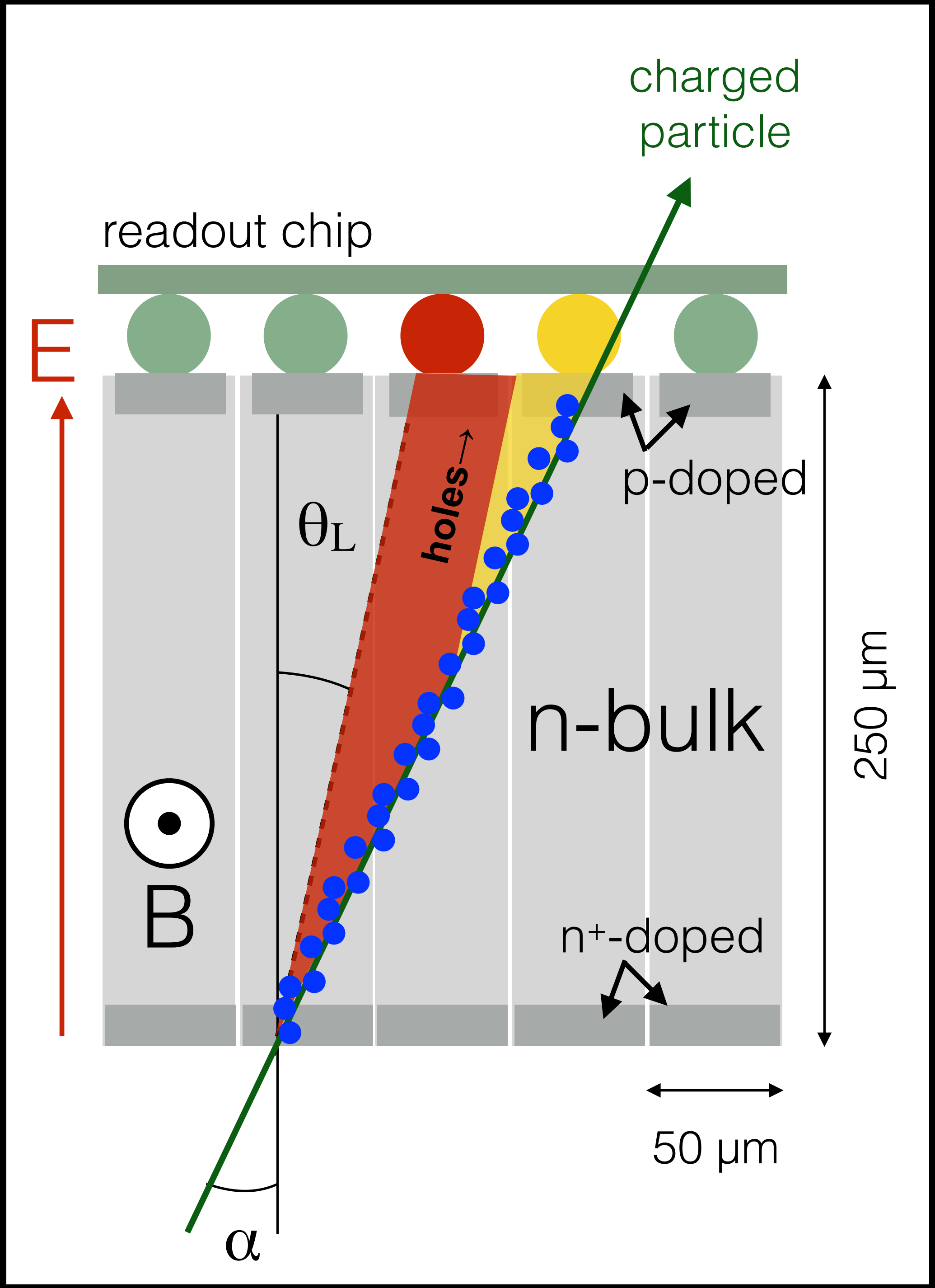
Semiconductors as Particle Detectors

- basic schema of a silicon detector
  - ➡ many reverse biased large diodes on a silicon wafer
    - allows for small structures, typical pitch is 50 μm
  - ➡ traversing charged particle ionises silicon
    - creates electron-hole pairs, drifting in E-field to electrodes leading to measurable signals in diodes



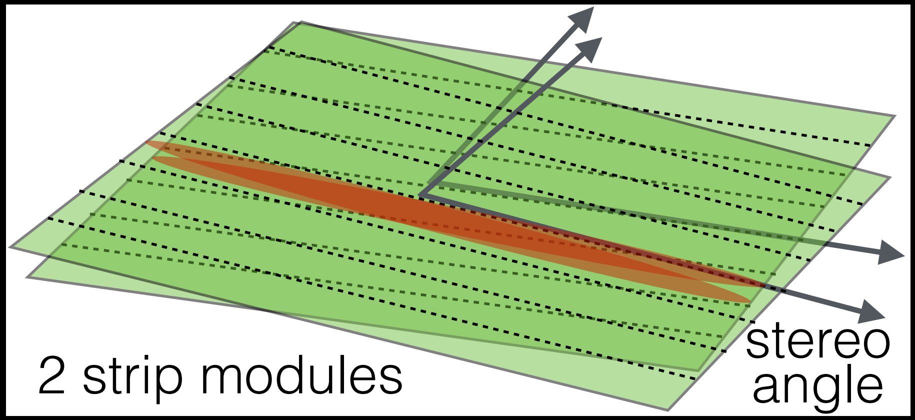
Semiconductors as Particle Detectors

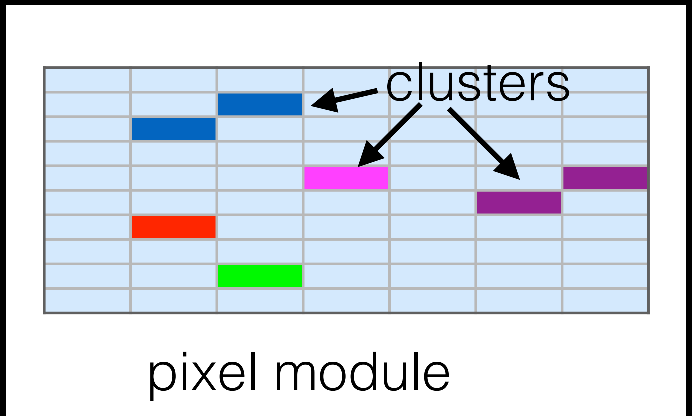
- basic schema of a silicon detector
  - ➡ many reverse biased large diodes on a silicon wafer
    - allows for small structures, typical pitch is 50 µm
  - ➡ traversing charged particle ionises silicon
    - creates electron-hole pairs, drifting in E-field to electrodes leading to measurable signals in diodes
    - Lorentz angle θ_L deflection in presence of B-field

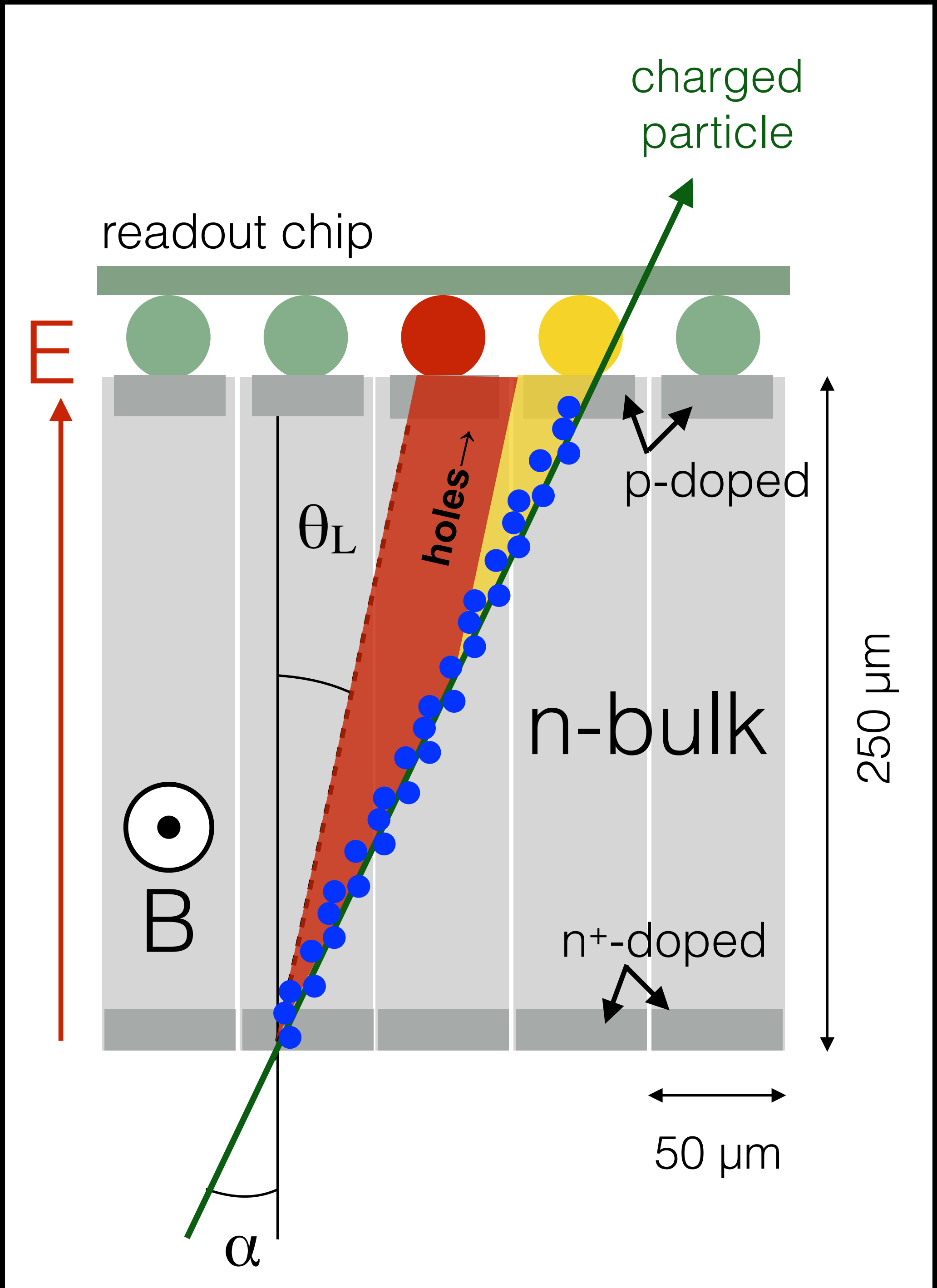


Semiconductors as Particle Detectors

- basic schema of a silicon detector
  - ➡ many reverse biased large diodes on a silicon wafer
    - allows for small structures, typical pitch is 50 µm
  - ➡ traversing charged particle ionises silicon
    - creates electron-hole pairs, drifting in E-field to electrodes leading to measurable signals in diodes
    - Lorentz angle θ_L deflection in presence of B-field
- 2 types: silicon strips and pixels
  - ➡ strip module: 50 µm pitch, wafers with ~6 cm diodes
    - needs 2 modules to measure both coordinates
  - ➡ pixel module: e.g. 50x400 µm pixel, analog readout
    - clusters measures precisely both coordinates

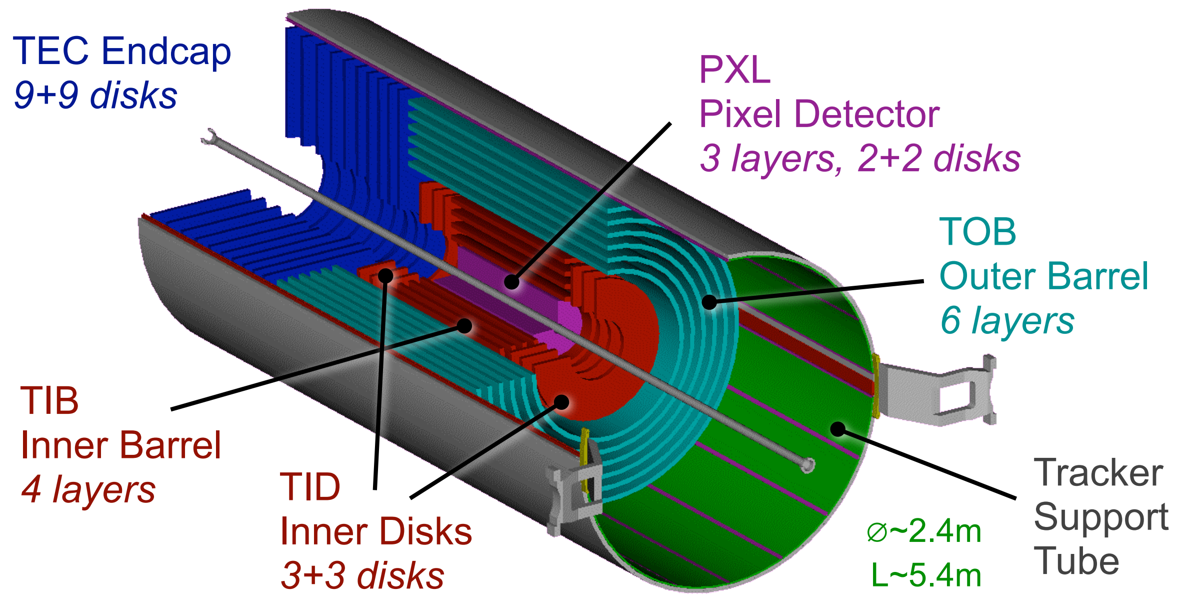




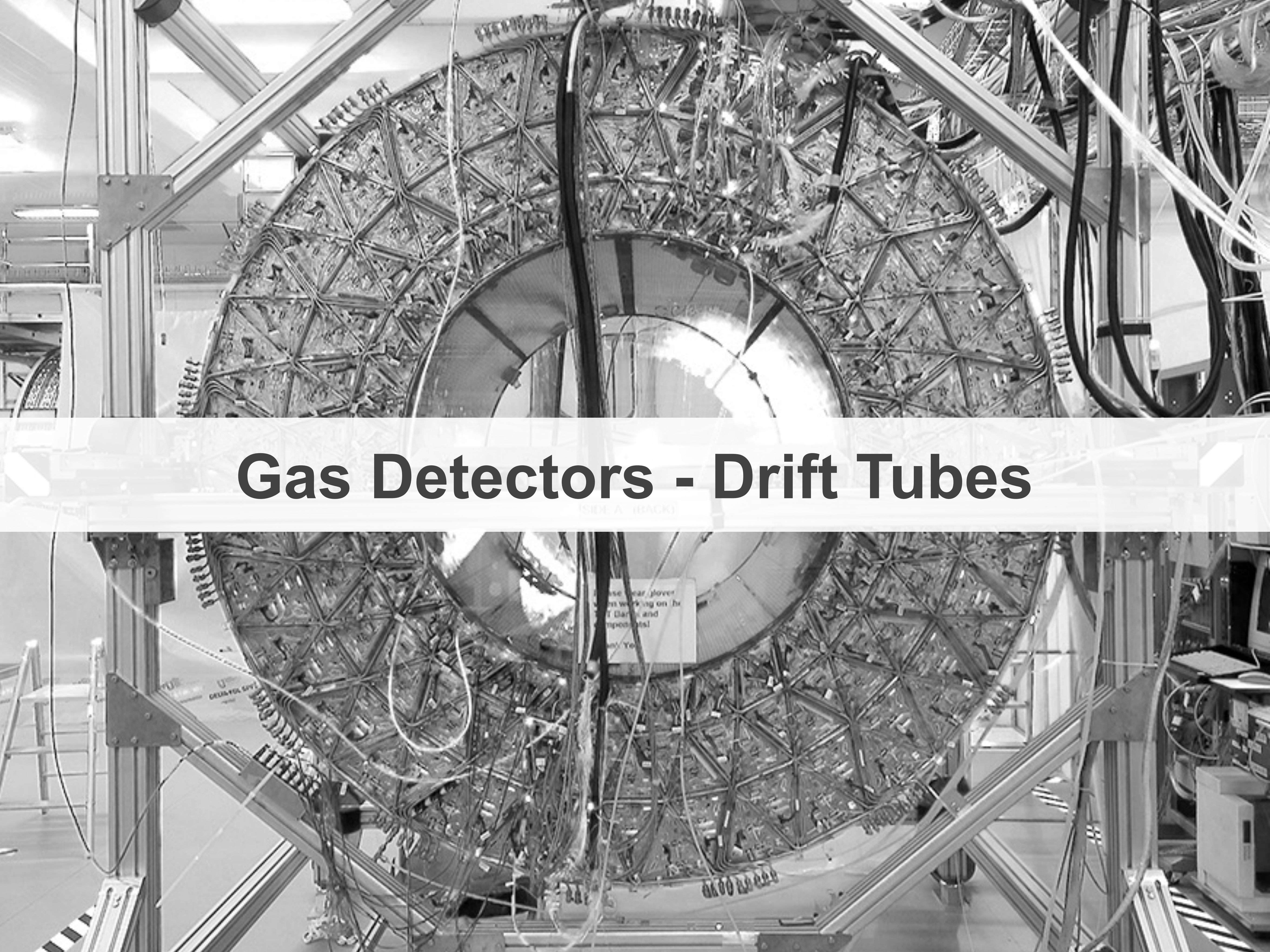


CMS Tracker

- largest silicon tracker ever built
  - ➡ **Pixels:** 66M channels, 100x150 μm² Pixel
  - ➡ **strip detector:** ~23m³, 210m² of Si area, 10.7M channels

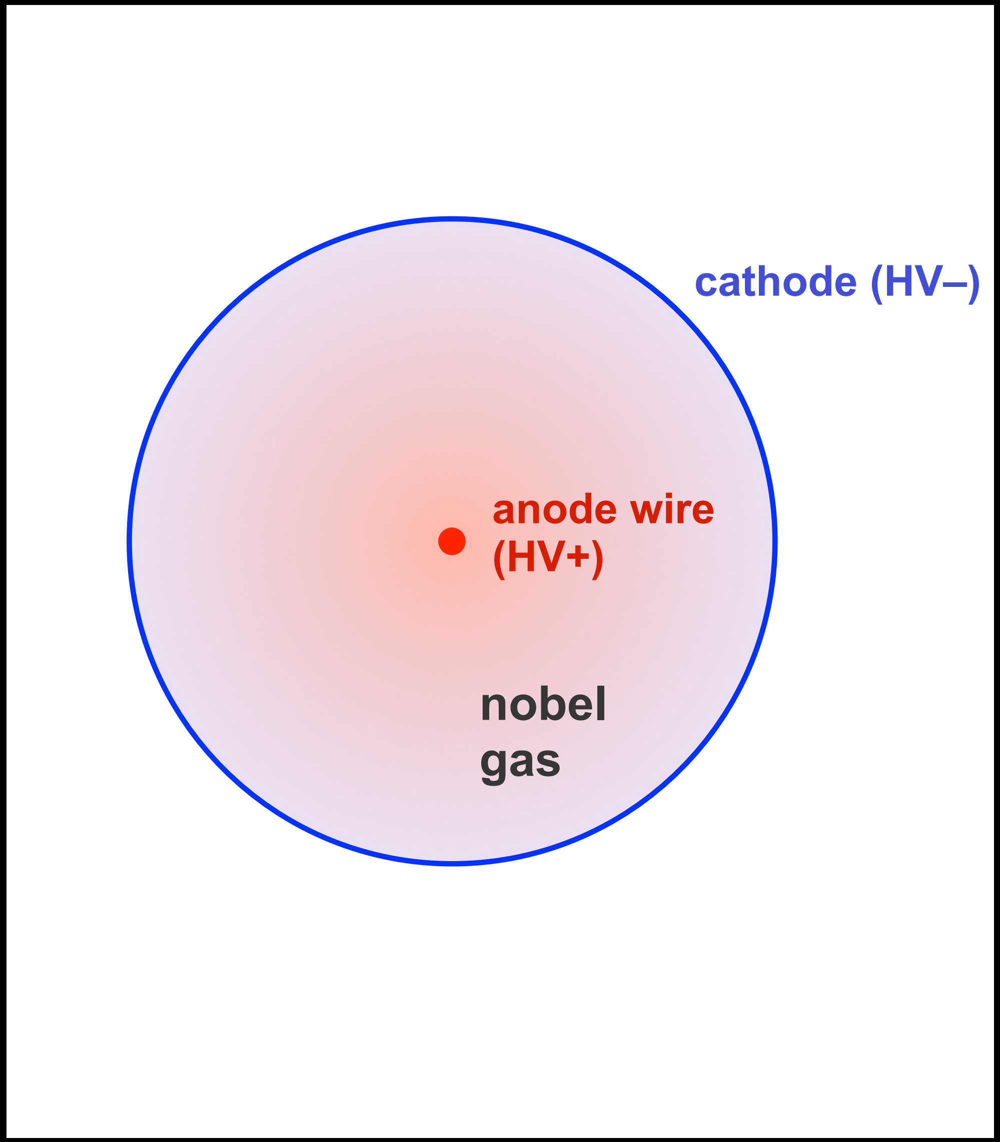


Gas Detectors - Drift Tubes



Classical Gas Detectors - Drift Tubes

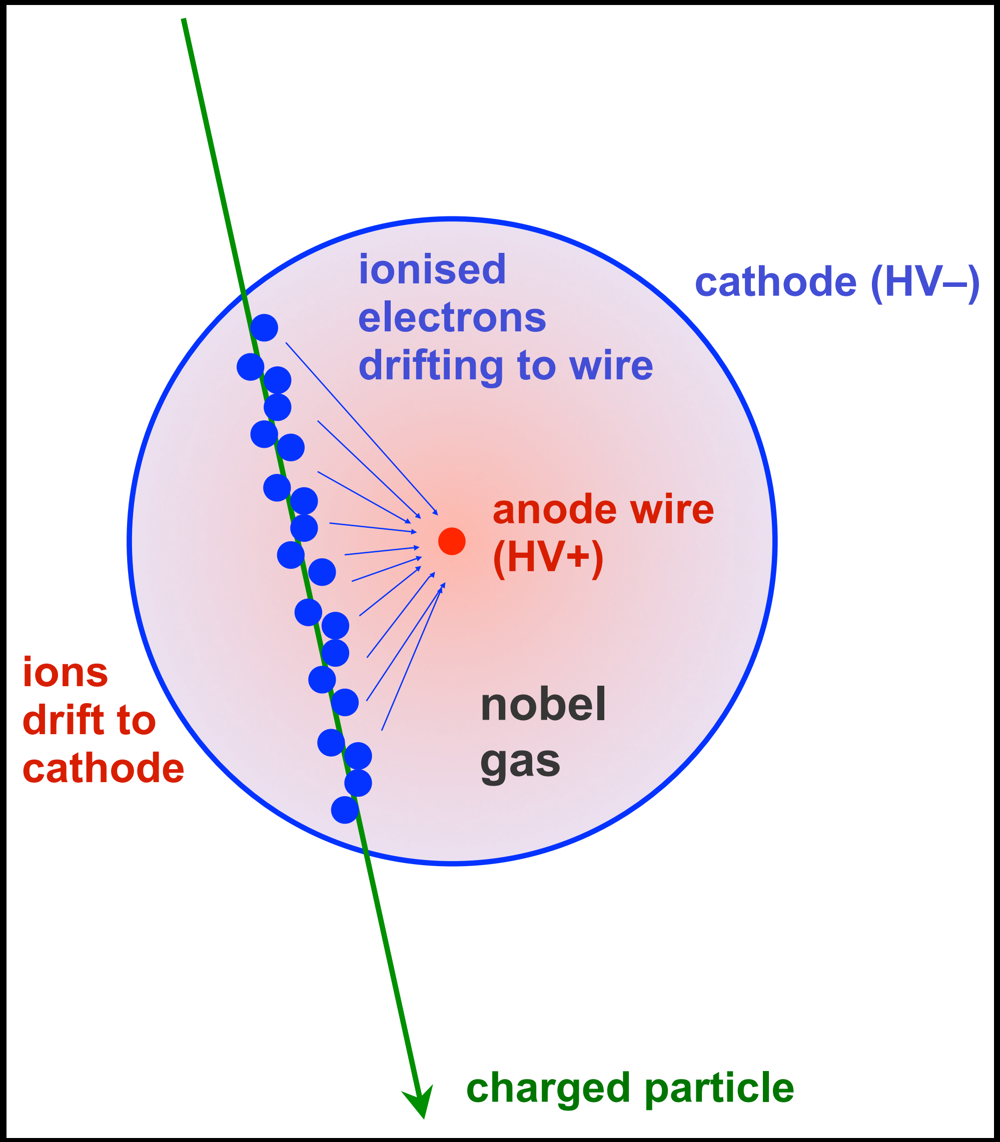
- detection technique for charged particles
  - ➡ used in muon systems and ATLAS TRT



TRT: Kapton tubes, ∅ = 4 mm
MDT: Aluminium tubes, ∅ = 30 mm

Classical Gas Detectors - Drift Tubes

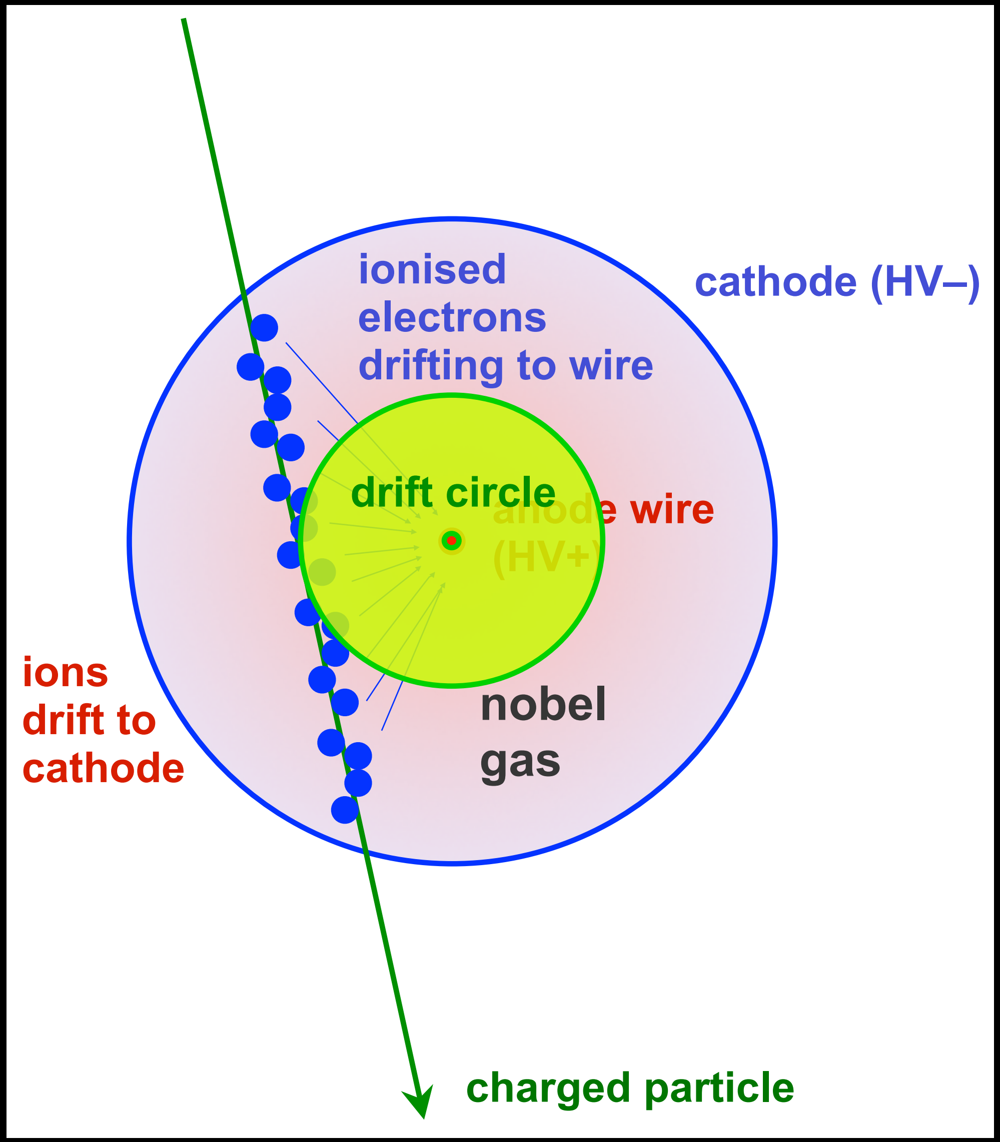
- detection technique for charged particles
  - ➡ used in muon systems and ATLAS TRT
- particles traversing tube ionises the gas
  - ➡ deposited charge drifts to anode wire in electric (E) field
    - charge amplification in high E-field in vicinity of wire leads to large signal pulse
    - Lorentz angle deflection in B-field (not shown)



TRT: Kapton tubes, ∅ = 4 mm
MDT: Aluminium tubes, ∅ = 30 mm

Classical Gas Detectors - Drift Tubes

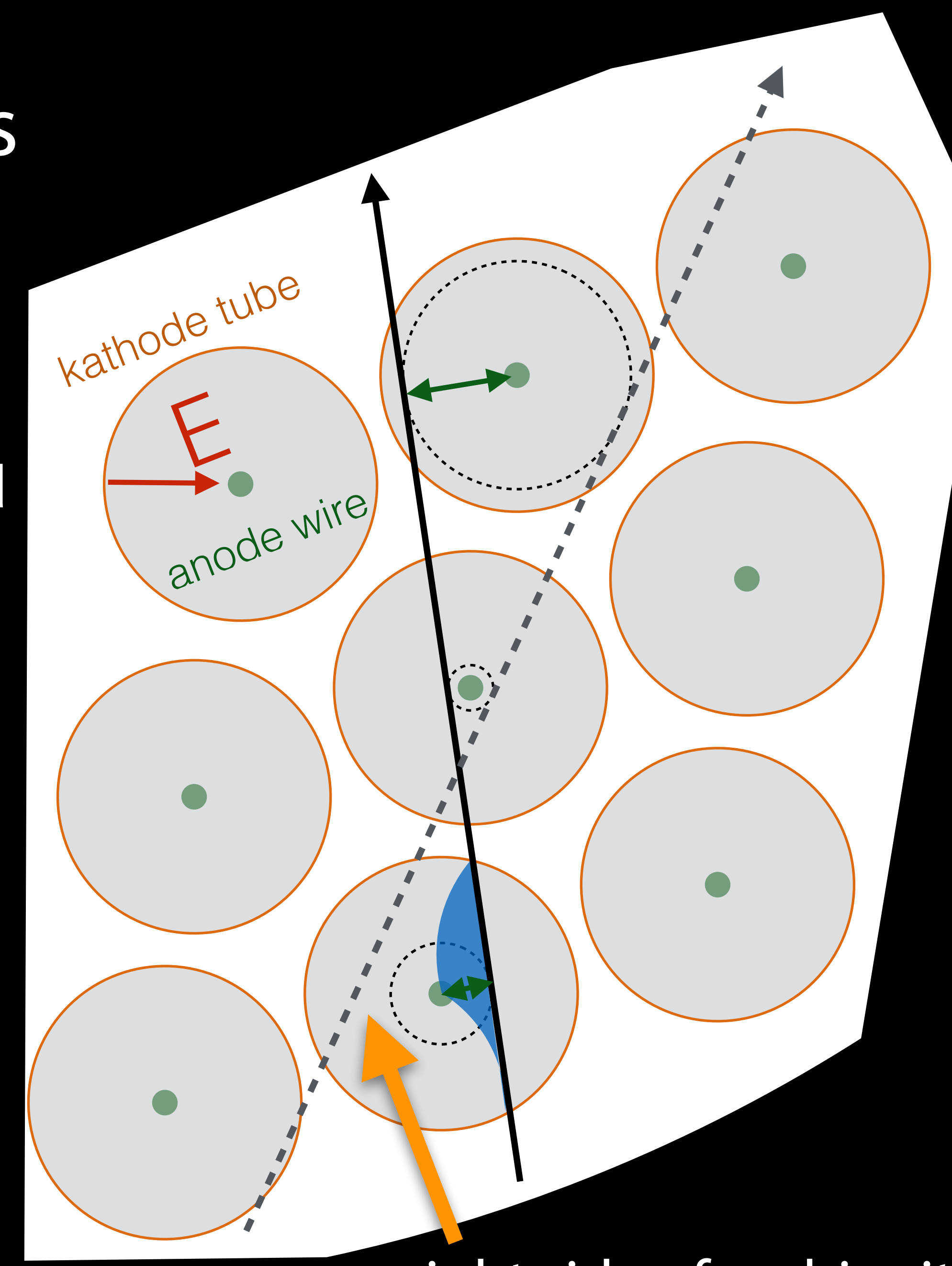
- detection technique for charged particles
  - ➡ used in muon systems and ATLAS TRT
- particles traversing tube ionises the gas
  - ➡ deposited charge drifts to anode wire in electric (E) field
    - charge amplification in high E-field in vicinity of wire leads to large signal pulse
    - Lorentz angle deflection in B-field (not shown)
  - ➡ measure time of signal pulse to determine drift circle
    - fast signal detection (v_D~30 ns/mm)
    - resolution of O(100 μm) on measured radius



TRT: Kapton tubes, ∅ = 4 mm
MDT: Aluminium tubes, ∅ = 30 mm

Classical Gas Detectors - Drift Tubes

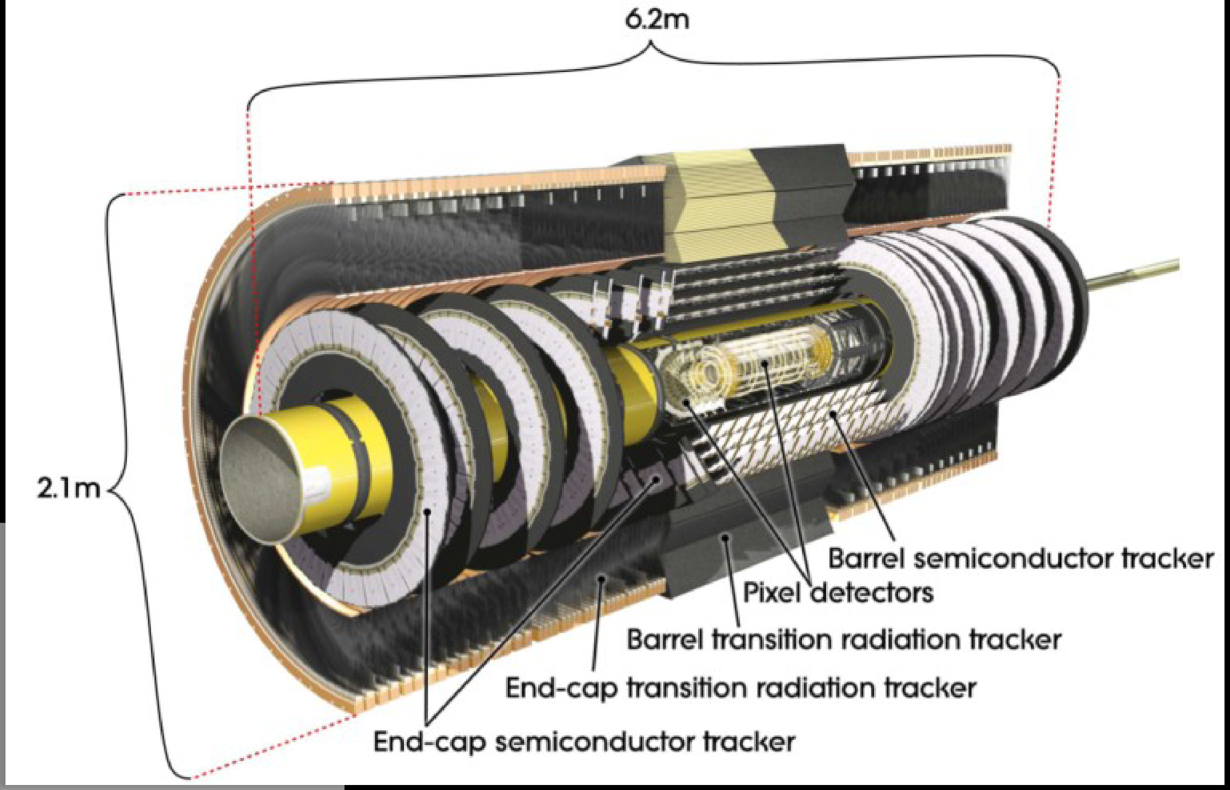
- detection technique for charged particles
  - ➡ used in muon systems and ATLAS TRT
- particles traversing tube ionises the gas
  - ➡ deposited charge drifts to anode wire in electric (E) field
    - charge amplification in high E-field in vicinity of wire leads to large signal pulse
    - Lorentz angle deflection in B-field (not shown)
  - ➡ measure time of signal pulse to determine drift circle
    - fast signal detection (v_D~30 ns/mm)
    - resolution of O(100 μm) on measured radius
- track reconstruction from drift circles
  - ➡ obtain drift radii from measured times
  - ➡ combined several measurements to find track
    - resolve left-right ambiguity (dotted line)
  - ➡ ATLAS TRT: as well electron identification using transition radiation

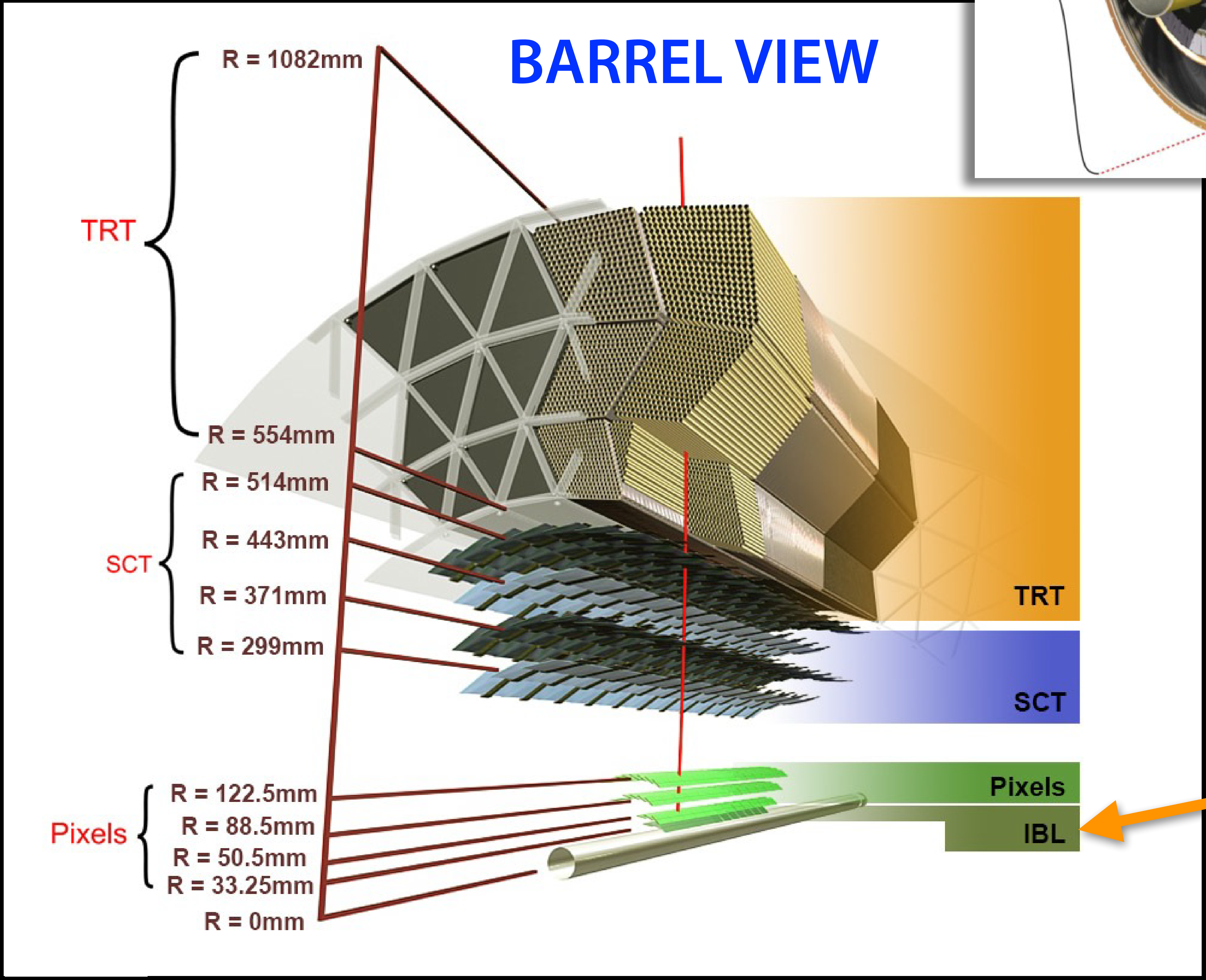


right side of ambiguity has large residual

ATLAS Inner Detector

- expanded view of barrel
  - ➡ IBL was installed 2014 !



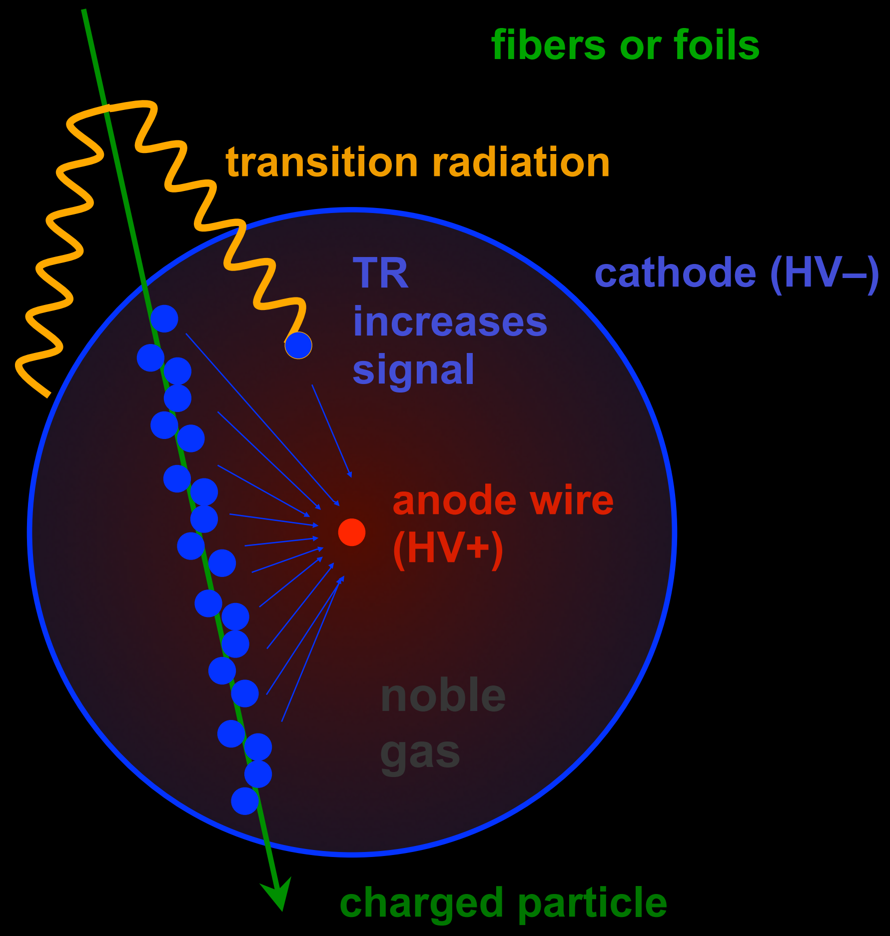


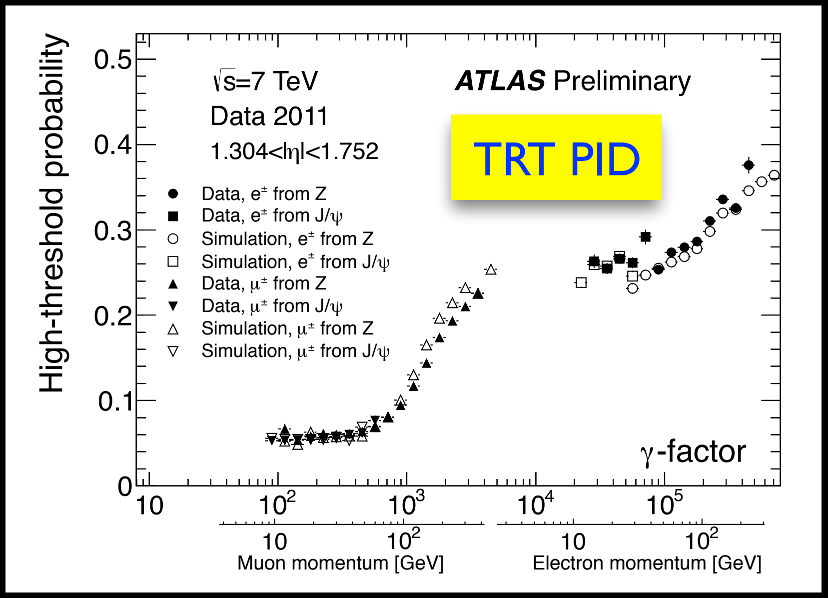
- barrel track passes:
  - ➡ 4(!) Pixel layers
  - ➡ 4x2 silicon Strips on stereo modules
  - ➡ ~36 TRT 4mm straws

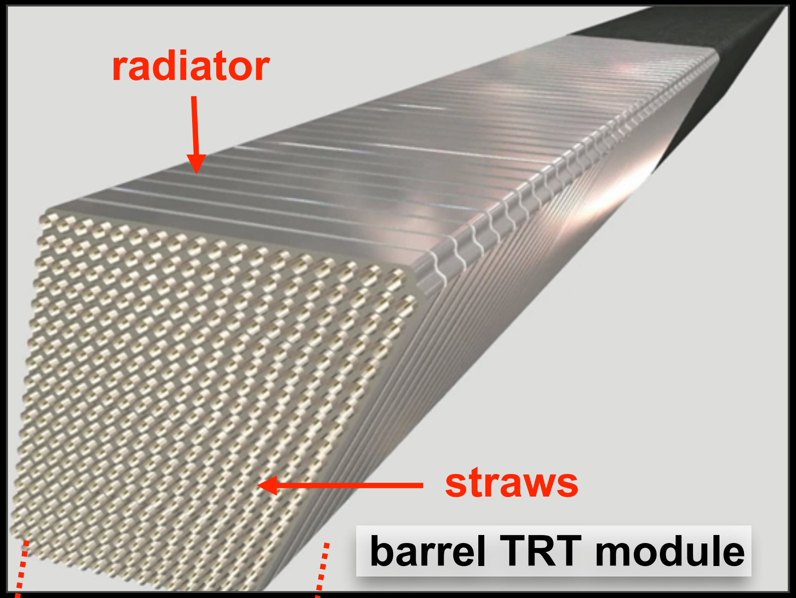
for completeness

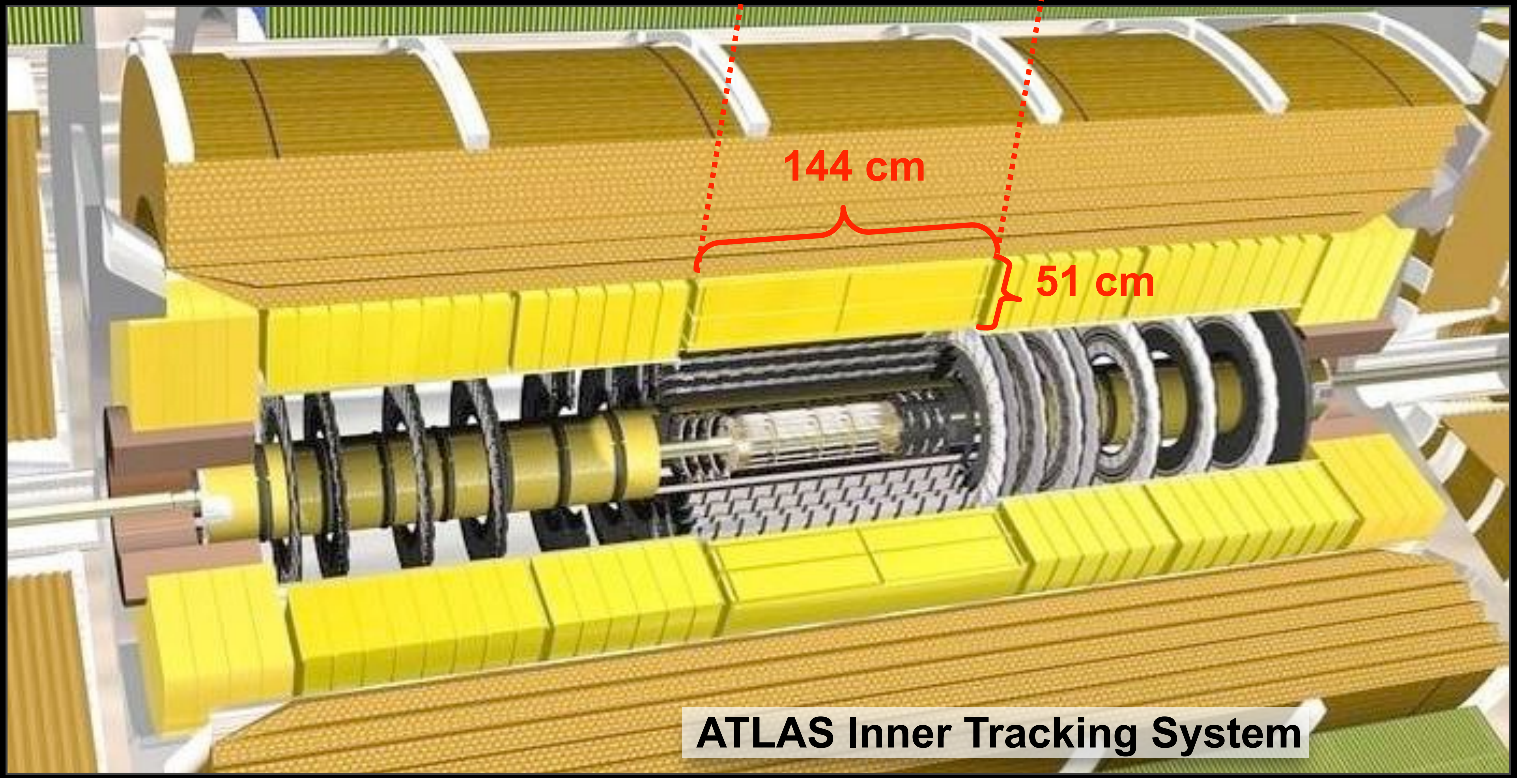
Electron Identification in the ATLAS TRT

➡ e/π separation via transition radiation: polymer (PP) fibers/foils interleaved with drift tubes









➡ electrons radiate → higher signal
- PID info by counting high-threshold hits component precisely

Charged Particle Trajectories and Extrapolation

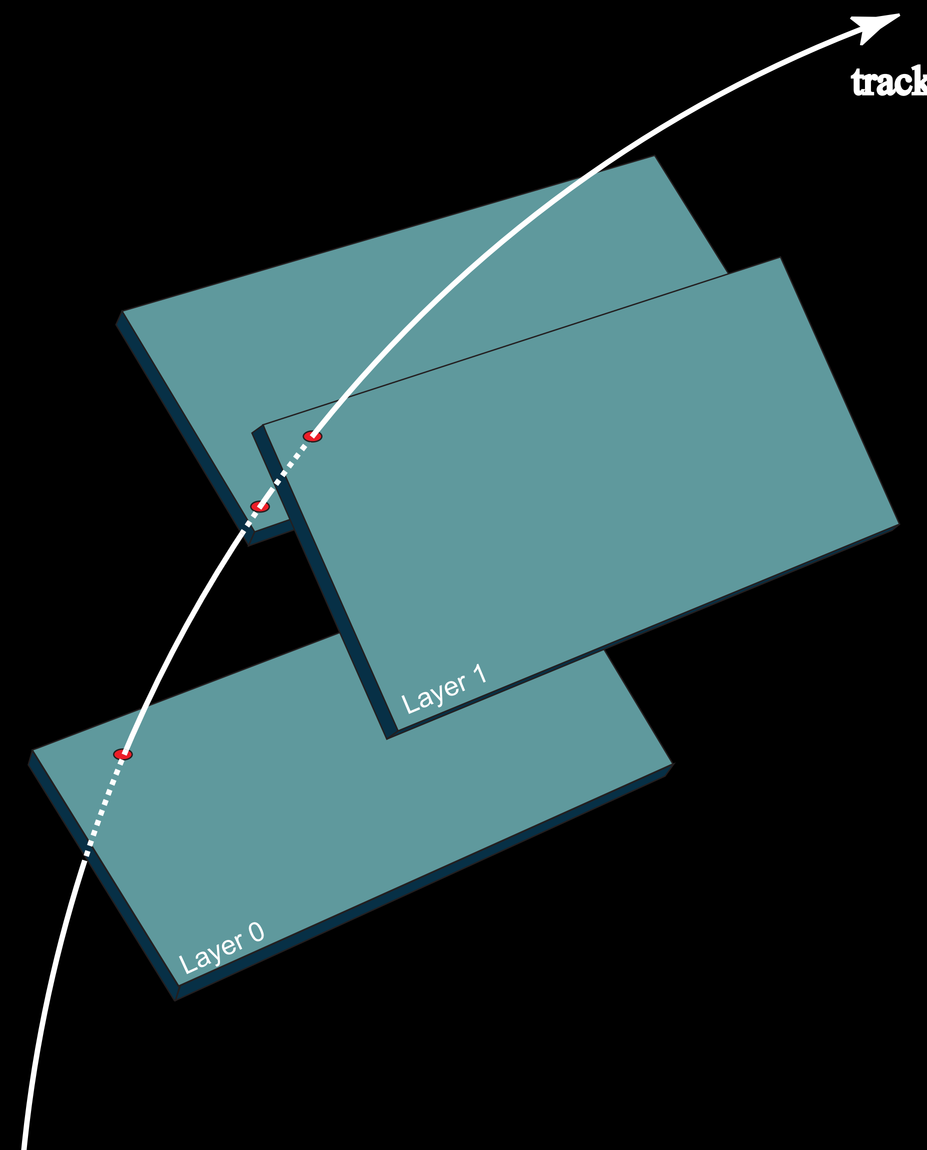
A Trajectory of a Charged Particle

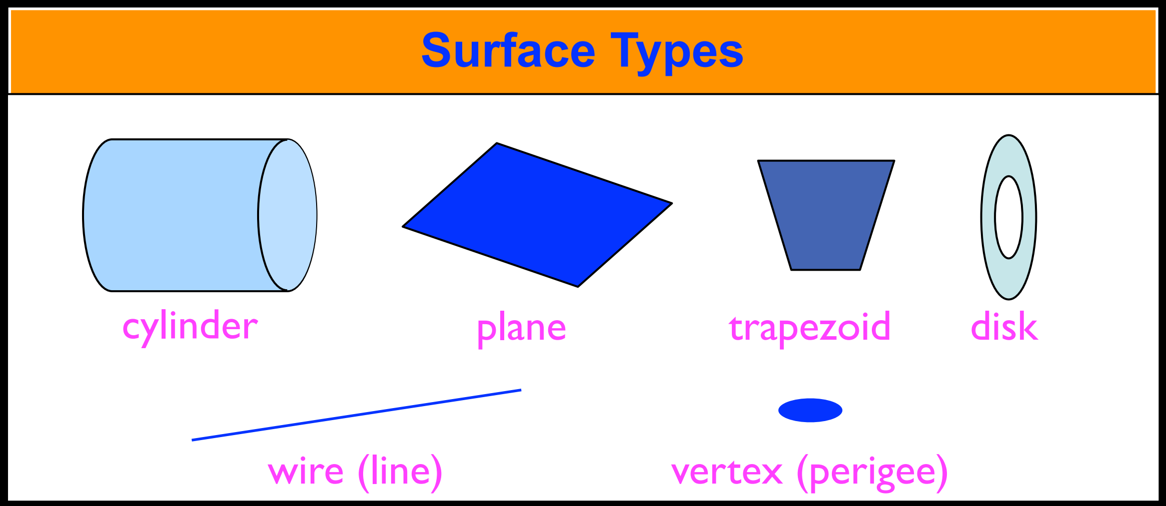
➡ in a solenoid B field a charged particle trajectory is describing a **helix**
- a circle in the plane perpendicular to the field (Rϕ)
- a path (not a line) at constant polar angle (θ) in the Rz plane

➡ a trajectory in space is defined by **5 parameters**
- the **local position** (l_1,l_2) on a plane, a cylinder, ..., on the surface or reference system
- the **direction** in θ and ϕ plus the **curvature** Q/P_T

➡ ATLAS choice:

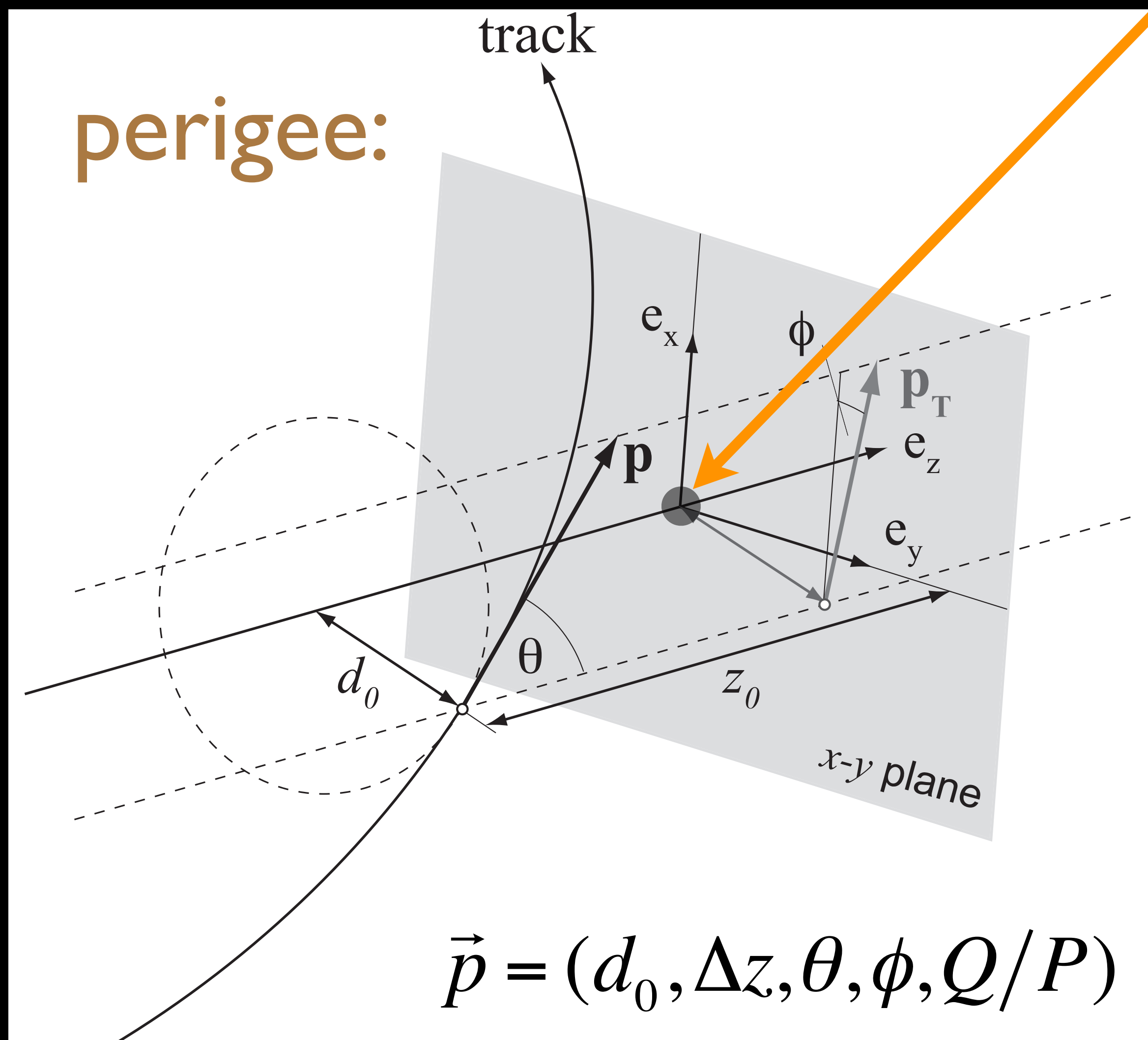
$$\vec{p} = (l_1, l_2, \theta, \phi, Q/P)$$





The Perigee Parameterization

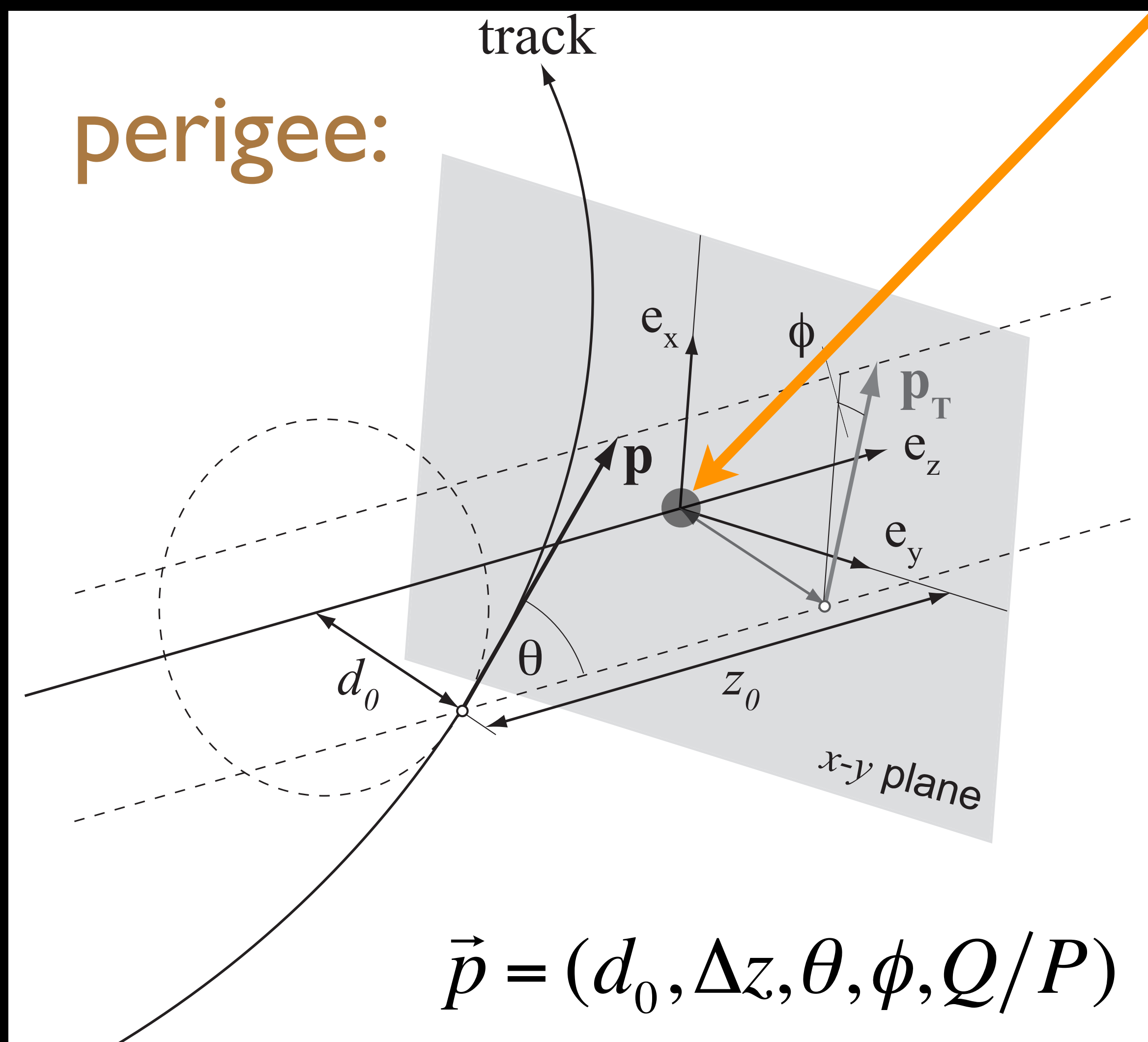
- helix representation w.r.t. a vertex

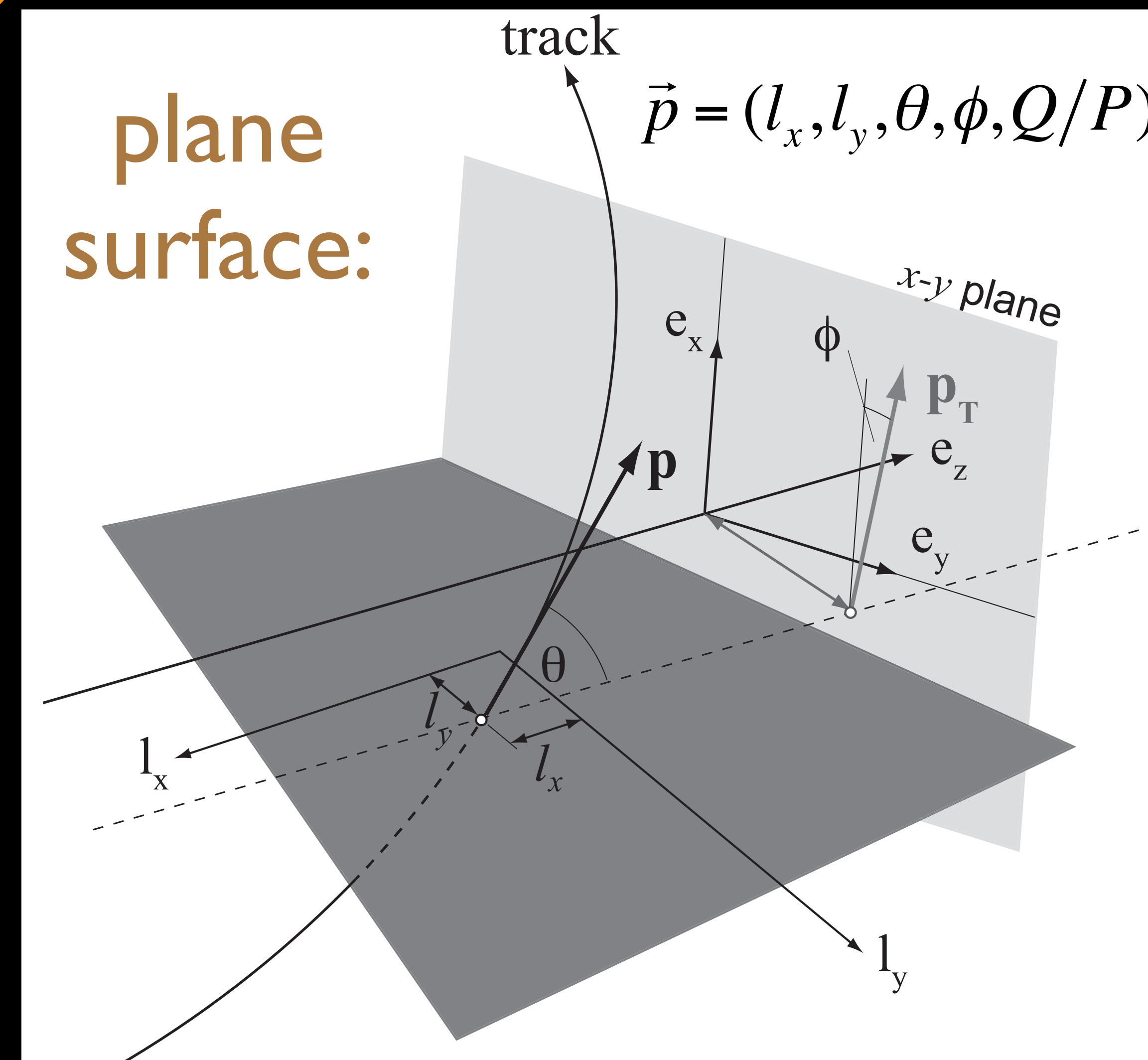


- commonly used
  - ➡ e.g. to express track parameters near the production vertex
  - ➡ alternative: e.g. on plane surface

The Perigee Parameterization

- helix representation w.r.t. a vertex





- commonly used
  - ➡ e.g. to express track parameters near the production vertex
  - ➡ alternative: e.g. on plane surface

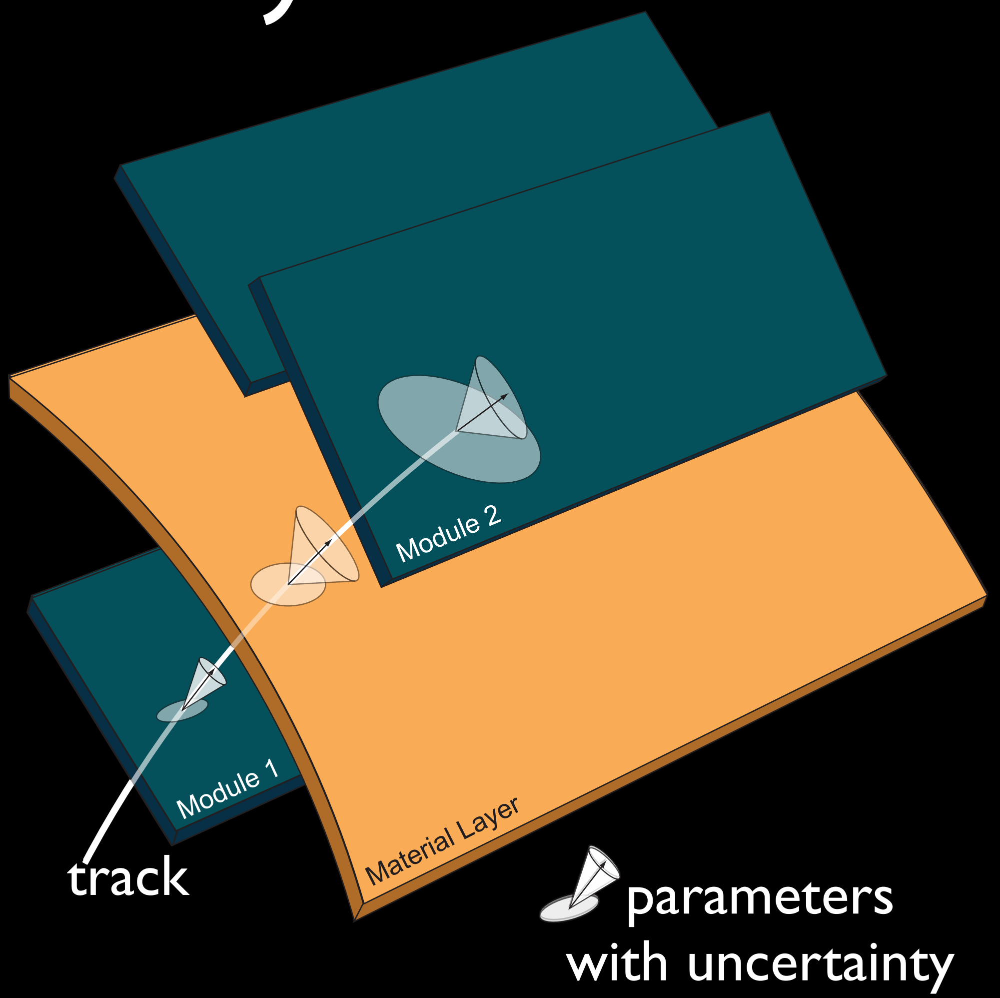
Following the Particle Trajectory

- basic problems to be solved in order to follow a track through a detector:
  - ➡ next detector module that it intersects ?
  - ➡ what are its parameters on this surface ?
    - what is the uncertainty of those parameters ?
  - ➡ for how much material do I have to correct for ?

- requires:
  - ➡ a detector geometry
    - surfaces for active detectors
    - passive material layers
  - ➡ a method to discover which is the next surface (navigation)
  - ➡ a propagator to calculate the new parameters and its errors
    - often referred to as "track model"

- for a constant B-field (or no field)
  - ➡ an analytical formula can be calculated for an intersection of a helix (or a straight line) on simple surfaces (plane, cylinder, vertex,...)

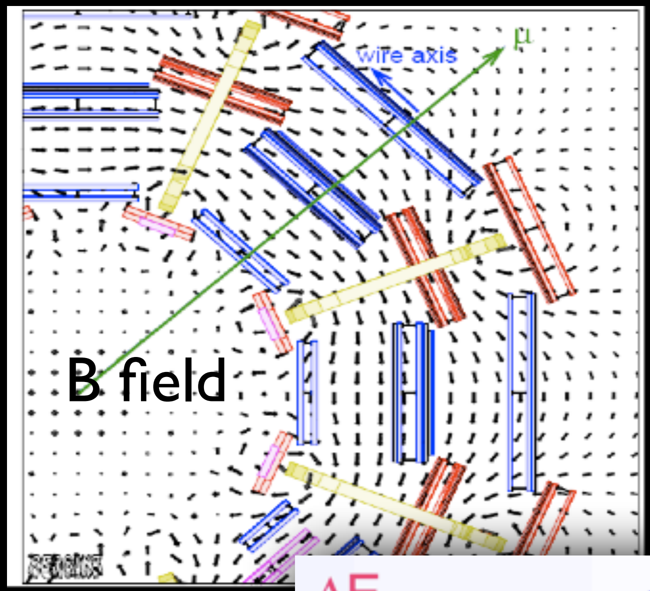
Following the Particle Trajectory



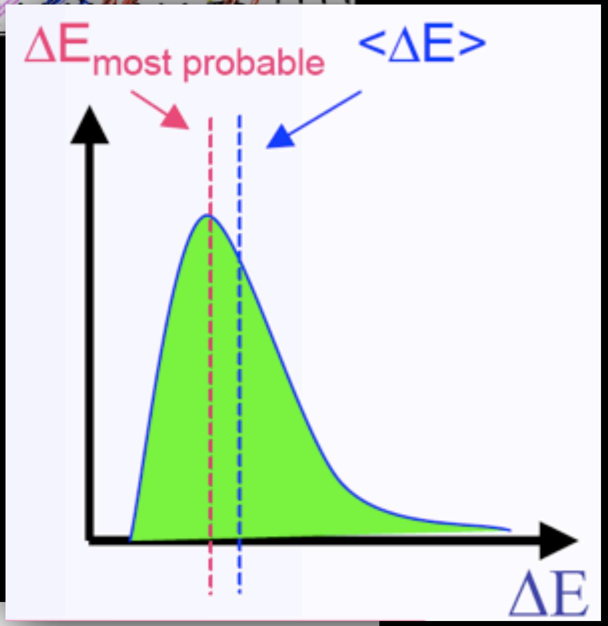
- basic problems to be solved in order to follow a track through a detector:
  - ➡ next detector module that it intersects ?
  - ➡ what are its parameters on this surface ?
    - what is the uncertainty of those parameters ?
  - ➡ for how much material do I have to correct for ?
- requires:
  - ➡ a detector geometry
    - surfaces for active detectors
    - passive material layers
  - ➡ a method to discover which is the next surface (navigation)
  - ➡ a propagator to calculate the new parameters and its errors
    - often referred to as "track model"
- for a constant B-field (or no field)
  - ➡ an analytical formula can be calculated for an intersection of a helix (or a straight line) on simple surfaces (plane, cylinder, vertex,...)

Effects of Material and realistic B-Field

- realistic non-homogeneous B-field
  - ➡ analytical helix propagation has to be replaced by numerical B-field integration along the path of the trajectory
  - ➡ in ATLAS and CMS a 4th order adaptive **Runge-Kutta-Nystrom** approach is used
  - ➡ propagates covariance matrix in parallel *(Bugge, Myrheim, 1981, NIM 179, p.365)*
    - for experts: muon reconstruction in ATLAS+CMS uses the STEP track model with continuous energy loss and multiple scattering



- energy loss
  - ➡ use most **probably energy loss** for x/X_0
  - ➡ correct momentum (curvature) and its covariance



- multiple scattering
  - ➡ increases **uncertainty on direction** of track
  - ➡ for given x/X_0 traversed add term to covariances of θ and ϕ on a material "layer"

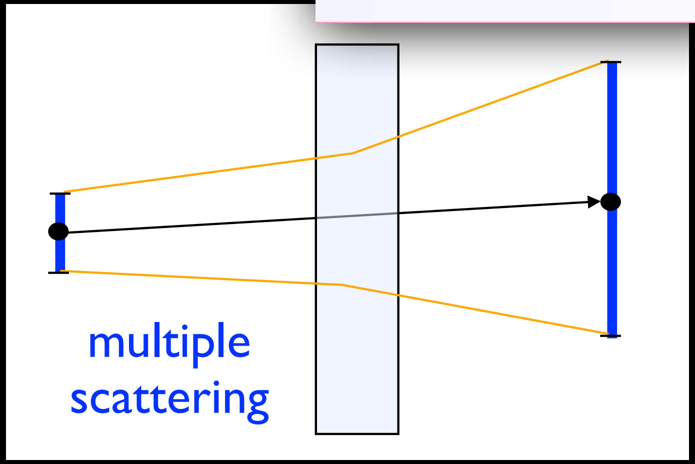


Illustration of Multiple Scattering Effect

- toy simulation
  - ➡ simulation of single particle traversing a set of individual thin material layers
    - single scattering steps accumulate

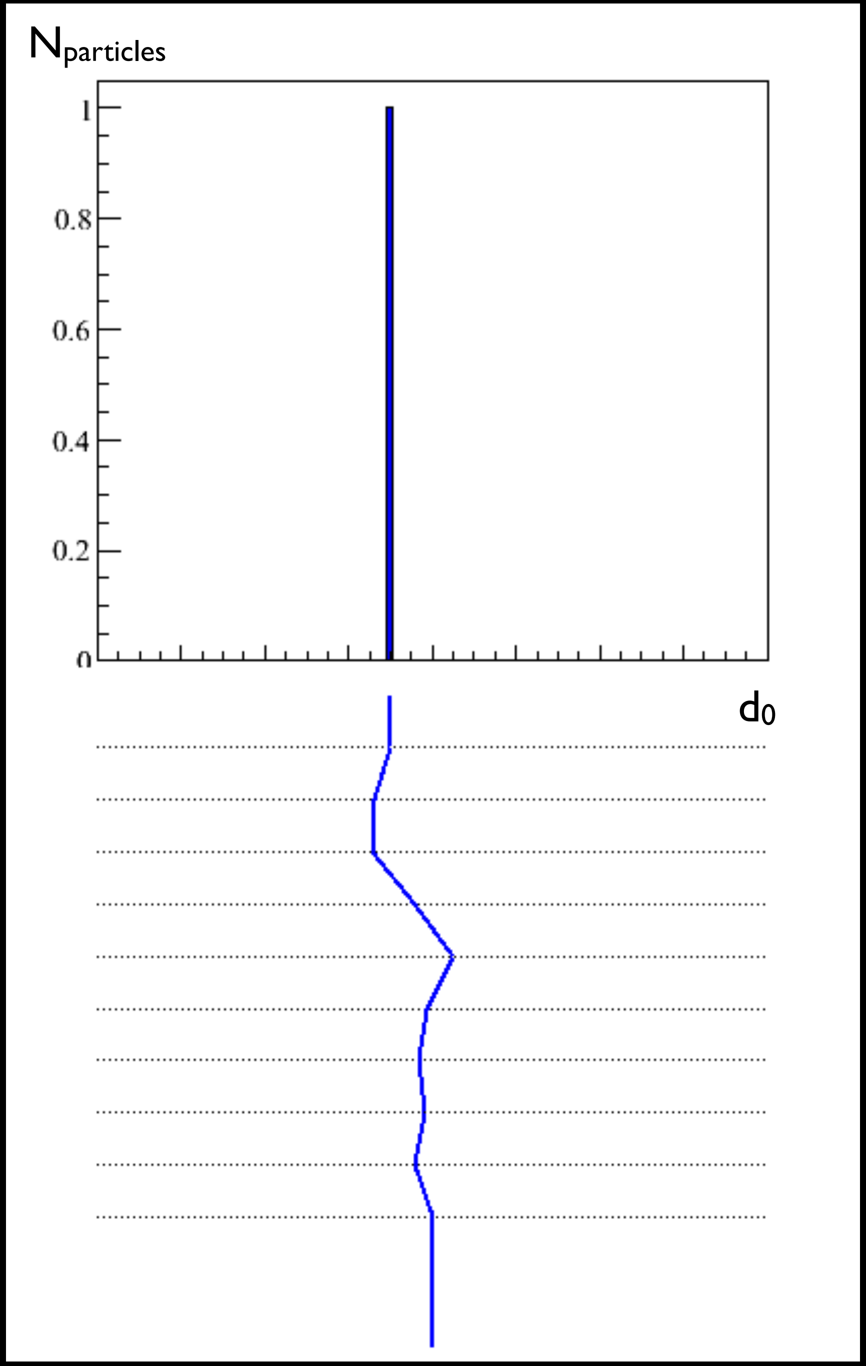


Illustration of Multiple Scattering Effect

- toy simulation
  - ➡ simulation of single particle traversing a set of individual thin material layers
    - single scattering steps accumulate
  - ➡ repeat N times:
    - central limit theorem predicts gaussian distribution

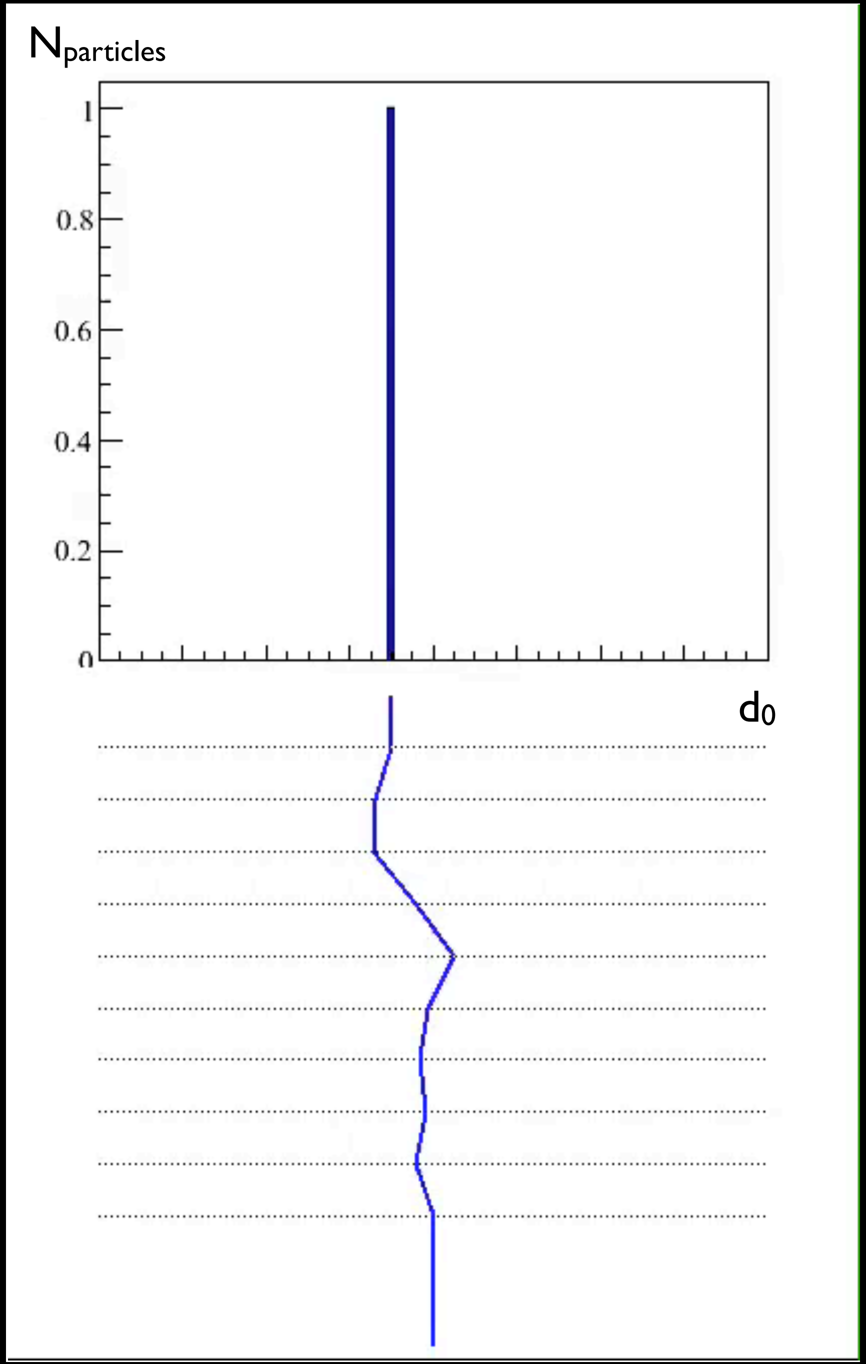
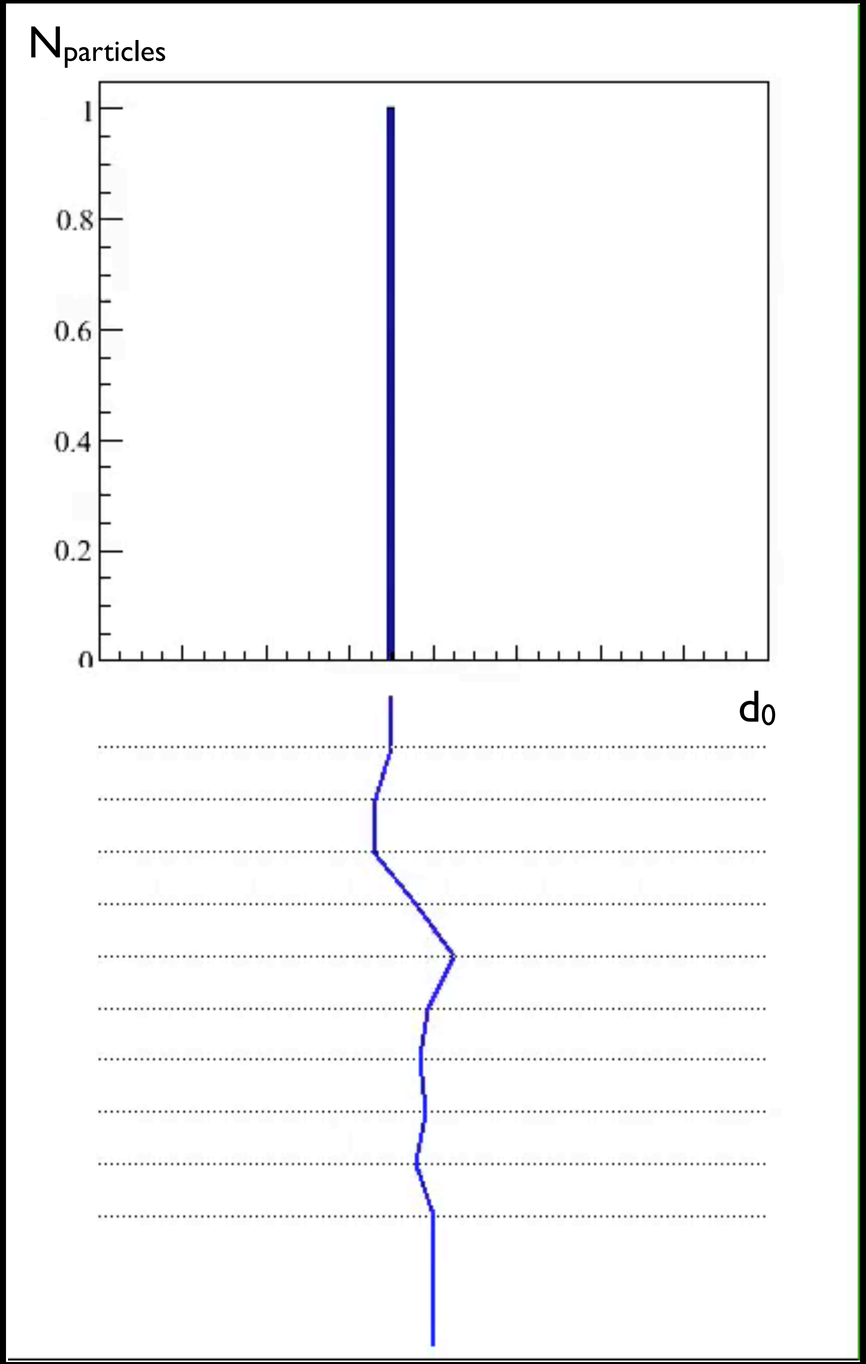


Illustration of Multiple Scattering Effect

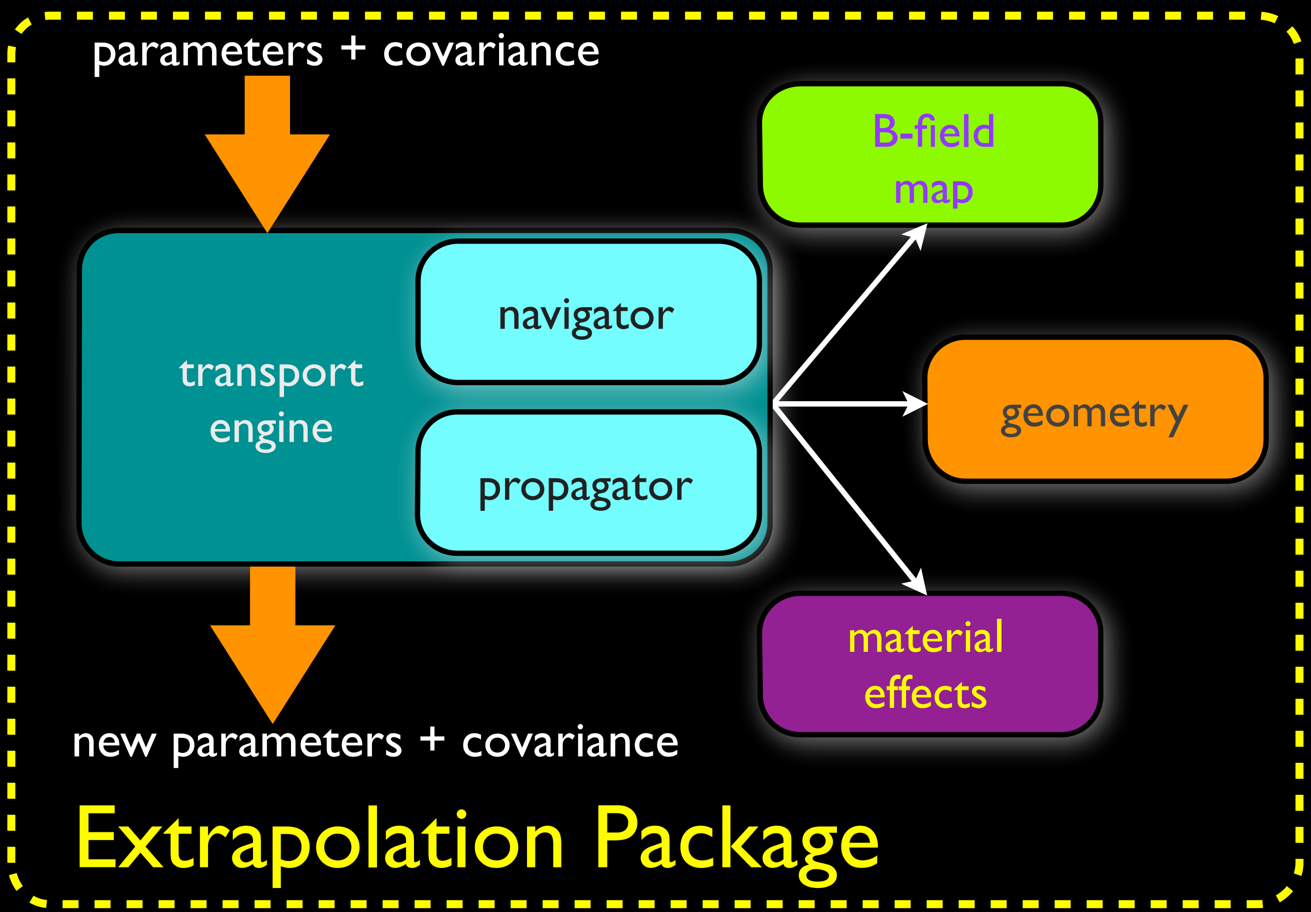
- toy simulation
  - ➡ simulation of single particle traversing a set of individual thin material layers
    - single scattering steps accumulate
  - ➡ repeat N times:
    - central limit theorem predicts gaussian distribution



- sometimes we experience the effect

The Track Extrapolation Package

- a **transport engine** used in tracking software
  - ➡ central tool for pattern recognition, track fitting, etc.
  - ➡ parameter transport from **surface to surface**, including covariance
  - ➡ encapsulates the track model, geometry and material corrections

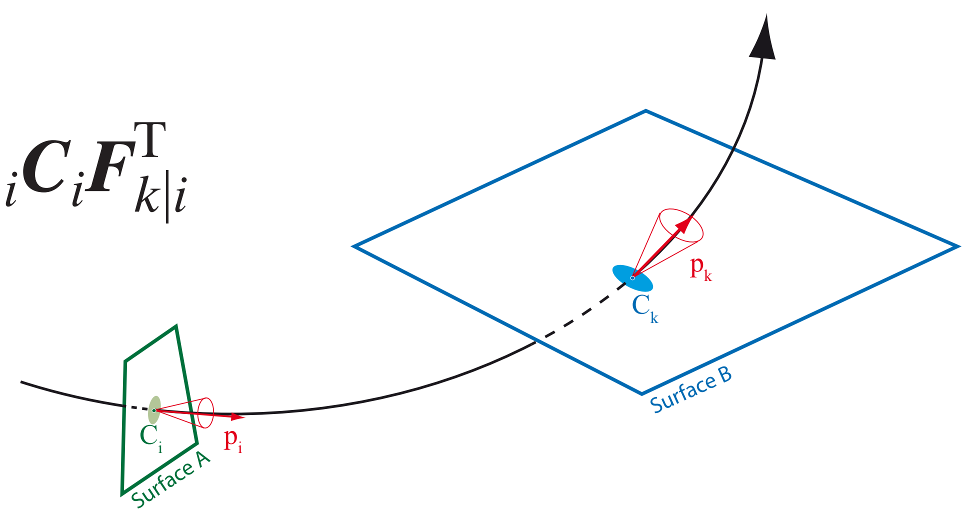


track following in mathematical terms:

$$q_k = f_{k|i}(q_i) \qquad \text{convariance: } C_k = F_{k|i} C_i F_{k|i}^{T}$$

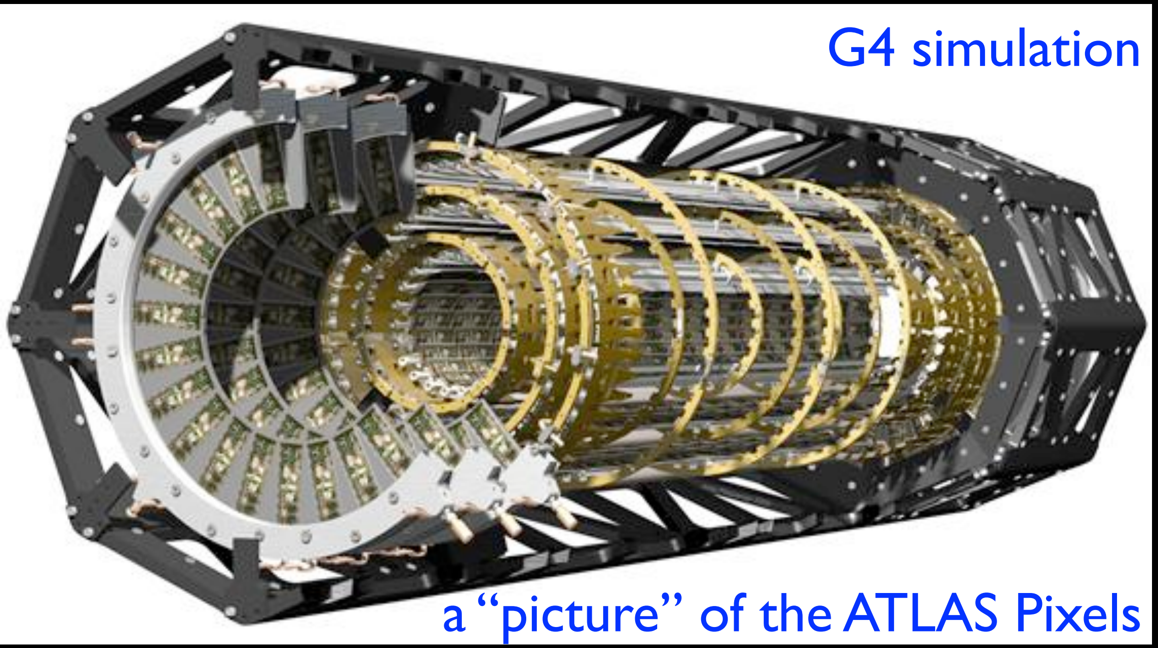
with: $f_{k|i}$ ~ track model

$$F_{k|i} = \frac{\partial q_k}{\partial q_i}$$ ~ Jacobi matrix



Detector Geometry

- interactions in detector material limiting tracking performance
  - ➡ LHC detectors are complex
    - require a very detailed description of their geometry
  - ➡ experiments developed geometry models (translation into G4 simulation)
    - huge number of volumes



- physics requirement to reach LHC goals (e.g. W mass)
  - ➡ control material close to beam pipe at % level

	model	placed volumes
ALICE	Root	4.3 M
ATLAS	GeoModel	4.8 M
CMS	DDD	2.7 M
LHCb	LHCb Det.Des.	18.5 M

CERN

Weighing Detectors during Construction

for completeness

- huge effort in experiments
  - ➡ put each individual detector part on balance and compare with model
    - CMS and ATLAS measured weight of their tracker and all of its components
  - ➡ correct the geometry implementation in simulation and reconstruction

CMS	estimated from measurements	simulation
active Pixels	2598 g	2455 g
full detector	6350 kg	6173 kg

Preliminary

ATLAS	estimated from measurements	simulation
Pixel package	201 kg	197 kg
SCT detector	672 ±15 kg	672 kg
TRT detector	2961 ±14 kg	2962 kg

example: ATLAS TRT measured before and after insertion of the SCT

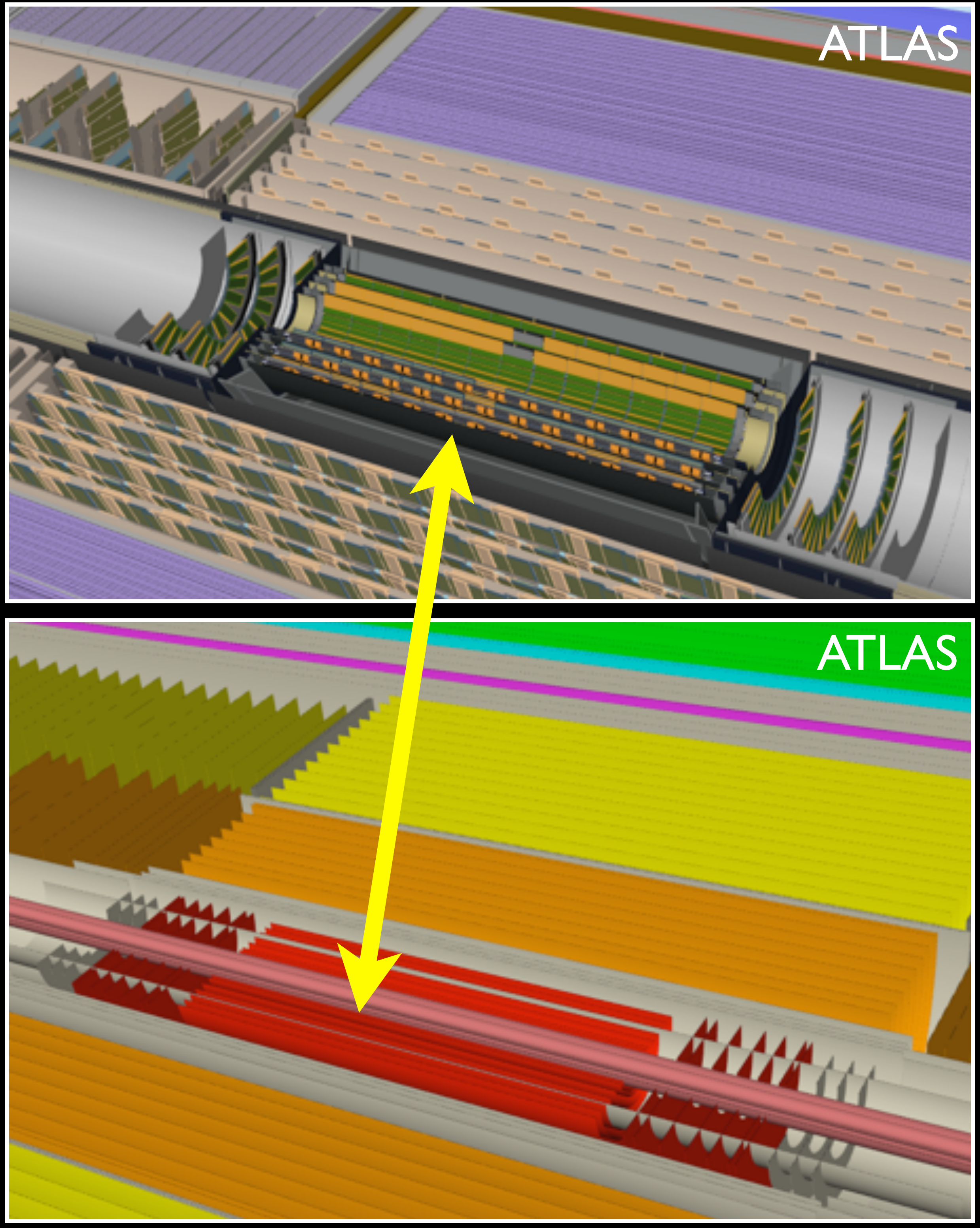
Date	ATLAS $\eta \approx 0$	ATLAS $\eta \approx 1.7$	CMS $\eta \approx 0$	CMS $\eta \approx 1.7$
1994 (Technical Proposals)	0.20	0.70	0.15	0.60
1997 (Technical Design Reports)	0.25	1.50	0.25	0.85
2006 (End of construction)	0.35	1.35	0.35	1.50

Full and Fast (Tracking) Geometries

- complex G4 geometries not optimal for reconstruction
  - ➡ simplified **tracking geometries**
  - ➡ material surfaces, field volumes
- reduced number of volumes
  - ➡ blending details of material onto simple surfaces/volumes
  - ➡ surfaces with 2D material density maps, templates per Si sensor...

	G4	tracking
ALICE	4.3 M	same *1
ATLAS	4.8 M	10.2K *2
CMS	2.7 M	3.8K *2
LHCb	18.5 M	30

*1 ALICE uses full geometry (TGeo)
*2 plus a surface per Si sensor



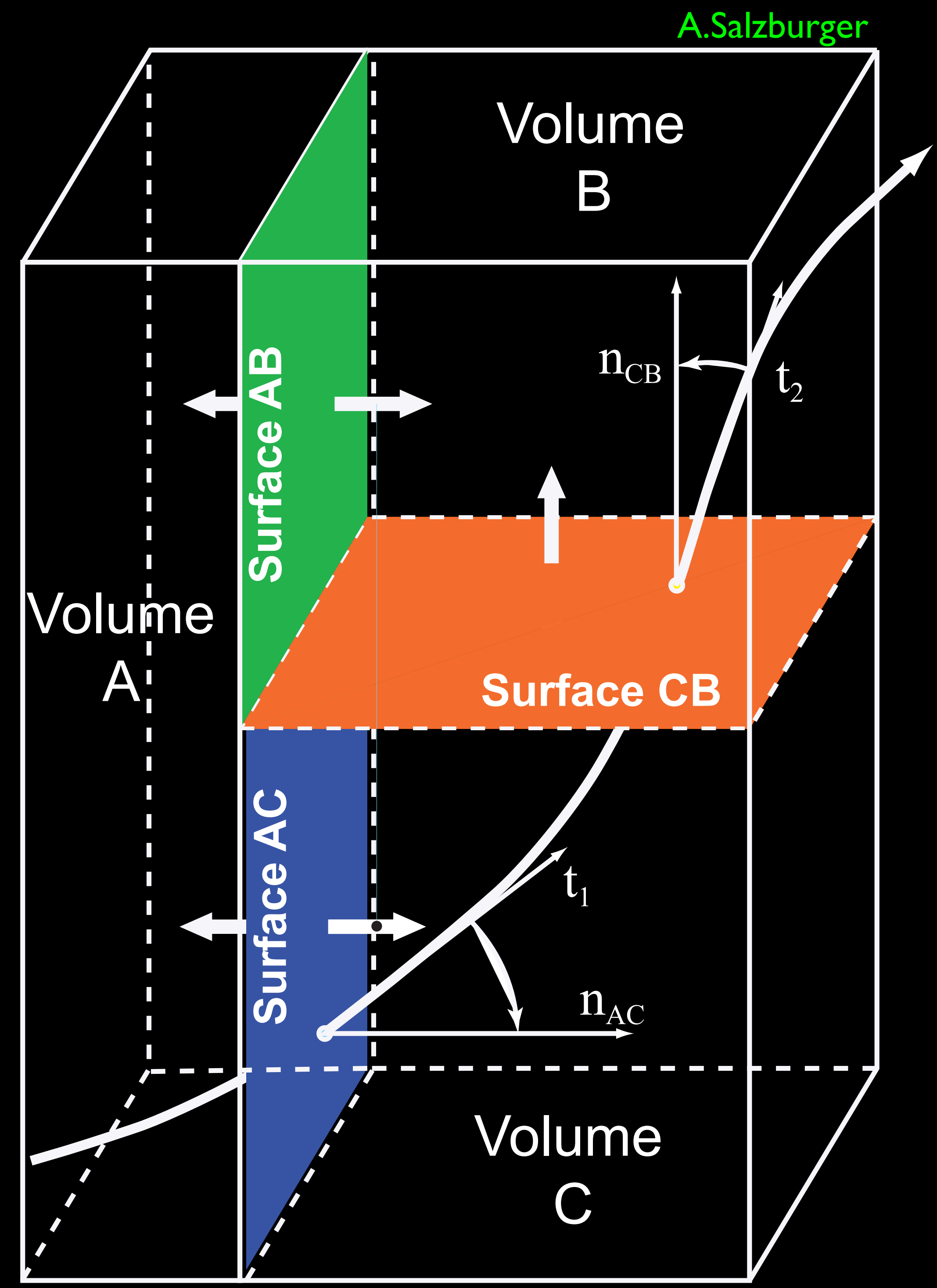
Embedded Navigation Schemes

- **embedded navigation** scheme in tracking geometries
  - ➡ G4 navigation uses voxelisation as generic navigation mechanism
  - ➡ **embedded navigation** for simplified models
    - used in pattern recognition, extrapolation, track fitting and fast simulation

- **example**: ATLAS
  - ➡ developed geometry of connected volumes
  - ➡ boundary surfaces connect neighbouring volumes to predict next step

ATLAS	G4	tracking	ratio
crossed volumes in tracker	474	95	5
time in SI2K sec	19.1	2.3	8.4

(neutral geantinos, no field lookups)



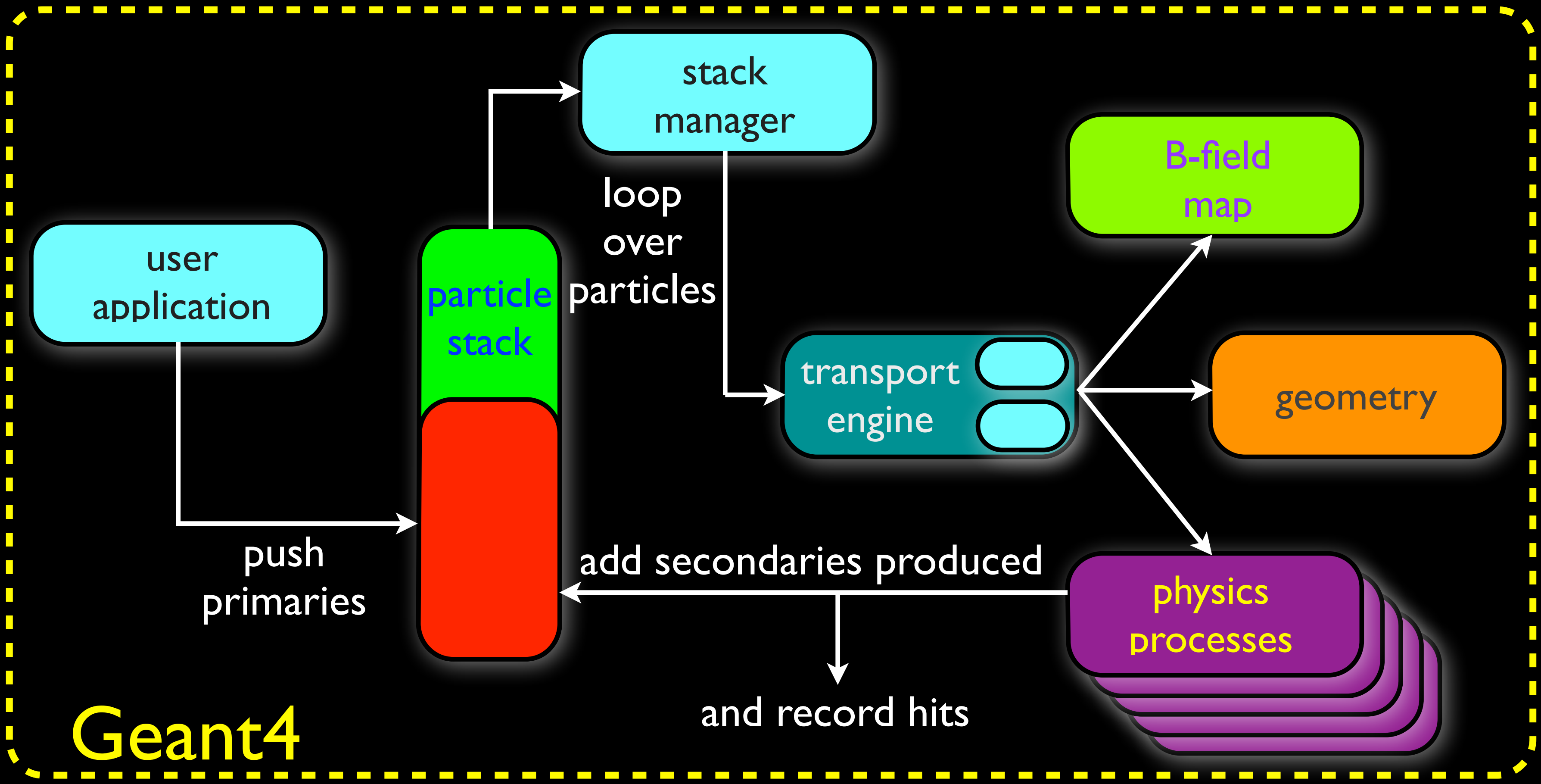
Detour: Simulation (Geant4)

for completeness

see lecture
by John Apostolakis

- Geant4 is based upon
  - ➡ **stack** to keep track of all particles produced and stack manager
  - ➡ **extrapolation system** to propagate each particle:
    - transport engine with navigation
    - geometry model
    - B-field
  - ➡ set of **physics processes** describing interaction of particles with matter
  - ➡ a user application interface, ...

same concept as for
track reconstruction



for completeness

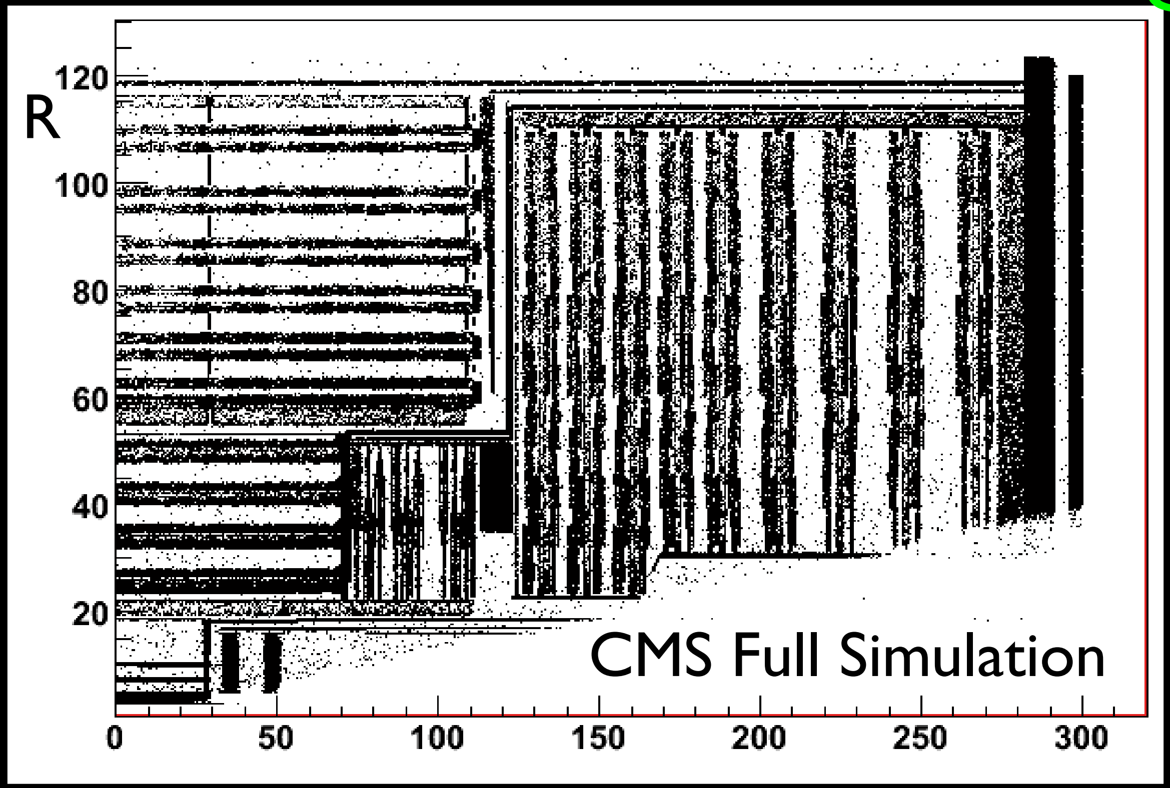
Fast Simulation

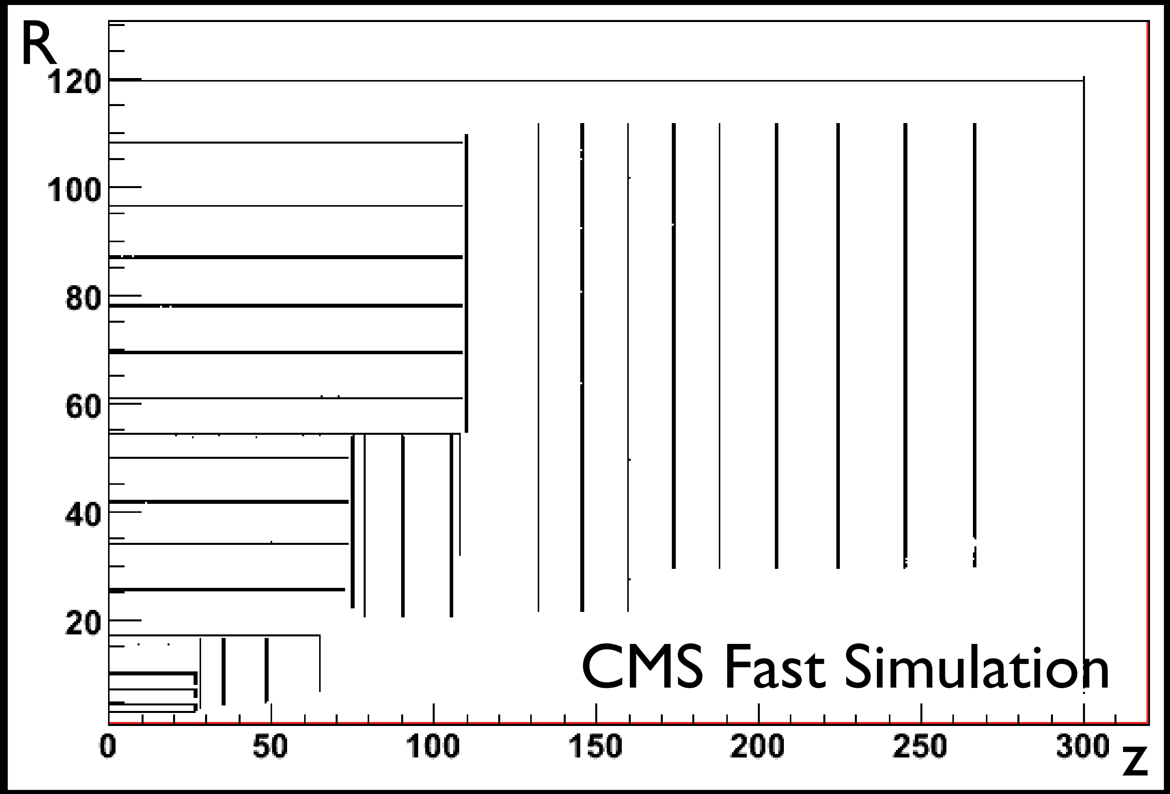
- CPU needs for full G4 exceeds computing models
  - ➡ simulation strategies of experiments mix full G4 and fast simulation

	G4	fast sim.
CMS	360	0.8
ATLAS	1990	7.4

- ttbar events, in kSI2K sec
- G4 differences: calo.modeling , phys.list, η cuts, b-field

- fast simulation engines
  - ➡ fast calo. simulation (parameterisation, showers libraries, ...)
  - ➡ simplified tracking geometries
  - ➡ simplify physics processes w.r.t. G4
  - ➡ output in same data model as full sim.
  - ➡ able to run full reconstruction (+trigger)





Back to Tracking: Track Fitting

Track Fitting

- measurements m_k of a track
  - ➡ in mathematical terms a model:

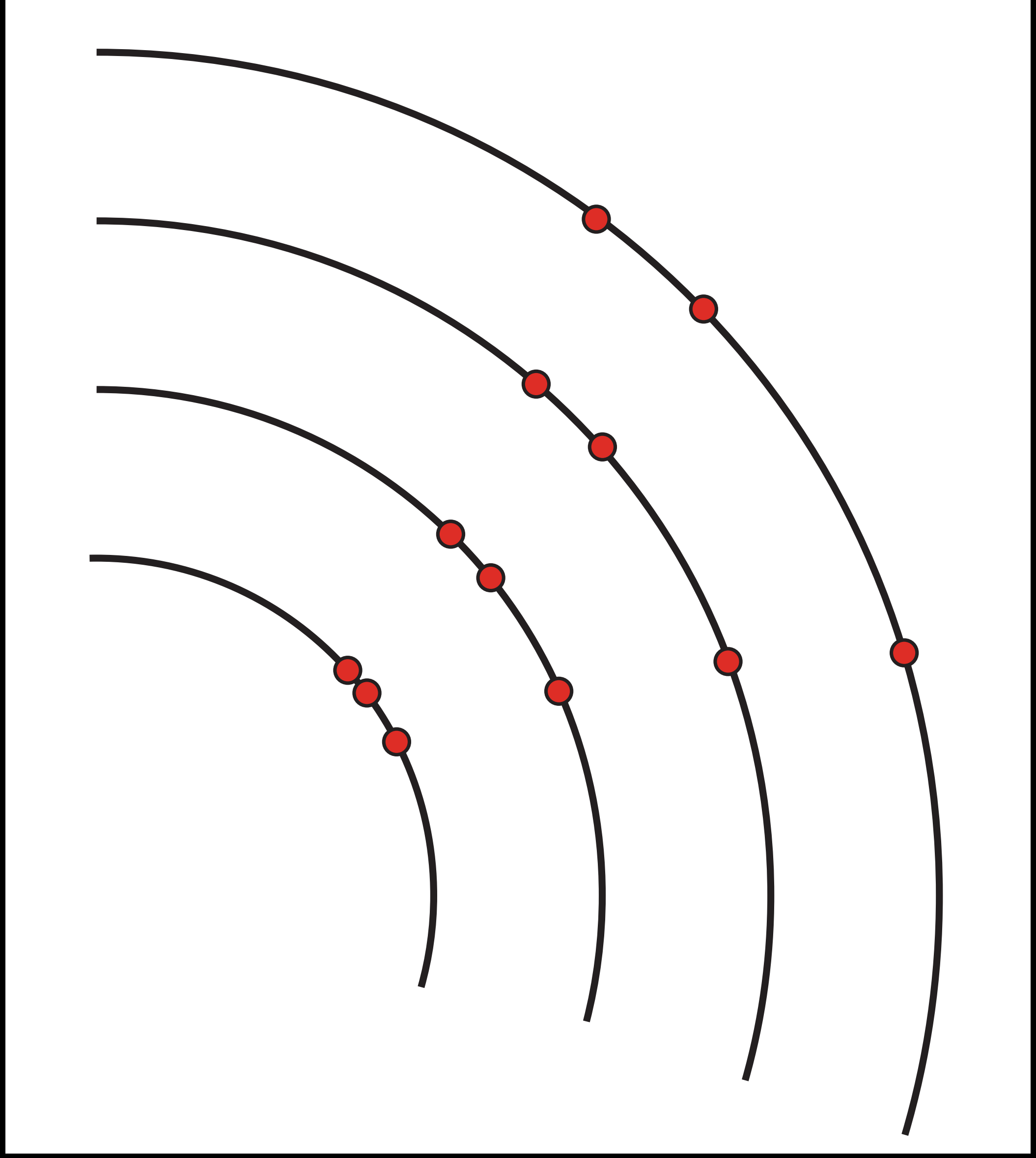
$$\boldsymbol{m}_k = \boldsymbol{h}_k(\boldsymbol{q}_k) + \boldsymbol{\gamma}_k$$

with: $\boldsymbol{h}_k$ ~ functional dependency of measurement on e.g. track angle

$\boldsymbol{\gamma}_k$ ~ error (noise term)

$$\boldsymbol{H}_k = \frac{\partial \boldsymbol{m}_k}{\partial \boldsymbol{q}_k}$$ ~ Jacobian, often contains only rotations and projections

  - ➡ in practice those m_k are clusters, drift circles, ...



Track Fitting

- measurements m_k of a track
  - ➡ in mathematical terms a model:

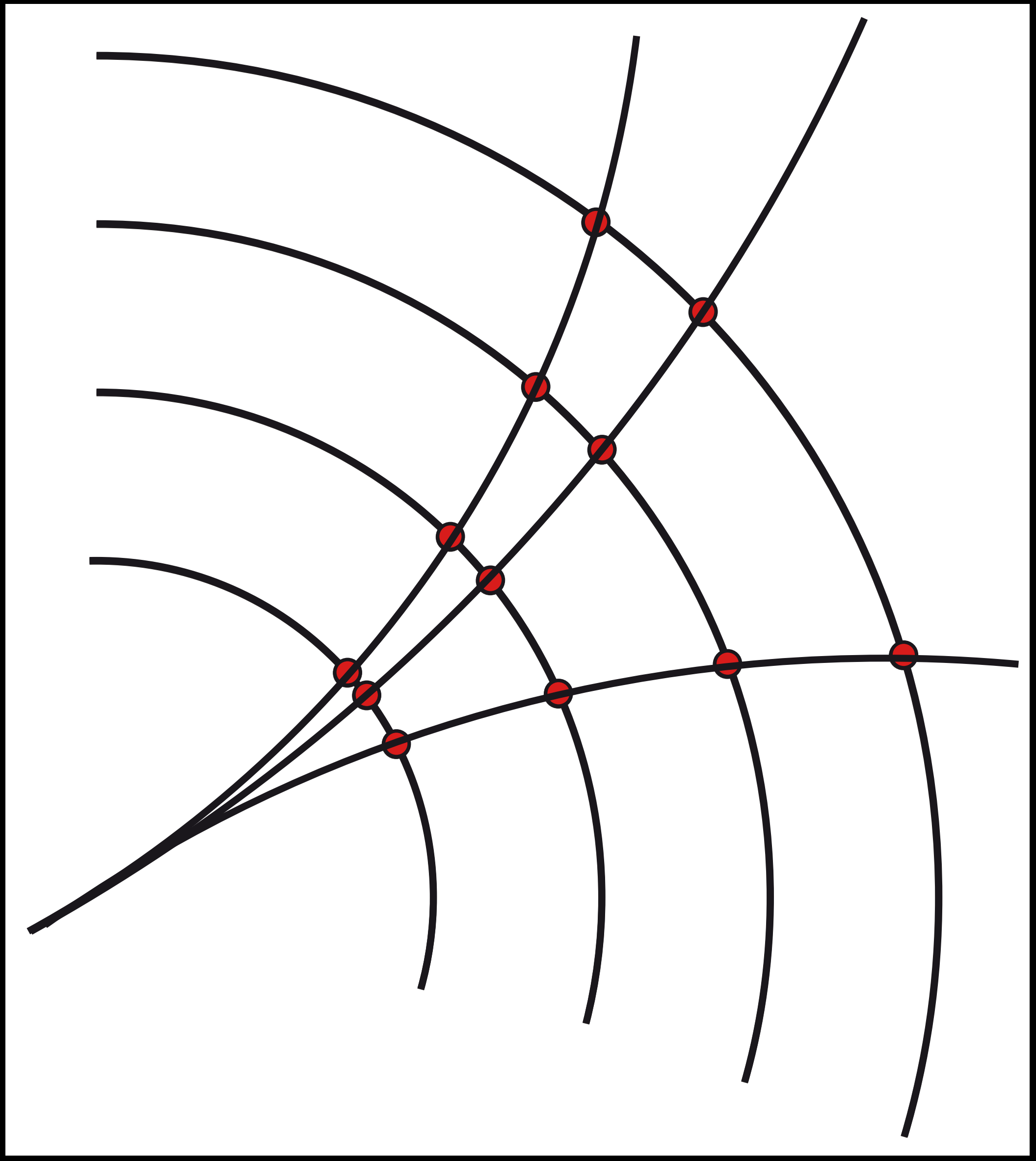
$$\boldsymbol{m}_k = \boldsymbol{h}_k(\boldsymbol{q}_k) + \boldsymbol{\gamma}_k$$

with: $\boldsymbol{h}_k$ ~ functional dependency of measurement on e.g. track angle

$\boldsymbol{\gamma}_k$ ~ error (noise term)

$\boldsymbol{H}_k = \frac{\partial \boldsymbol{m}_k}{\partial \boldsymbol{q}_k}$ ~ Jacobian, often contains only rotations and projections

  - ➡ in practice those m_k are clusters, drift circles, ...



- task of a track fit
  - ➡ estimate the track parameters from a set measurements
- examples for fitting techniques
  - ➡ Least Square track fit or Kalman Filter track fit
  - ➡ more specialised versions: Gaussian Sum Filter or Deterministic Annealing Filters

Classical Least Square Track Fit

- construct and minimise the χ^2 function:

Carl Friedrich Gauss is credited with developing the fundamentals of the basis for least-squares analysis in 1795 at the age of eighteen. **Legendre** was the first to publish the method, however.

$$\chi^2 = \sum_k \Delta m_k^T G_K^{-1} \Delta m_k \quad \text{with:} \quad \Delta m_k = m_k - d_k(p)$$

d_k contains measurement model and propagation of the parameters p :

$$\boldsymbol{d}_k = \boldsymbol{h}_k \circ \boldsymbol{f}_{k|k-1} \circ \cdots \circ \boldsymbol{f}_{2|1} \circ \boldsymbol{f}_{1|0}$$

G_k is the covariance matrix of m_k . Linearise the problem:

$$d_k(p_0 + \delta p) \cong d_k(p_0) + D_k \cdot \delta p \text{ + higher terms}$$

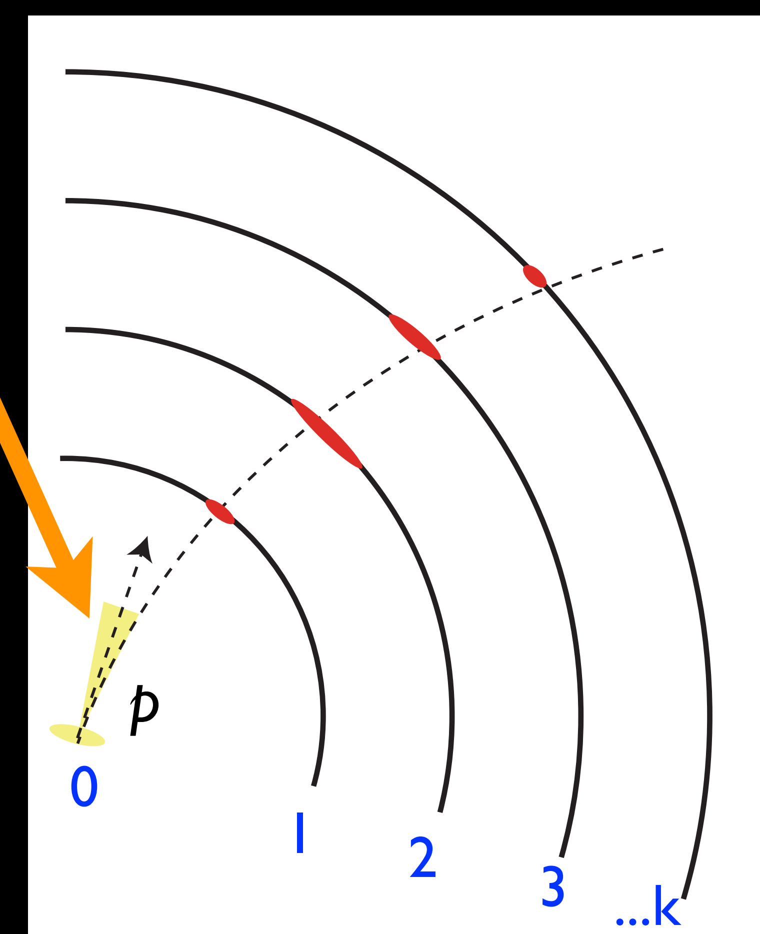
with Jacobian:

$$\boldsymbol{D}_k = \boldsymbol{H}_k \boldsymbol{F}_{k|k-1} \cdots \boldsymbol{F}_{2|1} \boldsymbol{F}_{1|0}$$

minimising the linearised χ^2 yields:

$$\frac{\partial \chi^2}{\partial p} = 0 \Rightarrow \delta p = \left(\sum_k D_k^T G_k^{-1} D_k \right)^{-1} \sum_k D_k^T G_k^{-1} (m_k - d_k(p_0))$$

and covariance of δp is: $C = \left(\sum_k D_k^T G_k^{-1} D_k \right)^{-1}$



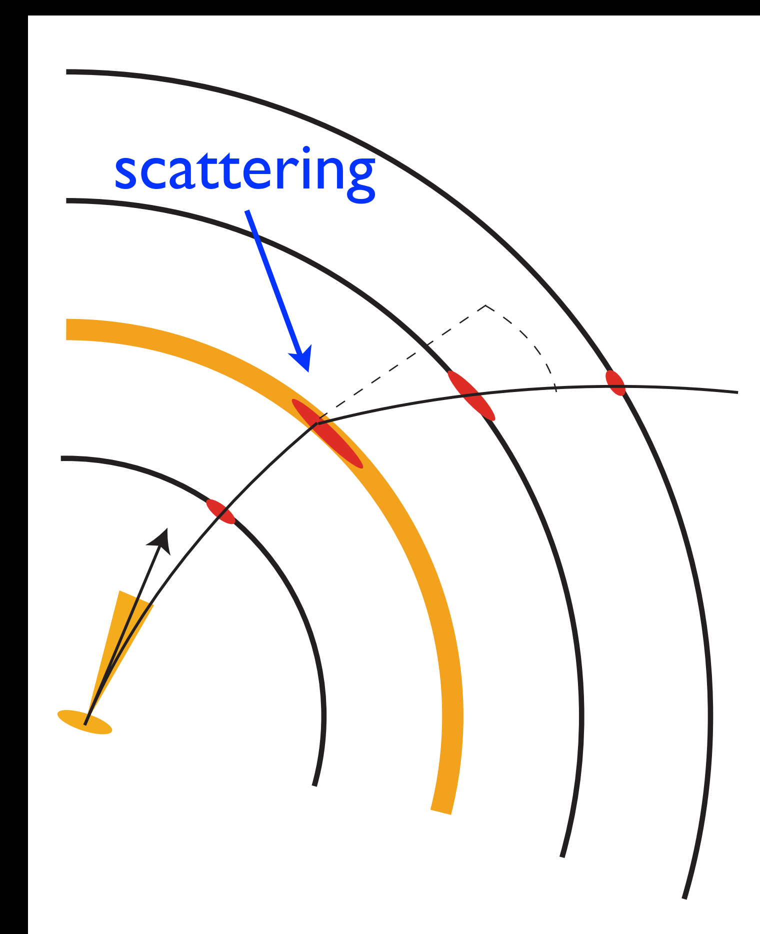
Classical Least Square Track Fit

for completeness

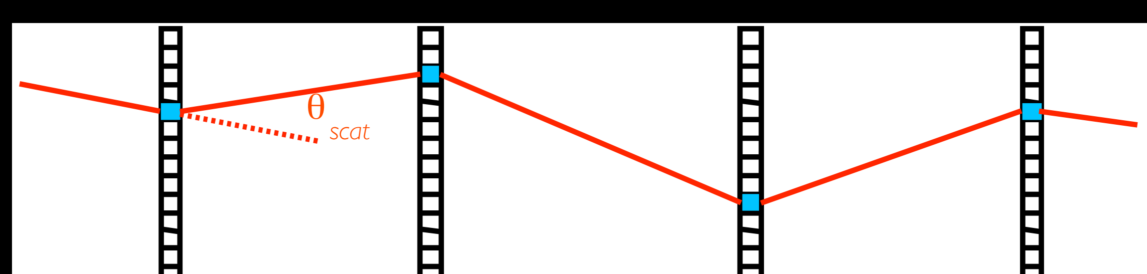
- material effects
  - ➡ can be absorbed in track model $f_{k|i}$, provided effects are small
  - ➡ for substantial multiple scatting, allows for **scattering angles** in the fit
- scattering angles
  - ➡ on each material surface, add 2 angles $\delta\theta_i$ as fee parameters to the fit
  - ➡ expected mean of those angles is 0 (!), their covariance Q_i is given by multiple scattering in x/X_0
- changes to χ^2 formula on previous slide

$$\chi^2 = \sum_k \Delta m_k^T G_K^{-1} \Delta m_k + \sum_i \delta\theta_i^T Q_i^{-1} \delta\theta_i$$

with: $\Delta m_k = m_k - d_k\left(p, \delta\theta_i\right)$



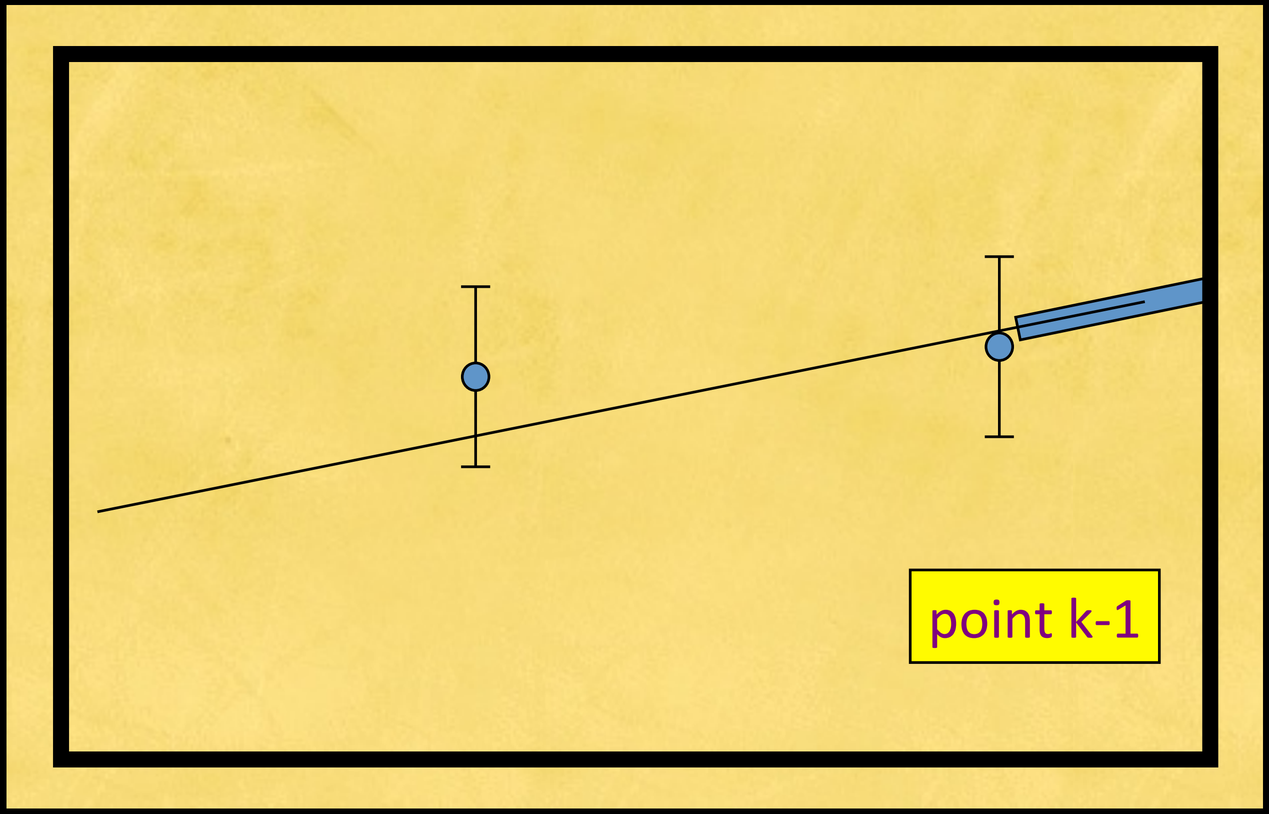
  - ➡ computationally expensive: *need to invert a (5+2*n) matrix*
  - ➡ advantage is that the fitted track precisely follows the particle trajectory: (e.g. for ATLAS muon reconstruction)



The Kalman Filter Track Fit

- a Kalman Filter is a progressive way of performing a least square fit
  ➡ mathematically equivalent

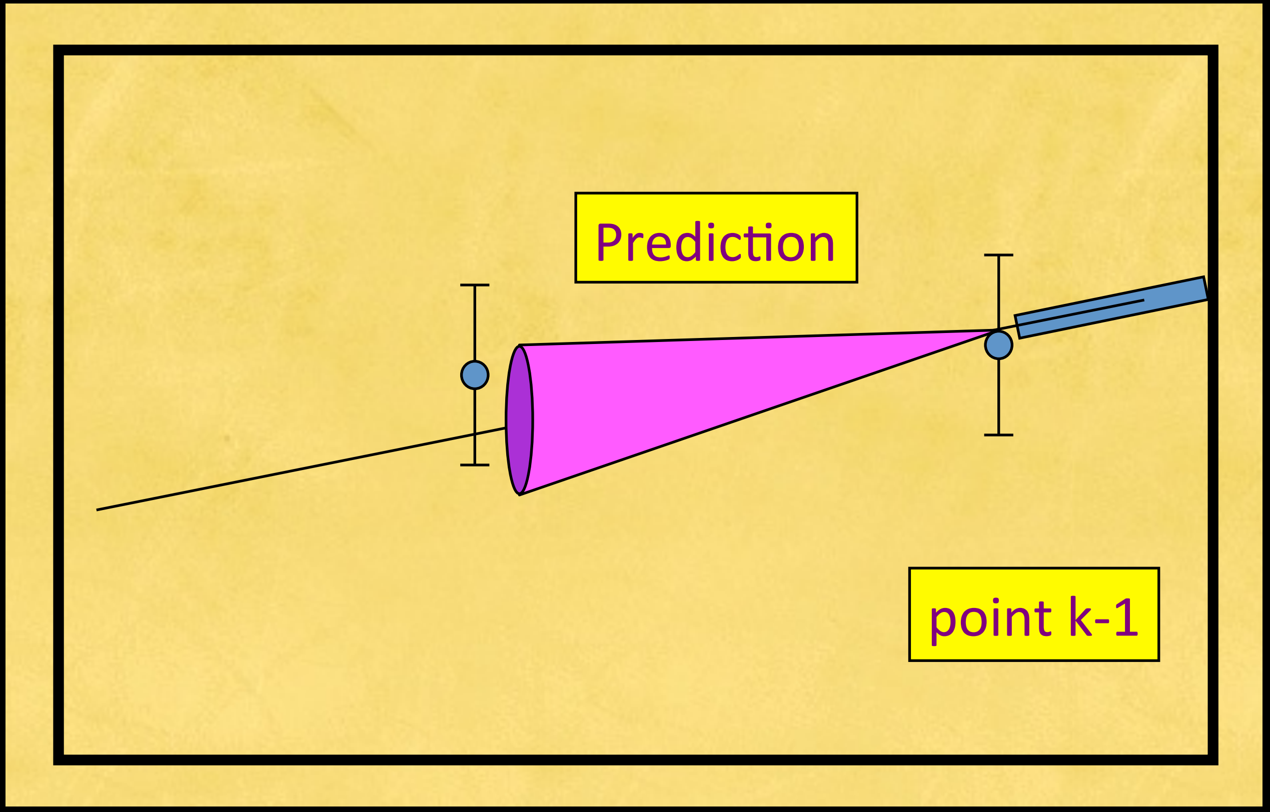
- how does the filter work ?
  1. trajectory parameters at point k-1



The Kalman Filter Track Fit

- a Kalman Filter is a progressive way of performing a least square fit
  - ➡ mathematically equivalent

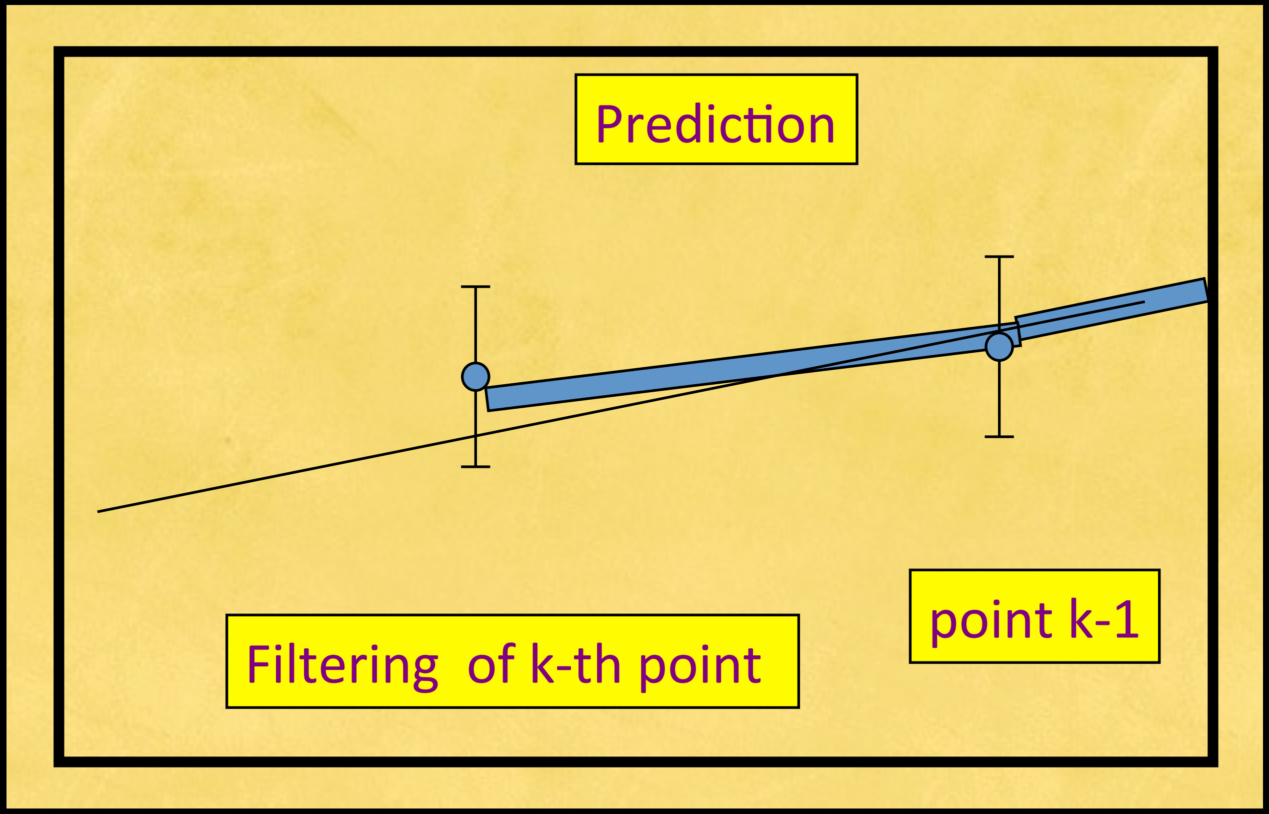
- how does the filter work ?
  1. trajectory parameters at point k-1
  2. propagate to point k to get predicted parameters
     (let's ignore material effects)



The Kalman Filter Track Fit

- a Kalman Filter is a progressive way of performing a least square fit
  ➡ mathematically equivalent

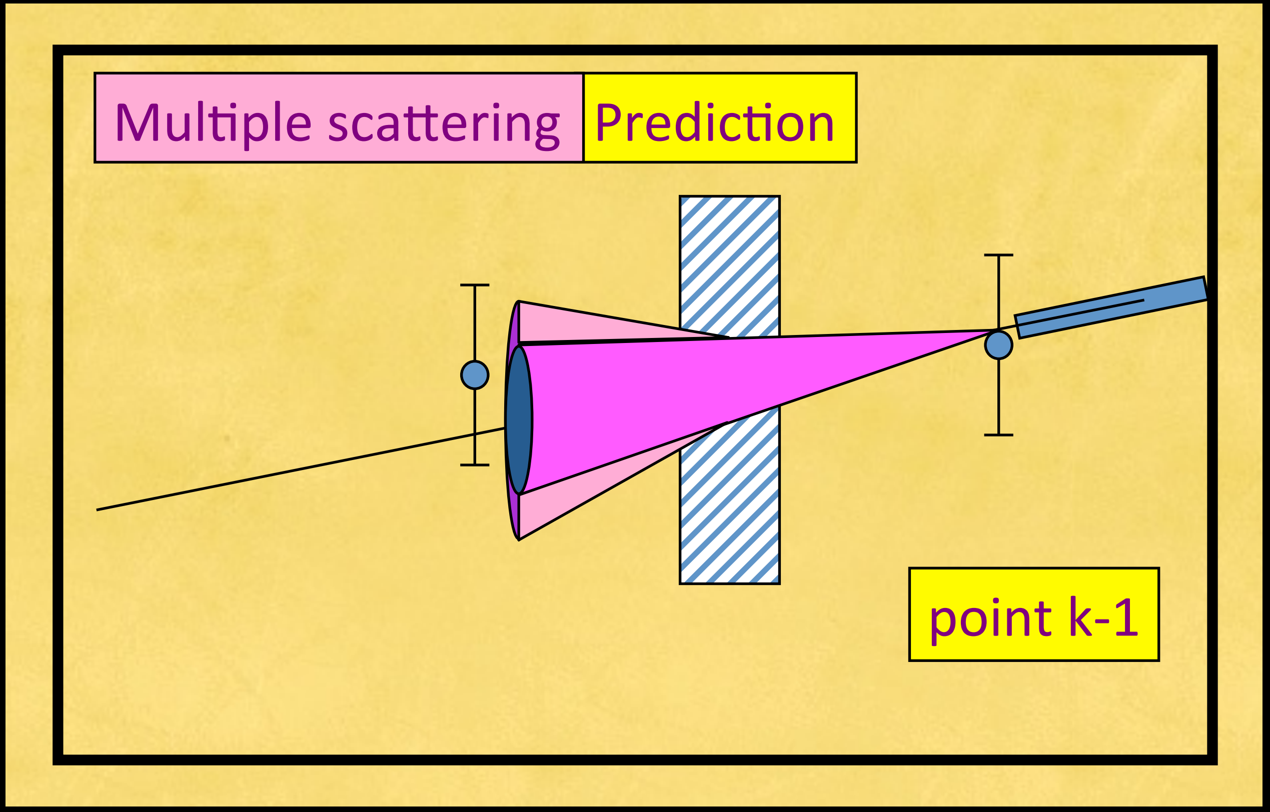
- how does the filter work ?
  1. trajectory parameters at point **k-1**
  2. propagate to point **k** to get predicted parameters
     (let's ignore material effects)
  3. update predicted parameters with measurement **k**
     (simple weighted mean or gain matrix update)
  4. and start over with 1.



The Kalman Filter Track Fit

- a Kalman Filter is a progressive way of performing a least square fit
  - ➡ mathematically equivalent

- how does the filter work ?
  1. trajectory parameters at point **k-1**
  2. propagate to point **k** to get predicted parameters
     (let's ignore material effects)
  3. update predicted parameters with measurement **k**
     (simple weighted mean or gain matrix update)
  4. and start over with 1.

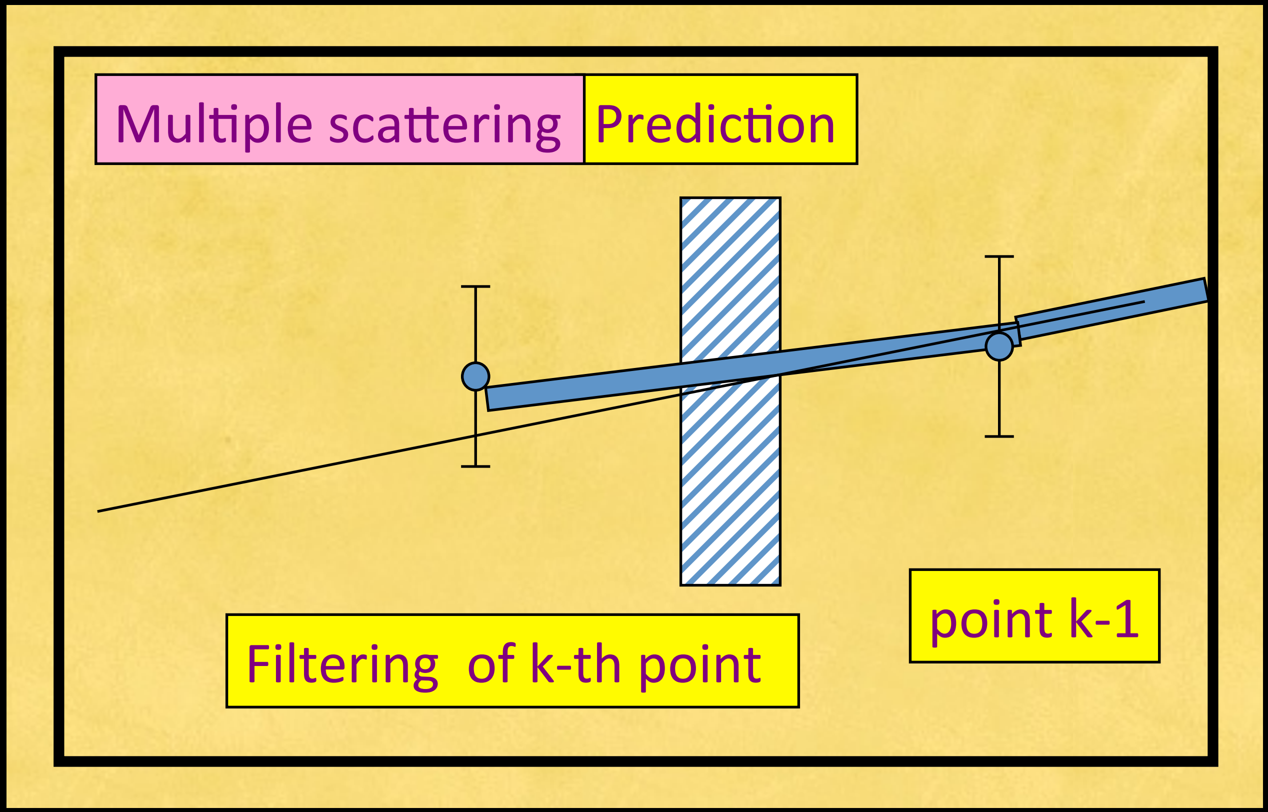


- material effects (multiple scattering and energy loss)
  - ➡ incorporated in the propagated parameters (prediction)

The Kalman Filter Track Fit

- a Kalman Filter is a progressive way of performing a least square fit
  ➡ mathematically equivalent

- how does the filter work ?
  1. trajectory parameters at point **k-1**
  2. propagate to point **k** to get predicted parameters
     (let's ignore material effects)
  3. update predicted parameters with measurement **k**
     (simple weighted mean or gain matrix update)
  4. and start over with 1.



- material effects (multiple scattering and energy loss)
  ➡ incorporated in the propagated parameters (prediction)
  ➡ and therefore enters into the updated parameters at point **k**

The Kalman Filter Track Fit

for completeness

- in mathematical terms:

1. propagate p_{k-1} and its covariance C_{k-1} :

$$q_{k|k-1} = f_{k|k-1}(q_{k-1|k-1})$$
$$C_{k|k-1} = F_{k|k-1} C_{k-1|k-1} F^{T}_{k|k-1} + Q_k$$

with Q_k ~ noise term (M.S.)

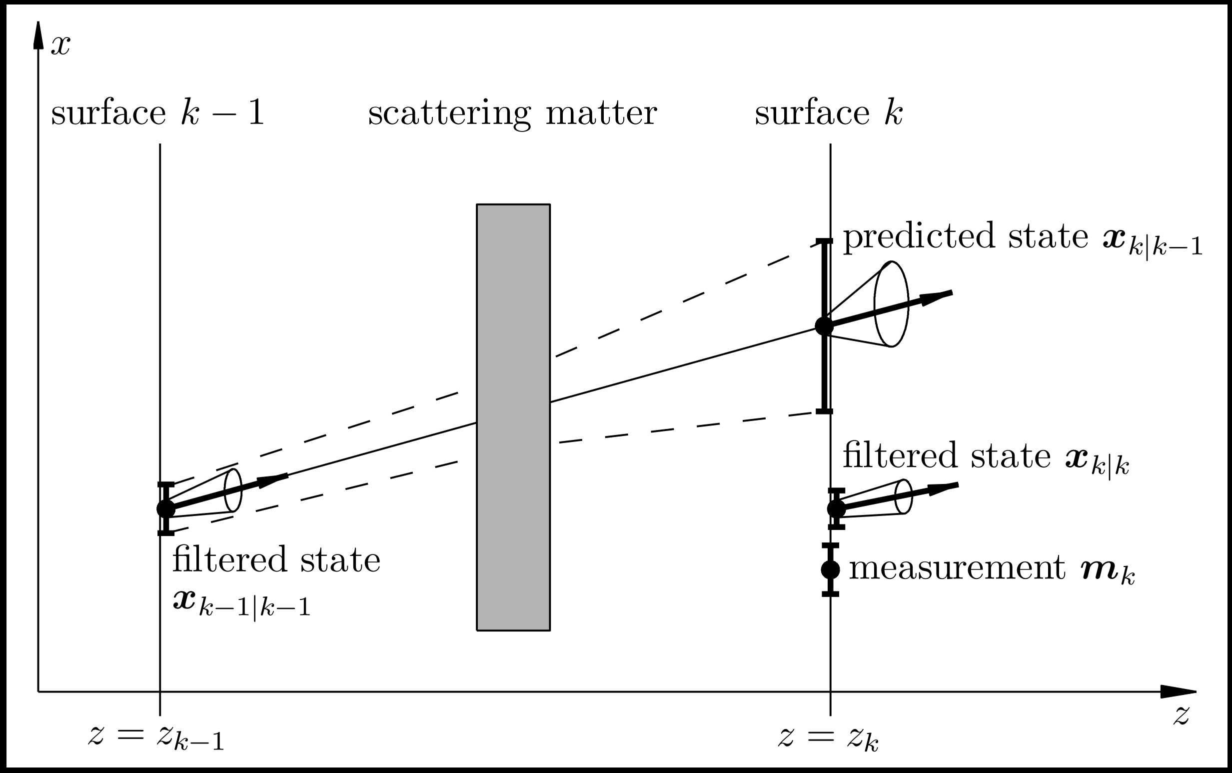
2. update prediction to get $q_{k|k}$ and $C_{k|k}$:

$$q_{k|k} = q_{k|k-1} + K_k[m_k - h_k(q_{k|k-1})]$$
$$C_{k|k} = (I - K_k H_k) C_{k|k-1}$$

with K_k ~ gain matrix :

$$K_k = C_{k|k-1} H^T_k (G_k + H_k C_{k|k-1} H^T_k)^{-1}$$

➡ alternative to gain matrix approach is a weighted mean to obtian $p_{k|k}$
- but requires to invert 5x5 matrix instead of a matrix of *rank(G_k)*



- Kalman Smoother:

➡ provides full information along track

proceeds from layer $k+1$ to layer k :

$$q_{k|n} = q_{k|k} + A_k(q_{k+1|n} - q_{k+1|k})$$
$$C_{k|n} = C_{k|k} - A_k(C_{k+1|k} - C_{k+1|n})A^T_k$$

with A_k ~ smoother gain matrix :

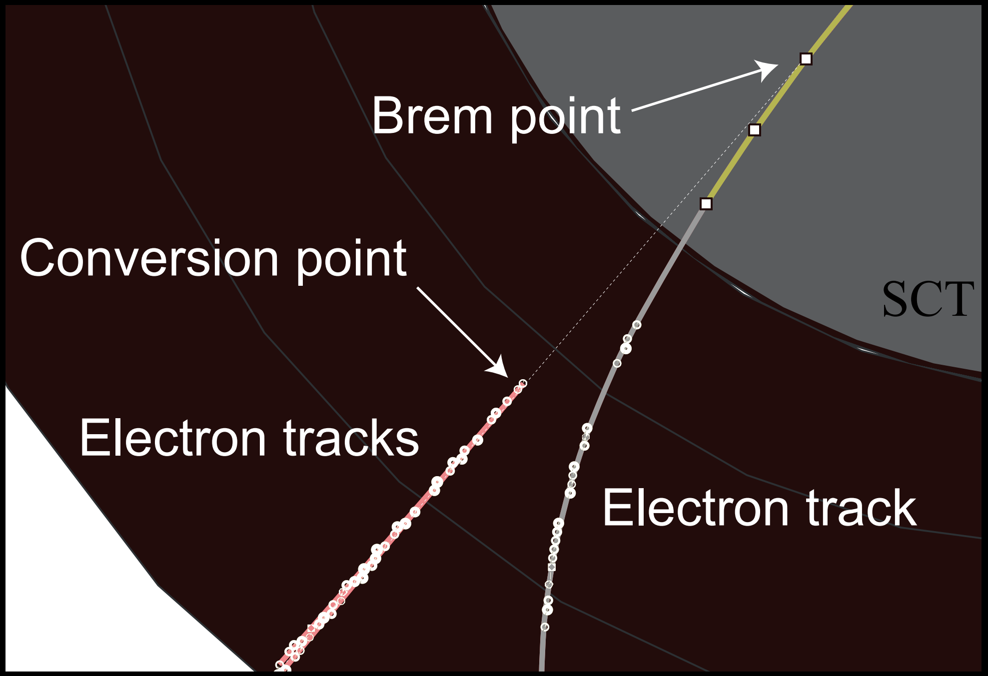
$$A_k = C_{k|k} F^T_{k+1|k} (C_{k+1|k})^{-1}$$

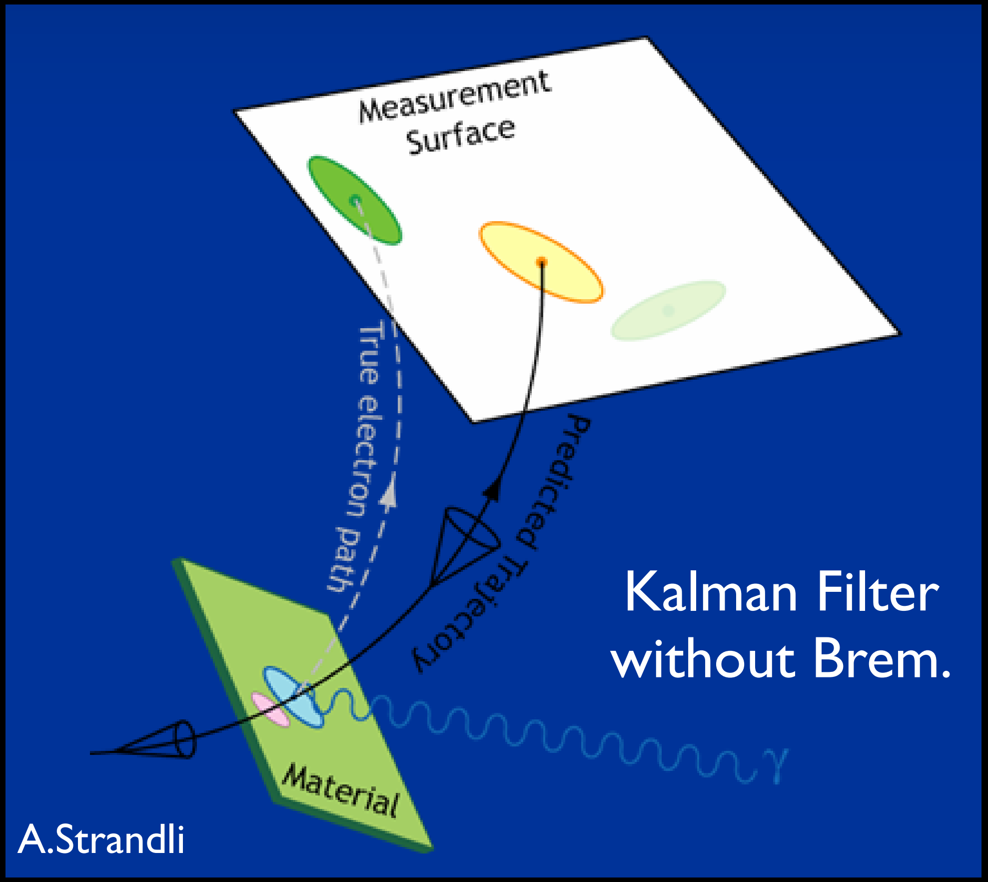
➡ equivalent: combine forw./back. filter

Brem. Fitting for Electrons

advanced techniques

- material in tracker
  - ➡ e-bremsstrahlung and γ-conversions
- electron efficiency limited
  - ➡ momentum loss due to bremsstrahlung leads to large changes in track curvature
  - ➡ fit is biased towards small momenta or fails completely
- techniques to allow for bremsstrahlung in track fitting
  - ➡ brem. point in Least Square track fit
  - ➡ Kalman Filter with dynamic noise adjustment
  - ➡ Gaussian Sum Filter

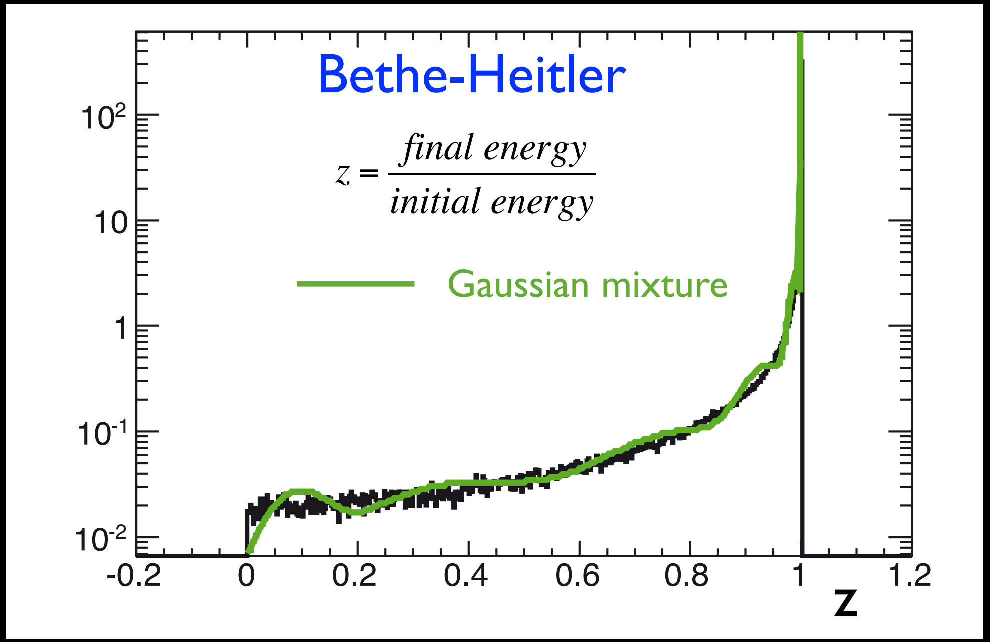


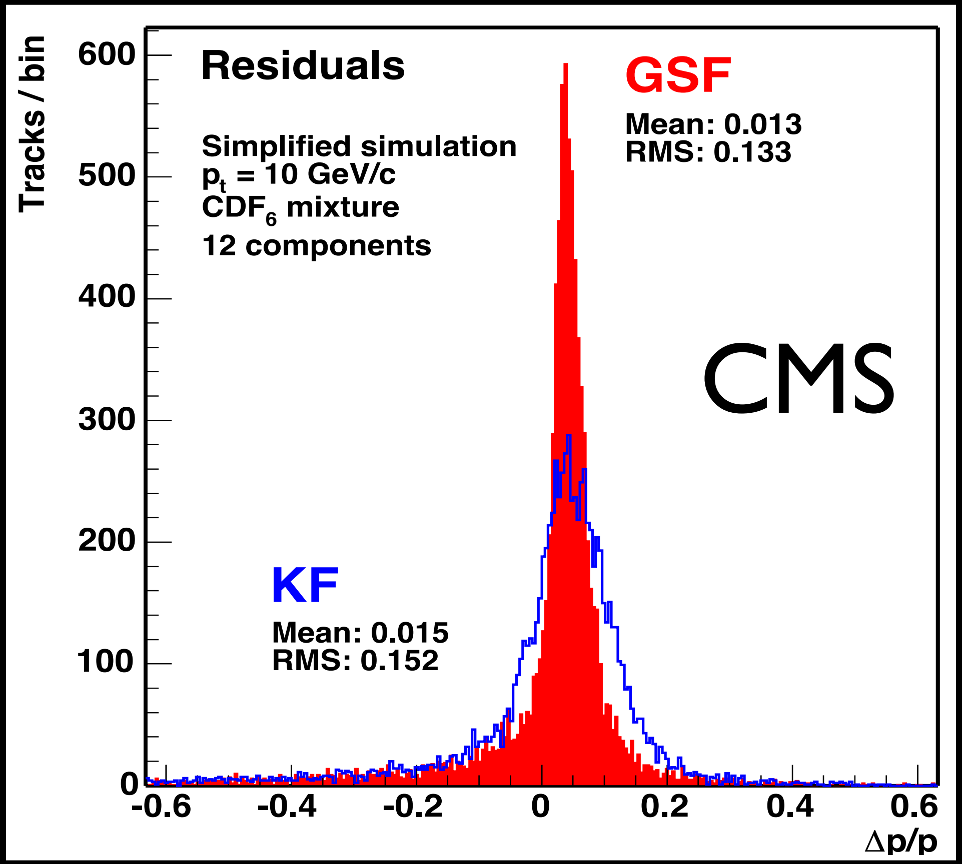


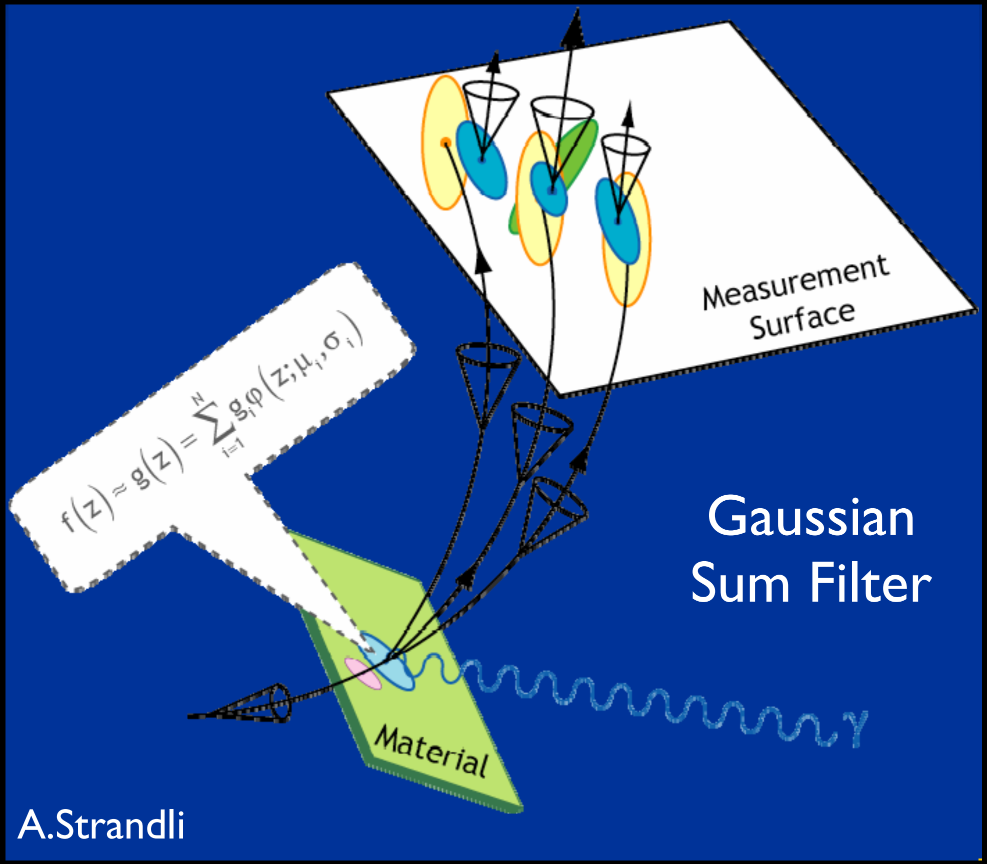
Gaussian Sum Filter

advanced techniques

- ➡ approximate Bethe-Heitler distribution as Gaussian mixture
  - state vector after material correction becomes sum of Gaussian components
- ➡ GSF resembles set of parallel Kalman Filters for N components
  - computationally expensive !
  - default electron fitter in CMS and ATLAS

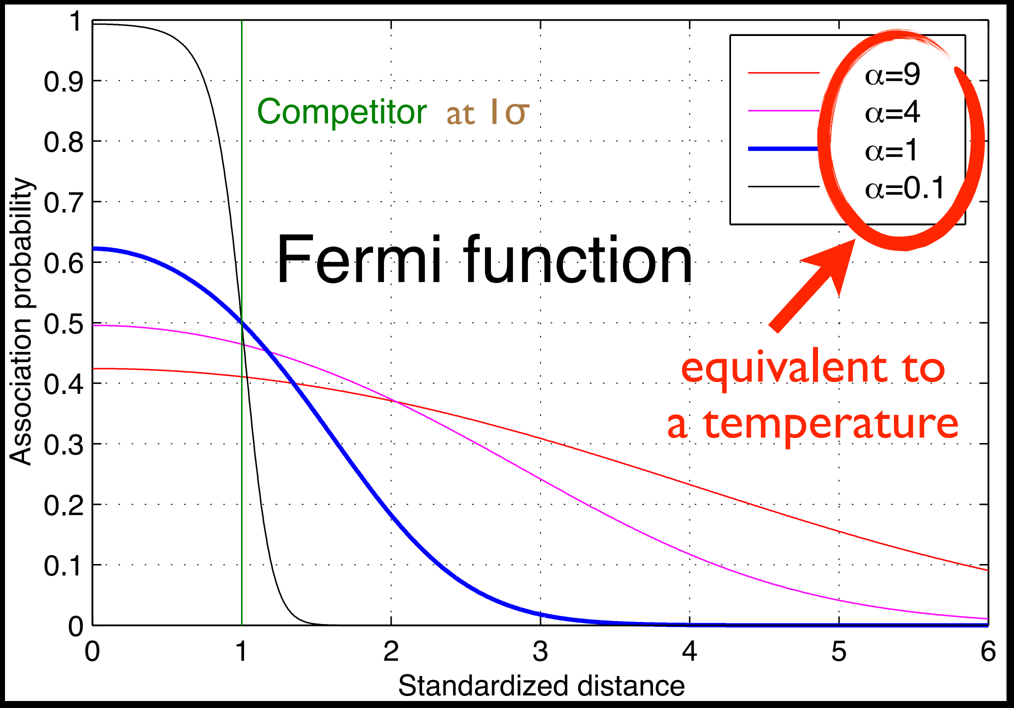






Deterministic Annealing Filters

- **robust technique**
  - ➡ developed for fitting with high occupancies
    - e.g. ATLAS TRT with high event pileup
    - reconstruction of 3-prong τ decays
  - ➡ can deal with several close by hits on a layer



- **adaptive fit**
  - ➡ multiply weight of each hit in layer with assignment probability:

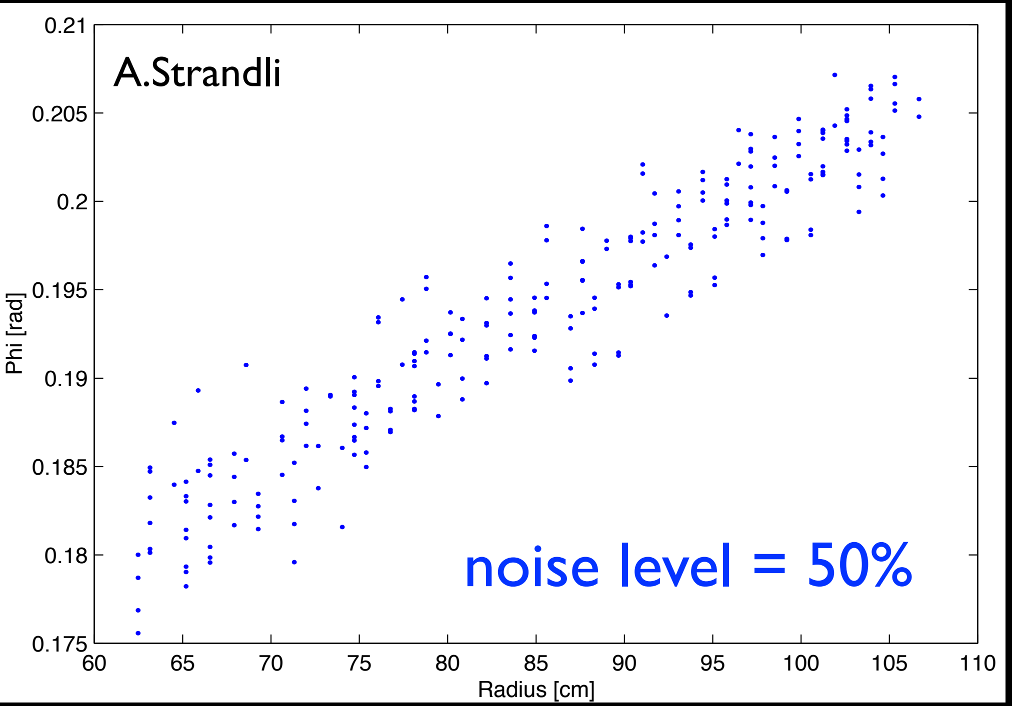
$$p_{ik} = \frac{\exp\left(-\hat{d}_{ik}^2/T\right)}{\sum_{j=1}^{n_k} \exp\left(-\hat{d}_{jk}^2/T\right)}$$

Boltzman factor

with: $\hat{d_{ik}} = d_{ik}/\sigma_k$

normalized distance

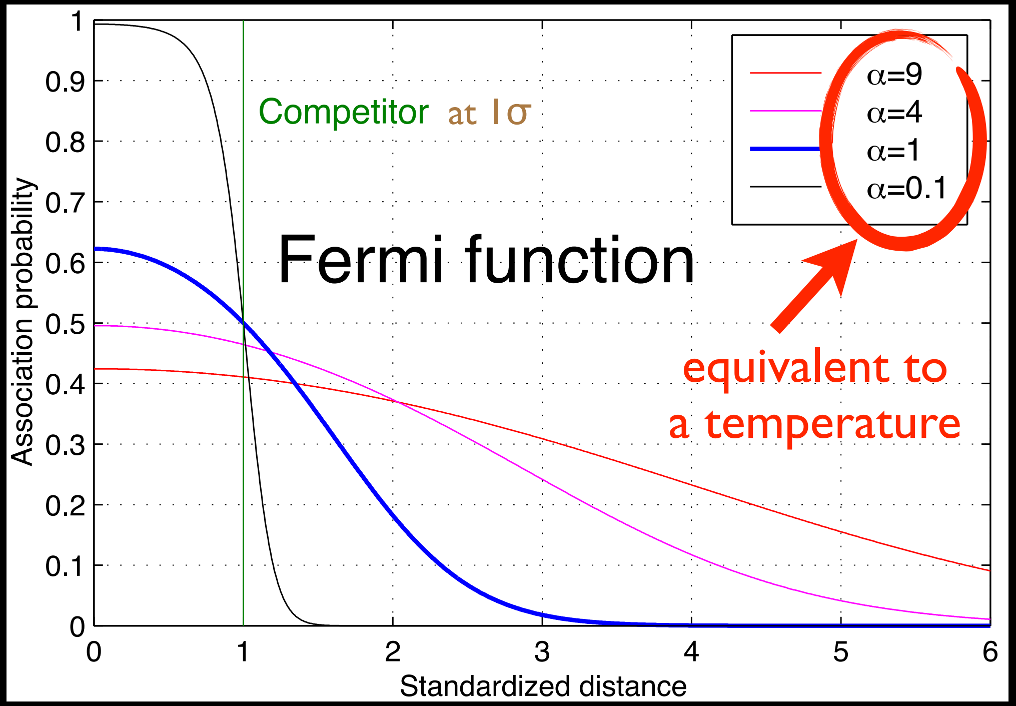
  - ➡ process decreasing temperature T is called annealing (iterative)
    - start at high T ~ all hits contribute same
    - at low T ~ close by hits remain
  - ➡ can be written as a Multi Track Filter



advanced techniques

Deterministic Annealing Filters

- robust technique
  - ➡ developed for fitting with high occupancies
    - e.g. ATLAS TRT with high event pileup
    - reconstruction of 3-prong τ decays
  - ➡ can deal with several close by hits on a layer



- adaptive fit
  - ➡ multiply weight of each hit in layer with assignment probability:

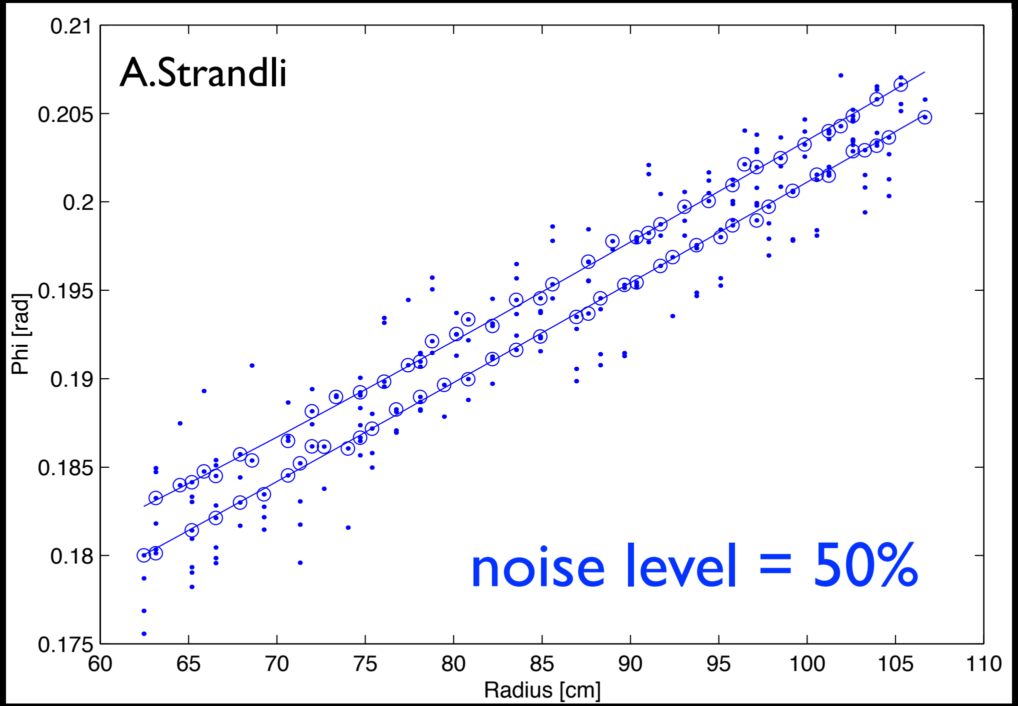
$$p_{ik} = \frac{\exp\left(-\hat{d}_{ik}^2/T\right)}{\sum_{j=1}^{n_k}\exp\left(-\hat{d}_{jk}^2/T\right)}$$

with: $\hat{d_{ik}} = d_{ik}/\sigma_k$

Boltzman factor

normalized distance

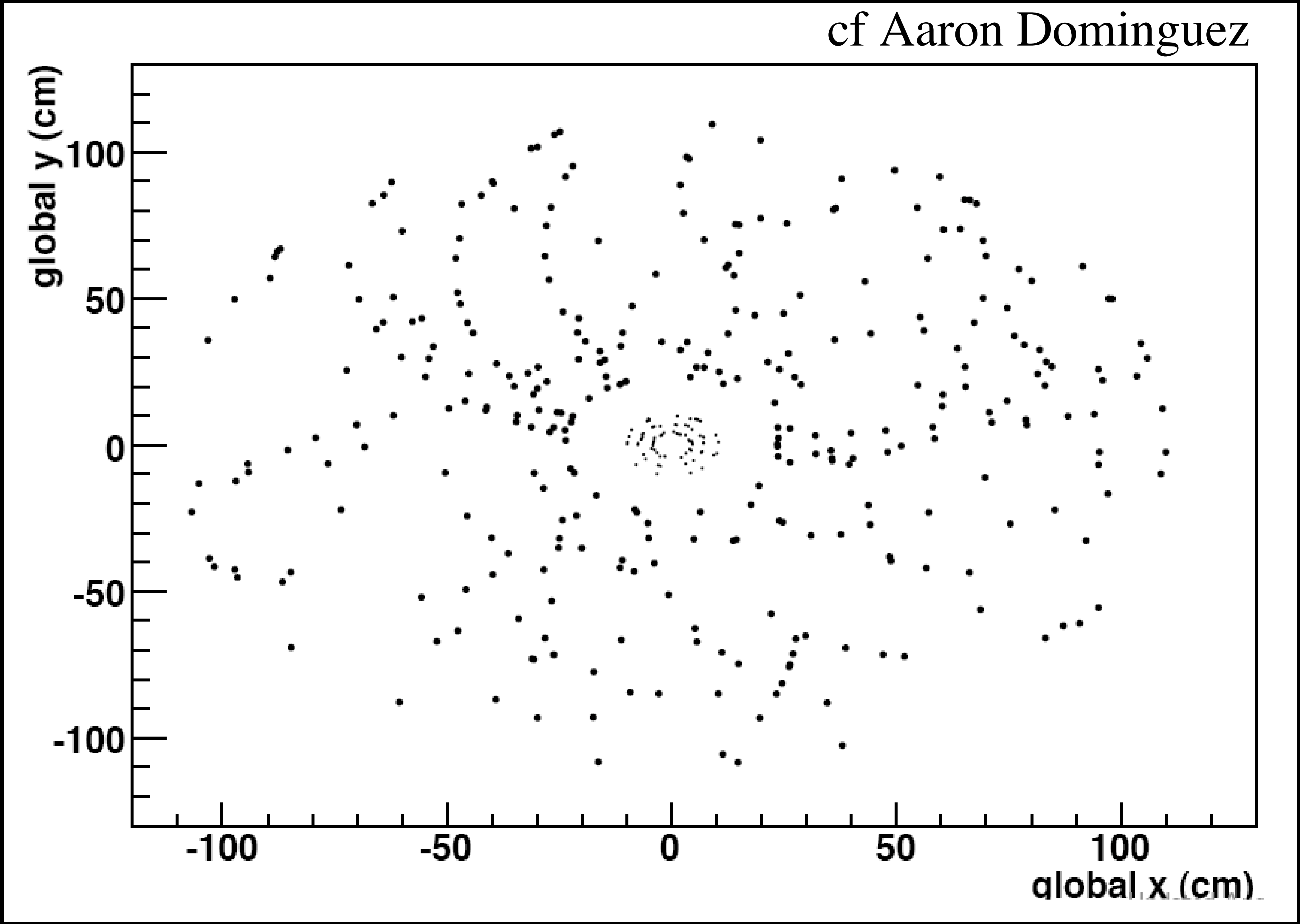
  - ➡ process decreasing temperature T is called annealing (iterative)
    - start at high T ~ all hits contribute same
    - at low T ~ close by hits remain
  - ➡ can be written as a Multi Track Filter



Track Finding

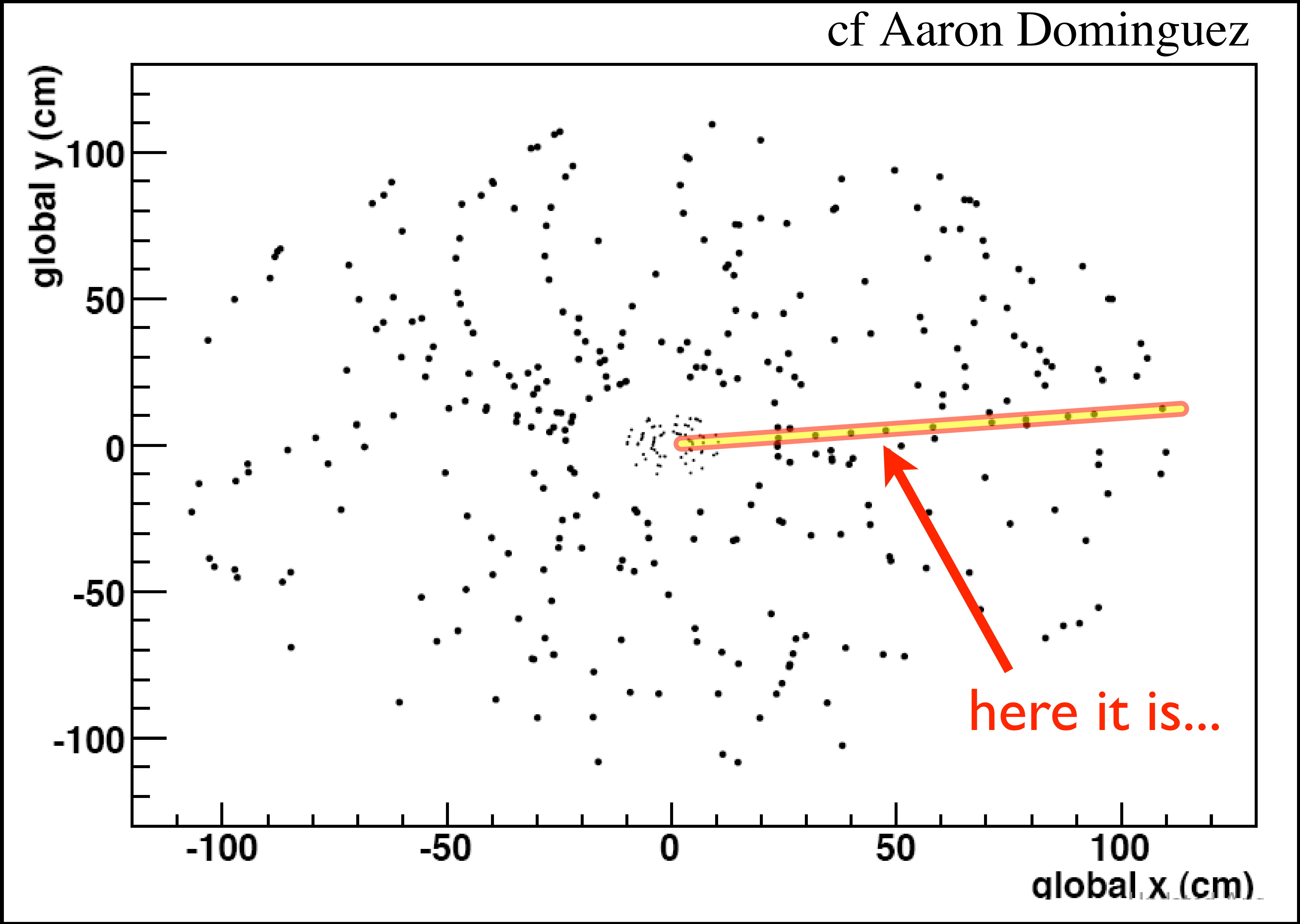
you saw this already !

Track Finding: Can you find the 50 GeV track?



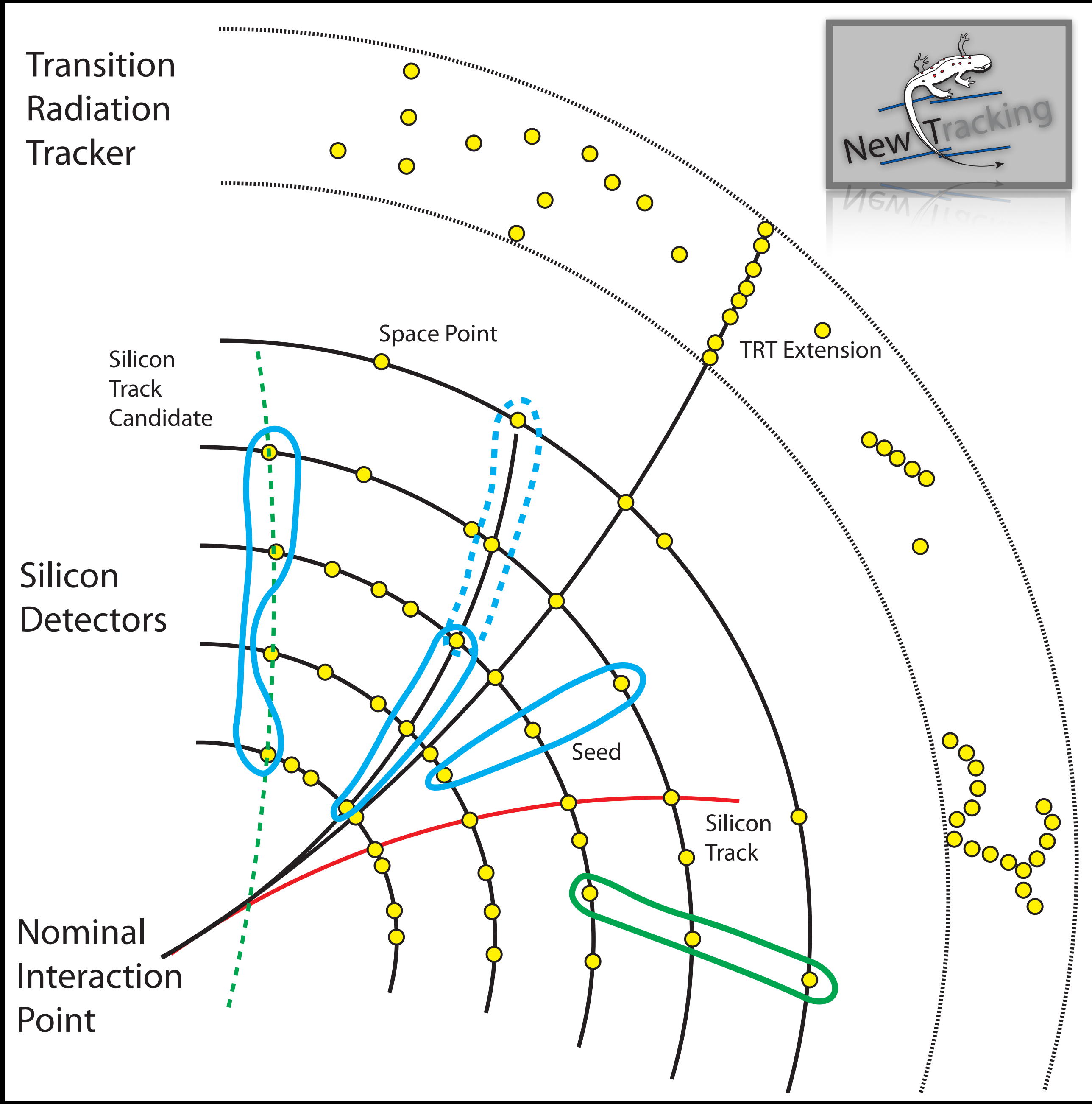
you saw this already !

Track Finding: Can you find the 50 GeV track?



Track Finding

- the task of the track finding
  - ➡ identify **track candidates** in event
  - ➡ cope with the combinatorial explosion of possible **hit combinations**
- different techniques
  - ➡ rough distinction: **local/sequential** and **global/parallel** methods
  - ➡ local method: generate **seeds and complete** them to track candidates
  - ➡ global method: **simultaneous clustering** of detector hits into track candidates
- some local methods
  - ➡ track road
  - ➡ track following
  - ➡ progressive track finding



- some global methods
  - ➡ conformal mapping
    - Hough and Legendre transform
  - ➡ adaptive methods
    - Elastic net, Cellular Automaton ...

      (will not discuss the latter)

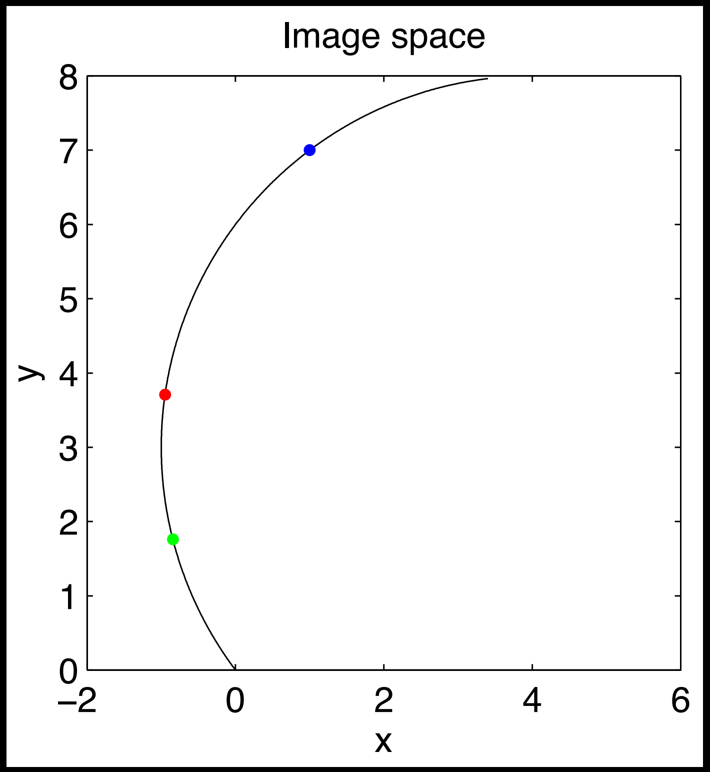
Conformal Mapping

- Hough transform
  - ➡ cycles through the origin in x-y transform into point in u-v

$$u = \frac{x}{x^2 + y^2}, \quad v = \frac{y}{x^2 + y^2}$$

$$\Rightarrow v = -\frac{x}{y} u + \frac{x^2 + y^2}{2y}$$

  - each hit becomes a straight line



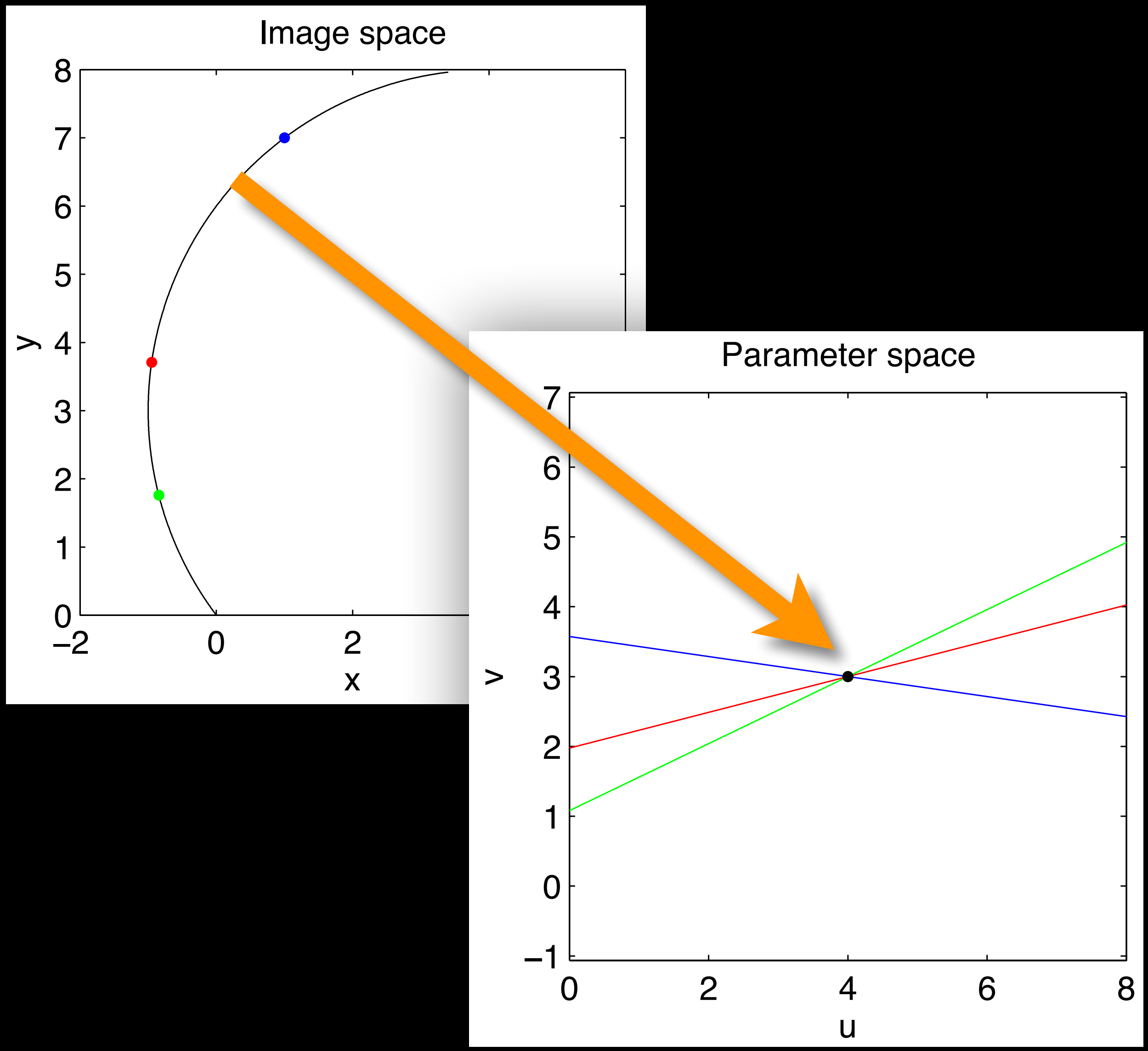
Conformal Mapping

- **Hough transform**
  - ➡ cycles through the origin in x-y transform into point in u-v

$$u = \frac{x}{x^2 + y^2}, \quad v = \frac{y}{x^2 + y^2}$$

$$\Rightarrow v = -\frac{x}{y}u + \frac{x^2 + y^2}{2y}$$

  - • each hit becomes a straight line
  - ➡ search for maxima (histogram) in **parameter space** to find track candidates



Conformal Mapping

● Hough transform

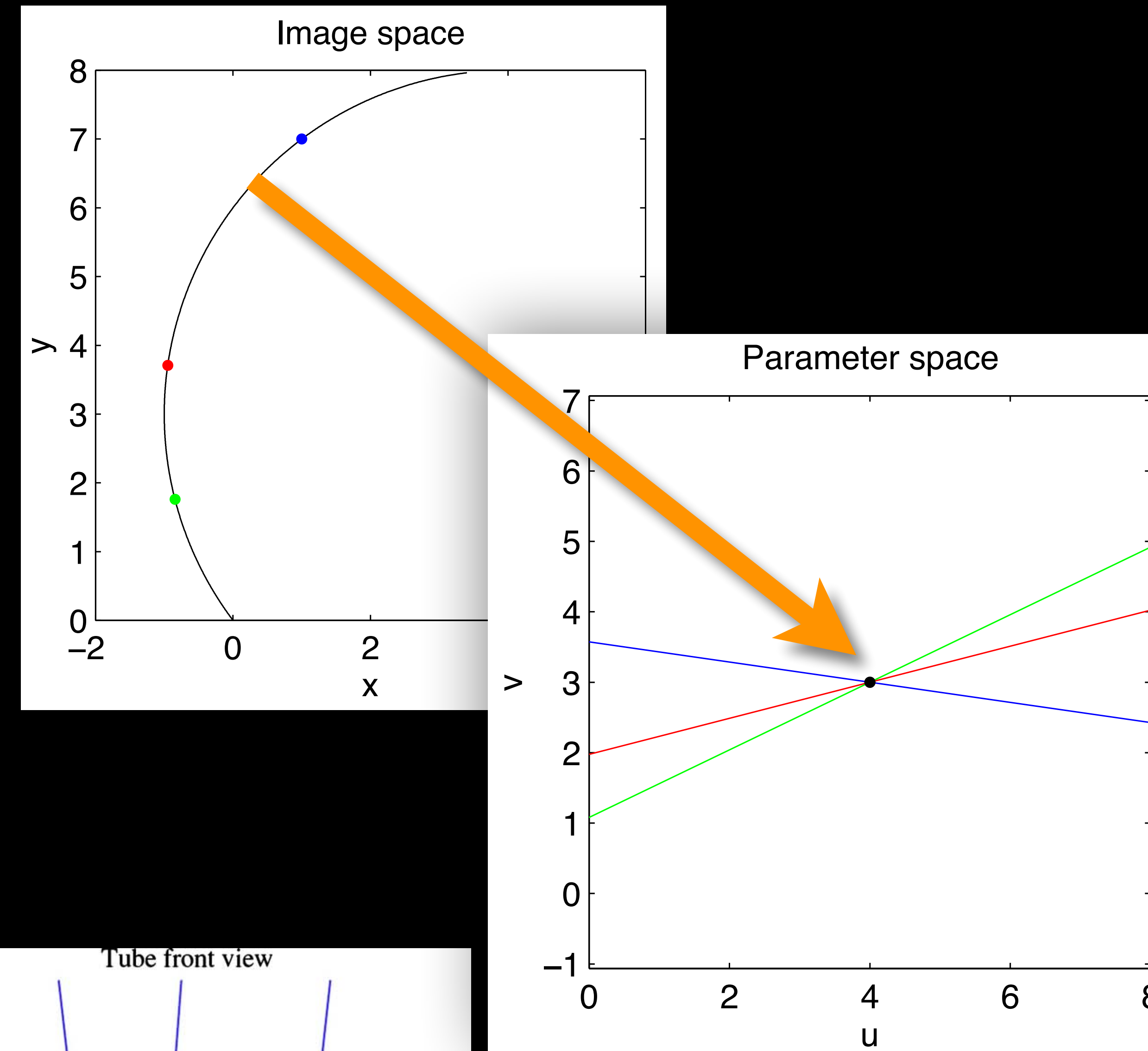
➡ cycles through the origin in x-y transform into point in u-v

$$u = \frac{x}{x^2 + y^2}, \quad v = \frac{y}{x^2 + y^2}$$

$$\Rightarrow \quad v = -\frac{x}{y}u + \frac{x^2 + y^2}{2y}$$

- each hit becomes a straight line

➡ search for maxima (histogram) in **parameter space** to find track candidates

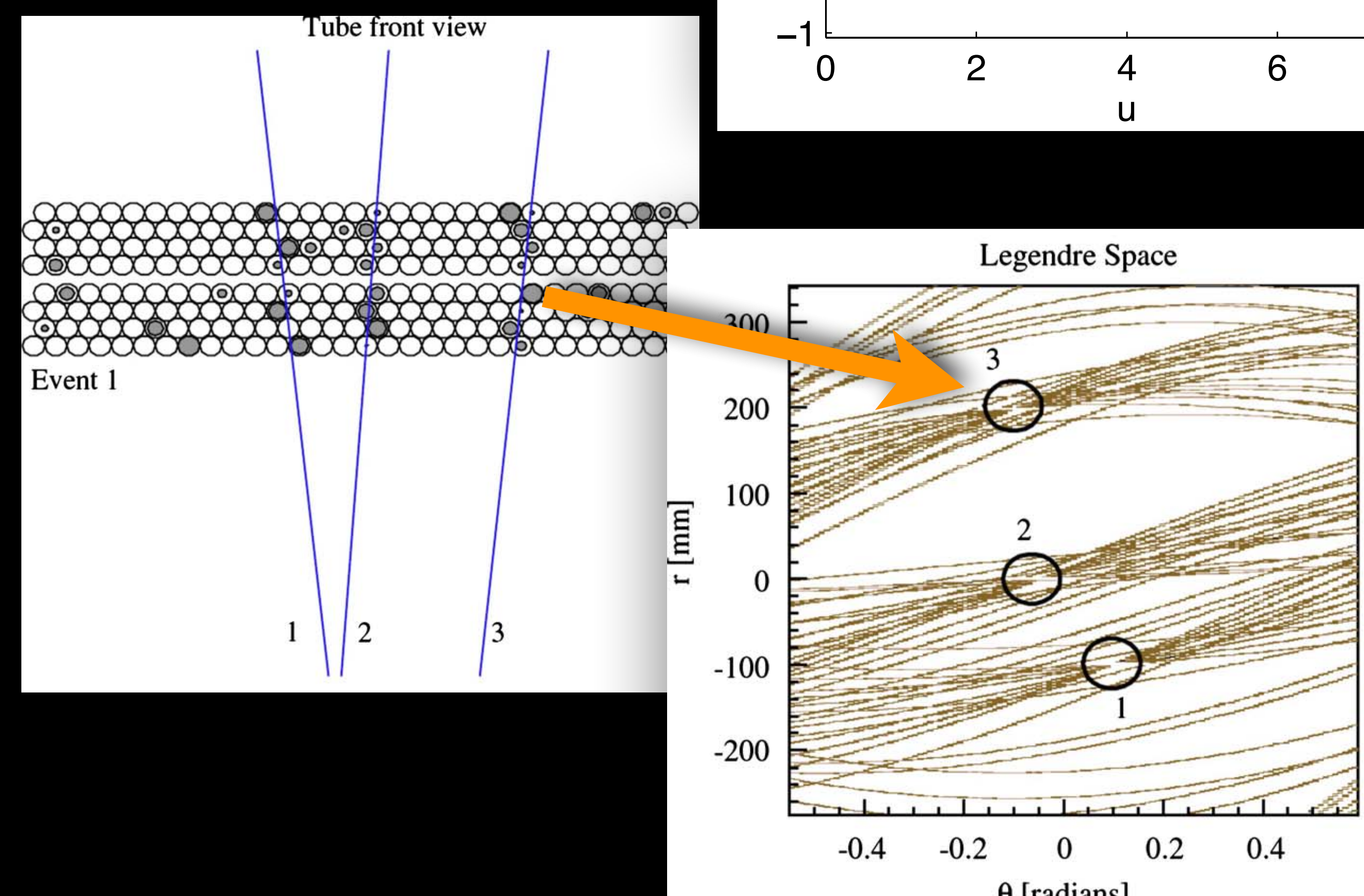


● Legendre transform

➡ used for track finding in drift tubes

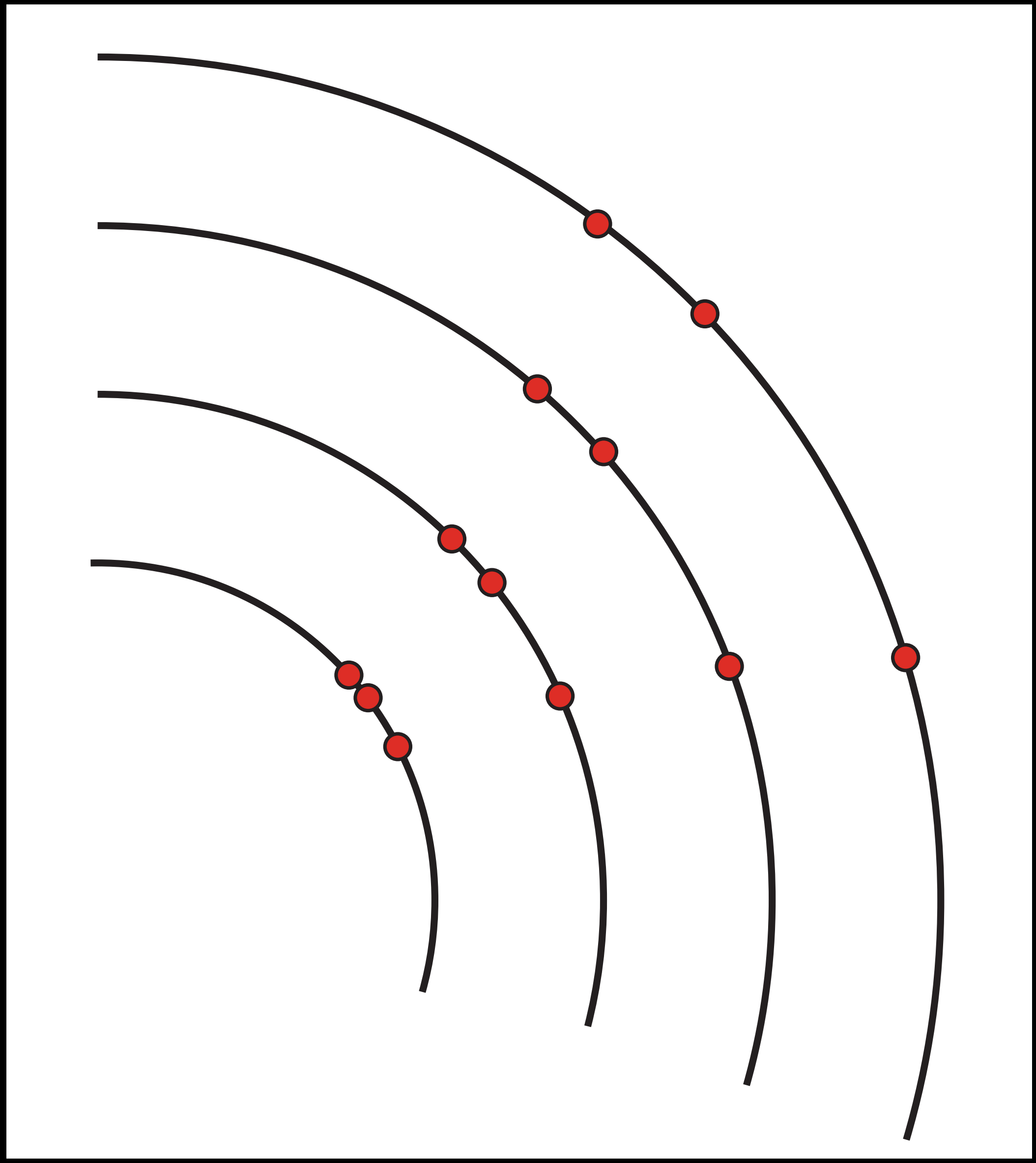
➡ drift radius is transformed into sine-curves in **Legendre space**

➡ solves as well L-R ambiguity



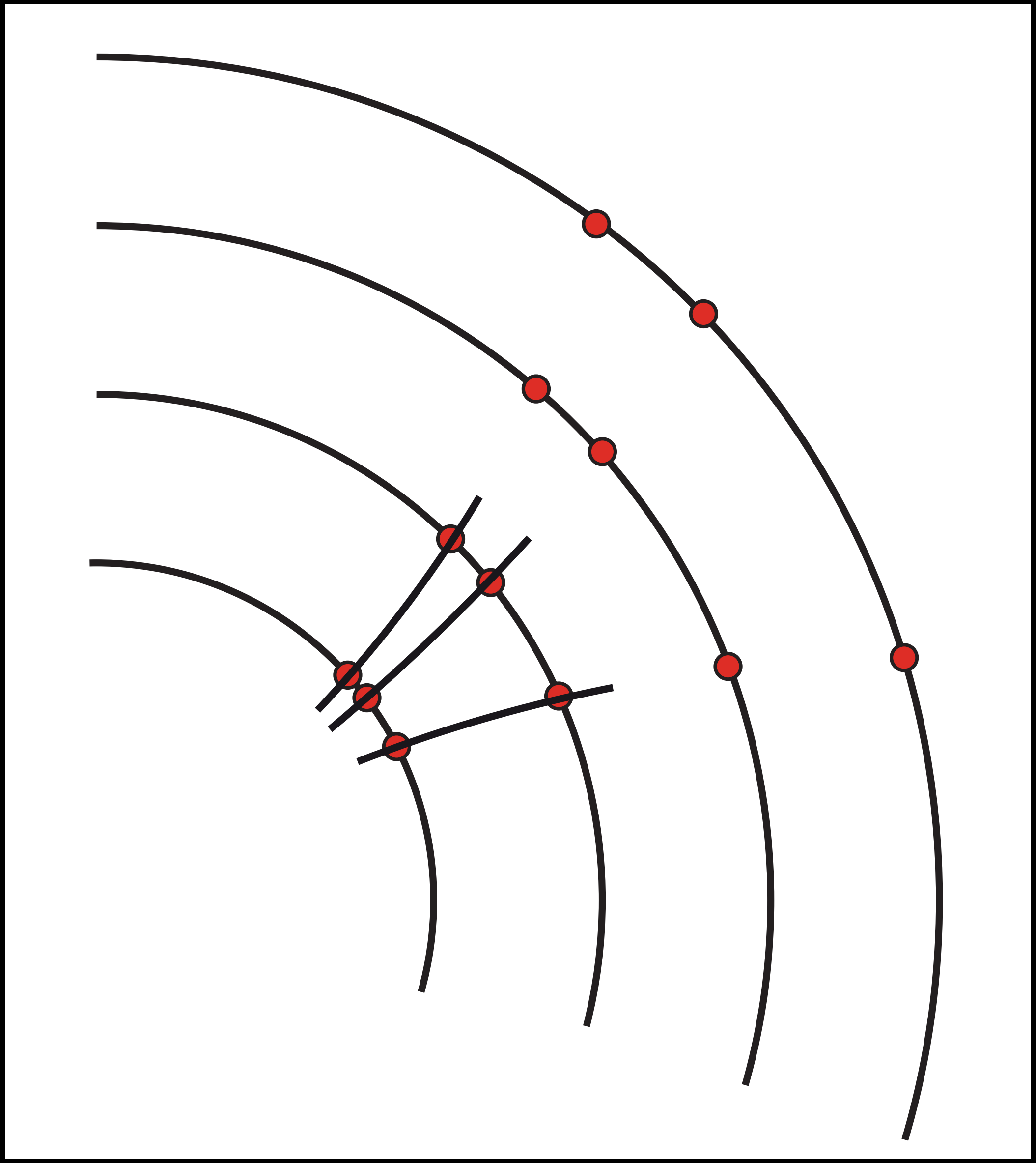
Local Track Finding

- Track Road algorithm



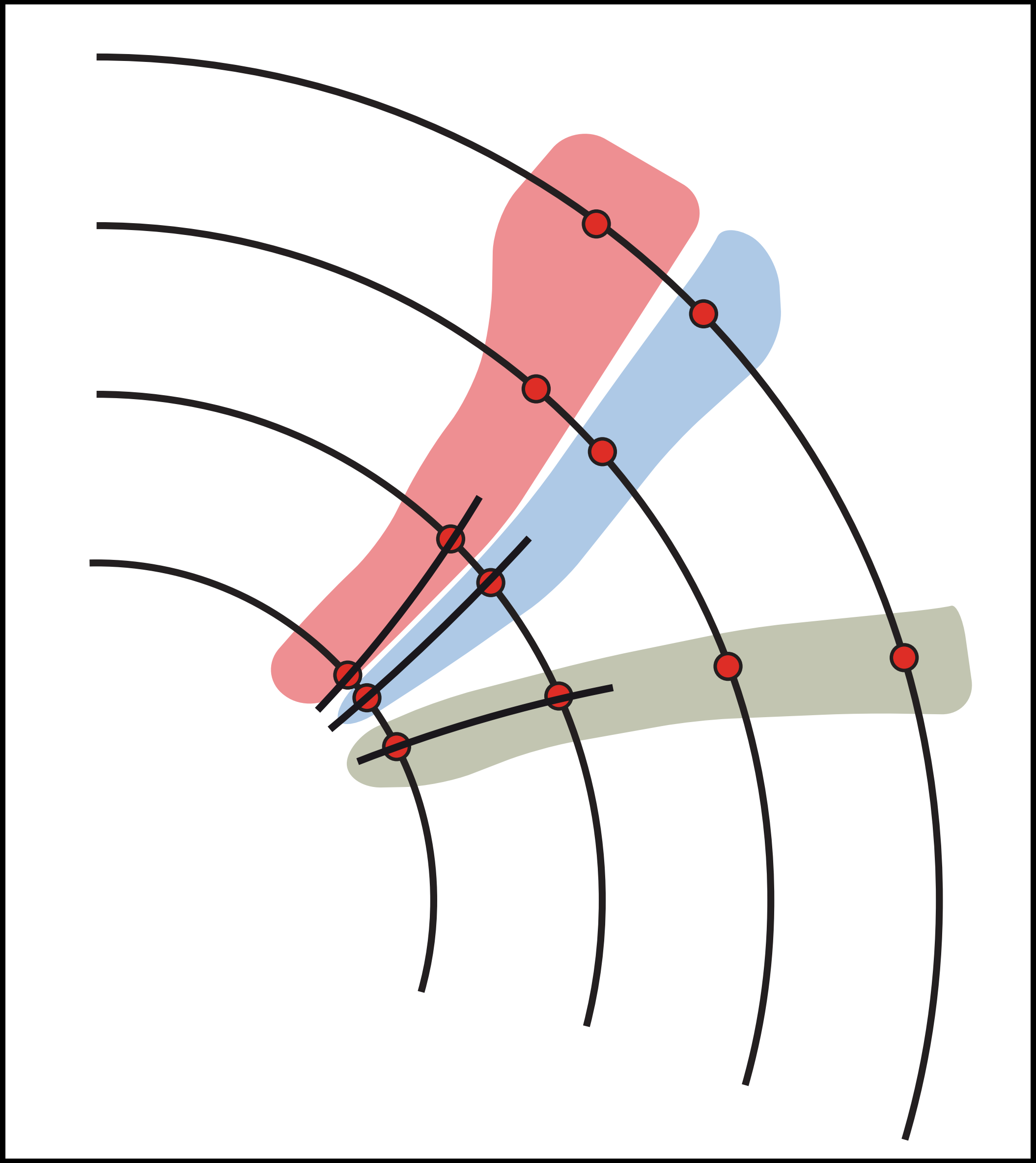
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits



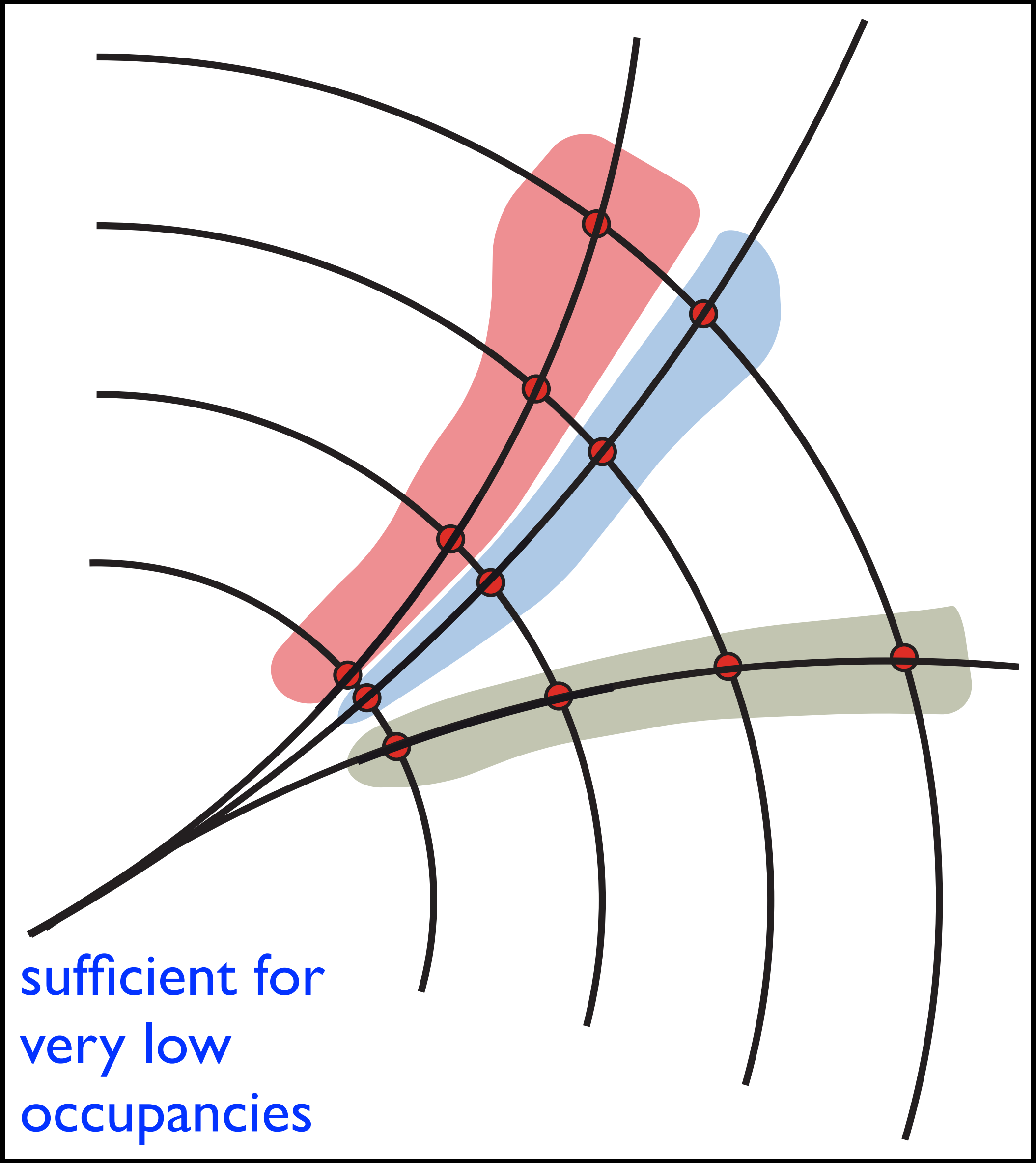
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory



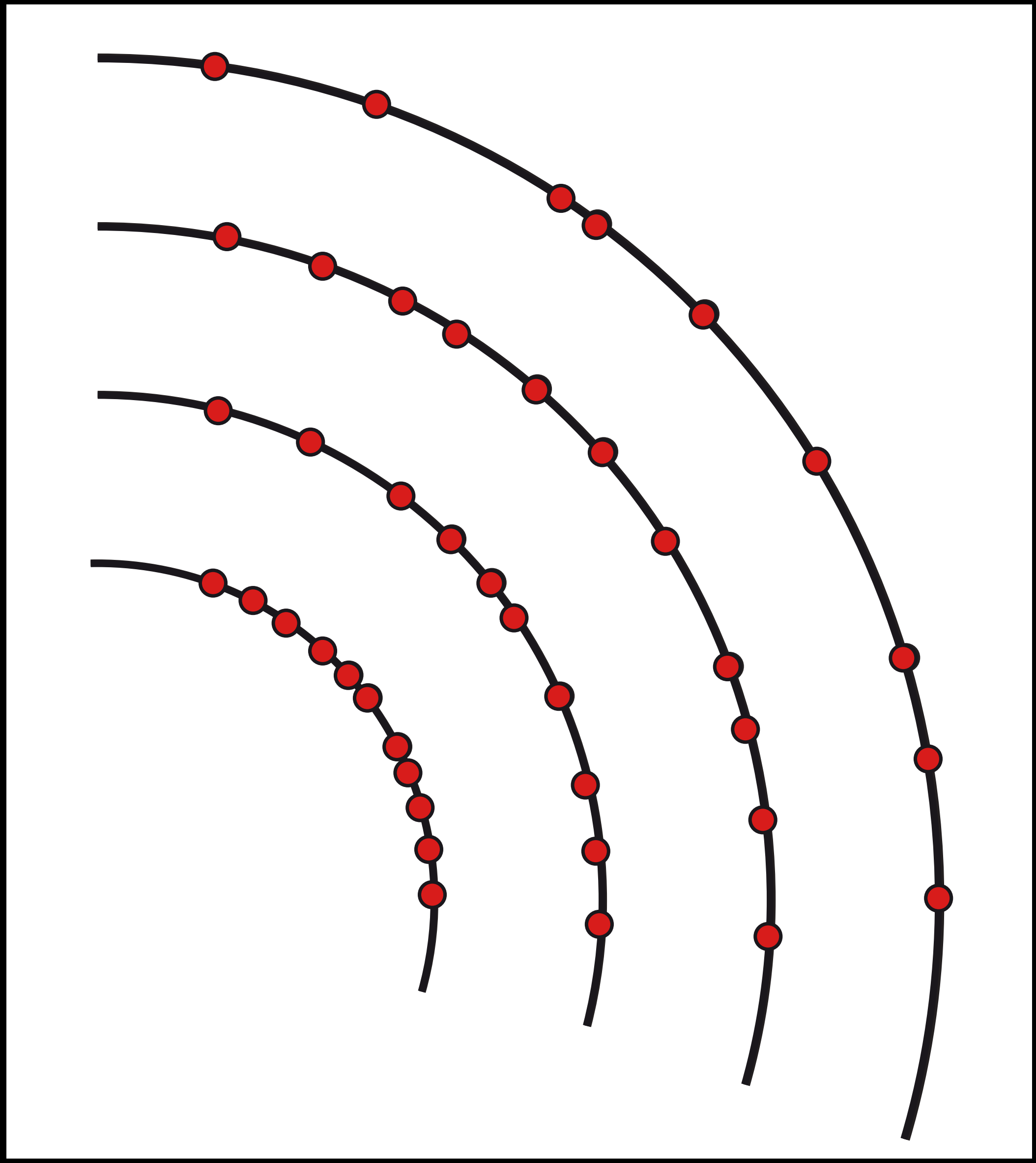
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**



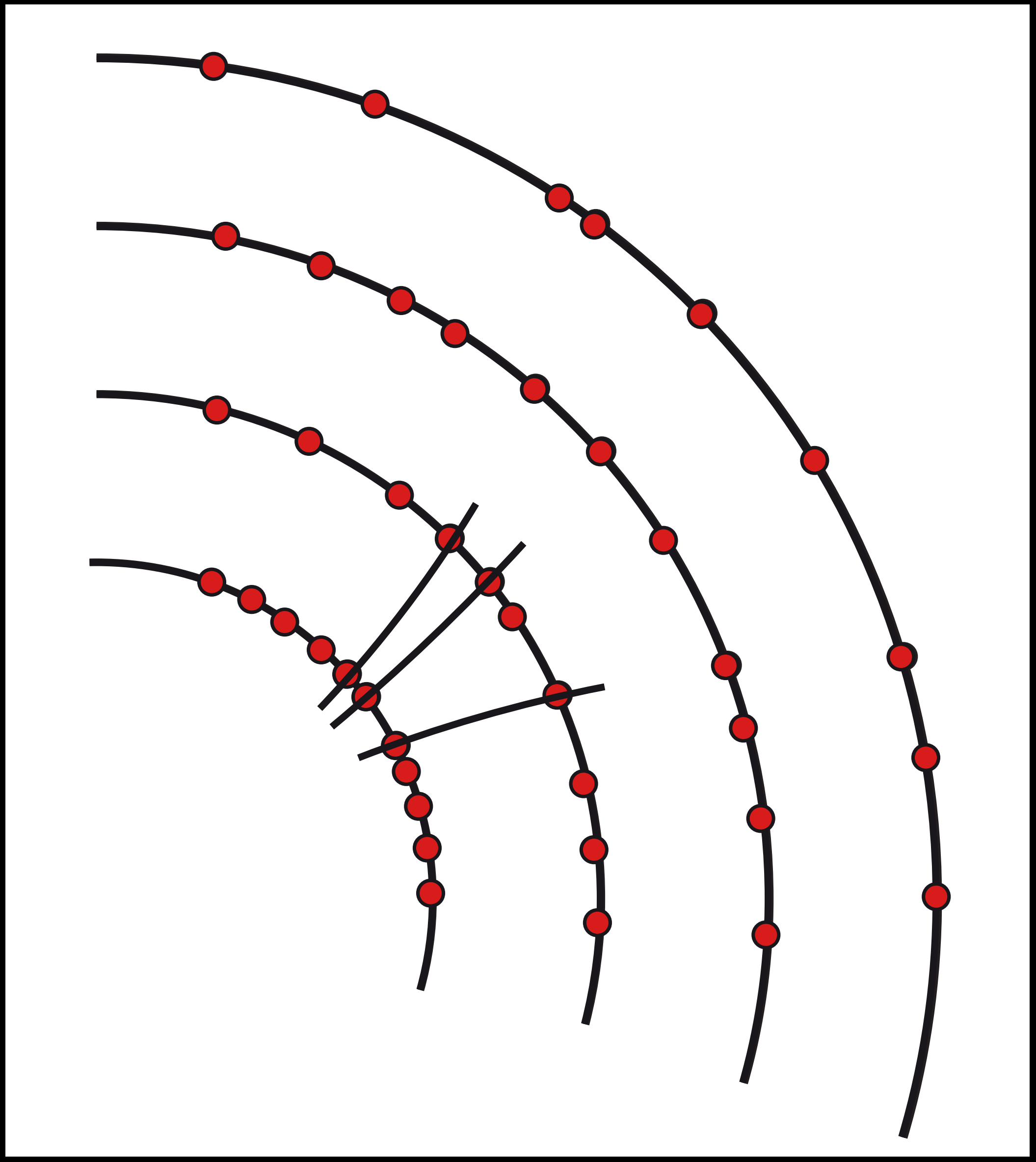
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Track Following



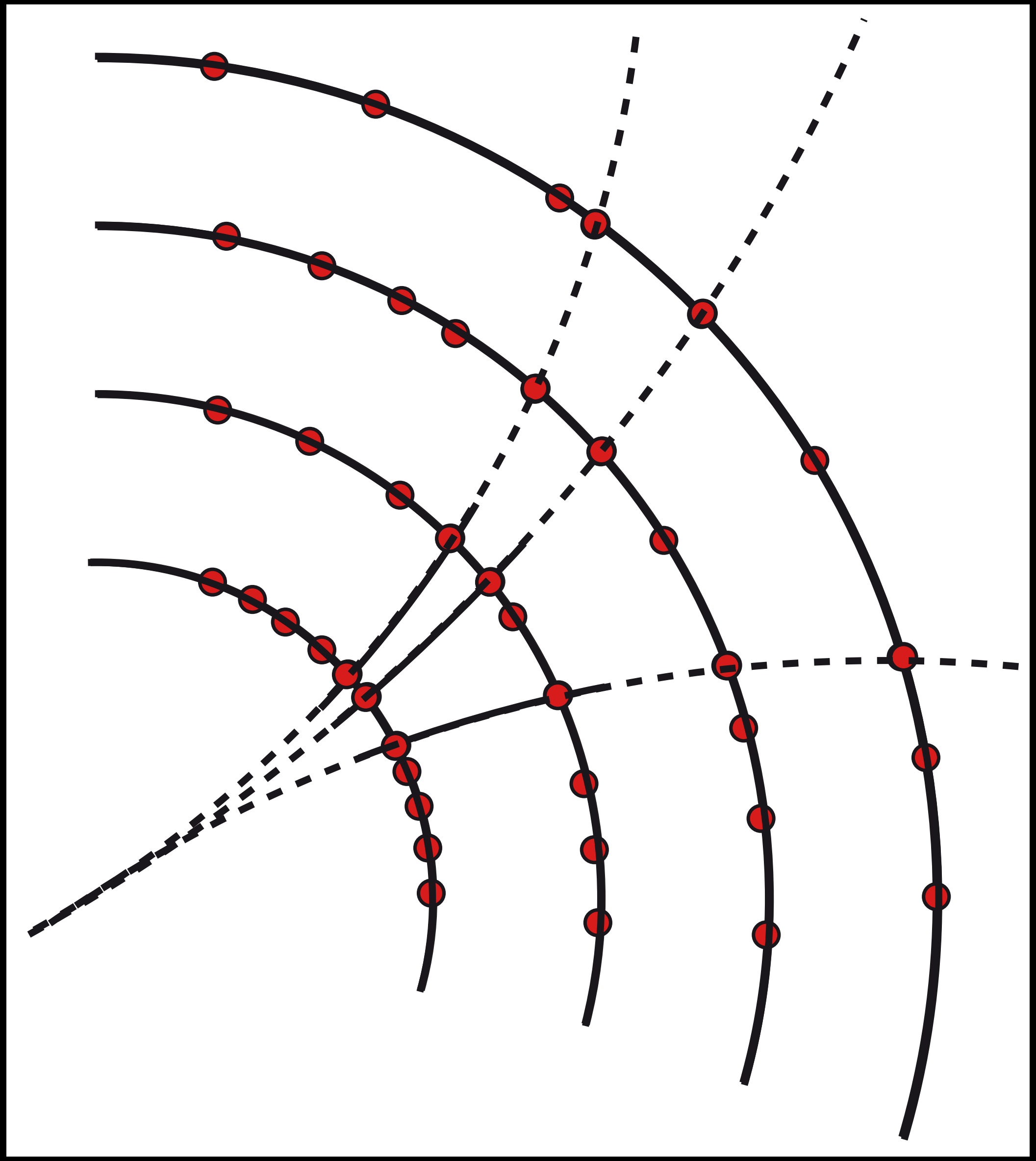
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Track Following
  - ➡ find **seeds** ~ combinations of 2-3 hits



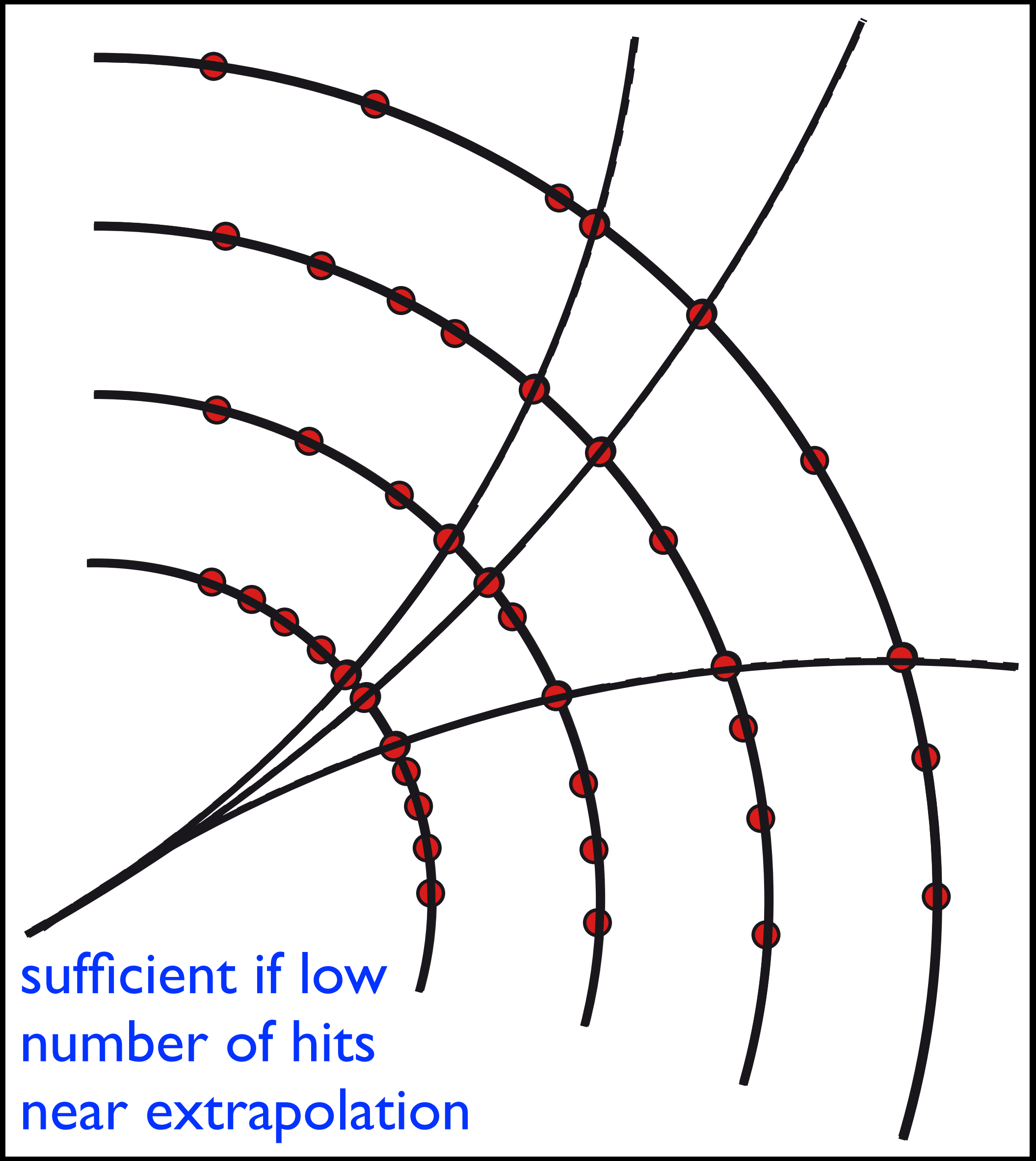
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Track Following
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory



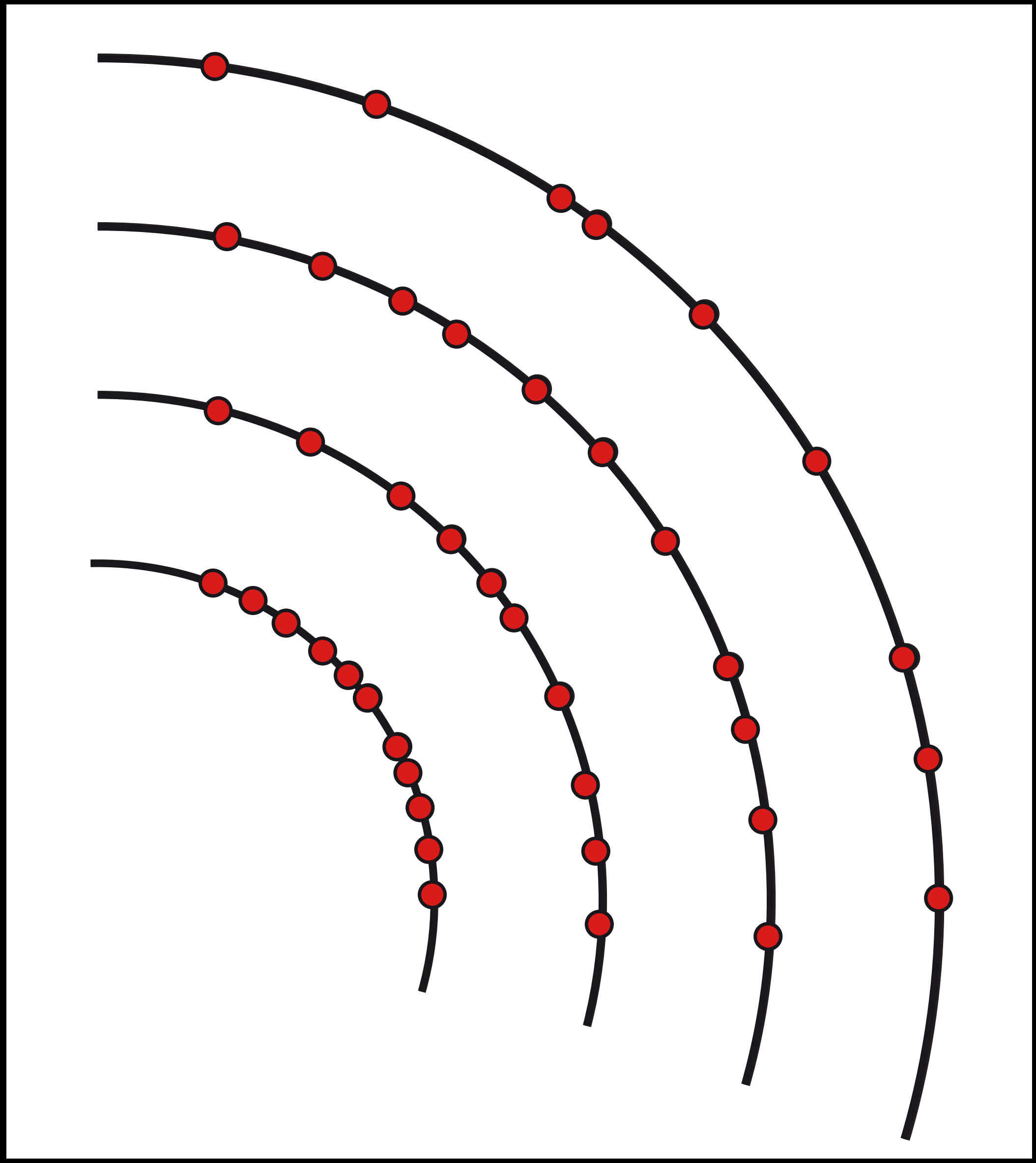
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Track Following
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**



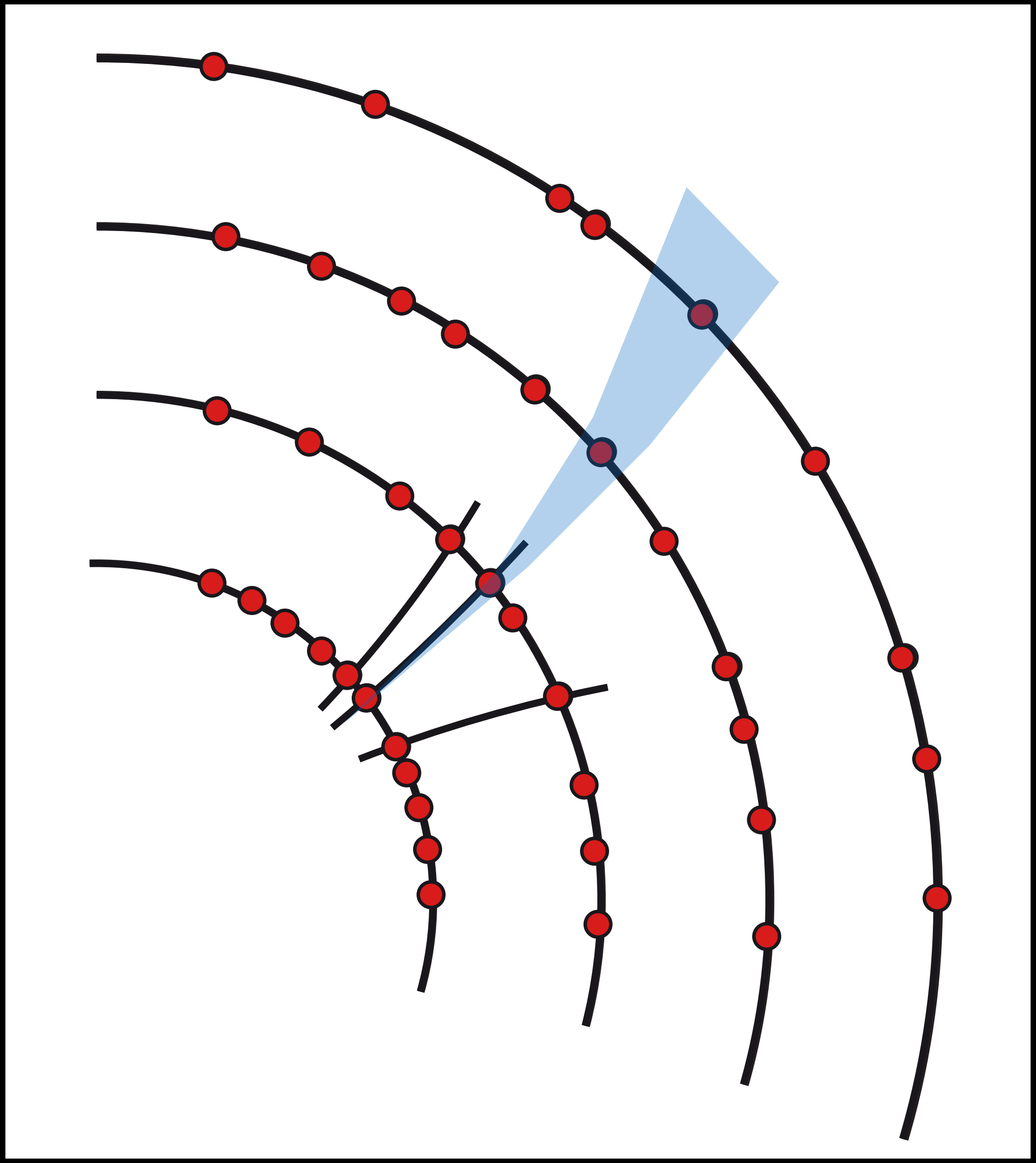
Local Track Finding

- **Track Road algorithm**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Track Following**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Progressive Track Finder**



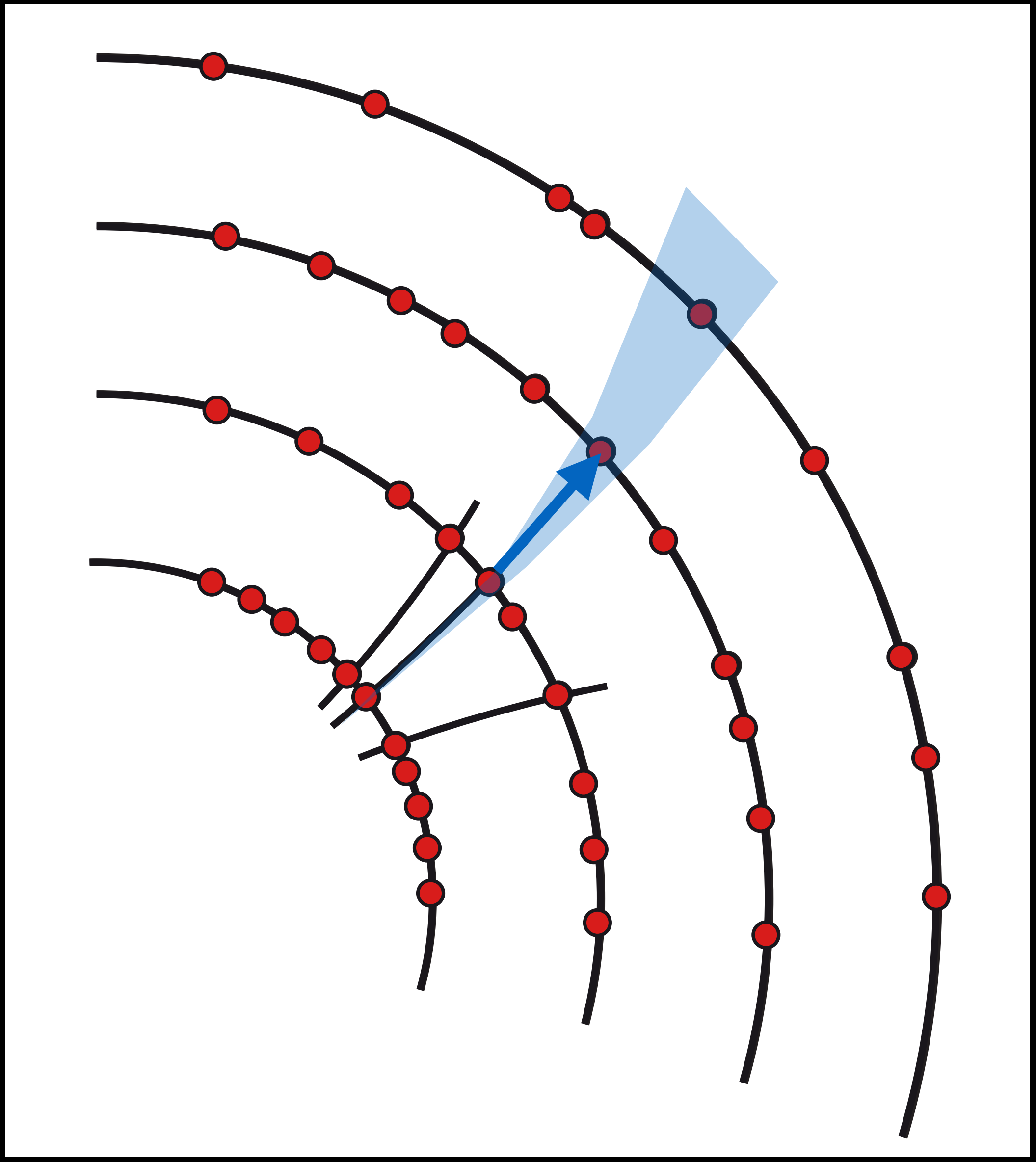
Local Track Finding

- **Track Road algorithm**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Track Following**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Progressive Track Finder**
  - ➡ find **seeds** ~ combinations of 2-3 hits



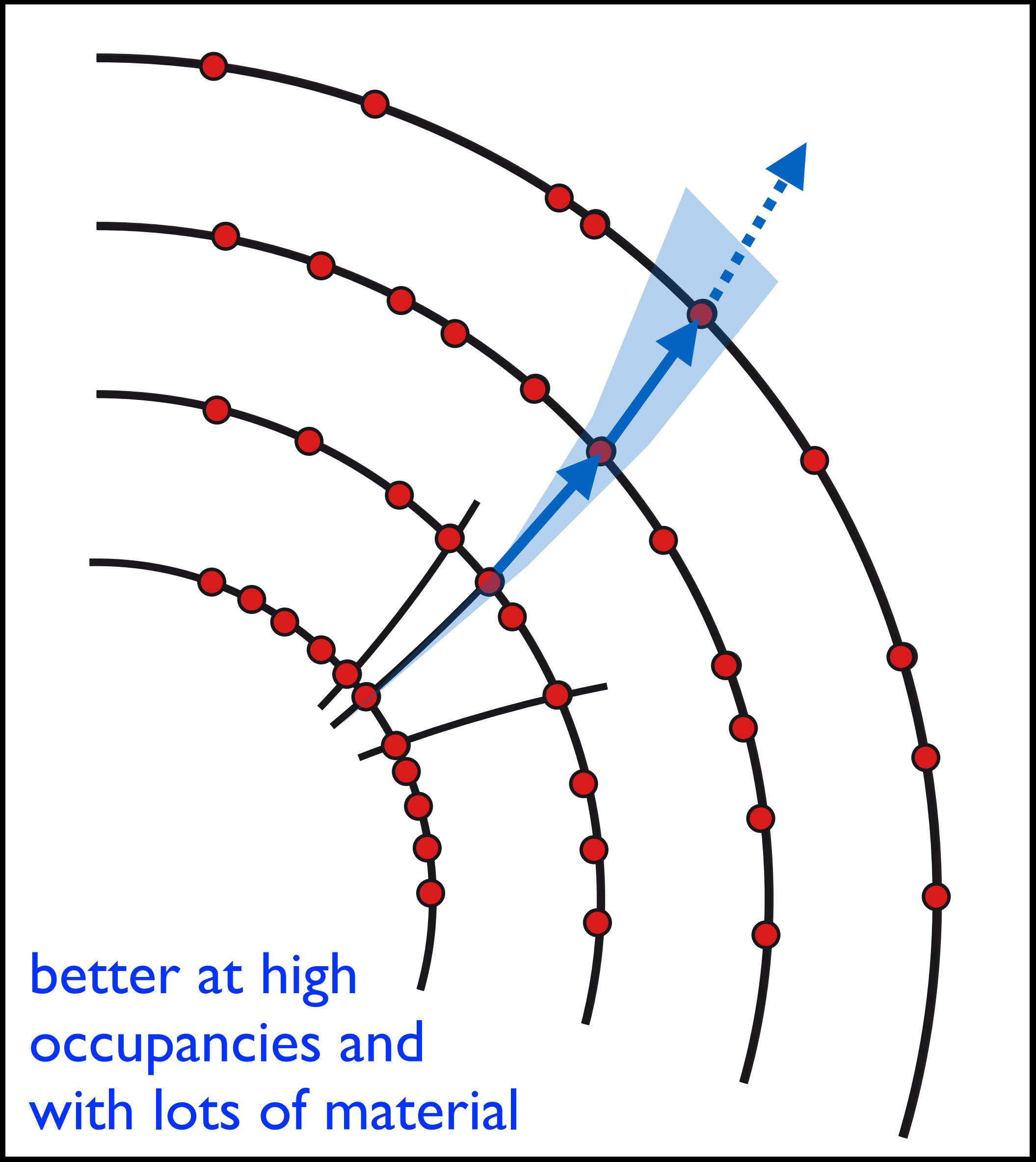
Local Track Finding

- **Track Road algorithm**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Track Following**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Progressive Track Finder**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** to next layer, find **best hit** and **update** trajectory



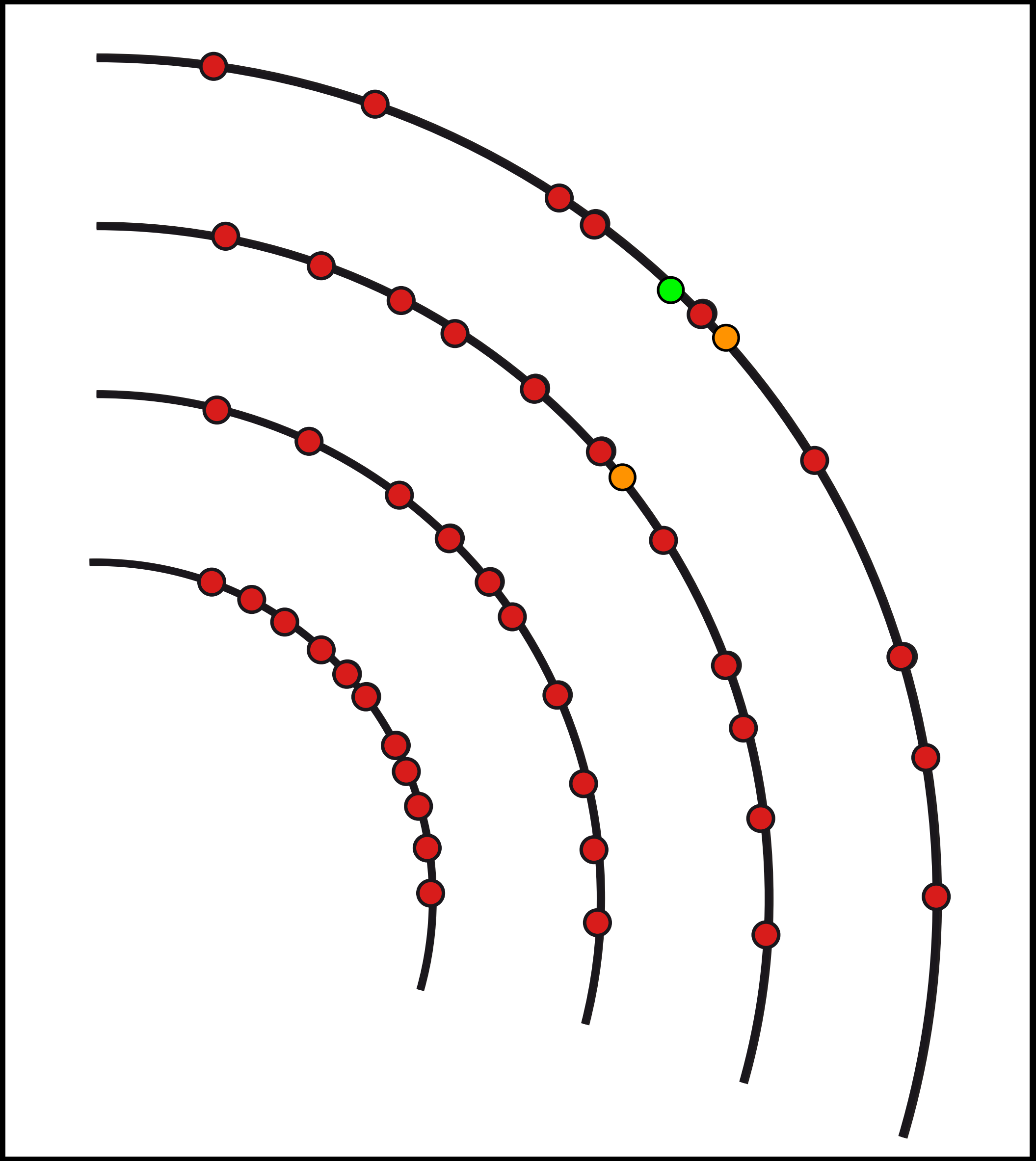
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Track Following
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Progressive Track Finder
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** to next layer, find **best hit** and **update** trajectory
  - ➡ repeat until last layers to obtain **candidates**



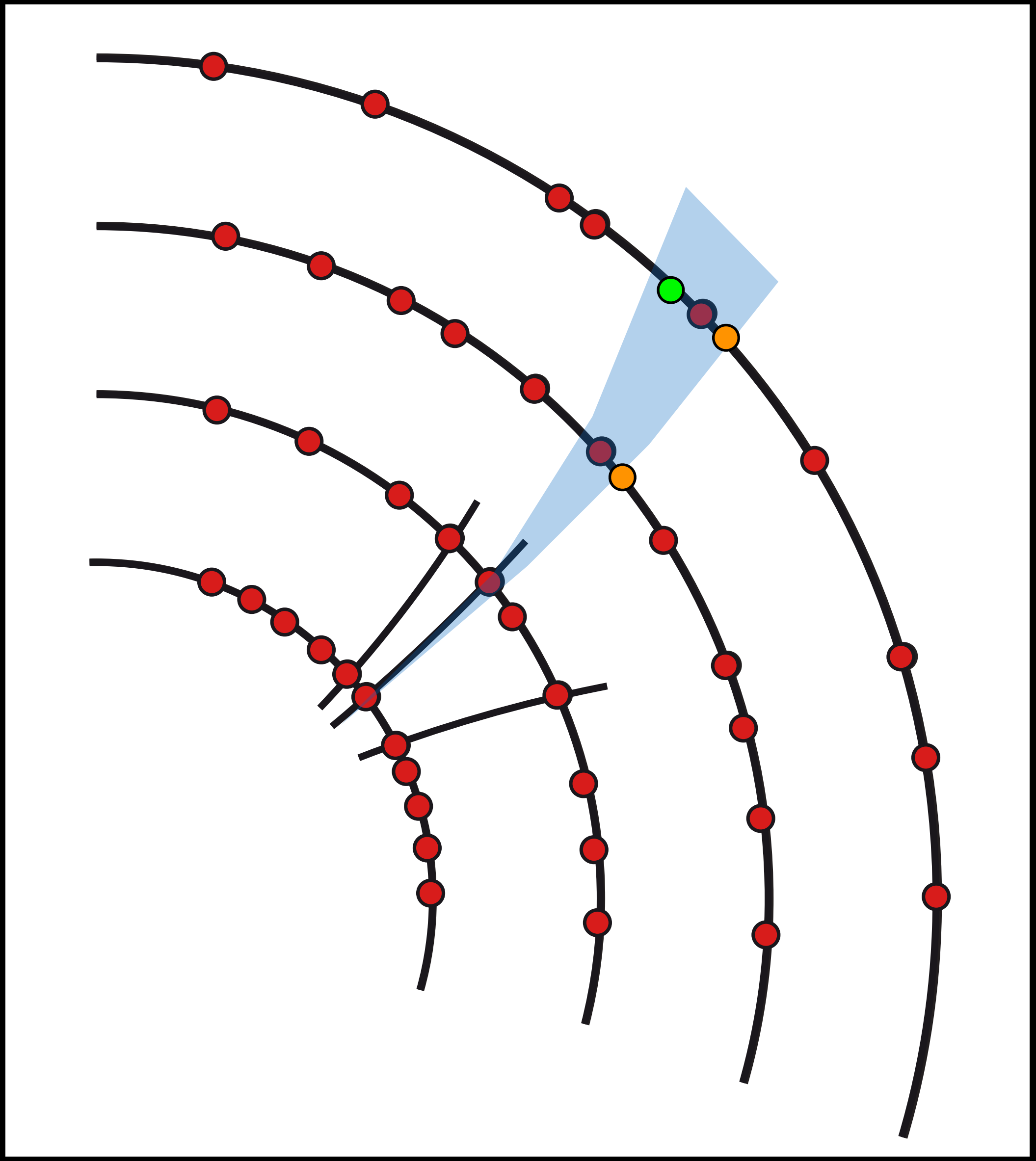
Local Track Finding

- Track Road algorithm
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Track Following
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- Progressive Track Finder
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** to next layer, find **best hit** and **update** trajectory
  - ➡ repeat until last layers to obtain **candidates**
- Combinatorial Kalman Filter



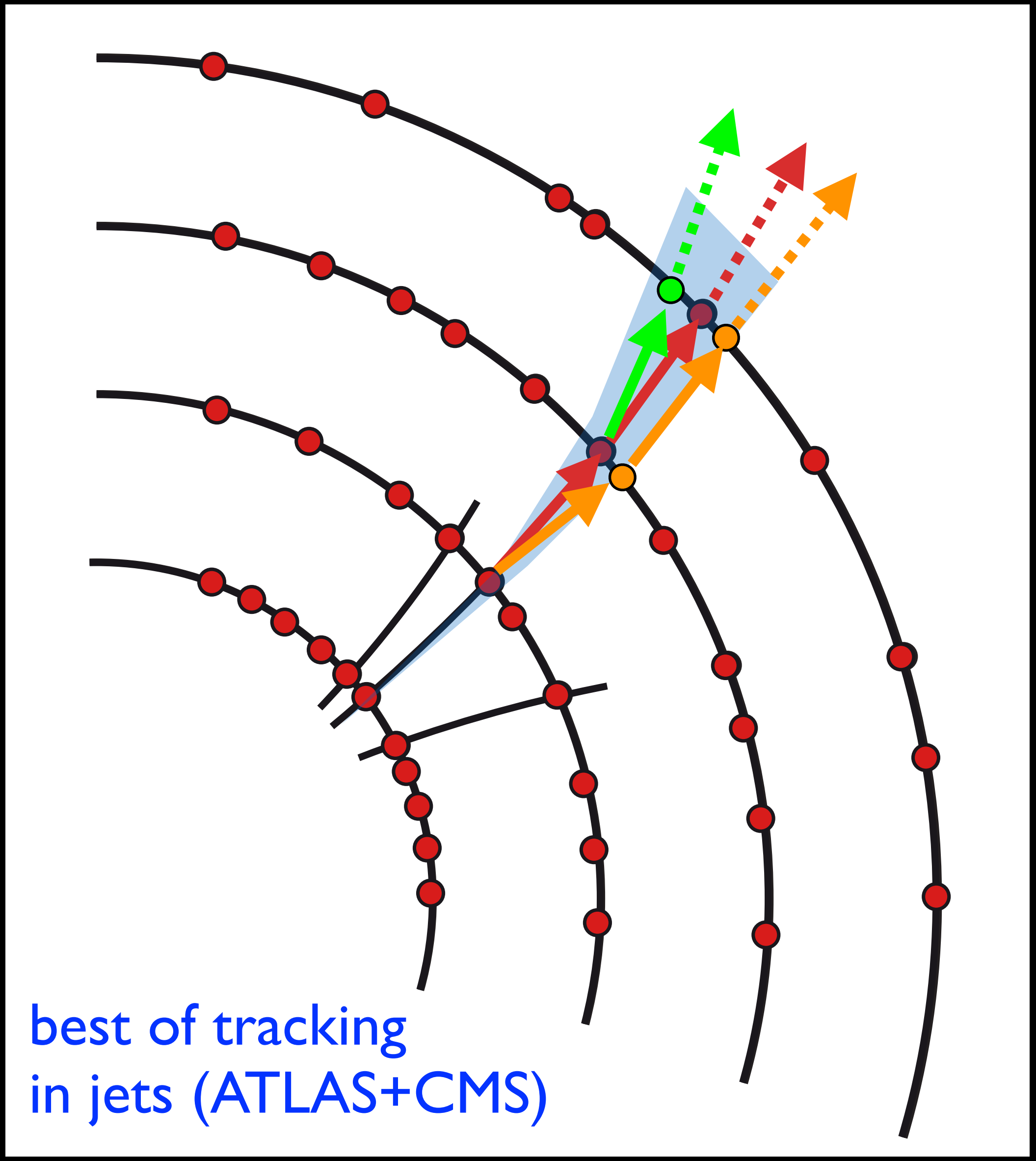
Local Track Finding

- **Track Road algorithm**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Track Following**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Progressive Track Finder**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** to next layer, find **best hit** and **update** trajectory
  - ➡ repeat until last layers to obtain **candidates**
- **Combinatorial Kalman Filter**
  - ➡ extension of a Progressive Track Finder for dense environments



Local Track Finding

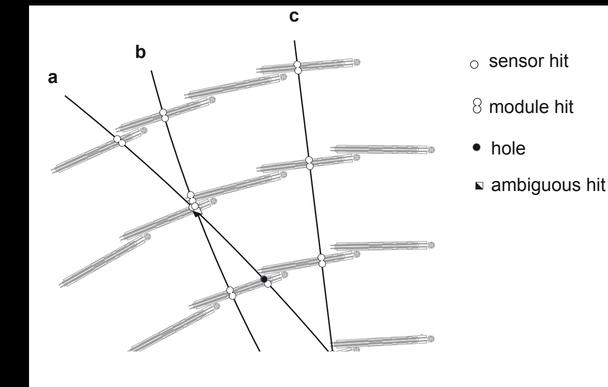
- **Track Road algorithm**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ build **road** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Track Following**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** along the likely trajectory
  - ➡ select **hits** on layers to obtain **candidates**
- **Progressive Track Finder**
  - ➡ find **seeds** ~ combinations of 2-3 hits
  - ➡ extrapolate **seed** to next layer, find **best hit** and **update** trajectory
  - ➡ repeat until last layers to obtain **candidates**
- **Combinatorial Kalman Filter**
  - ➡ extension of a Progressive Track Finder for dense environments
  - ➡ full **combinatorial exploration**, follow all hits to find all possible **track candidates**

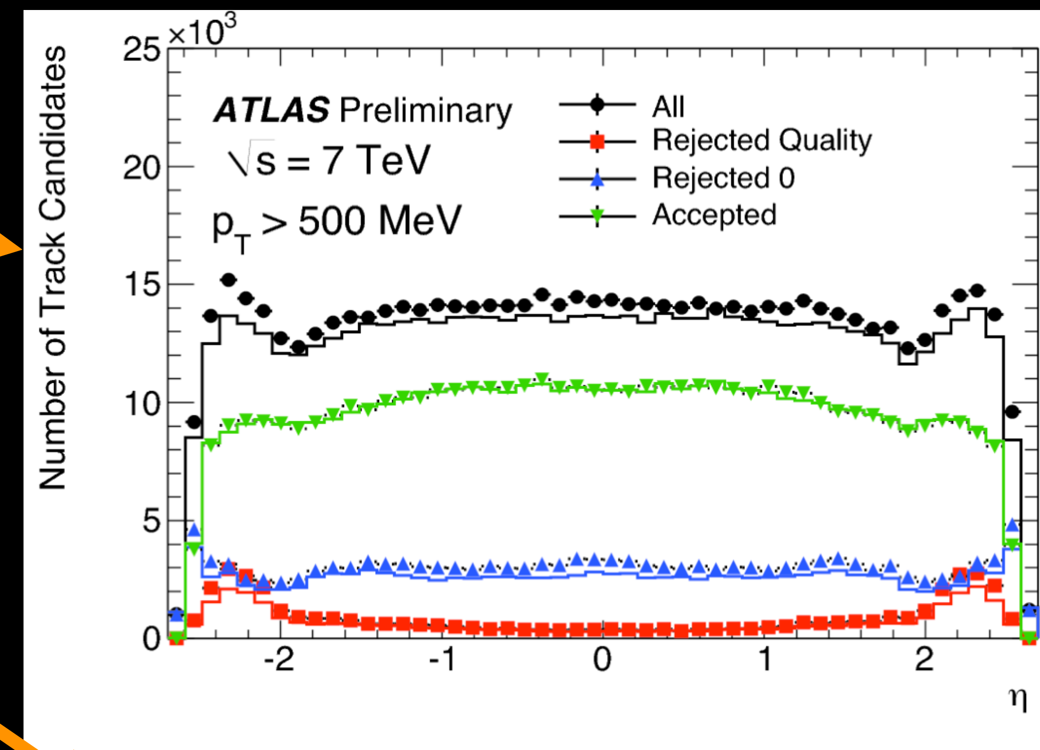


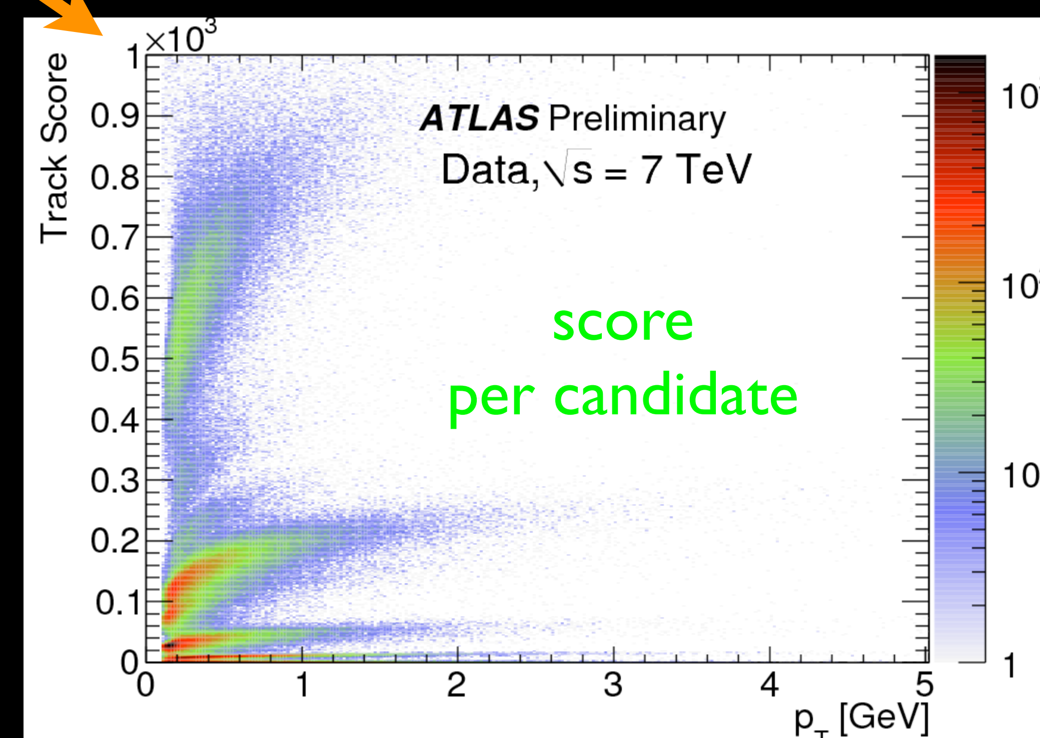
Ambiguity Solution

- track selection cuts
  - ➡ applied at every stage in reconstruction
  - ➡ still more candidates than final tracks
- task of ambiguity solution:
  - ➡ select good tracks and reject fakes
  - ➡ construct quality function ("score") for each candidate:
    1. hit content, holes
    2. number of shared hits
    3. fit quality...
  - ➡ candidates with best score win
  - ➡ if too many shared hits, create sub-tracks if if possible
  - ➡ in case of ATLAS: as well precise fit

- DELPHI (LEP), LC-Detector:
  - ➡ full recursive ambiguity processor
  - ➡ D.Wicke, M.E.



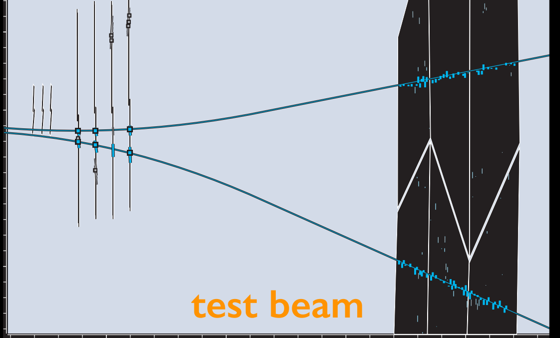


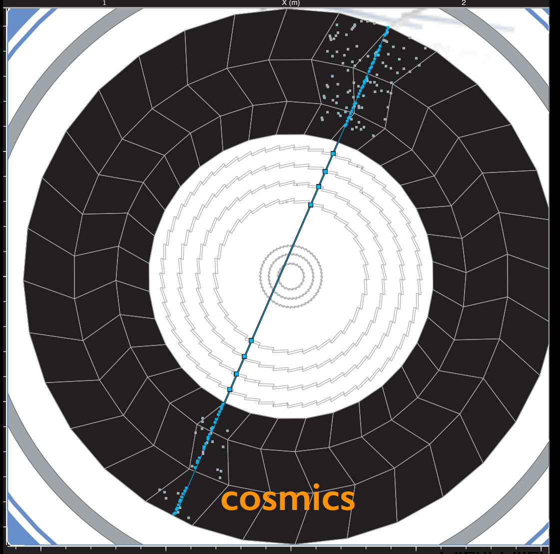


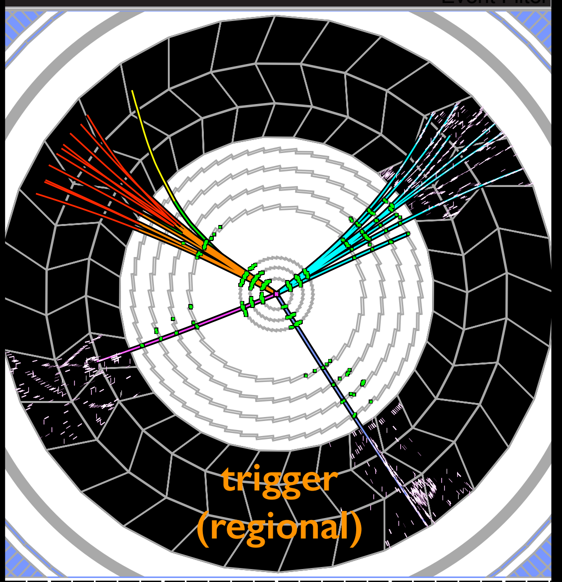
ATLAS Track Reconstruction

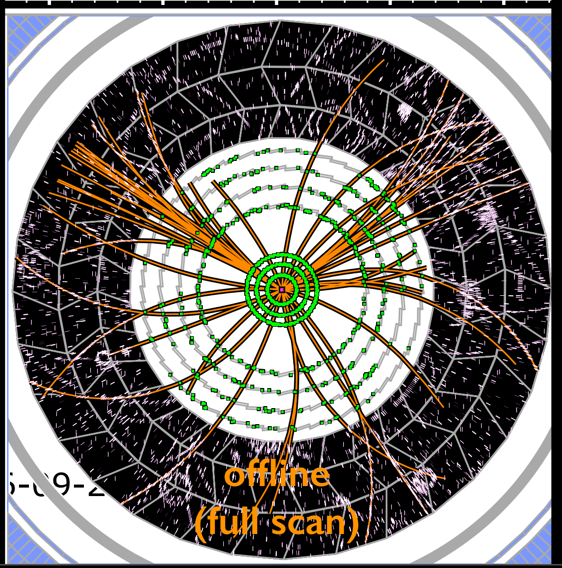
... and in Practice ?

- choice of reconstruction strategy depends on:
  - ➡ detector technologies
  - ➡ physics/performance requirements
  - ➡ occupancy and backgrounds
  - ➡ technical constraints (CPU, memory)
- even for same detector setup one looks at different types of events:
  - ➡ test beam
  - ➡ cosmics
  - ➡ trigger (regional)
  - ➡ offline (full scan)
- track reconstruction used by experiments
  - ➡ usually apply a **combination of different techniques**
  - ➡ often **iterative** ~ different strategies run one after the other to obtain best possible performance within resource constraints



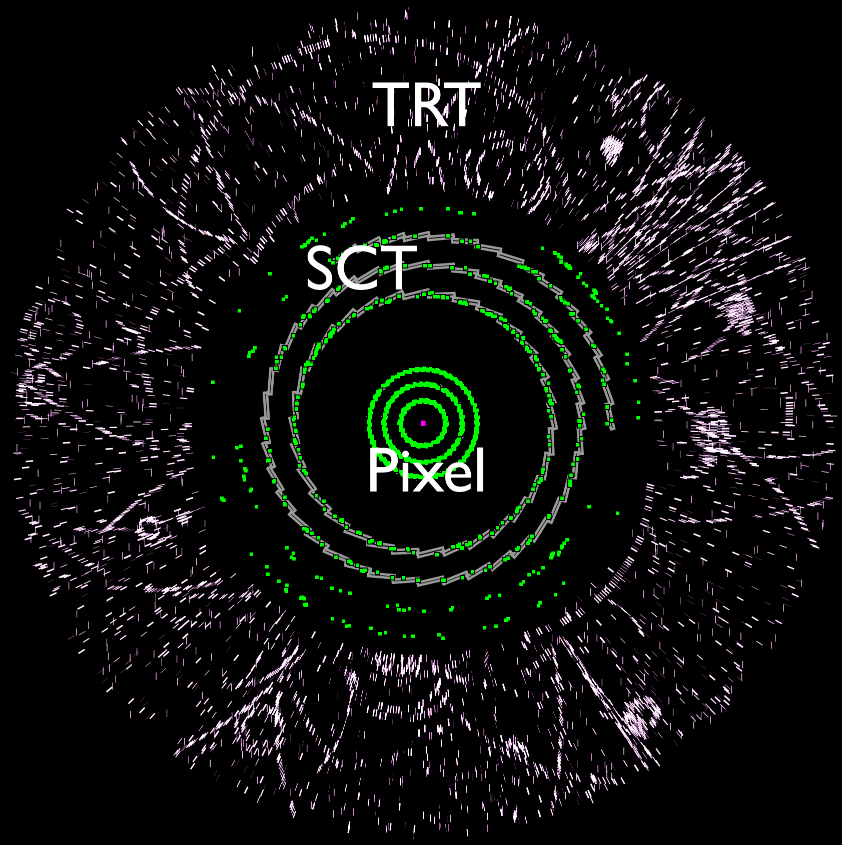






ATLAS NewTracking Software Chain

pre-precessing
- ➡ Pixel+SCT clustering
- ➡ TRT drift circle formation
- ➡ space points formation



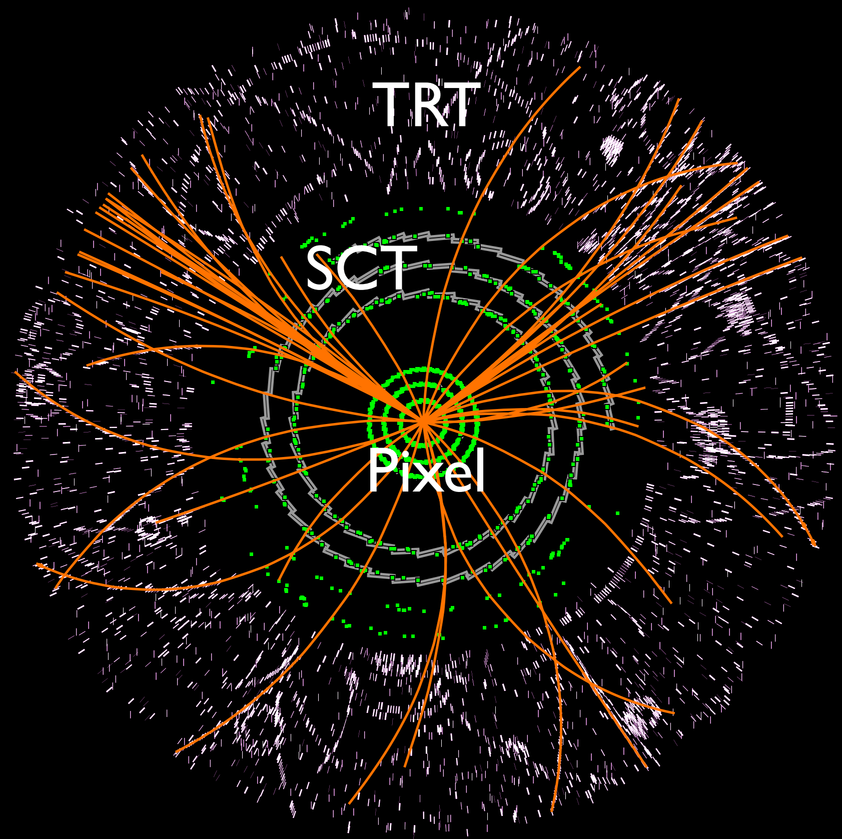
ATLAS NewTracking Software Chain

pre-precessing
- ➡ Pixel+SCT clustering
- ➡ TRT drift circle formation
- ➡ space points formation

combinatorial track finder
- ➡ iterative :
  1. Pixel seeds
  2. Pixel+SCT seeds
  3. SCT seeds
- ➡ restricted to roads
- ➡ bookkeeping to avoid duplicate candidates

ambiguity solution
- ➡ precise least square fit with full geometry
- ➡ selection of best silicon tracks using:
  1. hit content, holes
  2. number of shared hits
  3. fit quality...

extension into TRT
- ➡ progressive finder
- ➡ refit of track and selection



ATLAS NewTracking Software Chain

pre-precessing
- ➡ Pixel+SCT clustering
- ➡ TRT drift circle formation
- ➡ space points formation

combinatorial track finder
- ➡ iterative :
  1. Pixel seeds
  2. Pixel+SCT seeds
  3. SCT seeds
- ➡ restricted to roads
- ➡ bookkeeping to avoid duplicate candidates

ambiguity solution
- ➡ precise least square fit with full geometry
- ➡ selection of best silicon tracks using:
  1. hit content, holes
  2. number of shared hits
  3. fit quality...

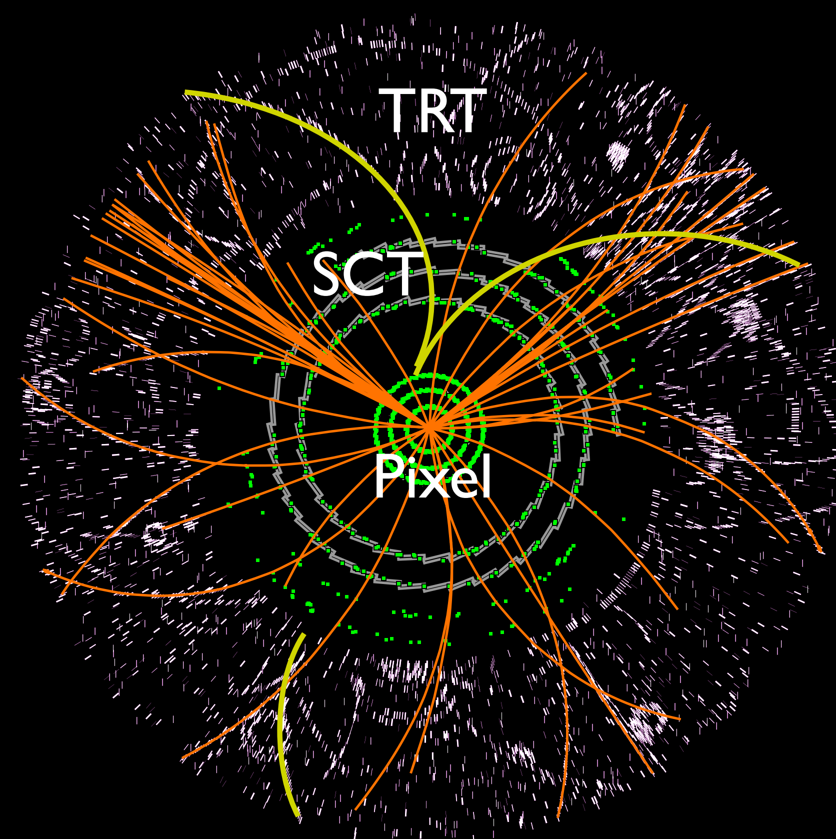
extension into TRT
- ➡ progressive finder
- ➡ refit of track and selection

TRT segment finder
- ➡ on remaining drift circles
- ➡ uses Hough transform

TRT seeded finder
- ➡ from TRT into SCT+Pixels
- ➡ combinatorial finder

ambiguity solution
- ➡ precise fit and selection
- ➡ TRT seeded tracks

standalone TRT
- ➡ unused TRT segments



ATLAS NewTracking Software Chain

pre-precessing
- ➡ Pixel+SCT clustering
- ➡ TRT drift circle formation
- ➡ space points formation

combinatorial track finder
- ➡ iterative :
  1. Pixel seeds
  2. Pixel+SCT seeds
  3. SCT seeds
- ➡ restricted to roads
- ➡ bookkeeping to avoid duplicate candidates

ambiguity solution
- ➡ precise least square fit with full geometry
- ➡ selection of best silicon tracks using:
  1. hit content, holes
  2. number of shared hits
  3. fit quality...

extension into TRT
- ➡ progressive finder
- ➡ refit of track and selection

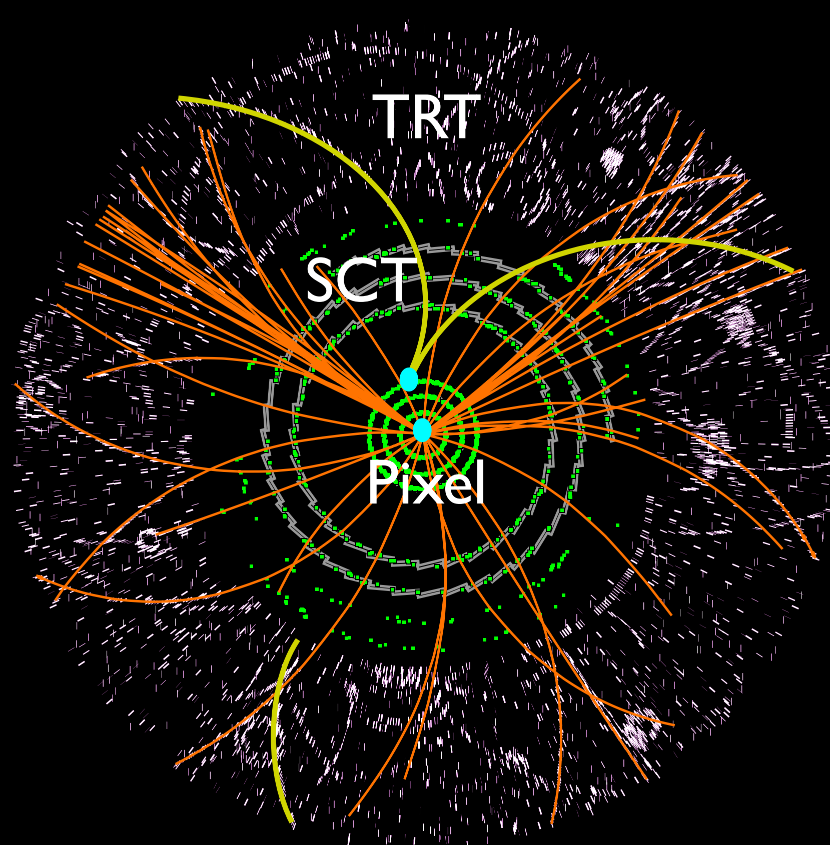
TRT segment finder
- ➡ on remaining drift circles
- ➡ uses Hough transform

TRT seeded finder
- ➡ from TRT into SCT+Pixels
- ➡ combinatorial finder

ambiguity solution
- ➡ precise fit and selection
- ➡ TRT seeded tracks

standalone TRT
- ➡ unused TRT segments

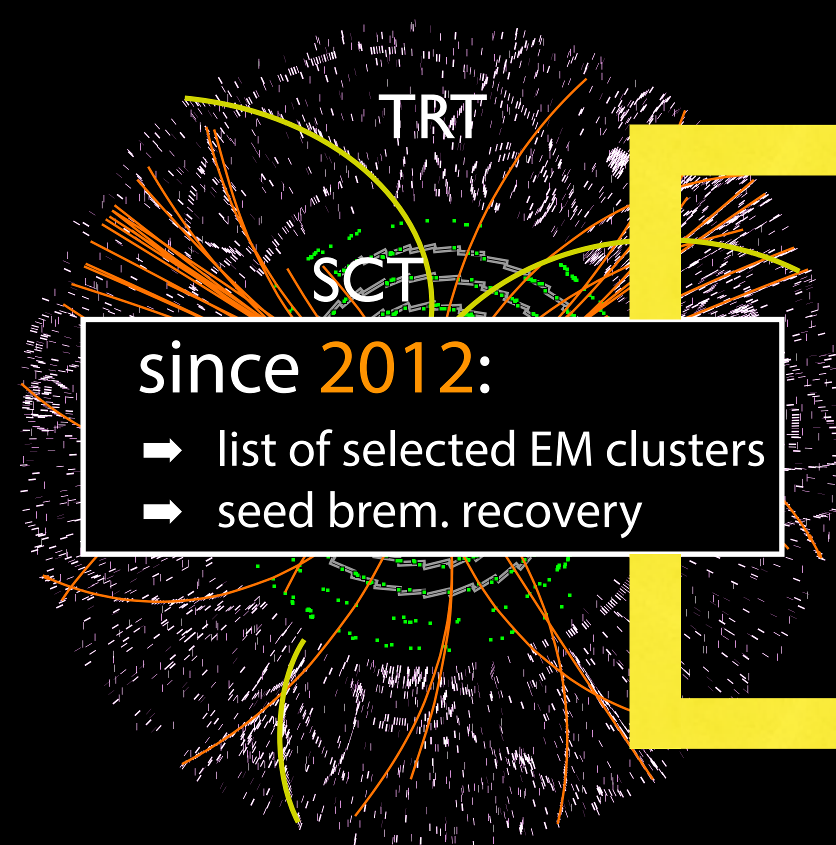
vertexing
- ➡ primary vertexing
- ➡ conversion and V0 search



ATLAS NewTracking Software Chain

pre-precessing
- ➡ Pixel+SCT clustering
- ➡ TRT drift circle formation
- ➡ space points formation

combinatorial track finder
- ➡ iterative :
  1. Pixel seeds
  2. Pixel+SCT seeds
  3. SCT seeds
- ➡ restricted to roads
- ➡ bookkeeping to avoid duplicate candidates



since 2012:
- ➡ list of selected EM clusters
- ➡ seed brem. recovery

ambiguity solution
- ➡ precise least square fit with full geometry
- ➡ selection of best silicon tracks using:
  1. hit content, holes
  2. number of shared hits
  3. fit quality...

extension into TRT
- ➡ progressive finder
- ➡ refit of track and selection

TRT segment finder
- ➡ on remaining drift circles
- ➡ uses Hough transform

TRT seeded finder
- ➡ from TRT into SCT+Pixels
- ➡ combinatorial finder

ambiguity solution
- ➡ precise fit and selection
- ➡ TRT seeded tracks

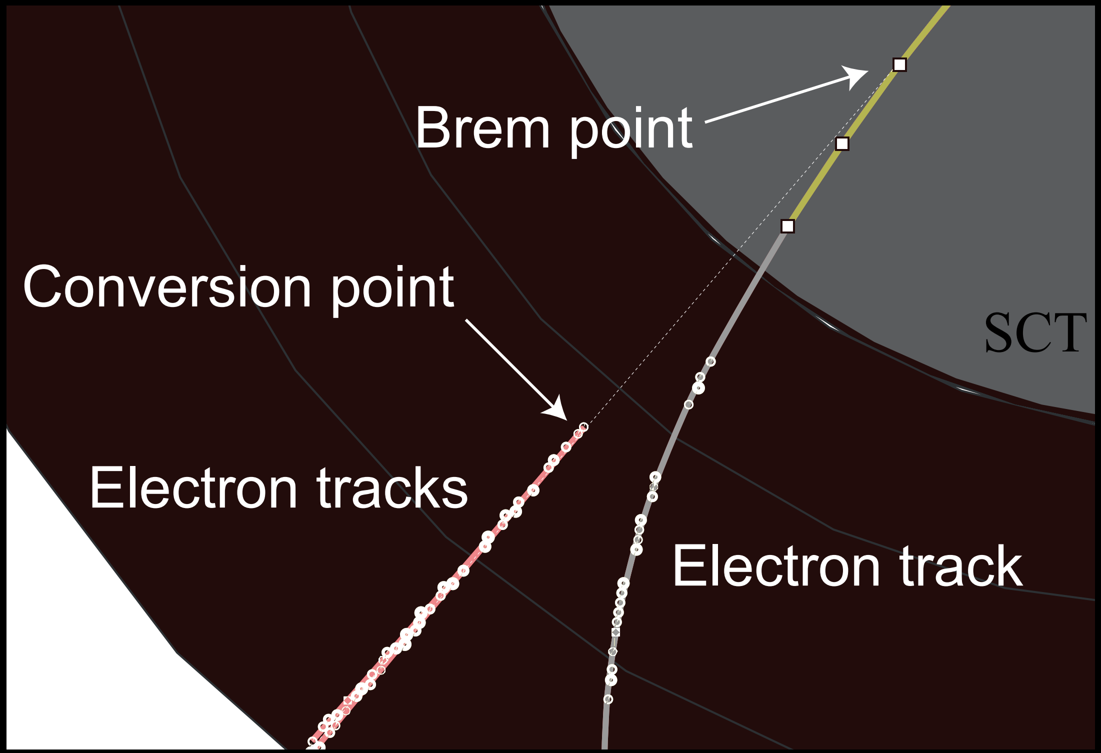
standalone TRT
- ➡ unused TRT segments

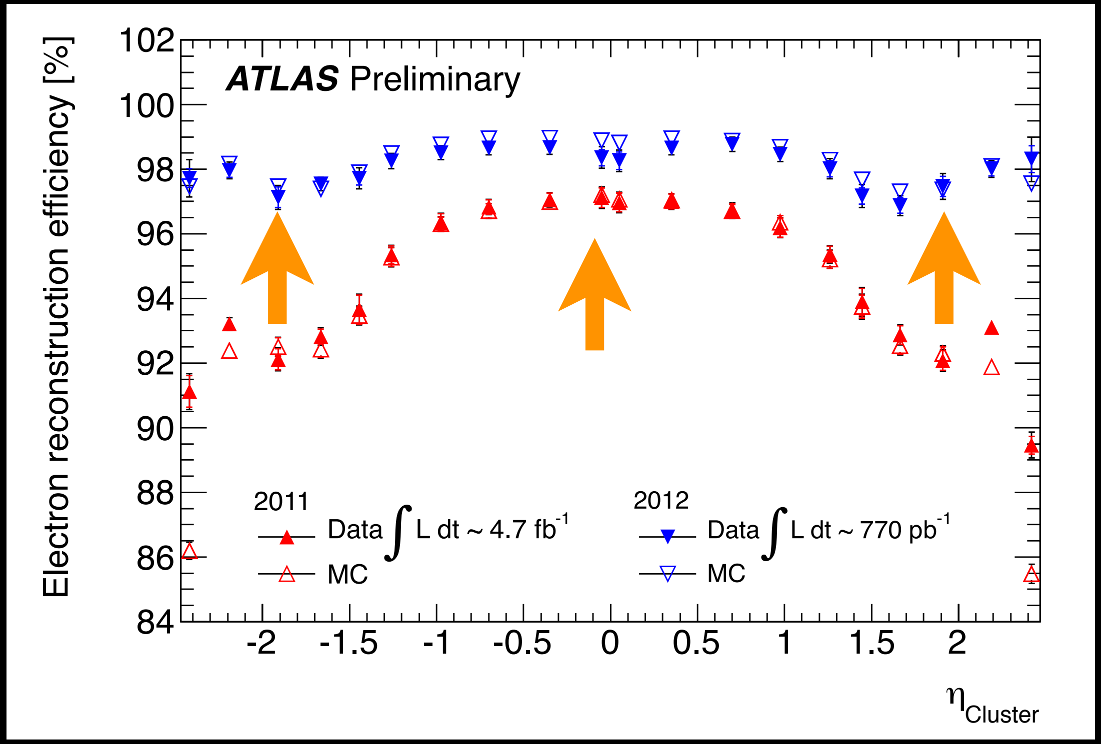
vertexing
- ➡ primary vertexing
- ➡ conversion and V0 search

Tracking with Electron Brem. Recovery

for completeness

- strategy for brem. recovery
  - ➡ restrict recovery to regions pointing to electromagnetic clusters (RoI)
  - ➡ pattern: allow for large energy loss in combinatorial Kalman filter
    - adjust noise term for electrons
  - ➡ global-χ^2 fitter allows for brem. point
  - ➡ adapt ambiguity processing (etc.) to ensure e.g. b-tagging is not affected
  - ➡ use full fledged Gaussian-Sum Filter in electron identification code





- tracking update deployed in 2012
  - ➡ improvements especially at low p_T (< 15 GeV)
    - limiting factor for $H \rightarrow ZZ^* \rightarrow 4e$
  - ➡ significant efficiency gain for Higgs discovery

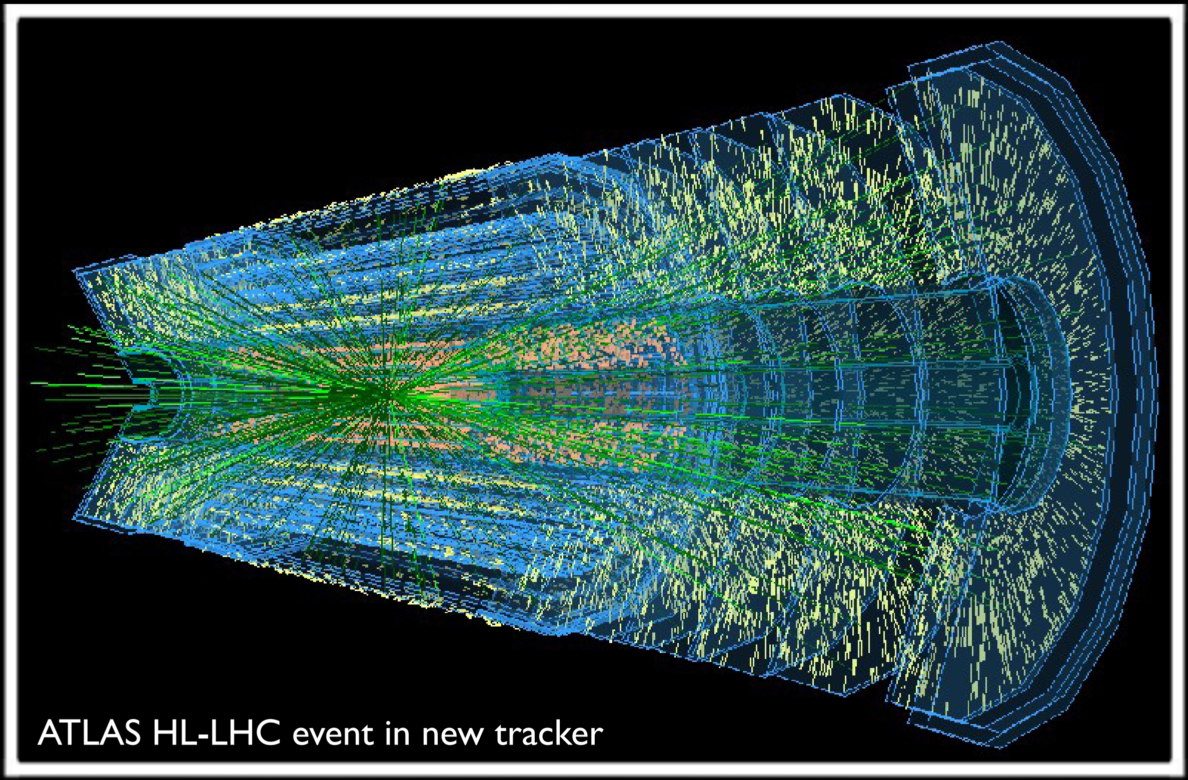
Let's Summarize...

- I introduced the reconstruction in a nutshell and why tracking is important for HEP computing

- I discussed briefly the principles of semiconductor trackers and drift tubes

- then we went over concepts and techniques for track extrapolation, fitting and finding

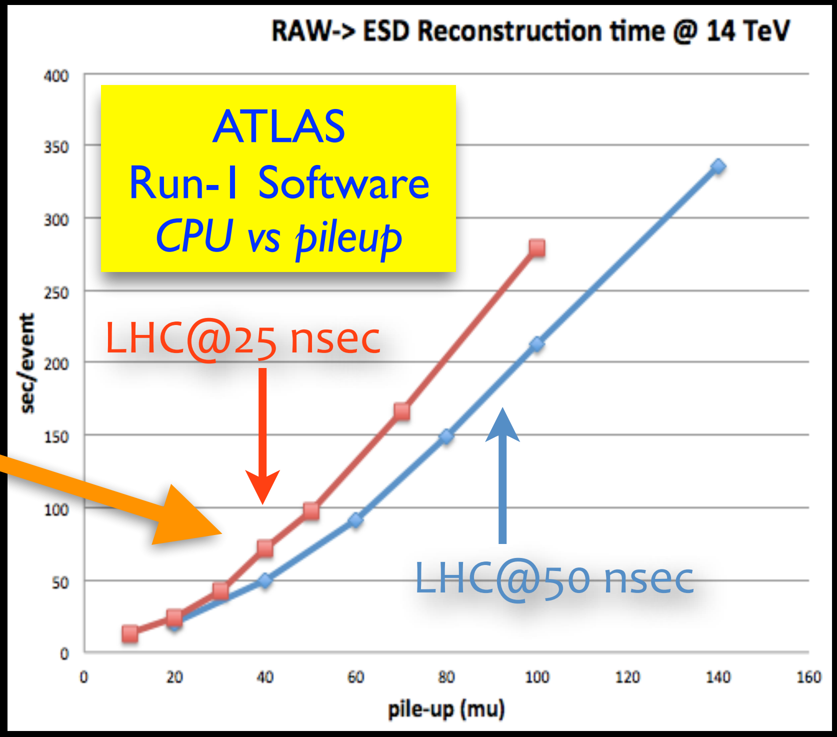
- and finally we saw how to put things together to implement the ATLAS Track Reconstruction



Bonus Slides...

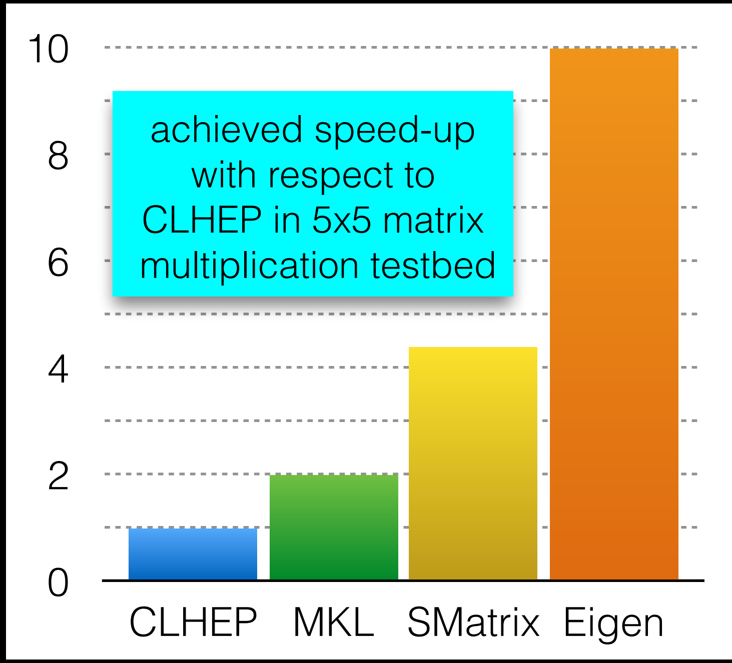
LS-1 Tracking Upgrades

...so what did we do about this so far ?



Tracking Developments towards Run-2

- ATLAS and CMS focus on technology and strategy to improve CURRENT algorithms
  - ➡ improve software technology, including:
    - simplify EDM design to be less OO ("hip" 10 years ago)
    - ATLAS migrated to Eigen - faster vector+matrix algebra (CMS was already using SMatrix)
    - vectorised trigonometric functions (CMS: VDT or ATLAS: intel math lib)
    - work on CPU hot spots (e.g. ATLAS replaced F90 by C++ for B-field service)
  - ➡ tune reconstruction strategy (very similar in ATLAS and CMS):
    - optimise iterative track finding strategy for 40 pileup
    - ATLAS modified track seeding to explore 4th Pixel layer
    - CMS added cluster-shape filter against out-of-time pileup
- hence, mix of SIMD and algorithm tuning
  - ➡ CMS made their tracking as well thread-safe



Tuning the Tracking Strategy

- optimal seeding strategy depends on level of pileup (ATLAS)
  - ➡ fraction of seeds to give a good track candidate:

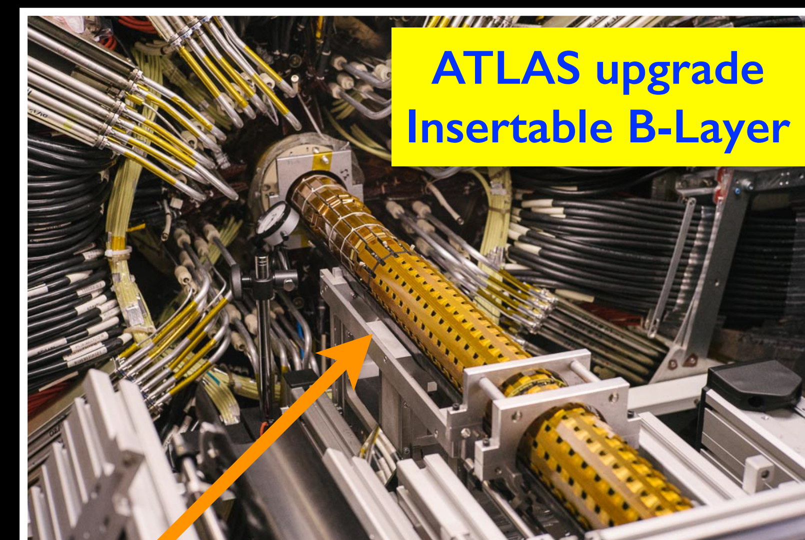
seed-triplets:
P = Pixel
S = Strips

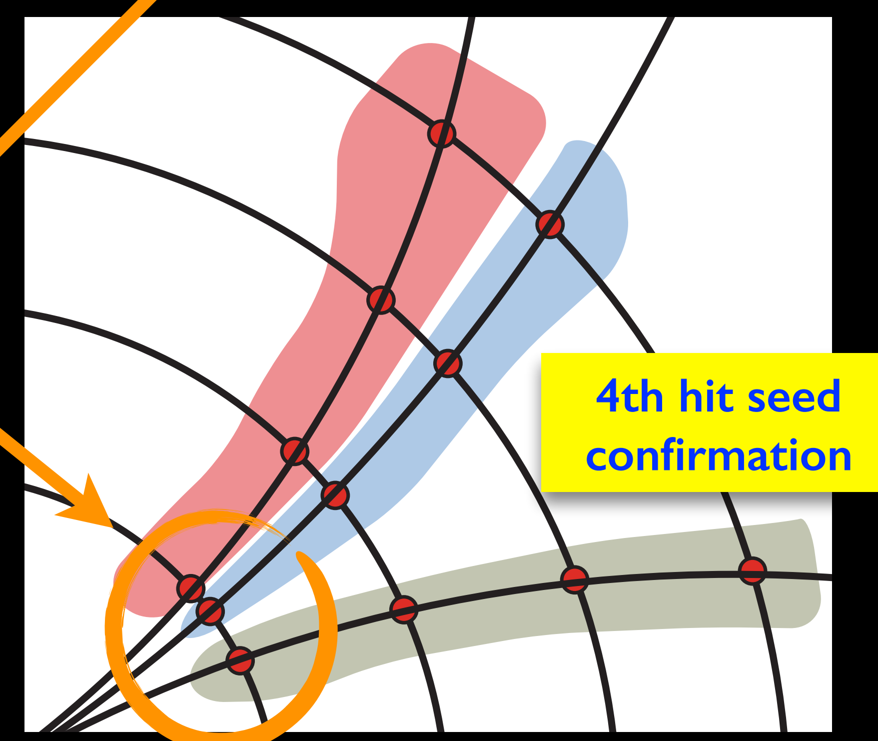
pileup	"PPP"	"PPS"	"PSS"	"SSS"
0	57%	26%	29%	66%
40	17%	6%	5%	35%

  - hence start with SSS at 40 pileup !
  - ➡ further increase good seed fraction using 4th hit

pileup	"PPP+1"	"PPS+1"	"PSS+1"	"SSS+1"
0	79%	53%	52%	86%
40	39%	8%	16%	70%

  - takes benefit from new Insertable B-Layer (IBL)





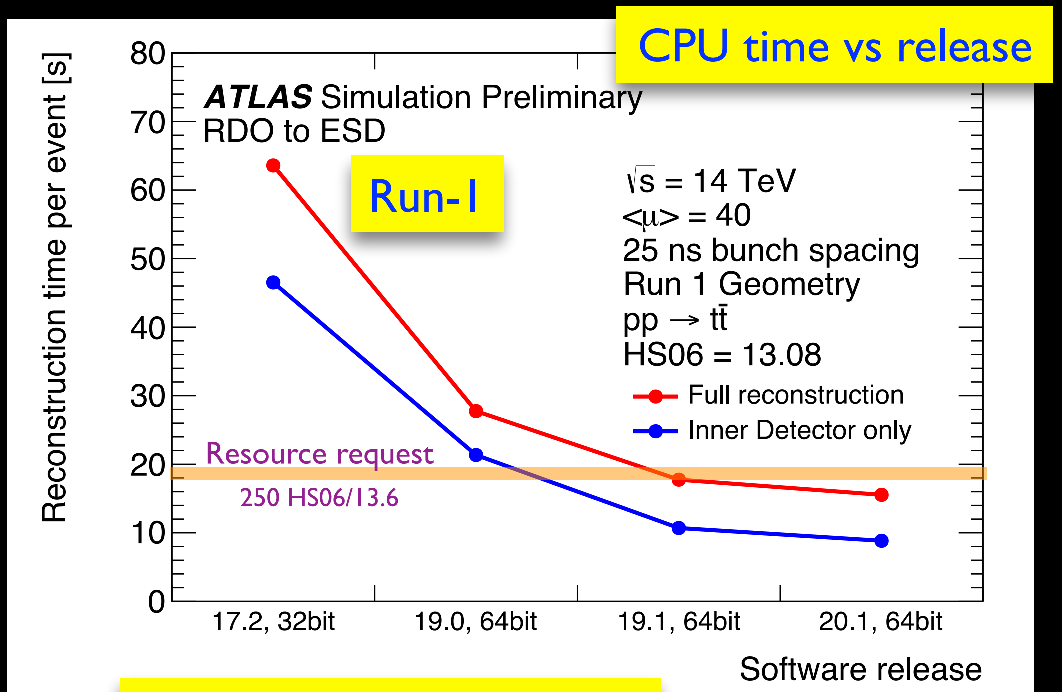
- final ATLAS Run-2 seeding strategy
  - ➡ significant speedup at 40 pileup (and 25 *ns*)

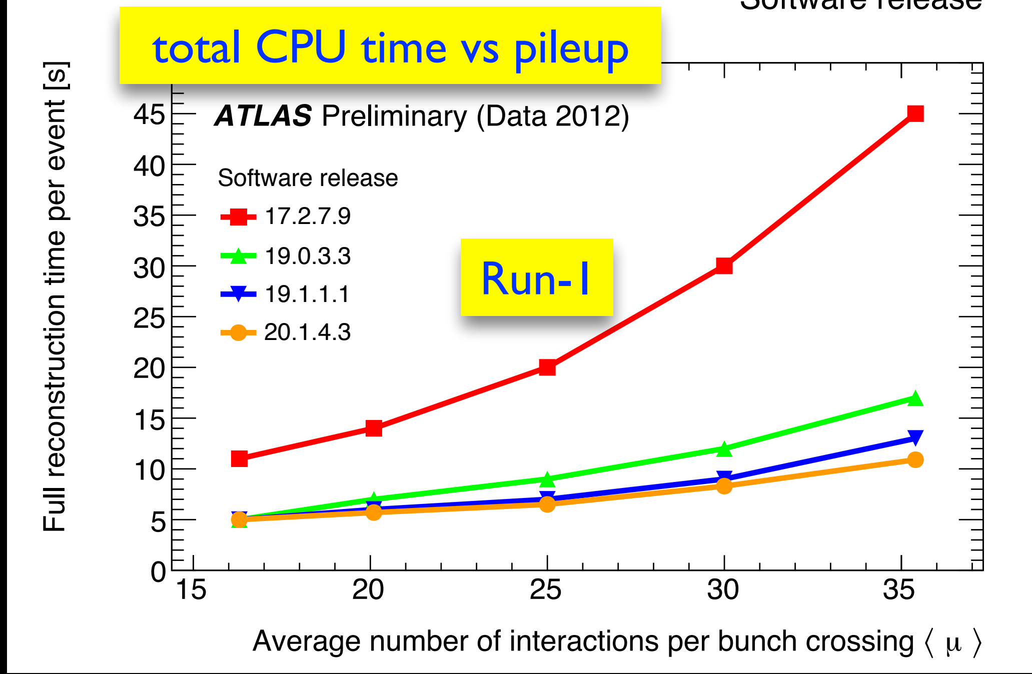
seeding	efficiency	CPU*
"Run-1"	94.0%	9.5 sec
"Run-2"	94.2%	4.7 sec

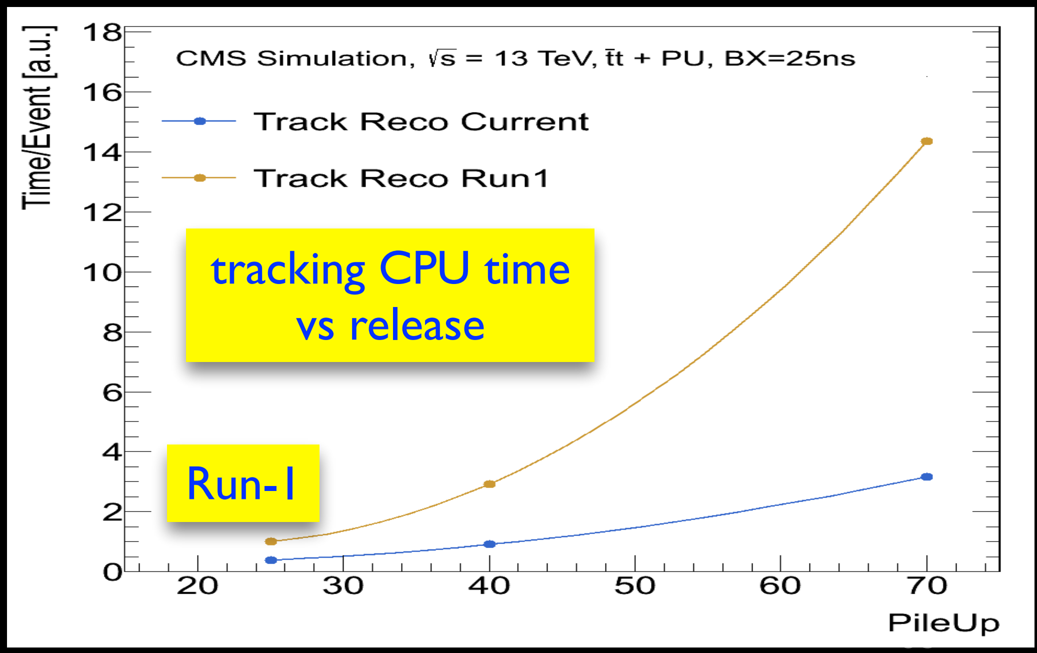
*on local machine

CPU for Reconstruction

- sum of tracking and general software improvements
  - ➡ improved software technology, including:
    - tracking related improvements
    - new 64 bit compilers, new tcmalloc
  - ➡ tune reconstruction strategy (very similar in ATLAS and CMS)
    - optimise track finding strategy for 40 pileup
    - faster versions of things like FastJet, ...
    - addressing other CPU hot spots in reconstruction



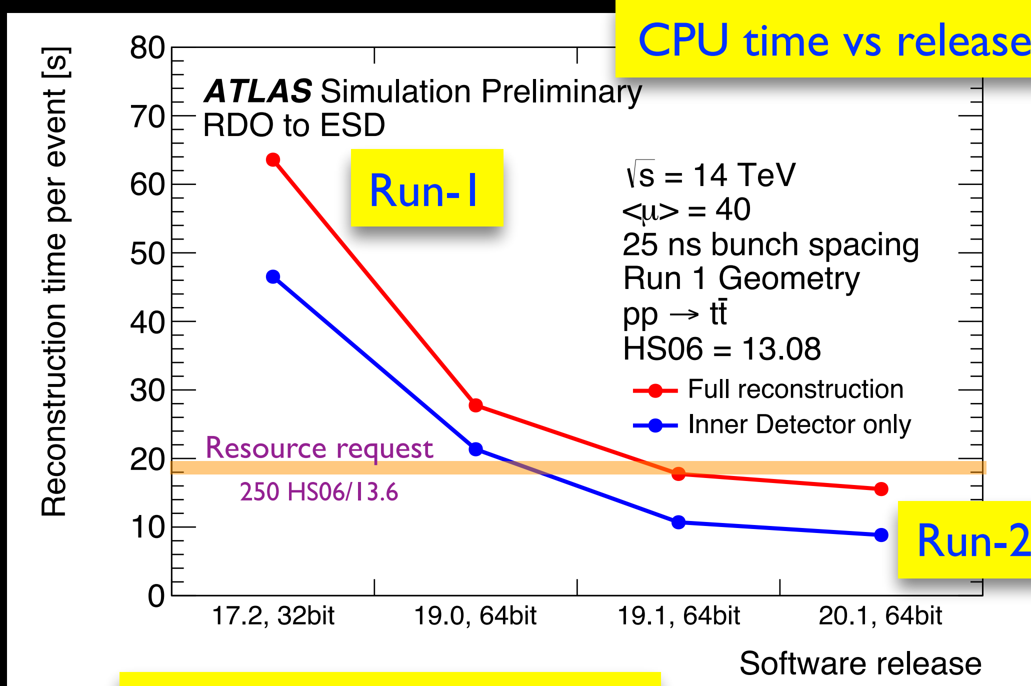


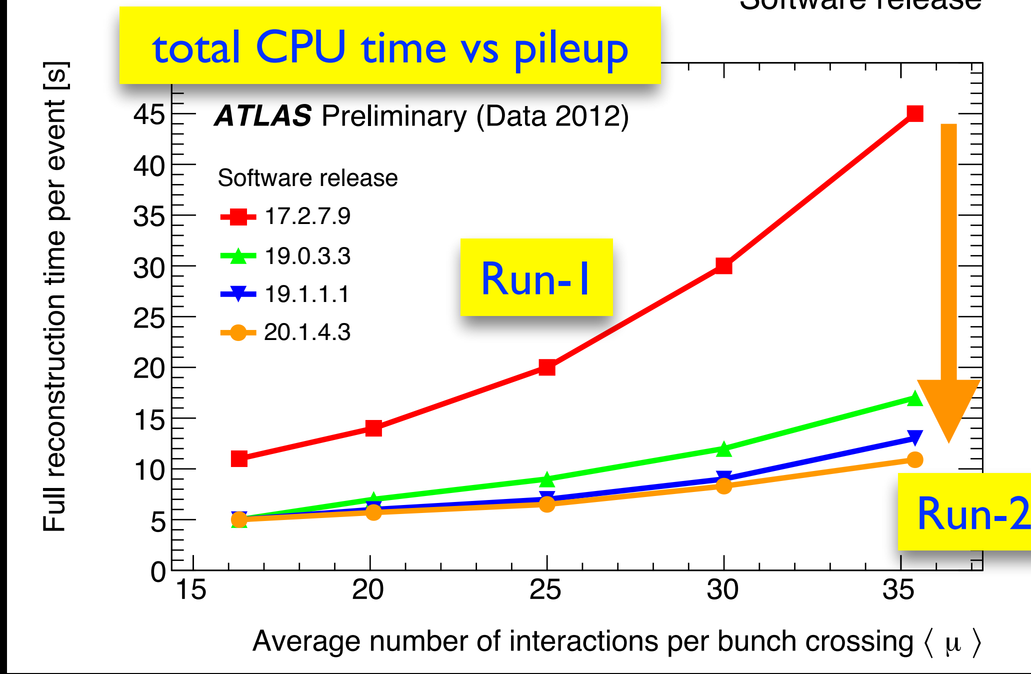


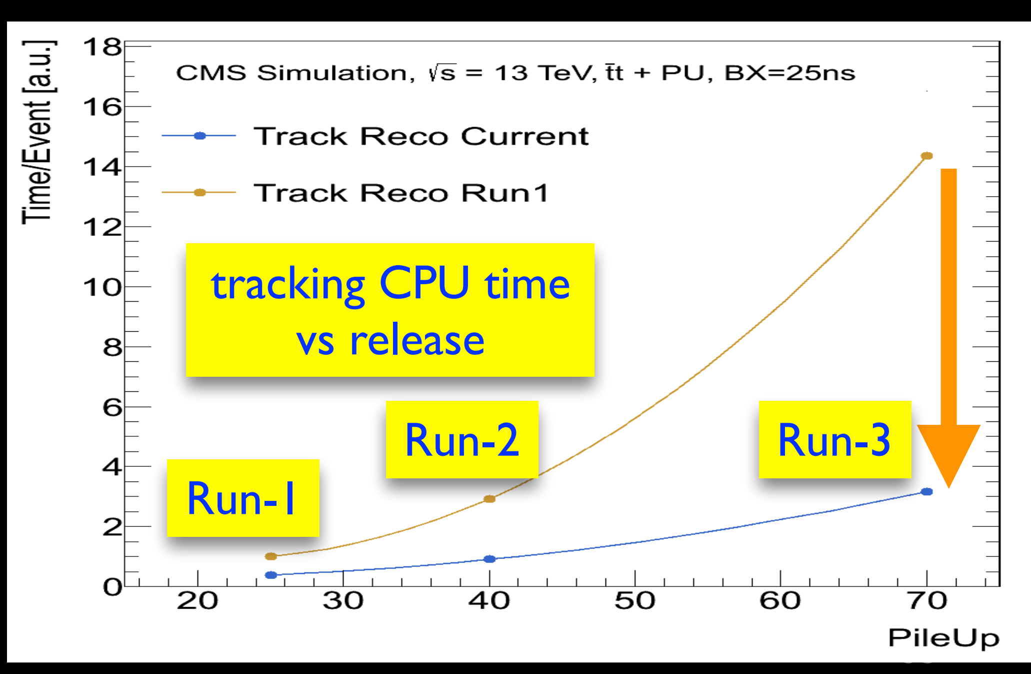
CPU for Reconstruction

- sum of tracking and general software improvements
  - ➡ improved software technology, including:
    - tracking related improvements
    - new 64 bit compilers, new tcmalloc
  - ➡ tune reconstruction strategy (very similar in ATLAS and CMS)
    - optimise track finding strategy for 40 pileup
    - faster versions of things like FastJet, ...
    - addressing other CPU hot spots in reconstruction

- huge gains achieved !
  - ➡ ATLAS reports overall factor > 4 in CPU time
    - touched >1000 packages for factor 5 in tracking
  - ➡ CMS reports overall factor > 2 in CPU time
    - on top of their 2011/12 improvements
    - as well dominated by tracking improvements
  - ➡ both experiments within 1 *kHz* Tier-0 budget
    - required to keep single lepton triggers

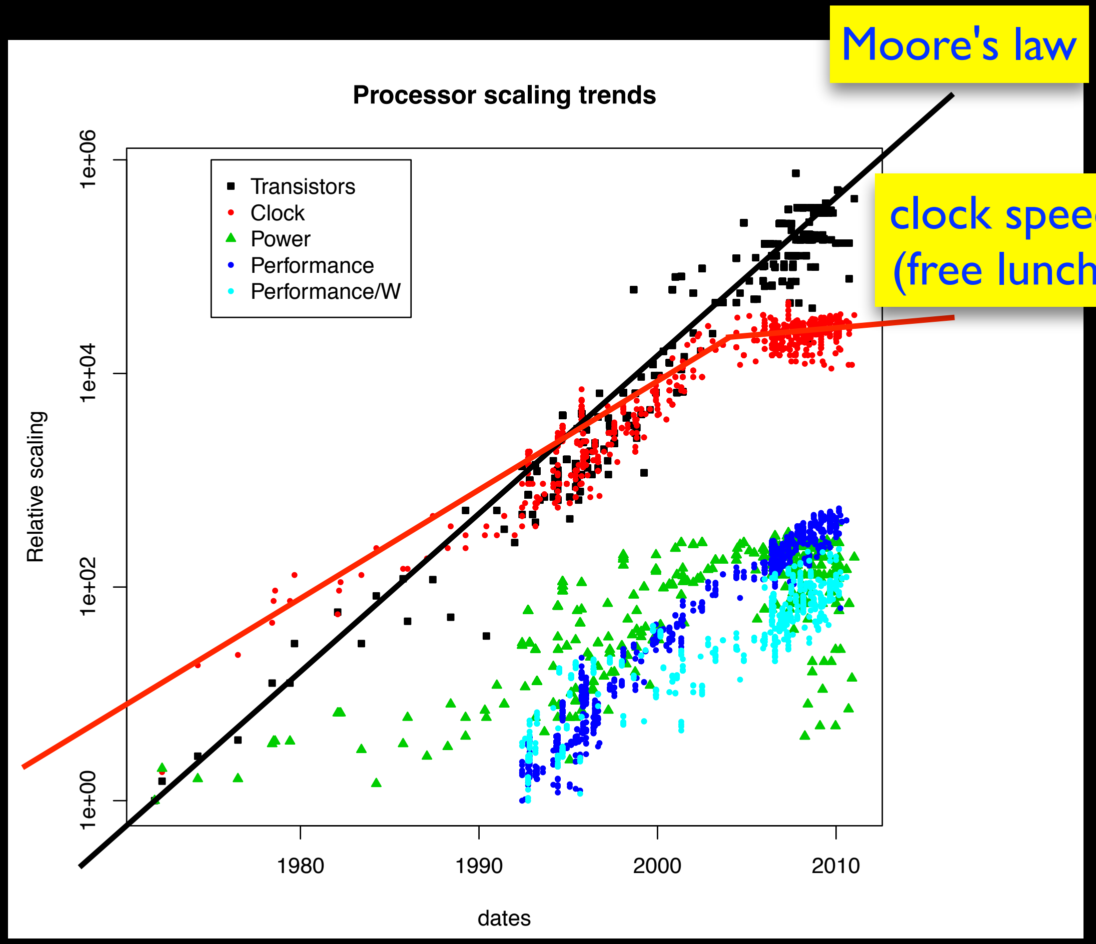




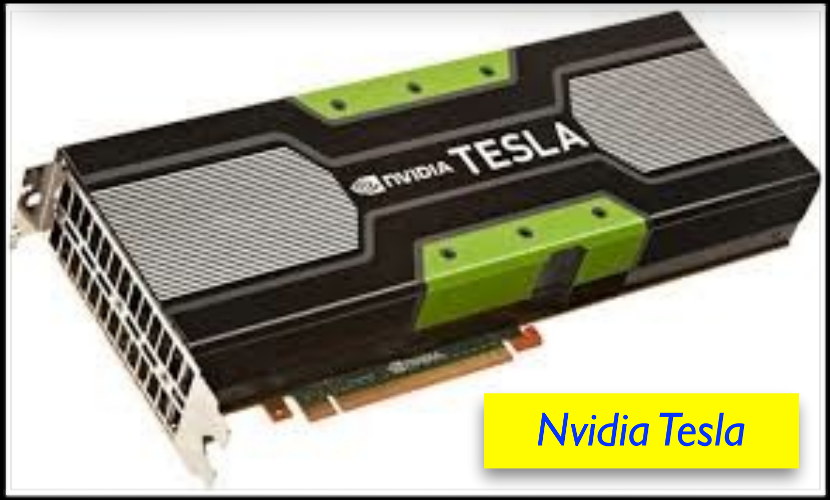


Technology Challenges

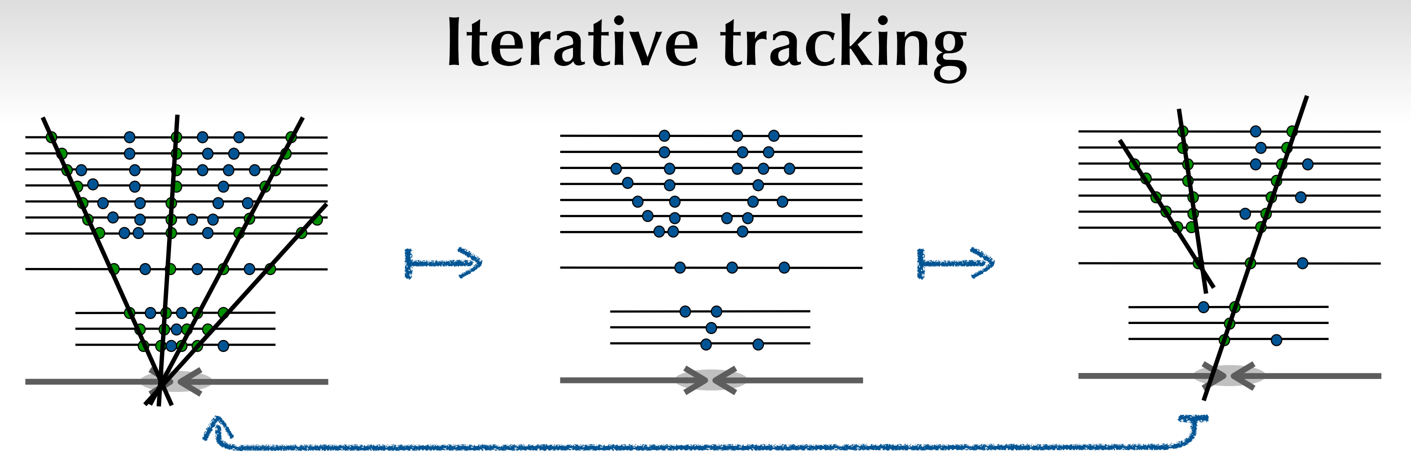
- Moore's law is still alive
  - ➡ number of transistors still doubles every 2 years
    - no free lunch, clock speed no longer increasing
  - ➡ lots of transistors looking for something to do:
    - vector registers
    - out of order execution
    - hyper threading
    - multiple cores
  - ➡ many-core processors, including GPGPUs
    - lots of cores with less memory
  - ➡ increase theoretical performance of processors
- challenge will be to adapt HEP software
  - ➡ hard to exploit theoretical processor performance
    - many of our algorithm strategies are sequential
  - ➡ need to parallelise applications (multi-threading)
    (GAUDI-HIVE and CMSSW multi-threading a step in this direction)
    - change memory model for objects, more vectorisation, ...



see G.Stewart, CHEP 2015



Massively parallel Tracking ?



- ATLAS/CMS tracking strategy is for early rejection
  - ➡ iterative tracking: avoid combinatorial overhead as much as possible !
    - early rejection requires strategic candidate processing and hit removal
  - ➡ not a heavily parallel approach, it is a SEQUENTIAL approach !
- implications for making it massively parallel ?
  - ➡ Armdahl's law at work:

$$\text{Time}_{||} = \text{Para} / \text{N} + \text{Seq}$$

  - ➡ iterative tracking: small parallel part Para, heavy on sequential Seq
    - hence, if we want to gain by a large N threads, we need to reduce Seq
- hence we need to re-think the algorithmic strategy
  - ➡ having concurrency in mind from the very start

Discussion ...