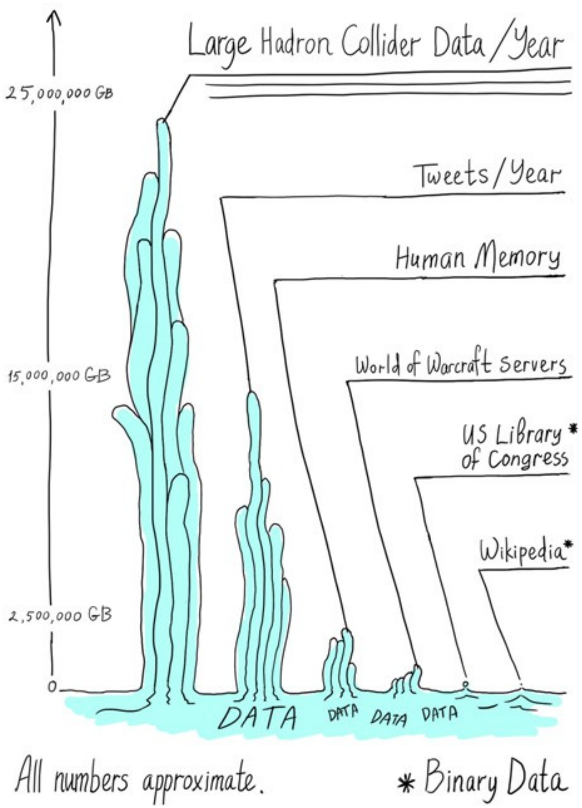
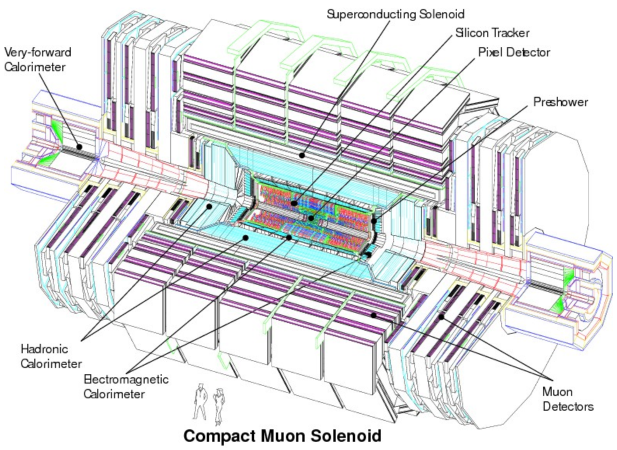
Challenges facing CMS data analysis in the 20's



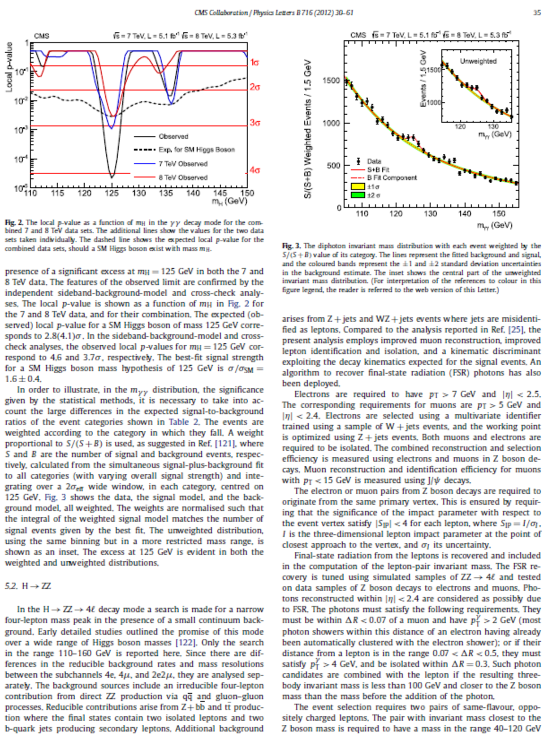
Andrea Giammanco (andrea.giammanco@cern.ch)
Centre for Cosmology, Particle Physics and Phenomenology
Louvain-la-Neuve, Belgium

How do we do that, in practice?



data

knowledge



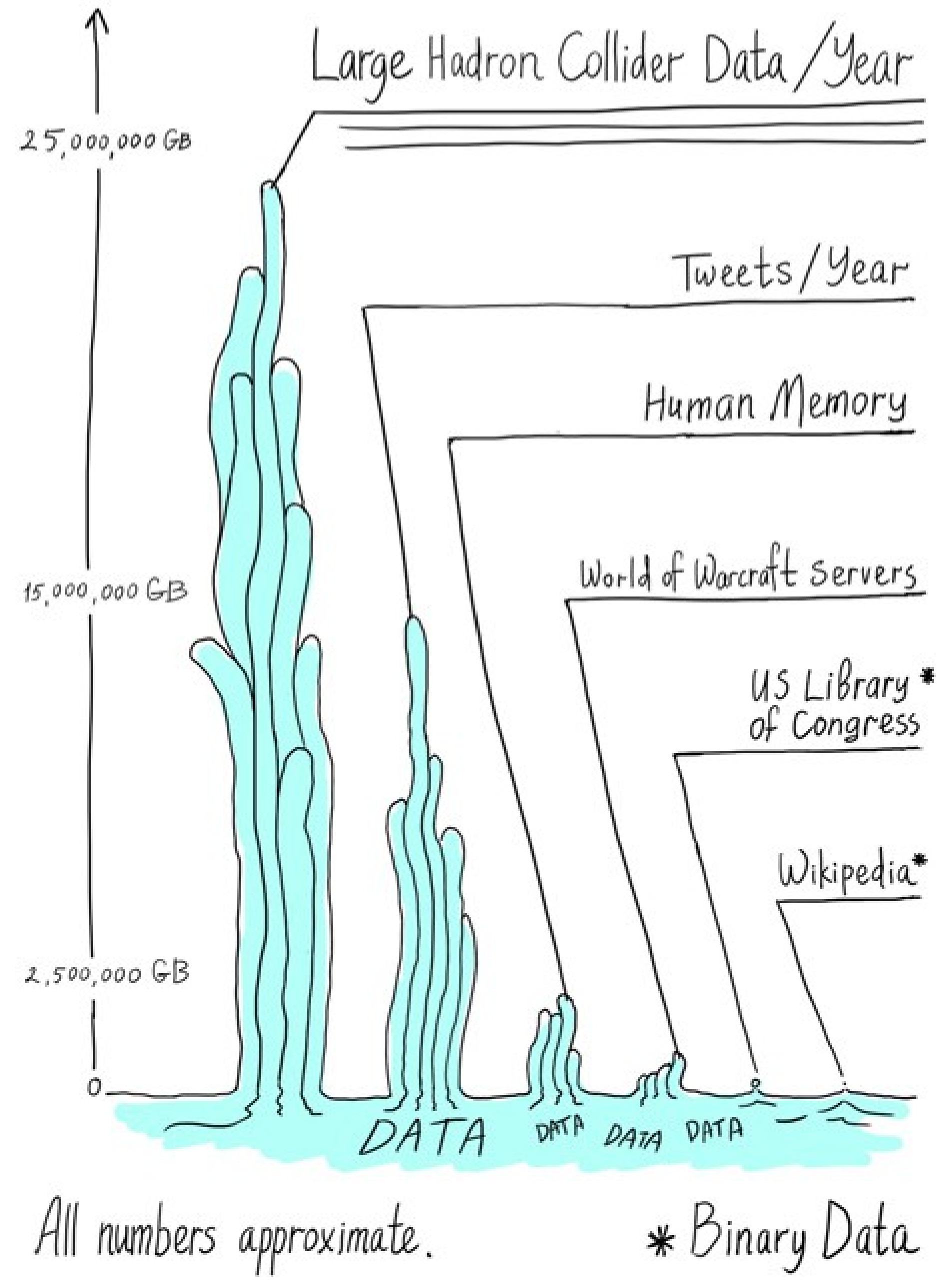
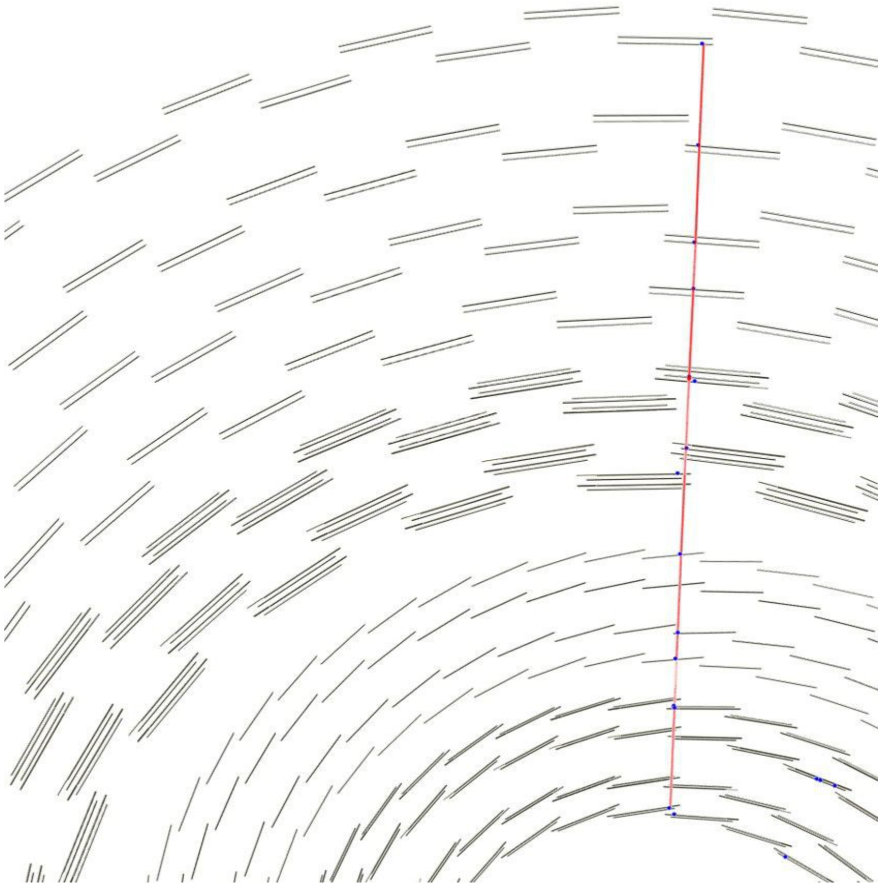


Image from http://www.symmetrymagazine.org/article/august-2012/particle-physics-tames-big-data

Example #1:
finding the tracks of particles

The CMS inner tracker

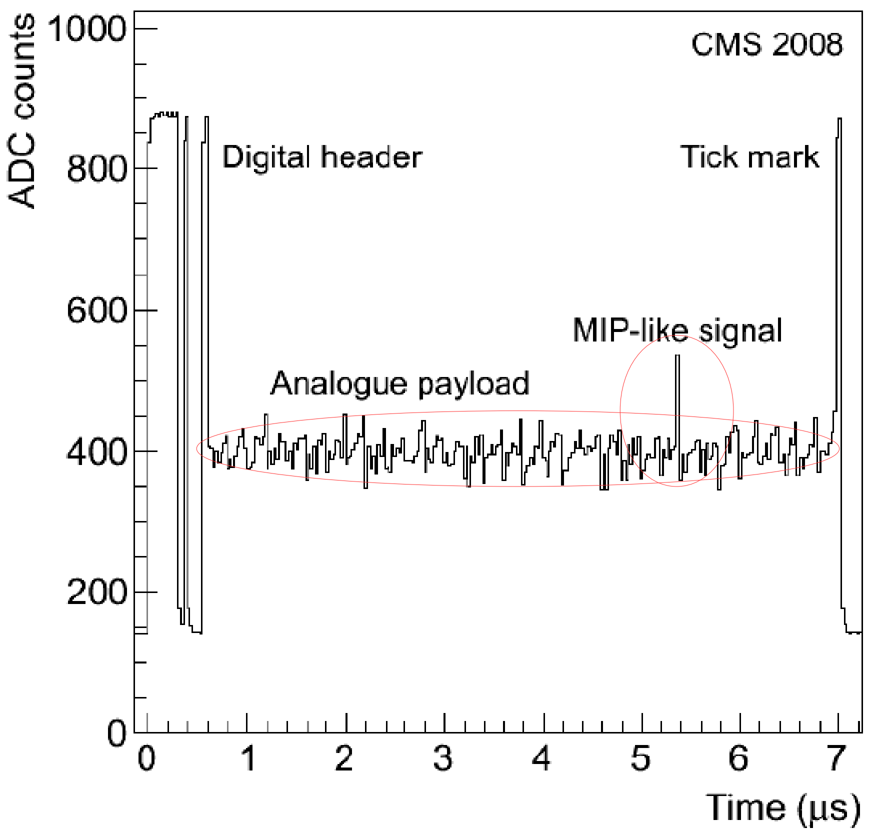
- Innermost part of CMS; a cylinder of 1.2 m radius (all CMS: 7.5 m)
- Electrically charged particles (and only them) give a signal each time they cross one of its layers
- Each layer is made of several modules, each module has hundreds of sensitive units (pixels or microstrips) with spatial resolution of O(0.1 mm)
- Its volume is only a fraction of all CMS, but it dominates the size of its raw data with its ~80 millions of sensitive units



Raw data from the tracker

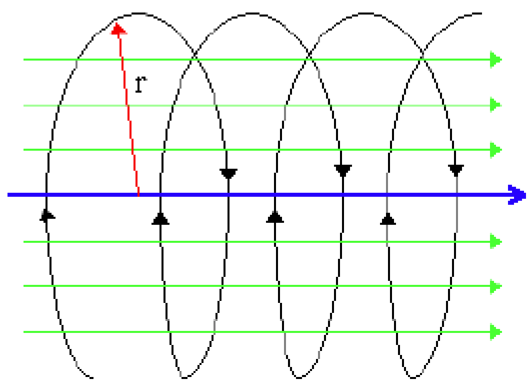
Example from one of the two technologies employed in the CMS tracker (microstrips):

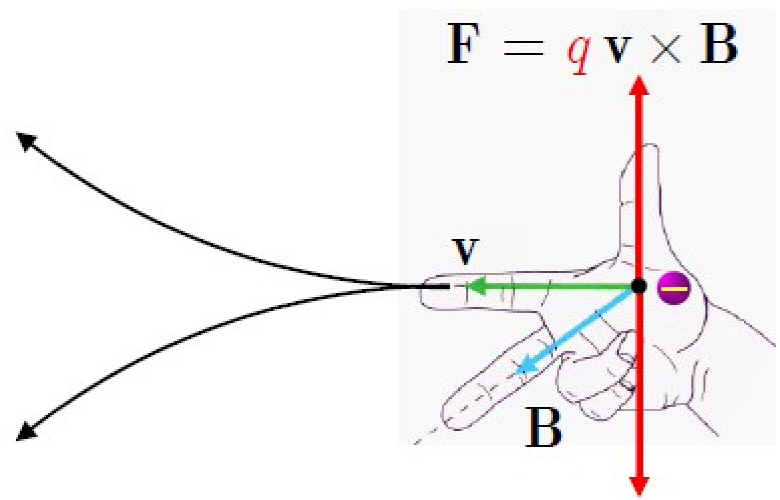
- A block of 128 microstrips is read-out by a single chip
- This chip sends as output a *data-frame* (see figure)
- Fluctuating part: electronic noise
- Passage of a particle gives a signal that sticks out of that noise: a *hit*
- From then on, we only process the *hits* and ignore other microstrips
- This is a case of *data reduction*



Data-frame; each bin between header and tick mark corresponds to the position of a strip

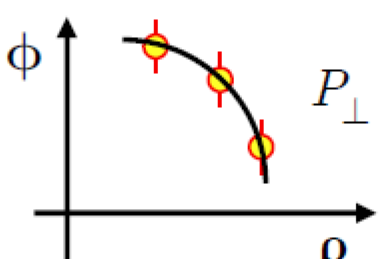
More data reduction: Tracking



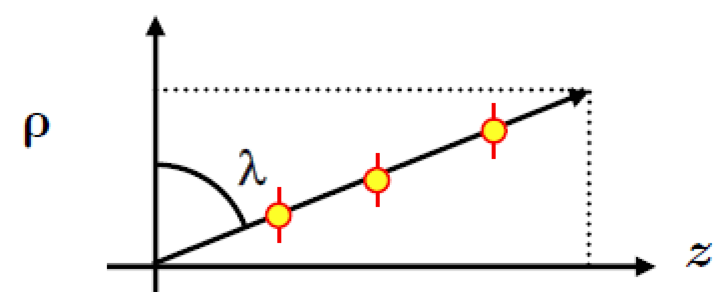


Solenoidal field along z: deflection in x-y (or $\rho-\phi$) plane

We sample the trajectory in a discrete number of crossings with the detector; from those crossings we must infer the trajectory

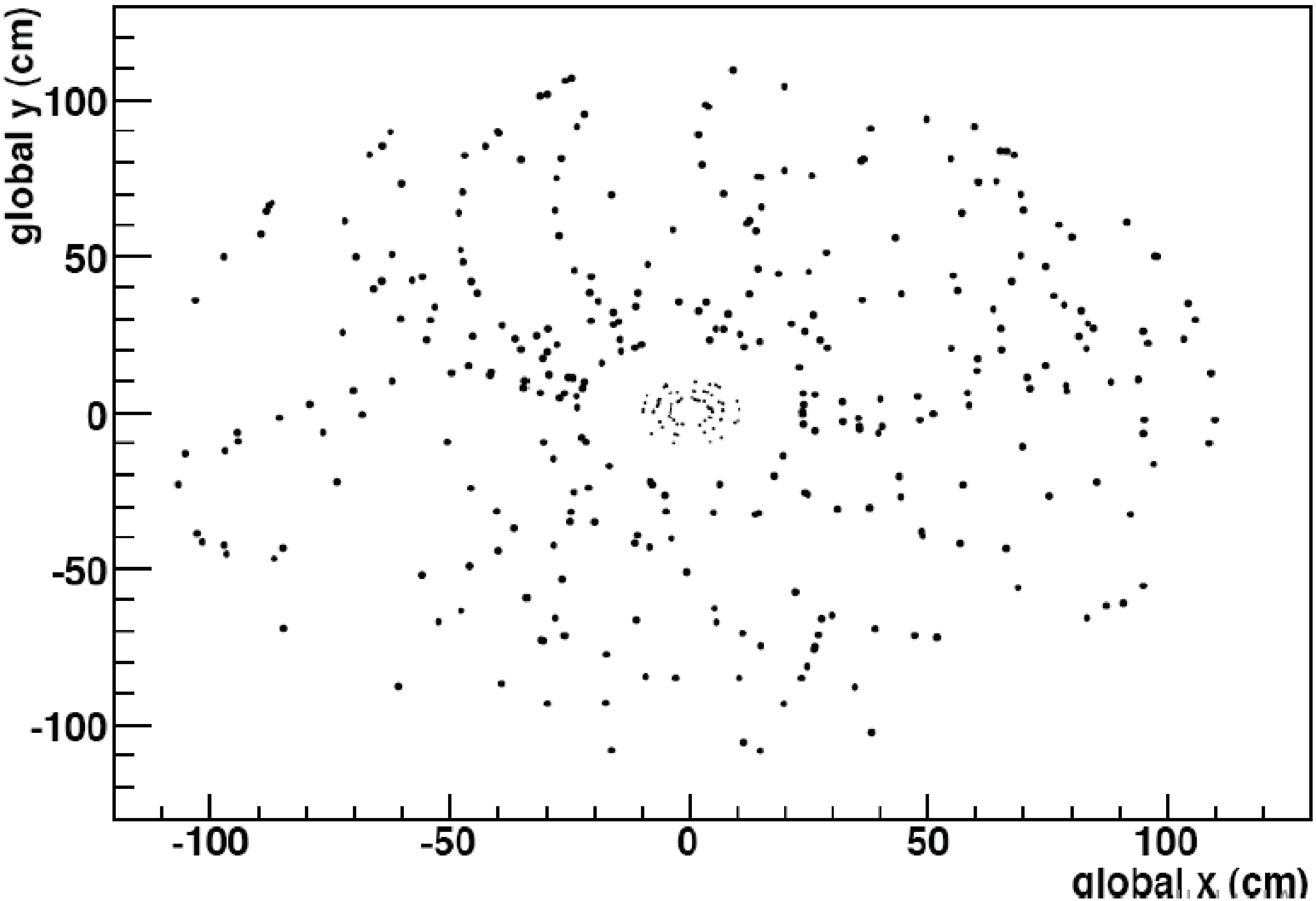


$$P_\perp = P \cos\lambda = 0.3BR$$



Find the track

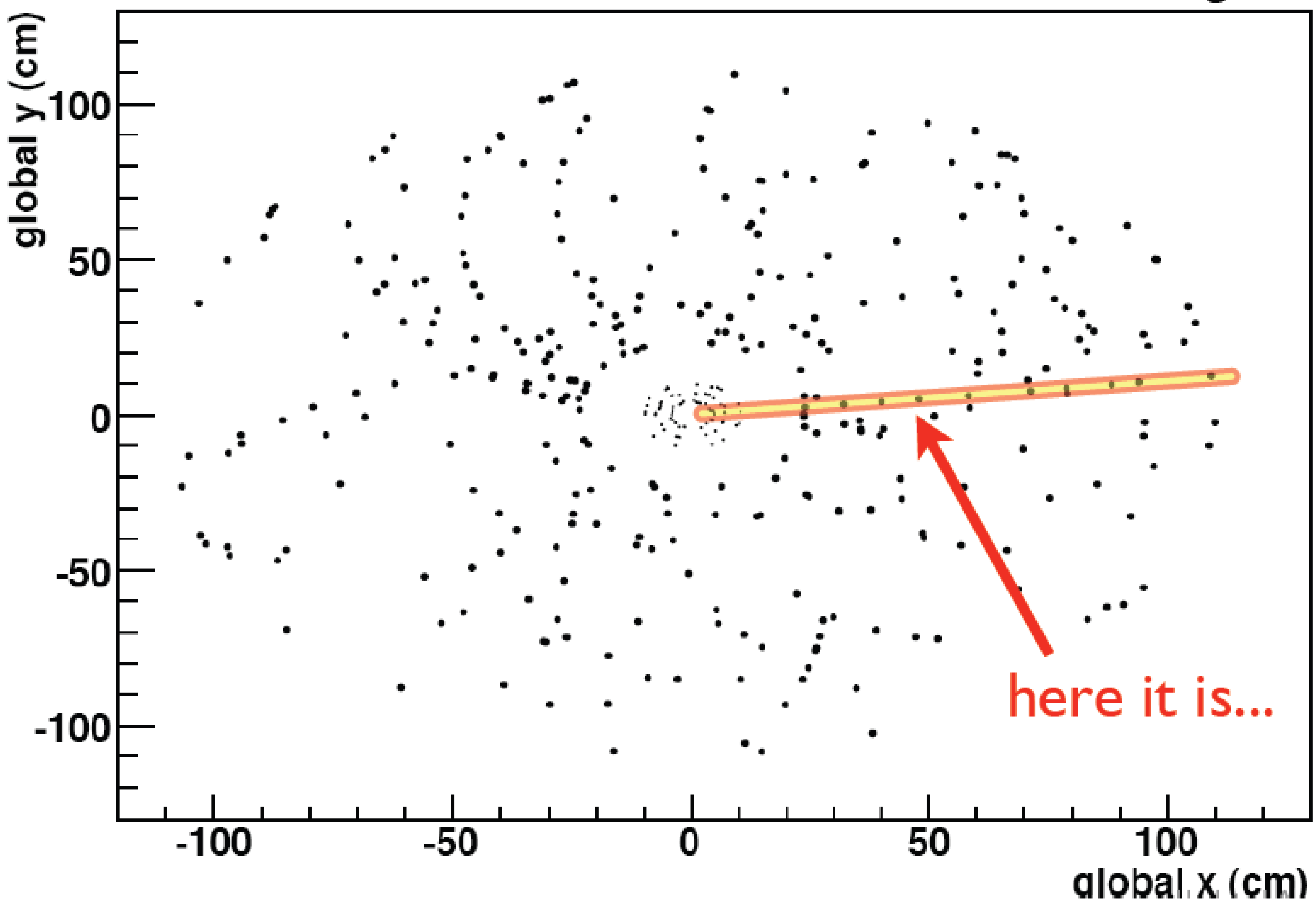
cf Aaron Dominguez



Where is the 50 GeV track? (Hint: it is very straight)

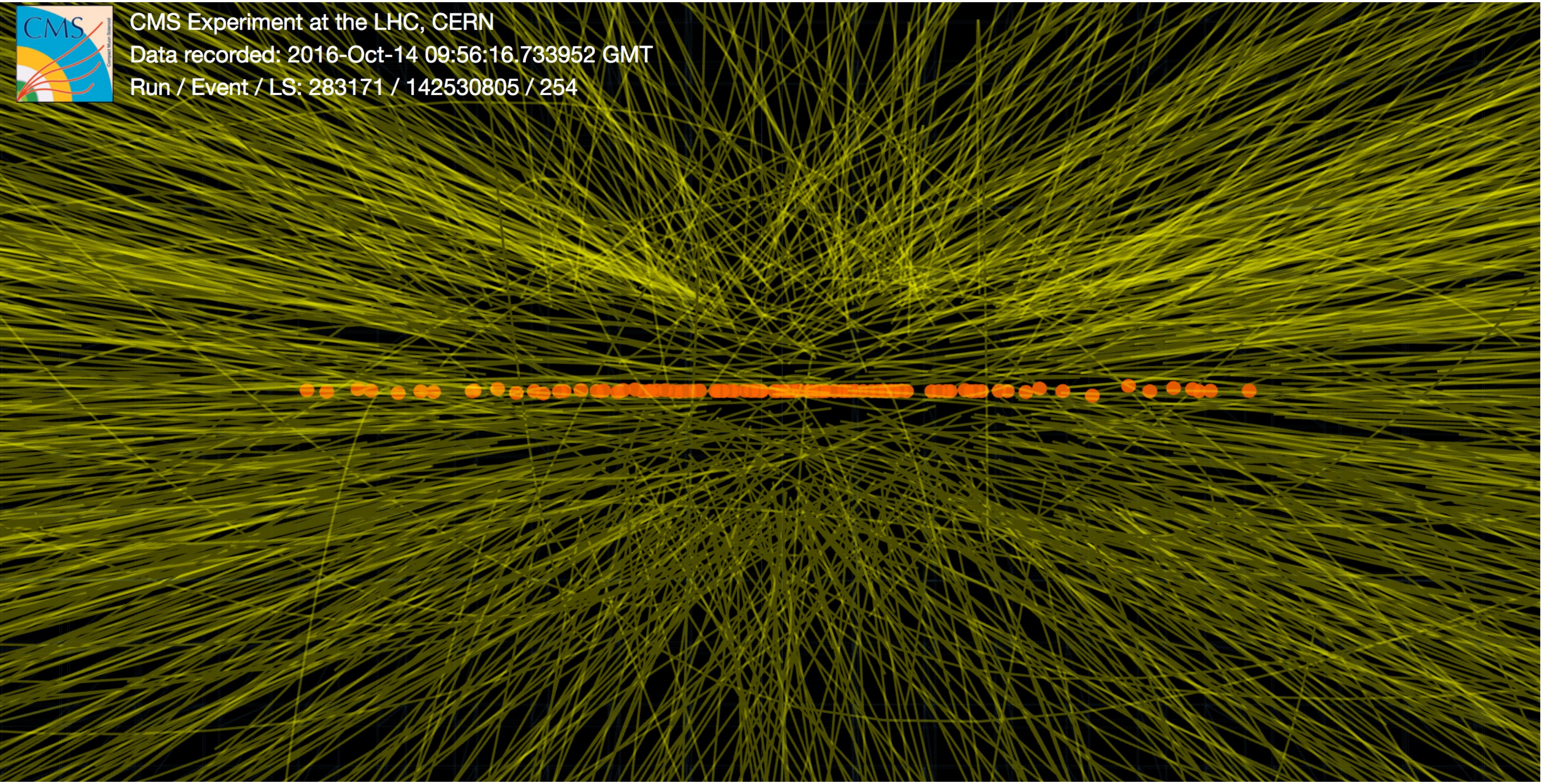
Find the track

cf Aaron Dominguez



These data are from Tevatron, a past accelerator operating at ~1/7 of LHC energy

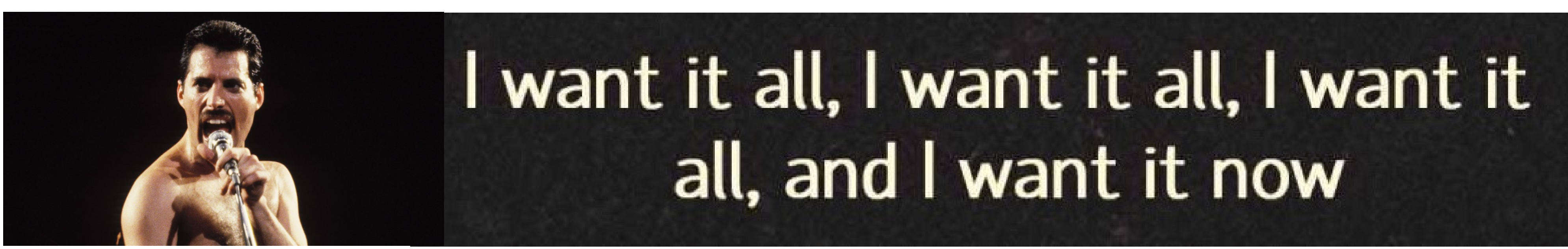
Tracking at LHC



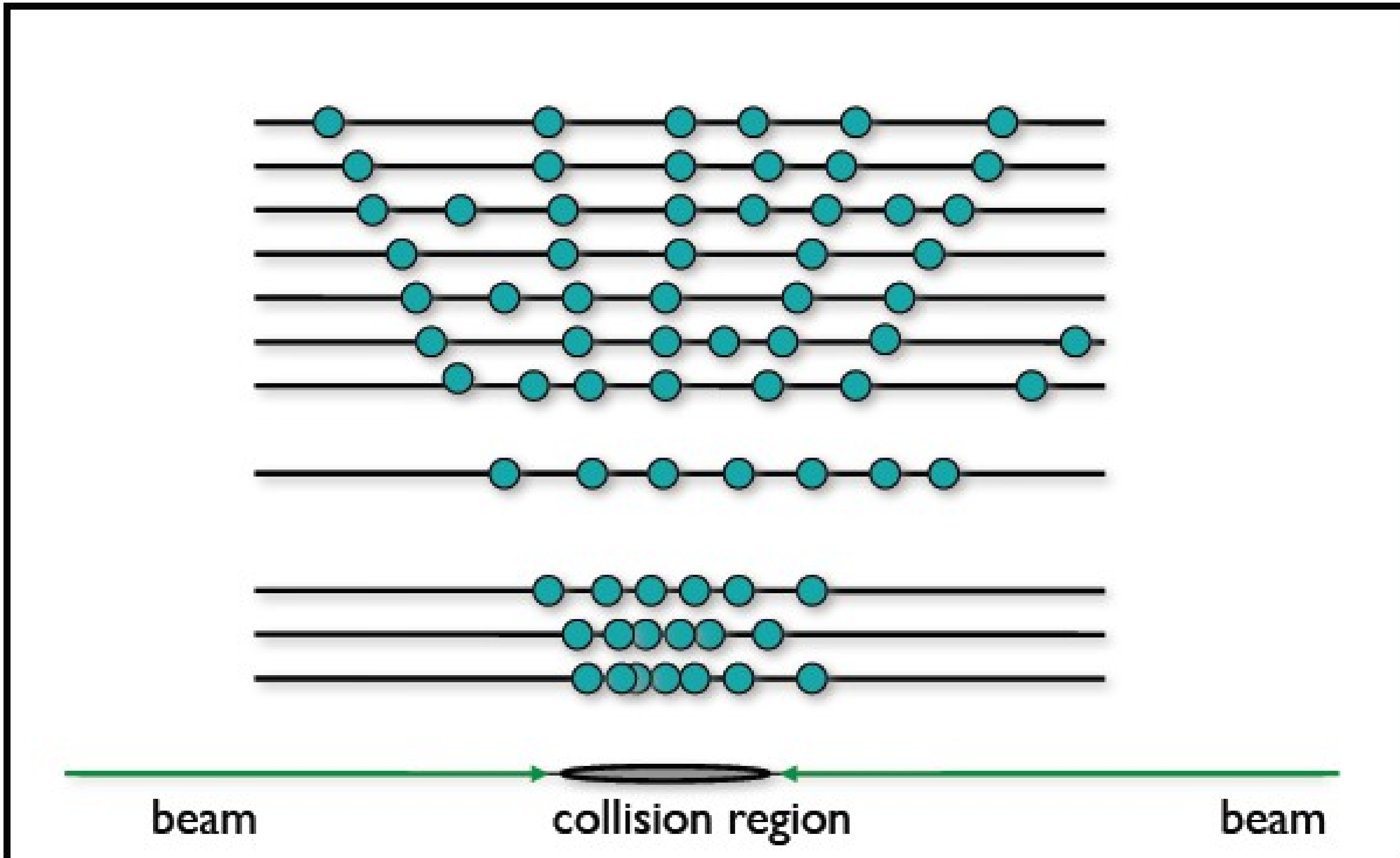
LHC achieves large intensities by very dense proton bunches (large number of protons, small volume) ⇒ several proton-proton interactions during each bunch crossing (*pileup*)

What we need

- We need track-finding to be *efficient*
  - Ideally, we would like to catch *all* true tracks
- We need the track sample to be very *pure*
  - Ideally, we would like *all* tracks that we reconstruct to be actual particles (and not fakes, i.e., wrong hit combinations)
- And it has to be *fast*
- To summarize:

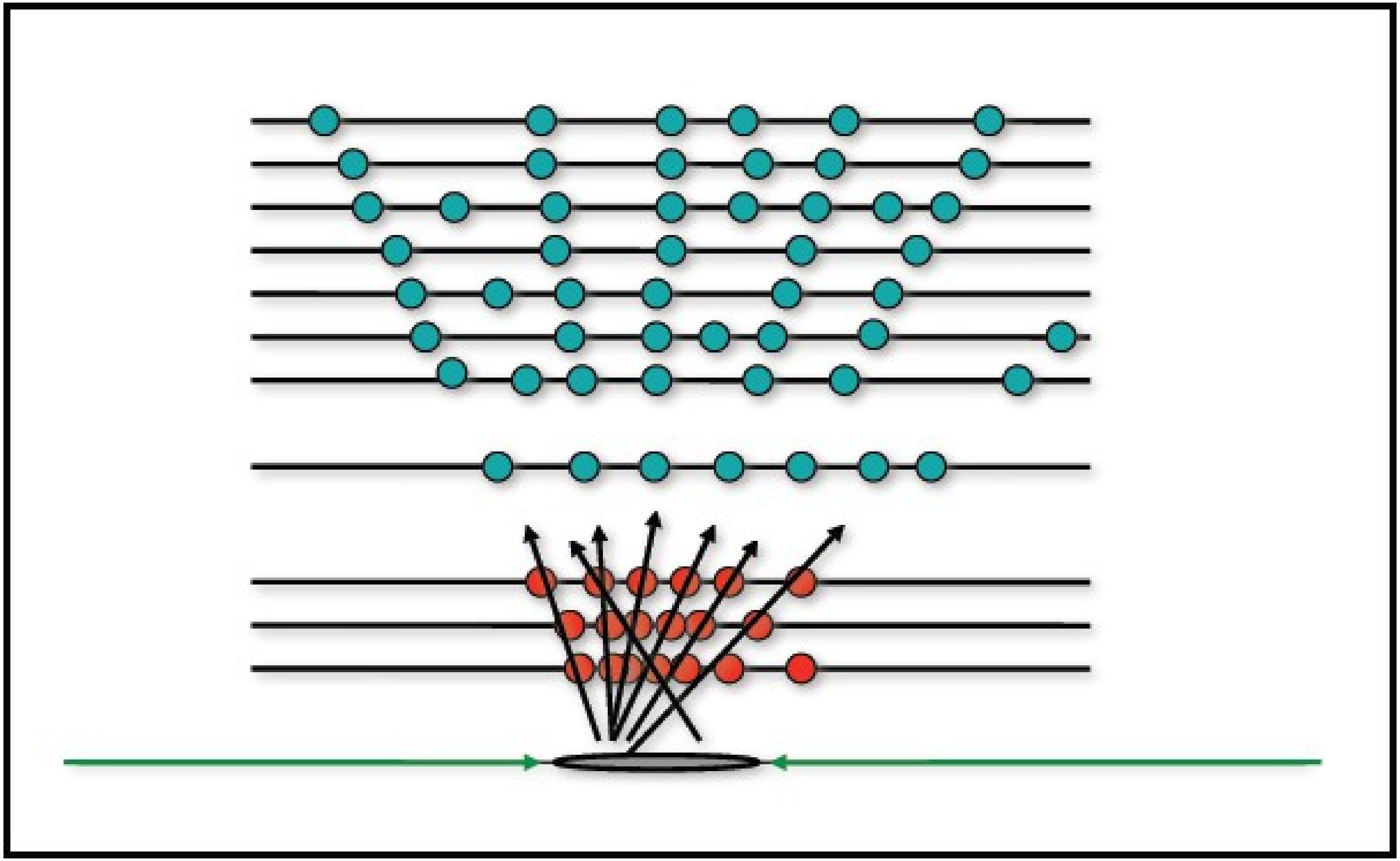


After local data reduction



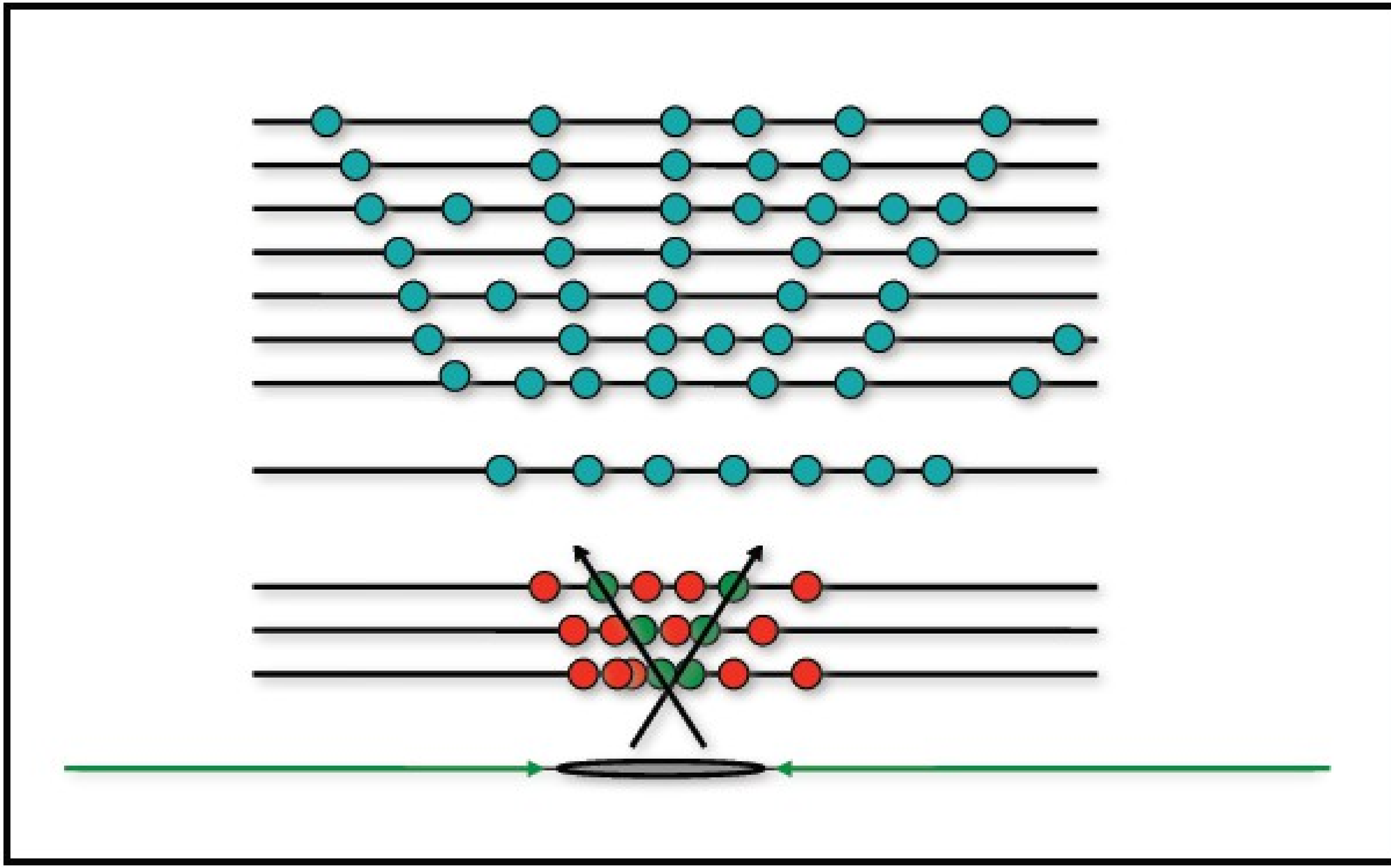
We start from a collection of hits, associated to a position and an uncertainty

Seeding



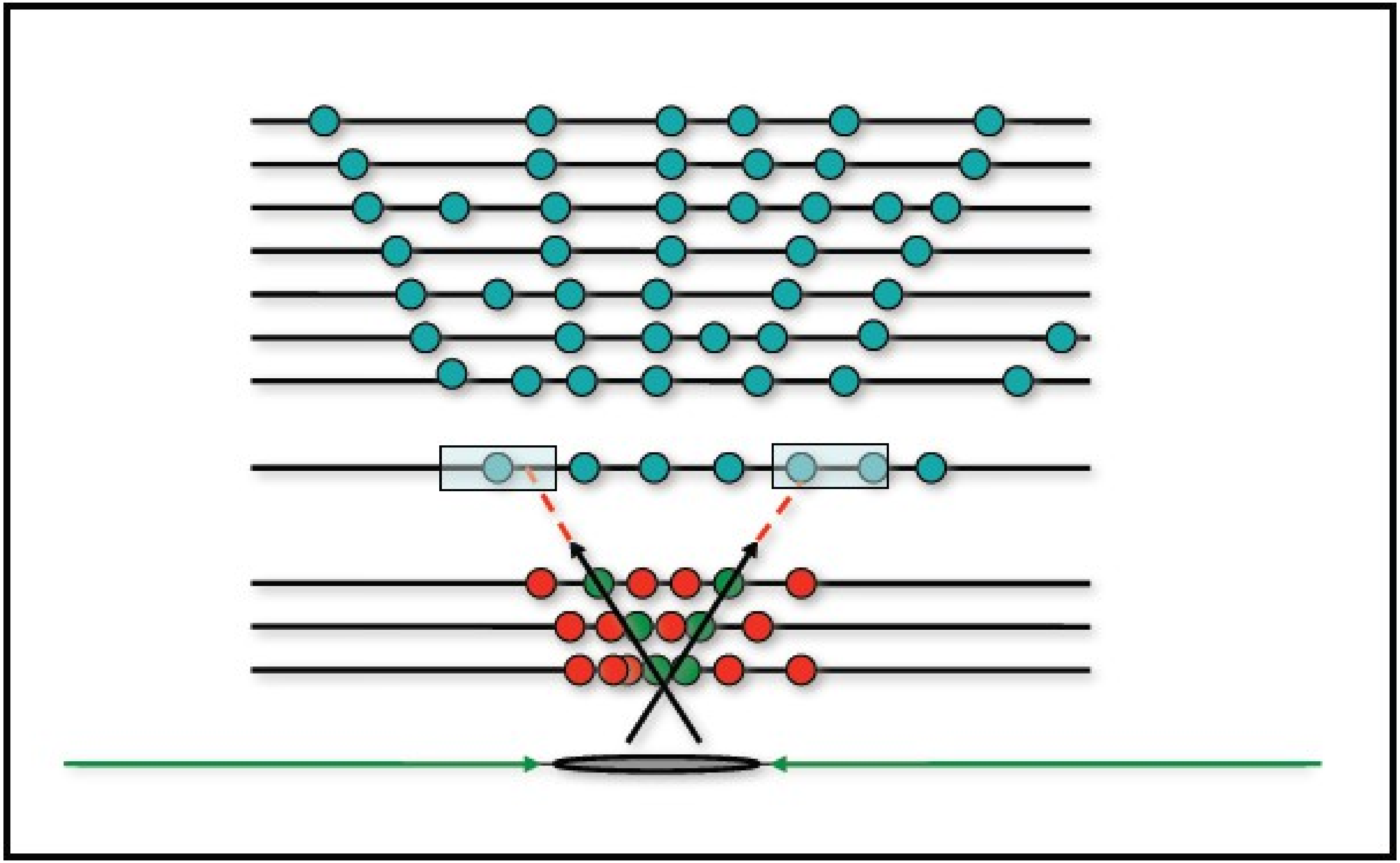
Fast fit to get initial trajectory, trying all combinations of hits in a small subset of layers

Trajectory building



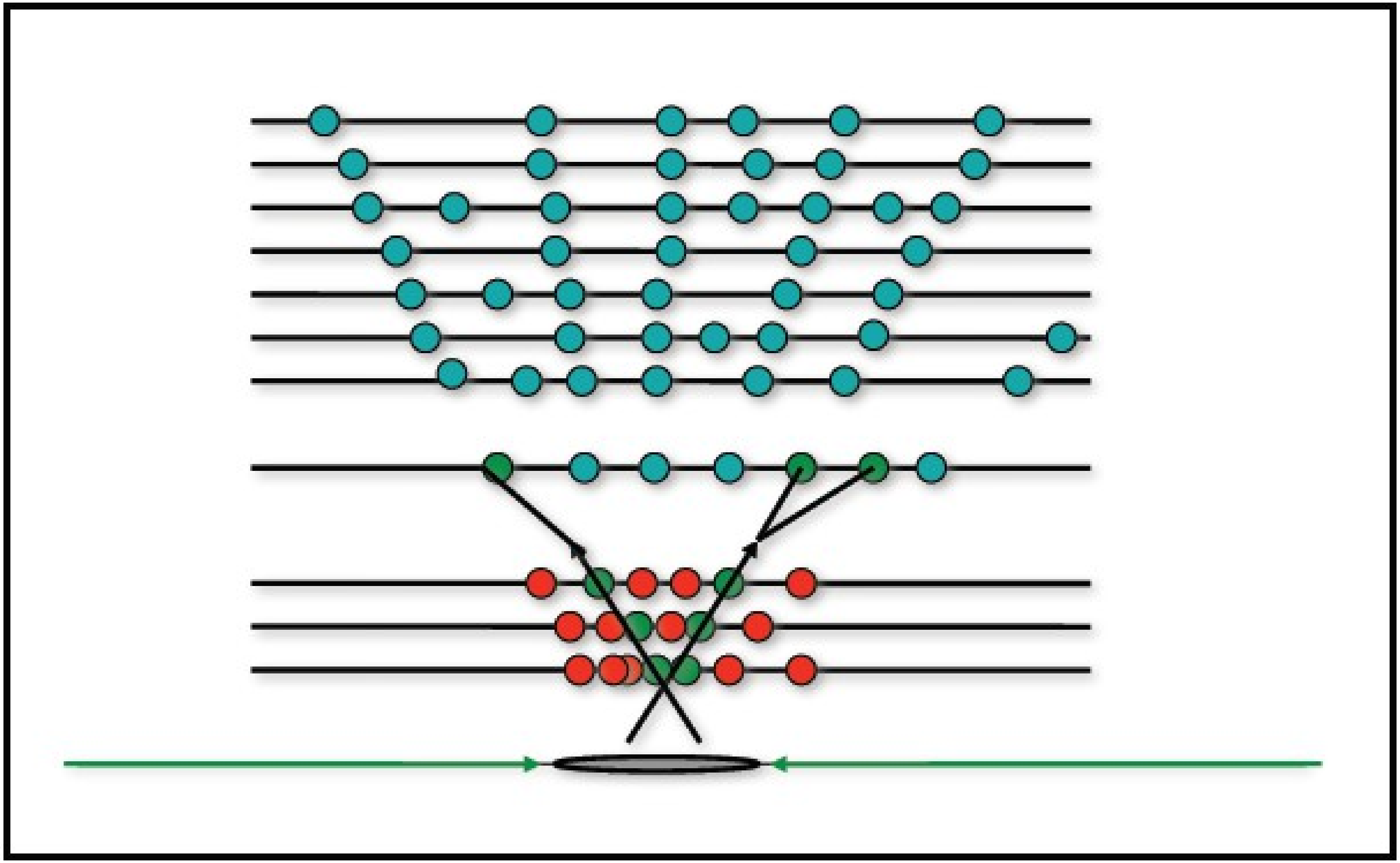
For illustration, let's consider these two seeds and let's see how trajectories are built from there.

Trajectory building

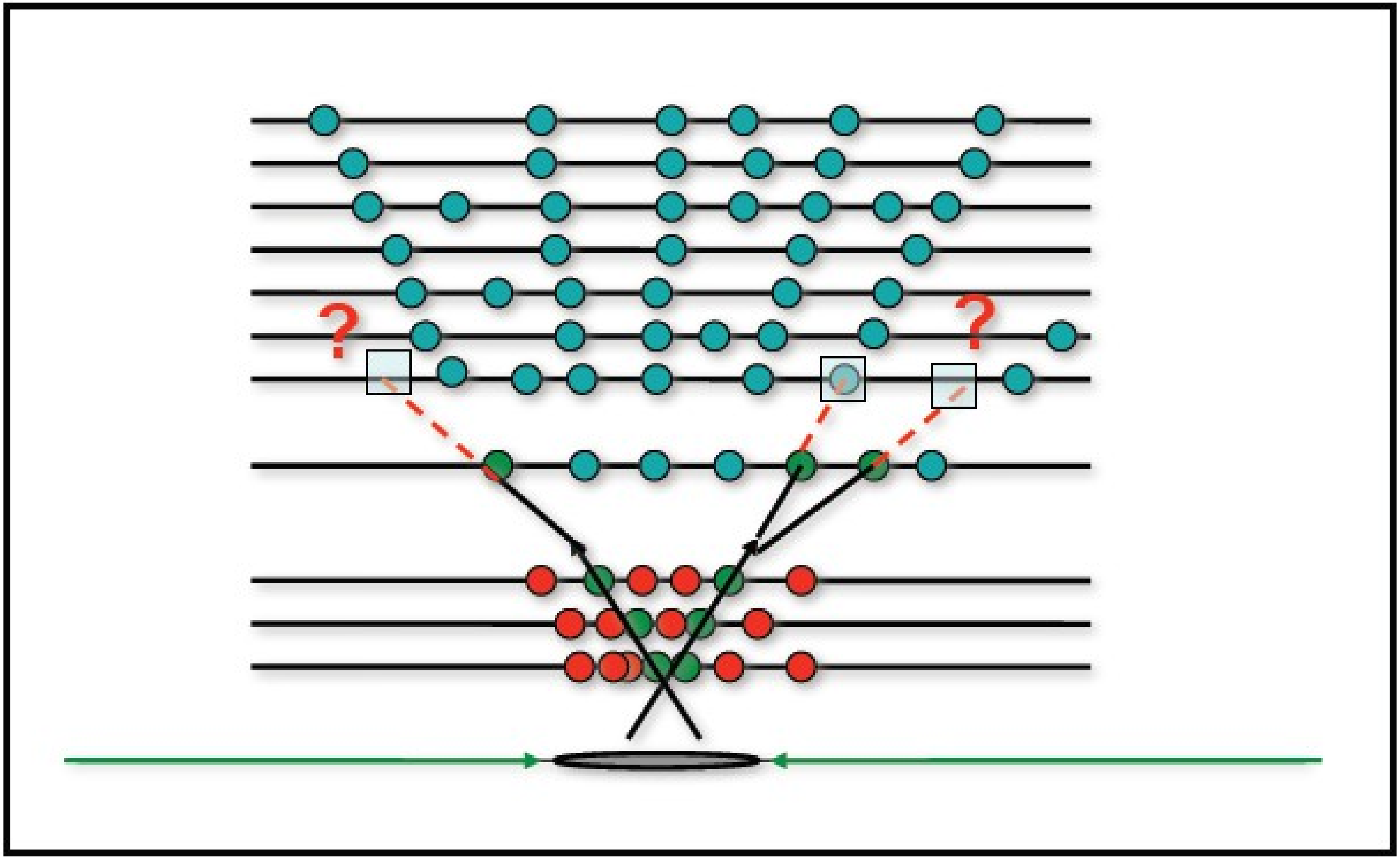


Trajectory is propagated from layer to layer taking into account the uncertainties on the hit positions, energy loss, multiple scattering

Trajectory building

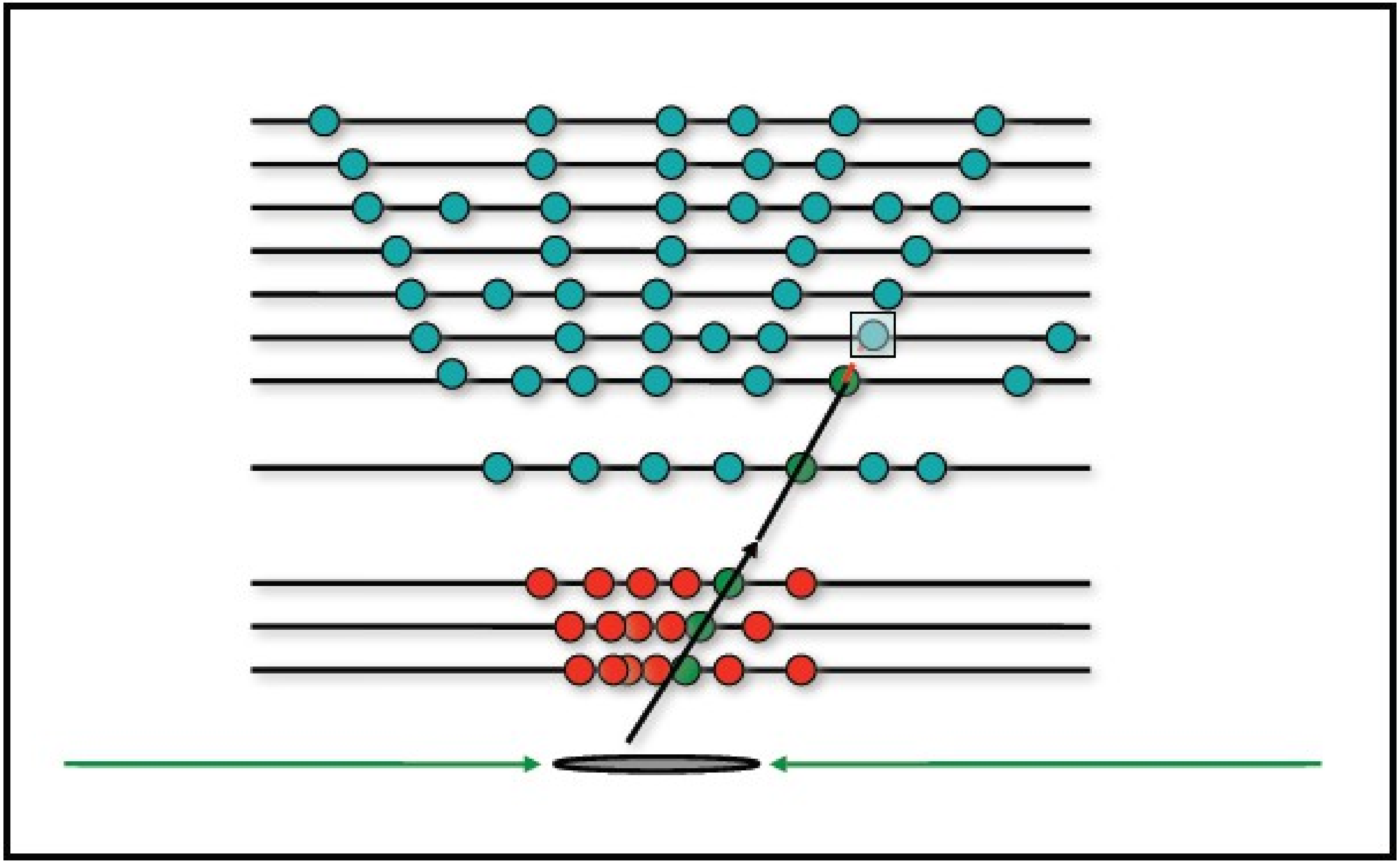


Trajectory building

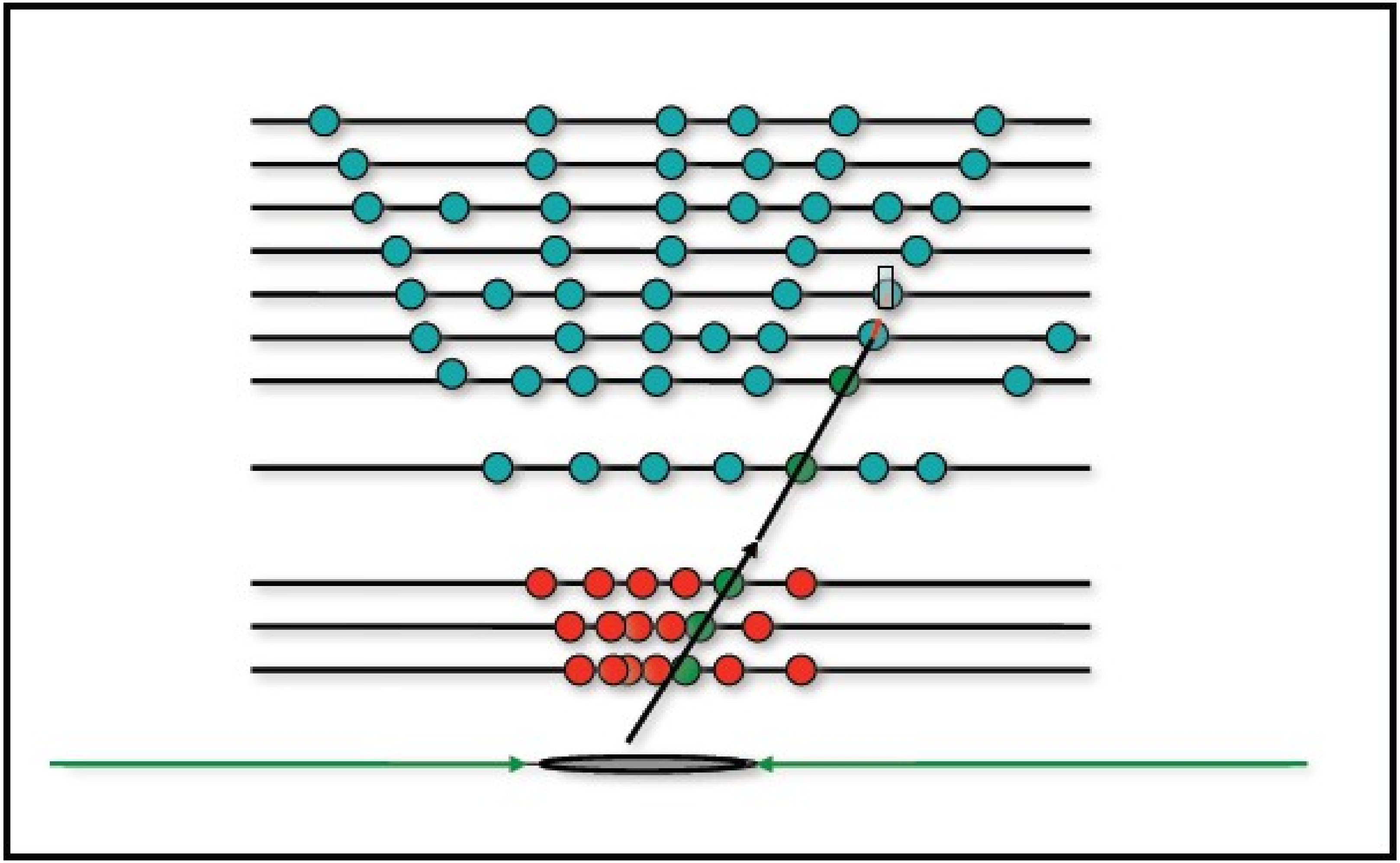


When no hits are found, track is probably fake

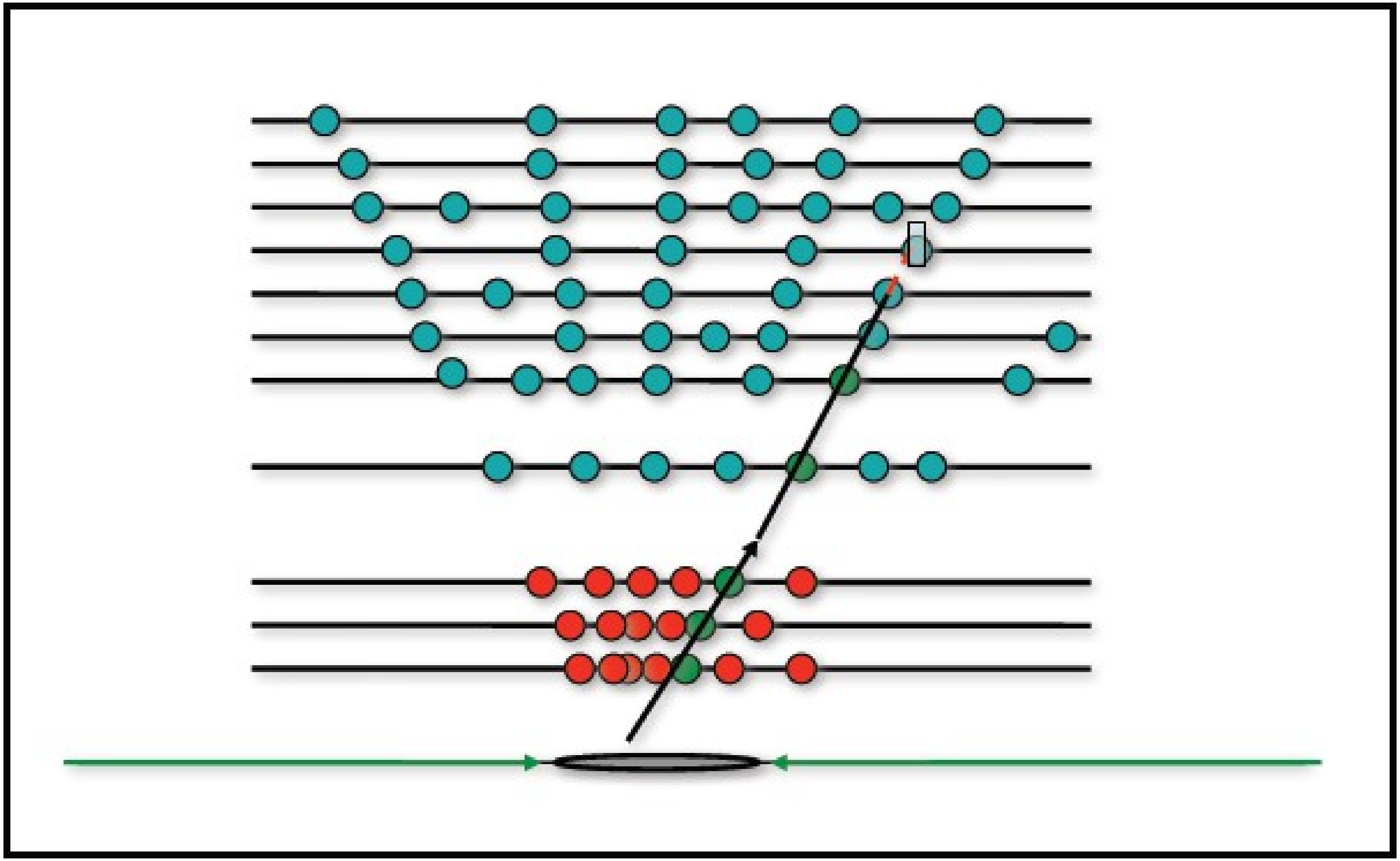
Trajectory building



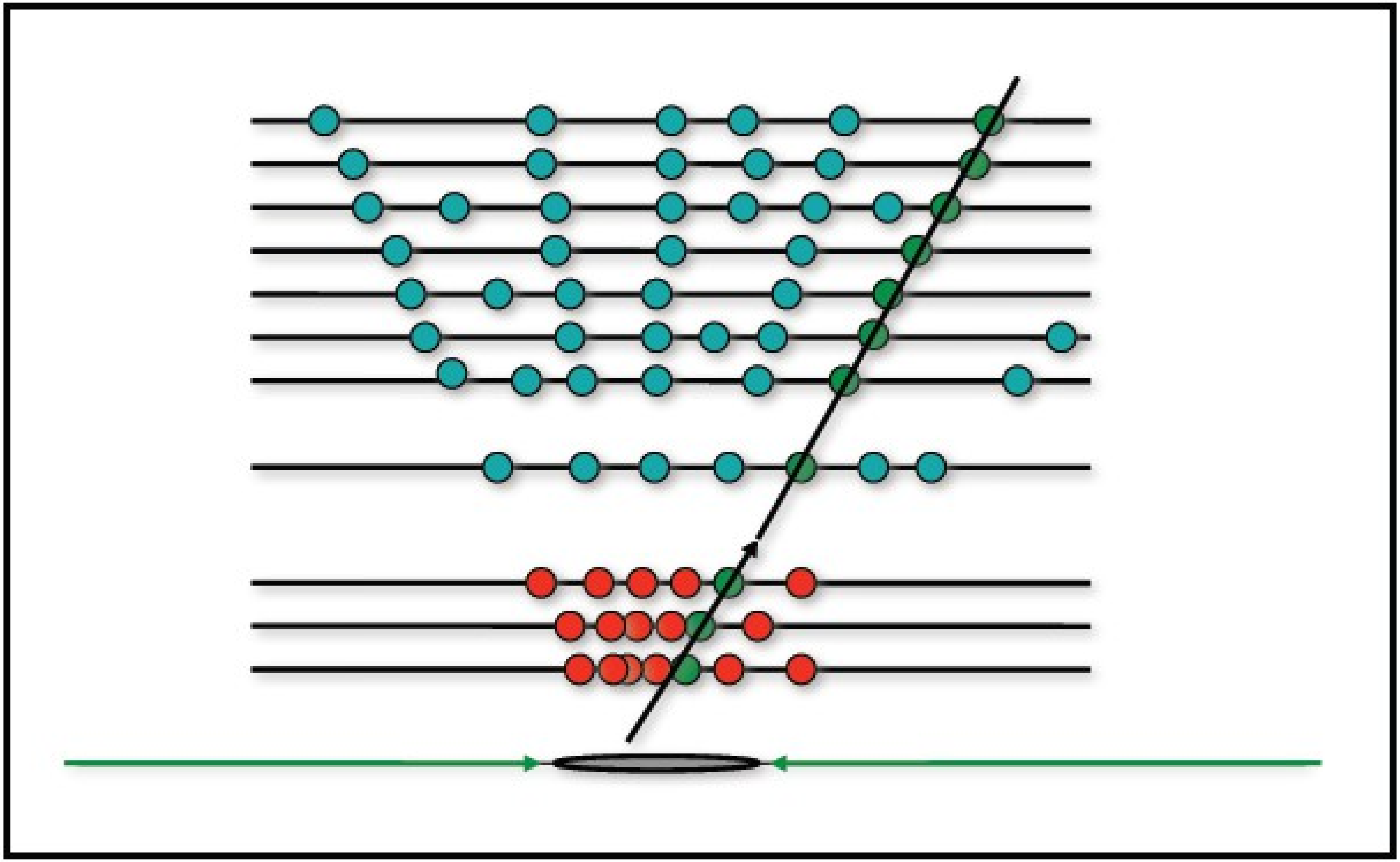
Trajectory building



Trajectory building



Trajectory building

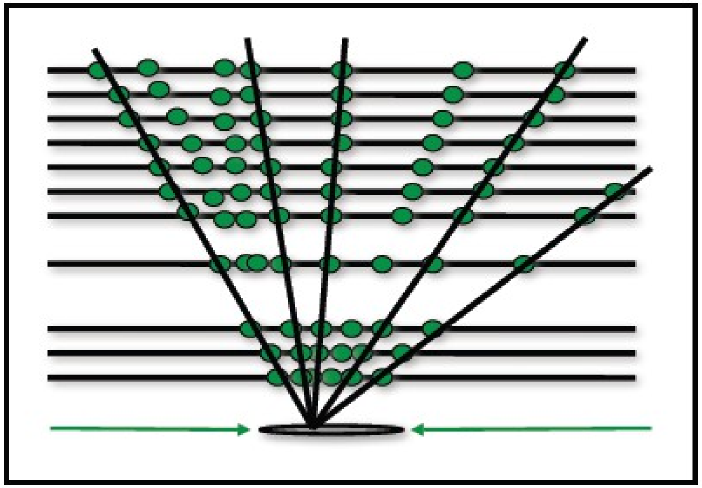


Now we have a track

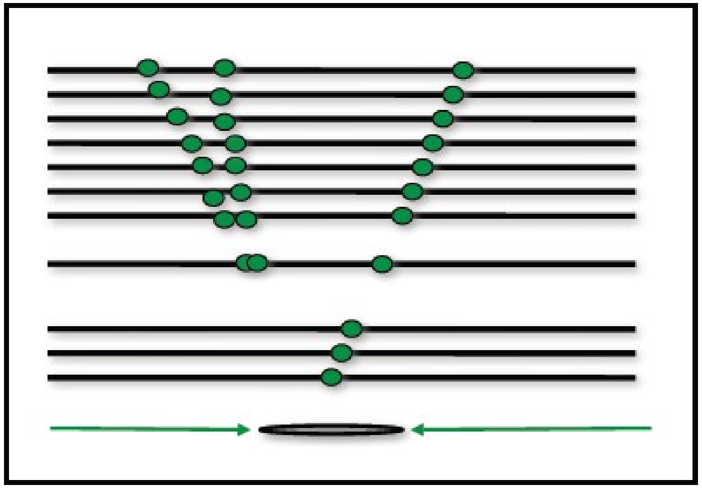
(Further refinements are applied, but I will not elaborate)

And then, iterate

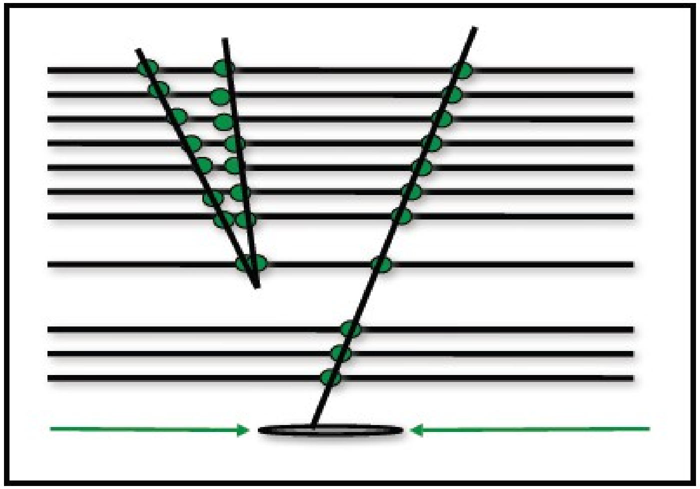
N-th step:



Remove associated hits:

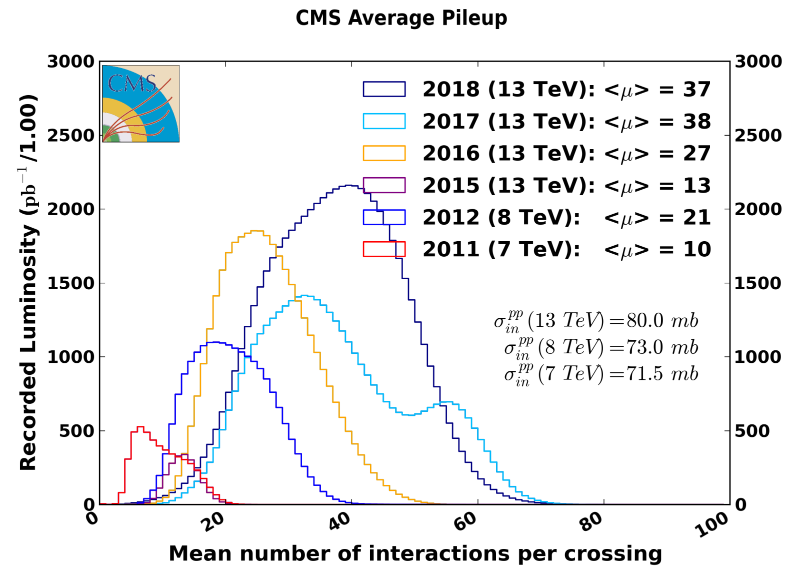


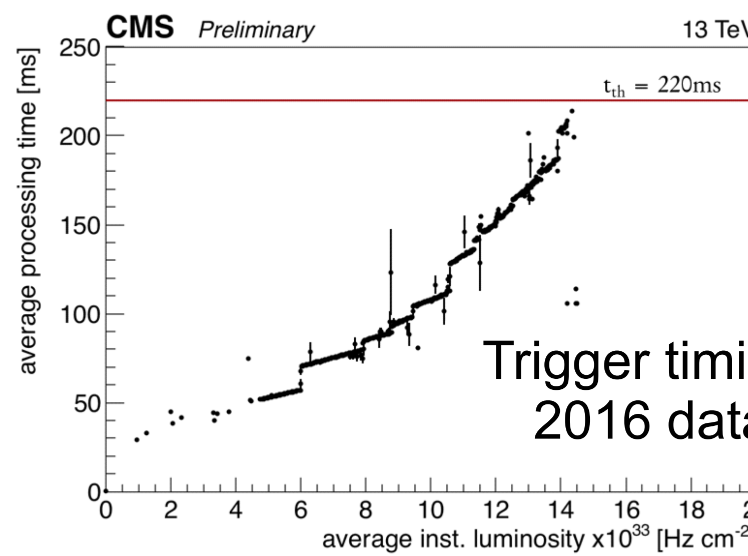
(N+1)-th step:



The Run II crisis

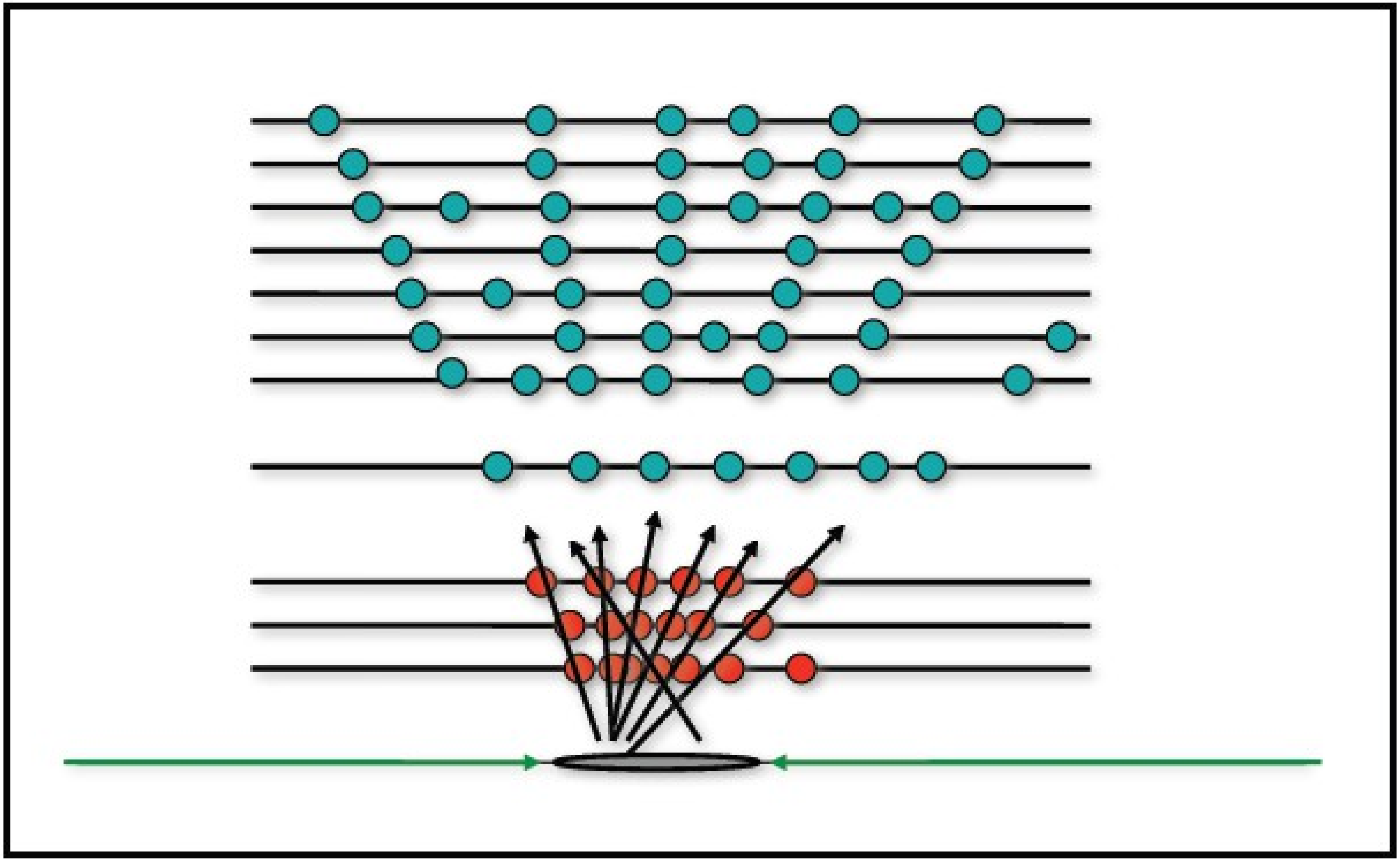
- Run II of LHC (2015-2018): larger collision energy (8 TeV → 13 TeV) and higher collision frequency
- Larger collision energy creates more particles per collision
- To reach the desired collision frequency, pileup had to increase too → even more particles per bunch crossing
- Issue: timing of the "seeding step" scales very badly with multiplicity
- Moreover, upgrade in early 2017 → one more inner layer (from 3 to 4) → more combinations → much slower seeding





Trigger timing, 2016 data

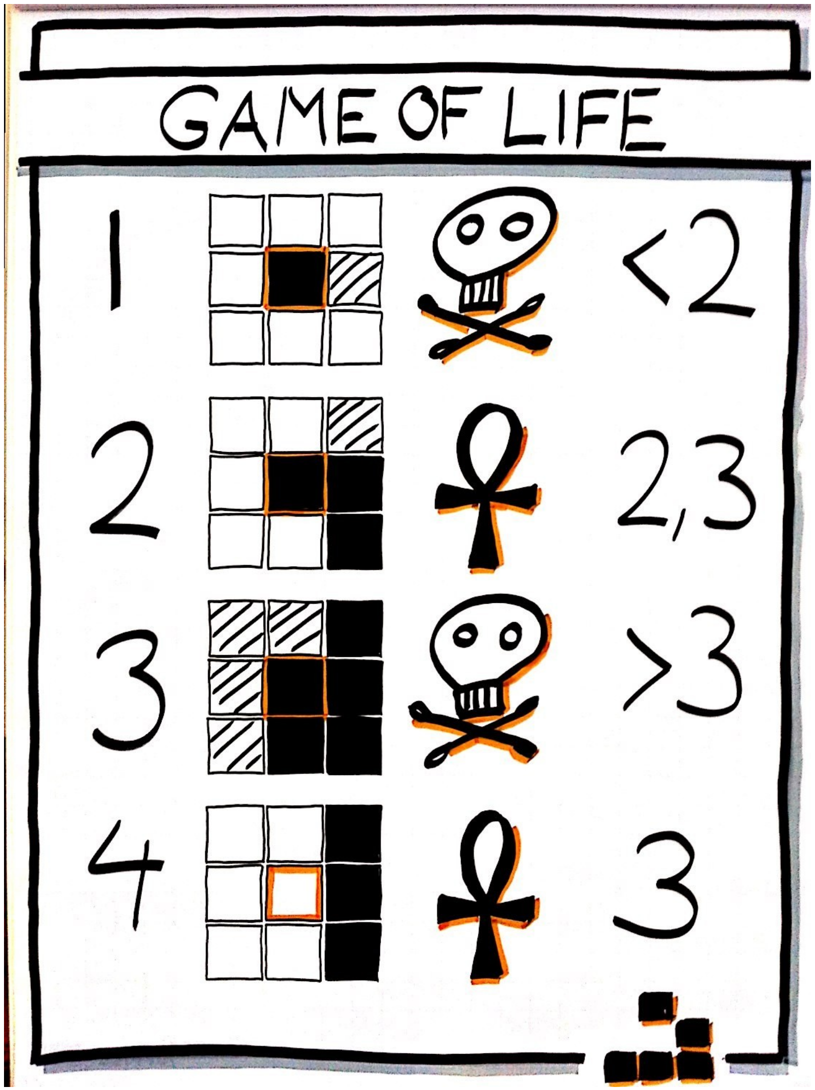
The main offender: Seeding



Fast fit to get initial trajectory, trying ***all combinations*** of hits in a small subset of layers...

Is that the smartest possible way?

Cellular Automata (CA)



In general, a CA consists of a regular grid of cells, each in a finite number of states.

For each cell, a set of cells called its neighborhood is defined.

An initial state (time t = 0) is selected by assigning a state for each cell.

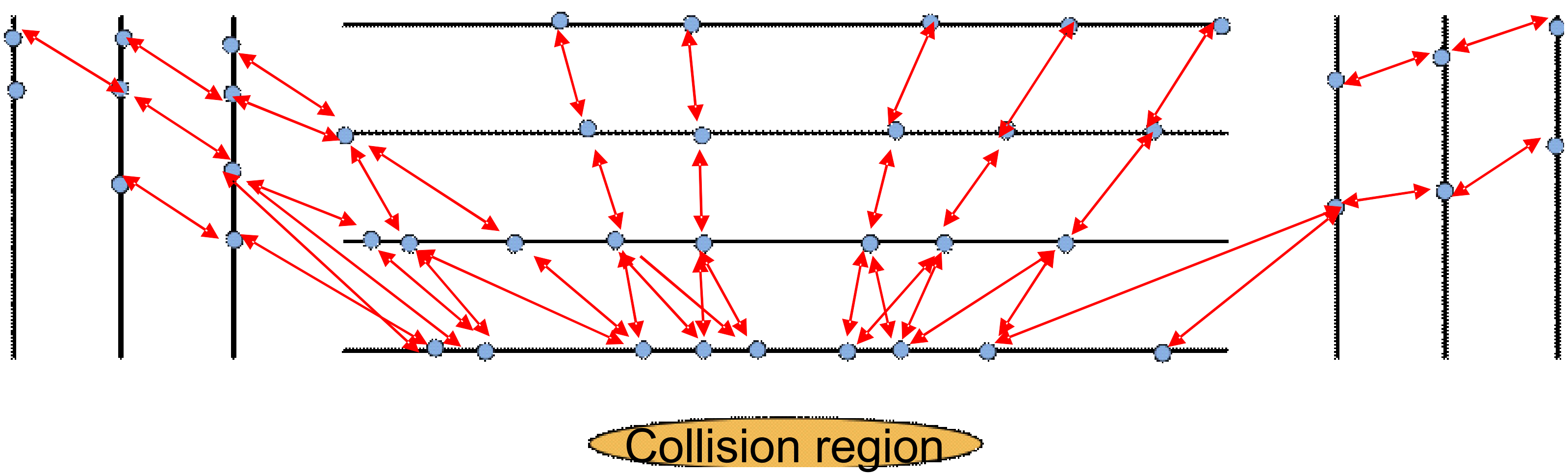
The new state of each cell depends from the current states of the cell and its neighborhood.

Famous example: Conway's Game of Life

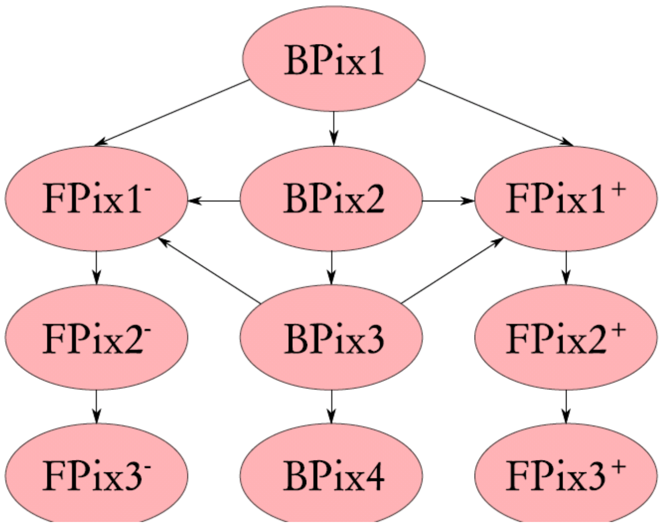
Picture from https://qualityswdev.com/2011/07/31/conways-game-of-life-in-scala/

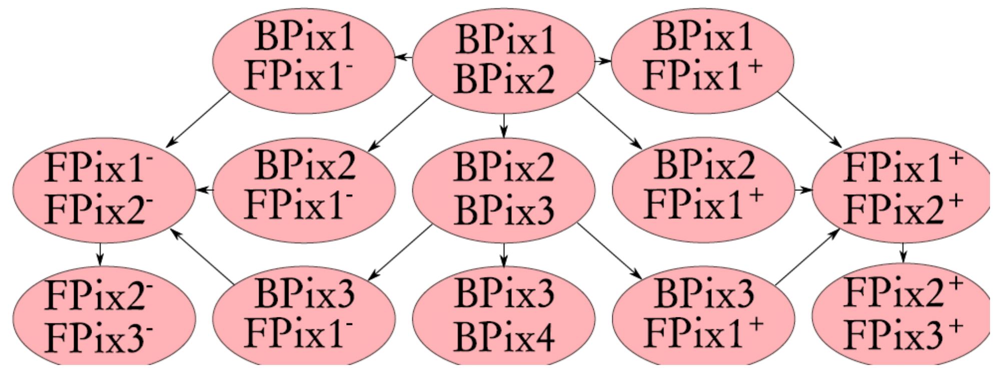
Cellular Automata (CA)

- Solution for seeding, chosen by CMS starting from 2017 operations
- A graph of all the possible connections between layers is created
- Doublets (“cells”) are created for each pair of layers
- Fast computation of the compatibility between two connected cells
- No knowledge of the world outside adjacent neighboring cells required, making it easy to parallelize

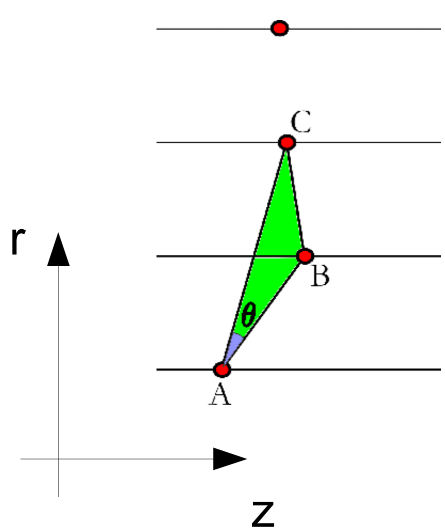


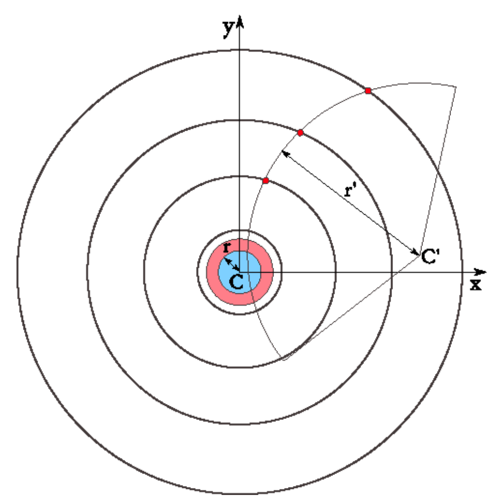
Cellular Automata (CA)





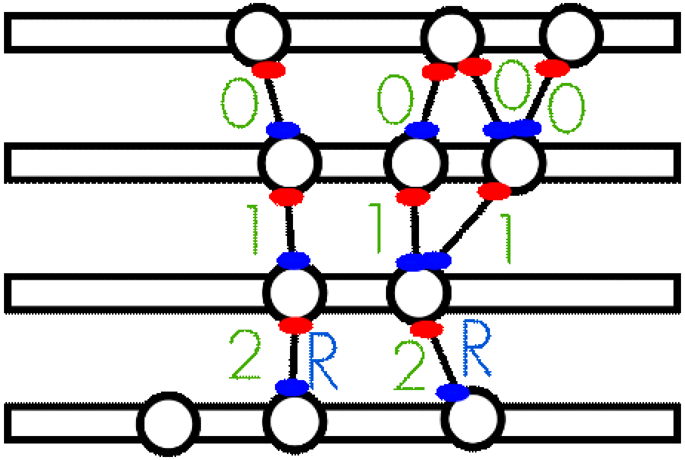
Degree of compatibility between hits is checked in r-z and x-y views:





Cellular Automata (CA)

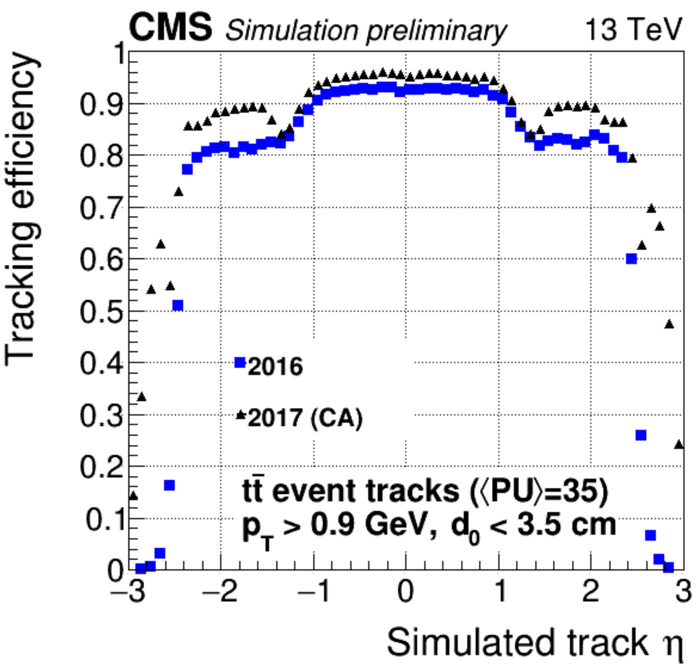
- If two cells satisfy all the compatibility requirements they are said to be neighbors and their state is set to 0
- In the evolution stage, their state increases in discrete generations if there is an outer neighbor with the same state
- At the end of the evolution stage the state of the cells will contain the information about the length
- If one is interested in quadruplets, pick a state 2 cell and for sure it is the start of a chain with at least 4 compatible hits
- [For a N-uplet, pick a state (N-2) cell]

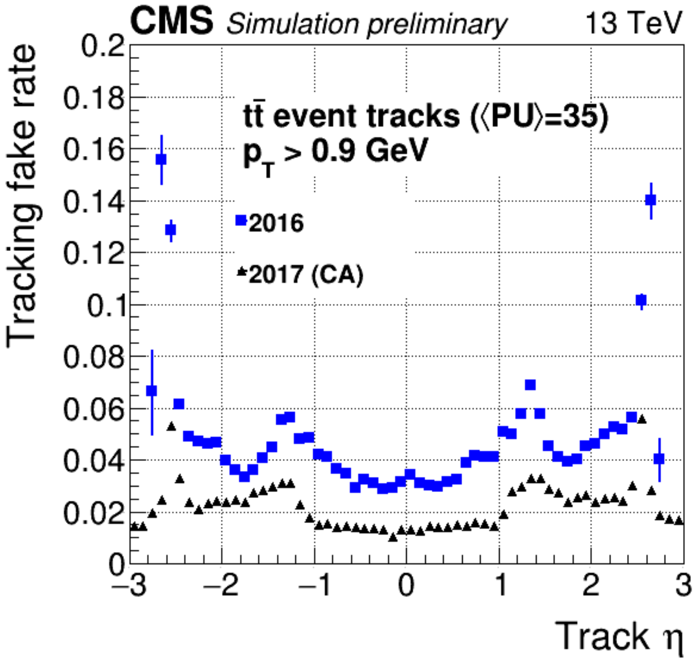


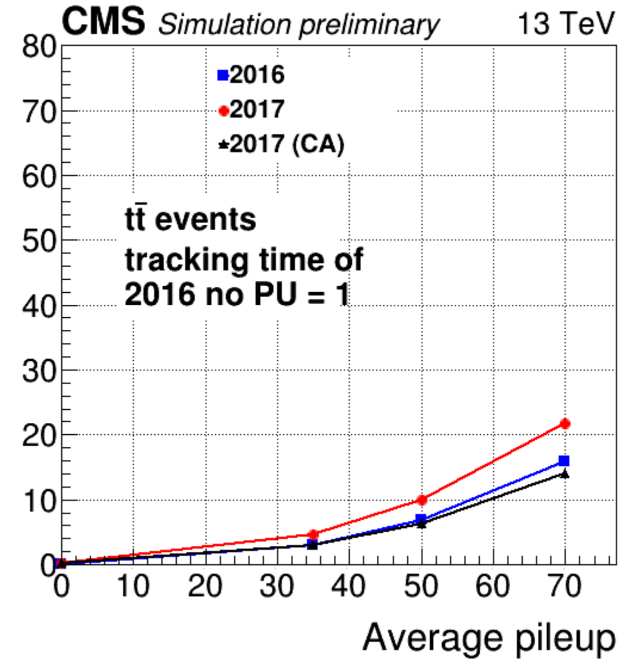
What we need, and what we got

- We need track-finding to be efficient
- We need the track sample to be very pure
- And it has to be fast

Automaton works well







And now we can parallelize...

Algorithm	Time per event [ms]: Average ± root mean square
2016 tracking on 2016 data	29.3 ± 13.1
2016 tracking on 2017 data	72.1 ± 25.7
Cellular Automaton on CPU	14.0 ± 6.2
Cellular Automaton on GPU	1.2 ± 0.9




CMS Thesis Award 2017

Felice Pantaleo

University of Hamburg

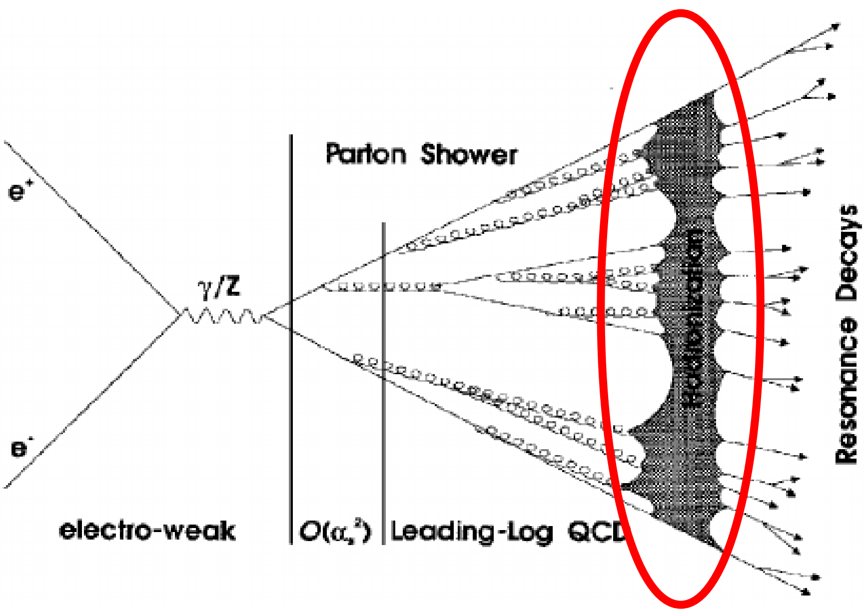
Title: New Track Seeding Techniques for the CMS Experiment

Even in a >3000 members collaboration, individuals can have a visible impact and get rewarded for thinking out of the box

Example #2:
finding the remnants of quarks

Quarks are always "dressed"

- You can't observe quarks directly
- QCD explanation: the attraction <u>increases</u> with r, so at some point the potential energy of the system is larger that $2m_q$



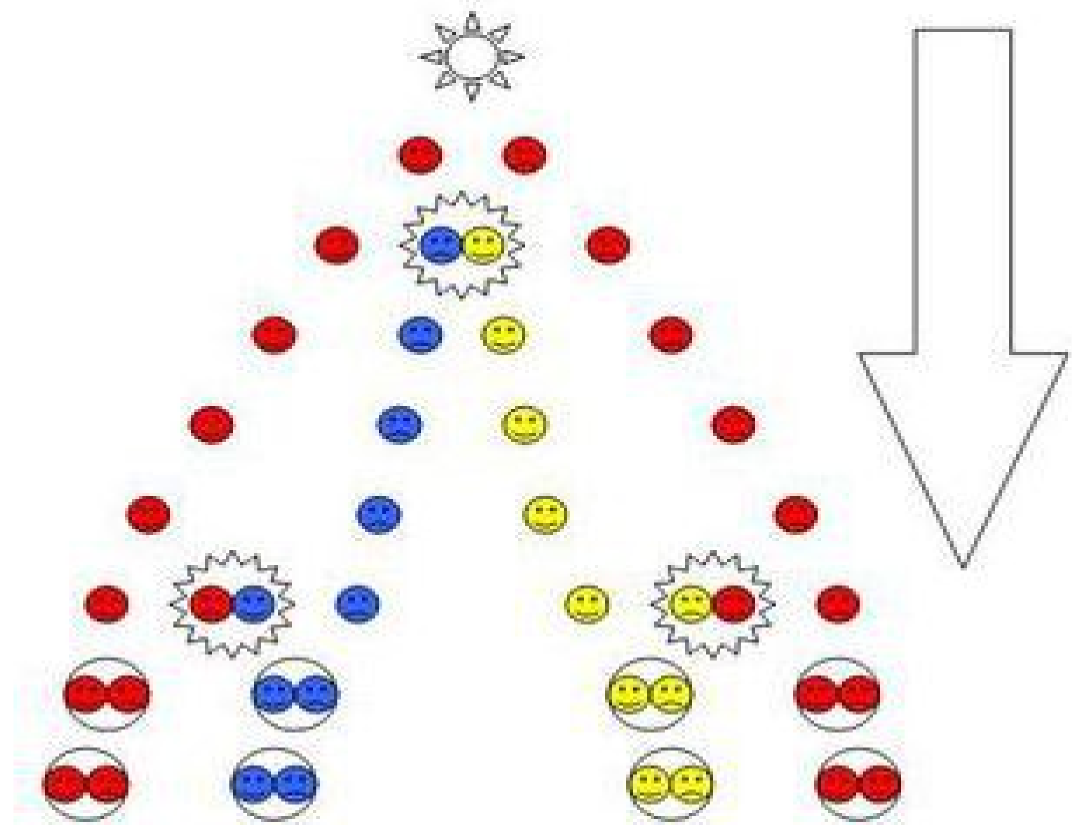
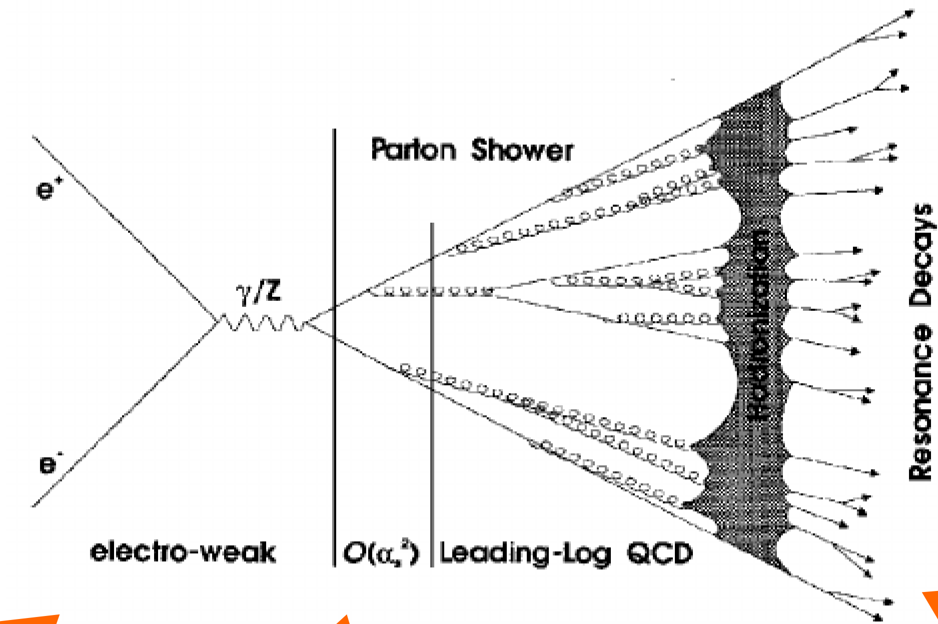


Image from Tommaso Dorigo

From quarks to observable particles



Leading order calculation (rather easy)

Higher-order corrections, computationally very intense

Here we need to cut corners, things get murky

Here we give up: parametrize with empirical models tuned to data

Final set of stable particles that we compare to data

Quarks create "jets"

Quarks are never naked, but we need to undress them

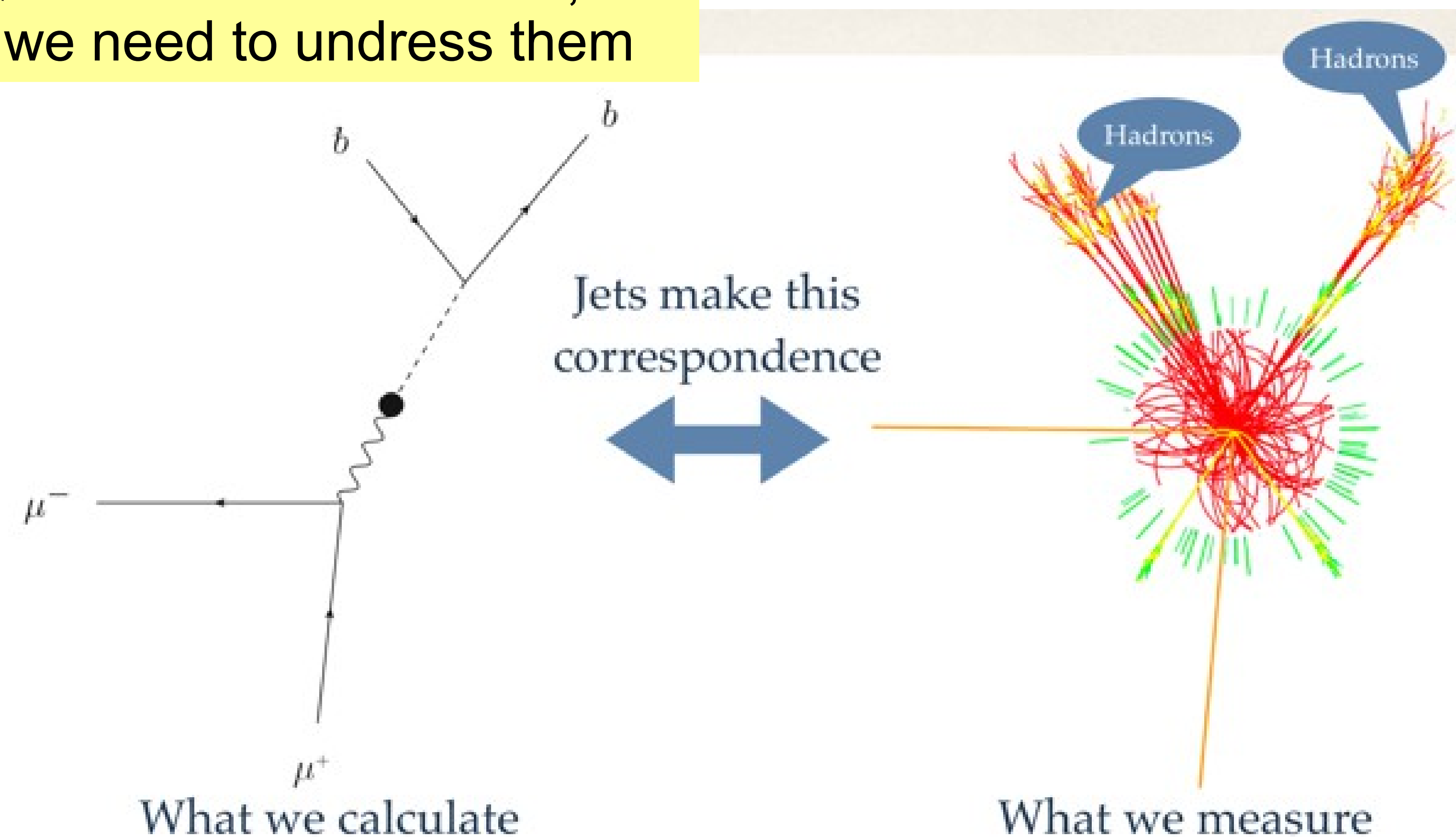
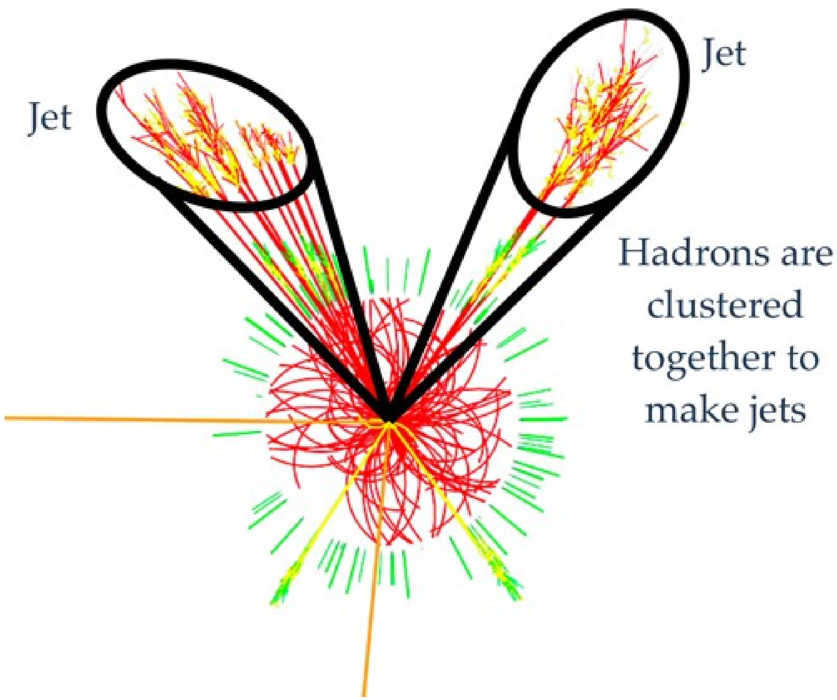


Illustration credit: Flip Tanedo

How to build jets

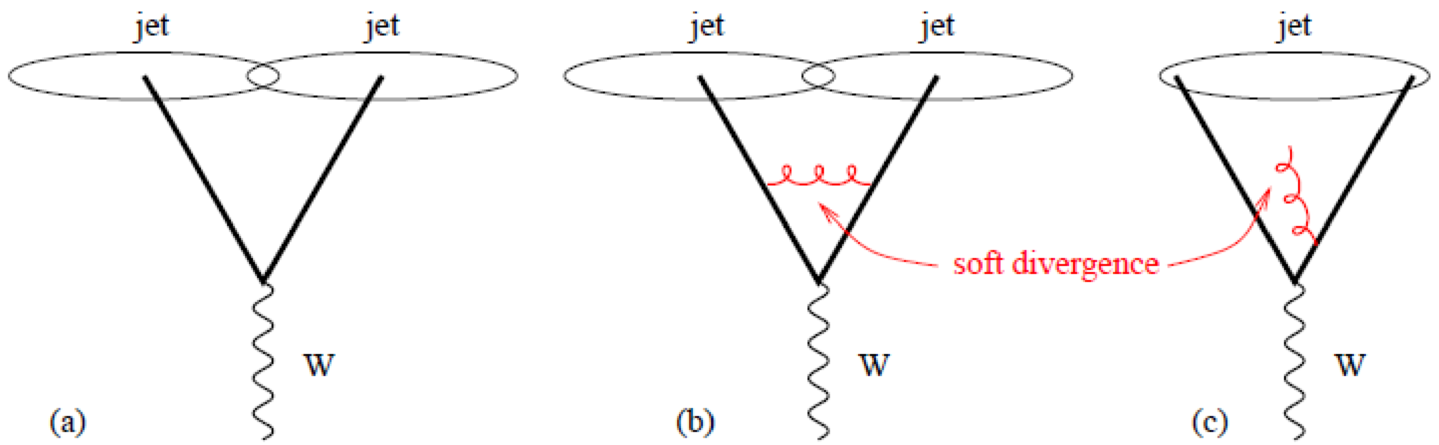


Two popular ways:

- **Cone-based algorithms:**
  - Use the highest-energy particles in the event as initial seeds
  - Sum momenta of all particles in a cone of fixed radius around each seed
  - Use those sum vectors as new seeds, and repeat until convergence
- **Clustering algorithms:**
  - Calculate distances d_{ij} (according to some metrics) between particles i and j, for all i,j, and distance d_{iB} between particle i and the beam axis
  - If $d_{ij}<d_{iB}$, combine i+j; else, call i a jet

Illustration credit: Flip Tanedo

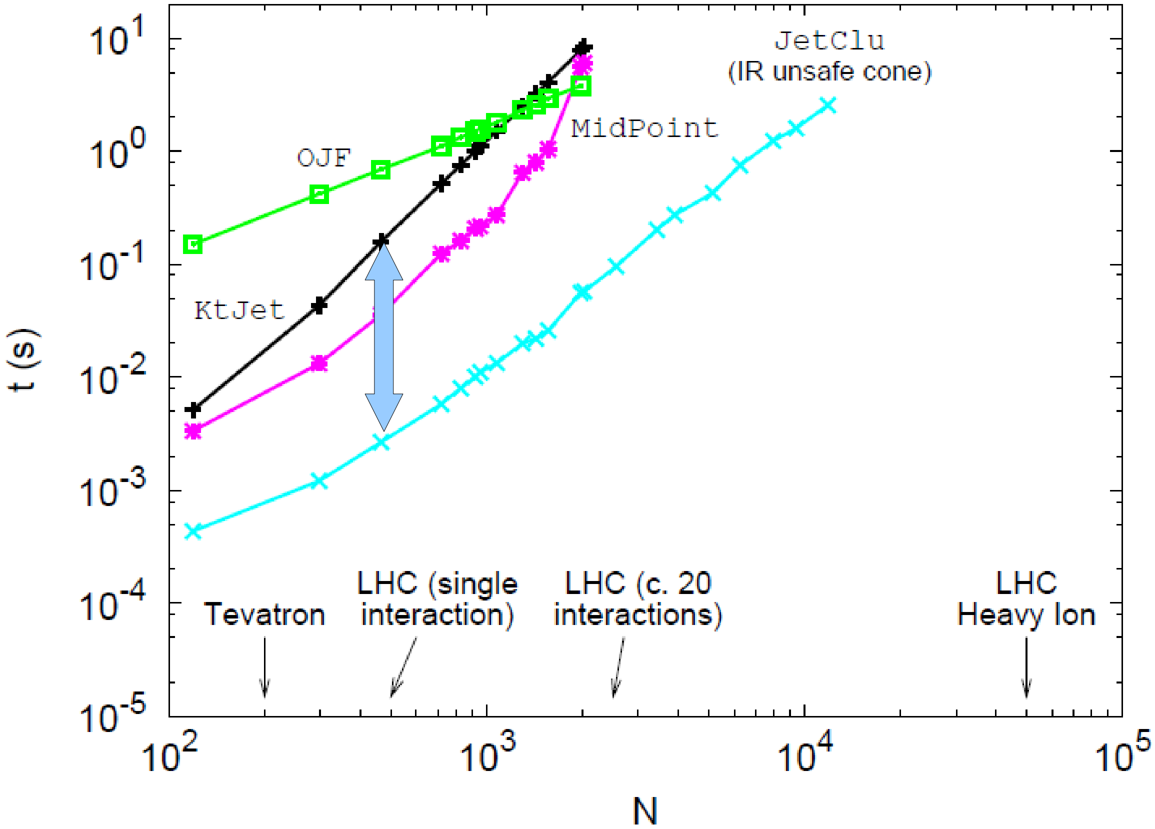
Infra-red (IR) stability



From G.Salam, arXiv:0906.1833 [hep-ph], Eur.Phys.J.C67 (2010) 637

A jet algorithm is said to be IR-unstable if the addition of a low-momentum particle (with arbitrarily low momentum) can change the outcome of the jet finding, making the theory-experiment comparison quite ill-defined

Fast and wrong, or right and slow?



- The blue curve is for a cone algorithm
  - IR-unstable...
  - ...but a lot faster
- The black curve is for a clustering algorithm
  - IR-stable...
  - ...but much slower
  - Gets worse as N grows: finding minimal value of d_{ij}, d_{iB} for all i,j is a $O(N^2)$ operation done N times
  - *(Really?)*

Jet finding with Voronoi cells

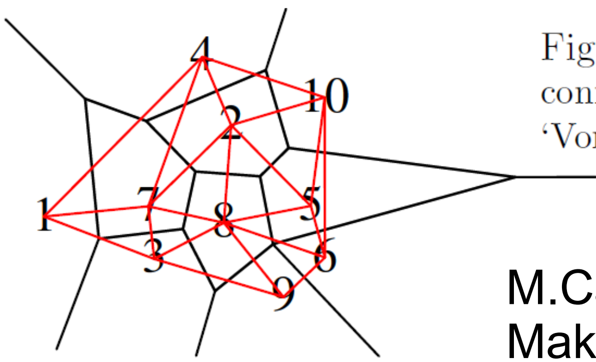
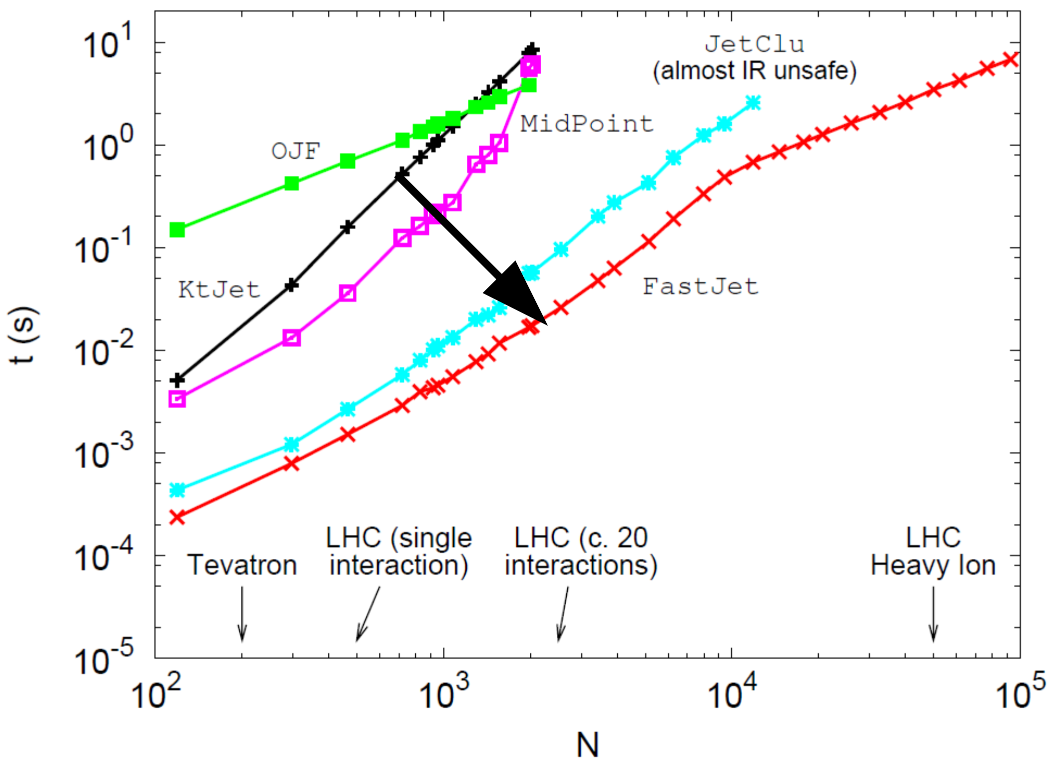


Figure 1: The Voronoi diagram for ten random points. The Delaunay triangulation (red) connecting the ten points is also shown. In this example the points 1, 4, 2, 8 and 3 are the 'Voronoi' neighbours of 7, and 3 is its nearest neighbour.

M.Cacciari, G.Salam, arXiv:hep-ph/0512210, Phys.Lett.B641 (2006) 57
Making use of work by Dirichlet (1850) and Voronoi (1908)



O(N^3) became O(N lnN)

Example of a *Paradigm Shift*: as soon as the authors of that paper released their code, cone algorithms became a thing of the past

Higher level analysis

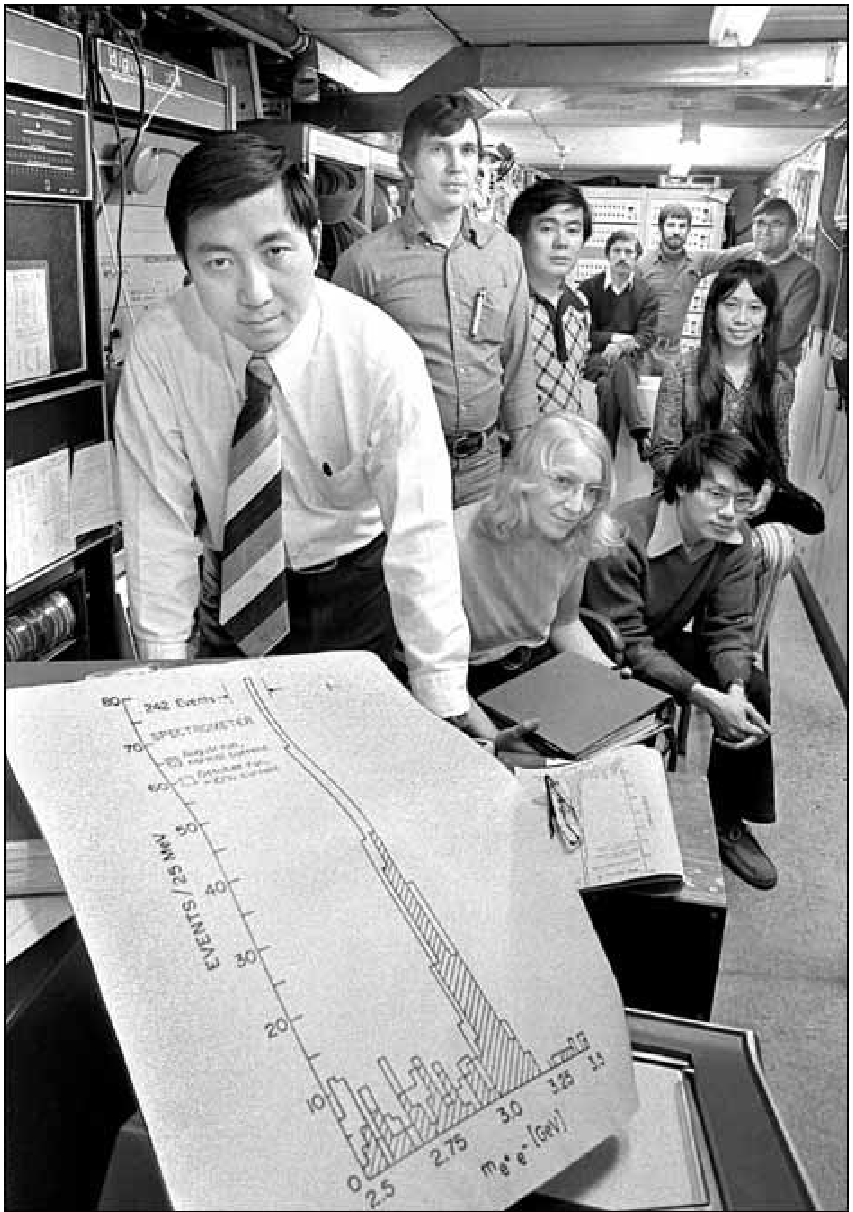
Higher level analysis

- All that we have seen so far is run centrally in CMS
- Now that the complexity of the problem is reduced to a small set of jets and other high-level objects (e, μ, τ, γ), you can start the very last bit of analysis, e.g., your PhD thesis
- It may look very different, depending on the question you want to address, e.g.:
  - Search for a new particle, for which you have a model
  - Search for new particles, as model-independently as possible
  - Measure a certain quantity, and compare it with models
  - Measure a certain quantity, for which there is no expectation (e.g., a fundamental parameter of Nature)

Hypothesis testing

- Quantify the agreement of data with a null hypothesis H_0 (e.g., the Standard Model)
  - In case we only test H_0, methods may resemble to what is elsewhere called Anomaly Detection
- Or quantify which one is best between H_0 or H_1, e.g.:
  - H_0 = only backgrounds exist, and behave as in SM
  - H_1 = like H_0 but also the Higgs exists and behaves as in SM
- Or select which sub-set of $\{H_i\}$ is consistent with data
  - $\{H_i\}$ is often a continuum, e.g.: $m=10.0\pm1.0$ GeV, meaning that $9.0<m<11.0$ GeV is the 68% confidence interval for m

Anomaly detection, the way we prefer it



Dream of every particle physicist:

- Study a simple feature of data, e.g., some invariant mass
- Find a spectacular anomaly with a clear interpretation, e.g., a peak rising from a smooth background
- Get a Nobel Prize

(Or at least get it awarded to your boss, or to some theorist who predicted it.)

Anomaly detection, the tough way

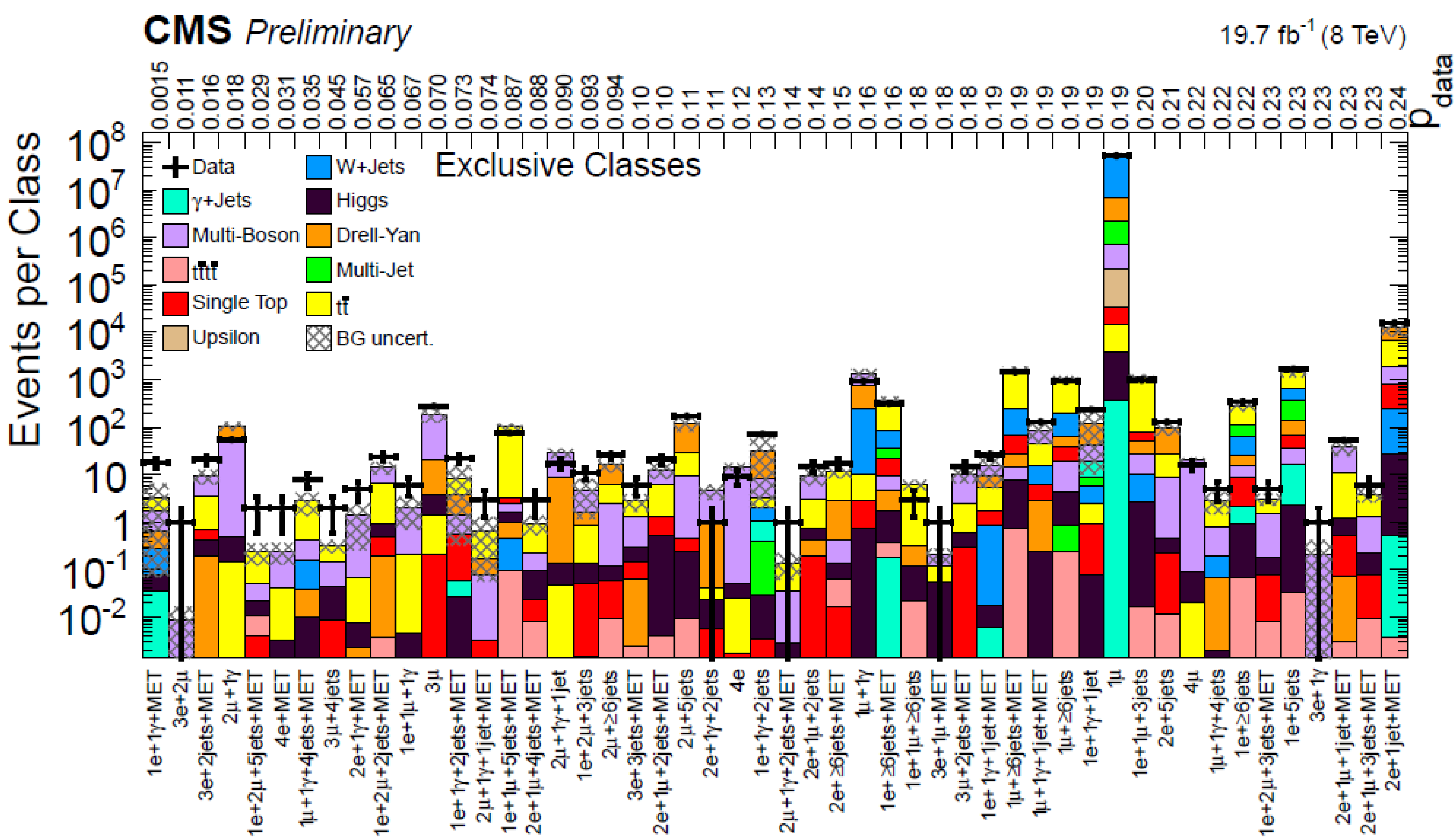


Figure 2: The 50 most significant exclusive event classes, considering only the total number of events.

CMS Collaboration, CMS-PAS-EXO-14-016

Anomaly detection, the tough way

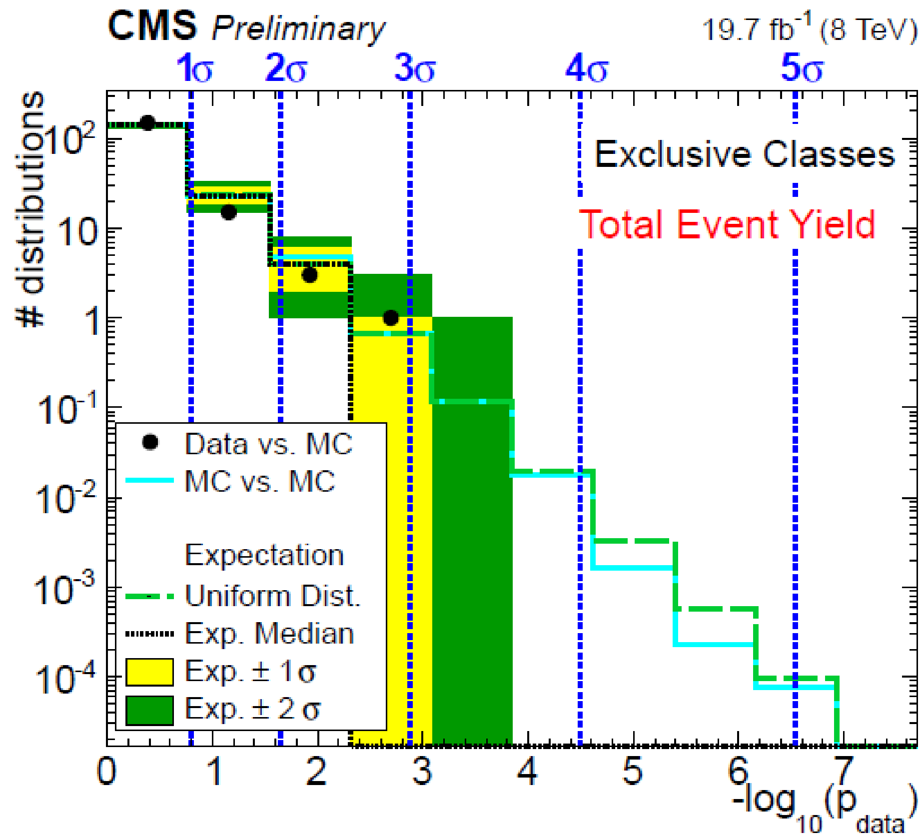


Figure 5: Distribution of p-values for exclusive event classes in the scan of total event yield. Black markers represent the measured data compared to the SM MC expectation. The histogram labeled "MC vs. MC" represents the comparison of the SM MC expectation to pseudo-data generated under the SM-only hypothesis. As a further comparison, the expectation from the uniform distribution is given, where the individual components are explained in Sec. 3.5. In the first bin 148 distributions are observed, with $139^{+4}_{-5}(1\sigma)^{+9}_{-10}(2\sigma)$ expected from the SM.

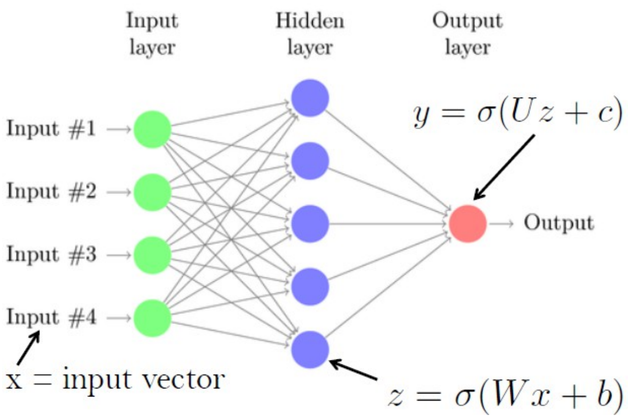
MC: Monte Carlo

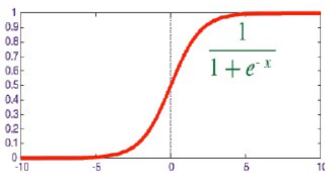
(If you don't know what σ and p-value are, just ask, I have a backup slide)

CMS Collaboration, CMS-PAS-EXO-14-016

Machine Learning (ML)

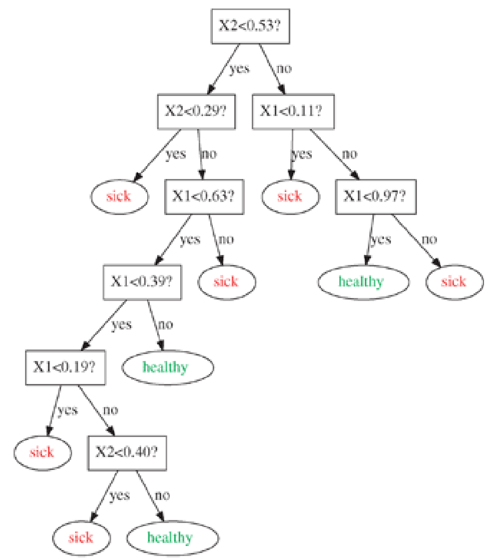
Neural Network:

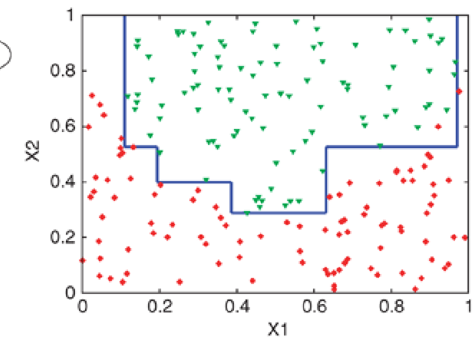




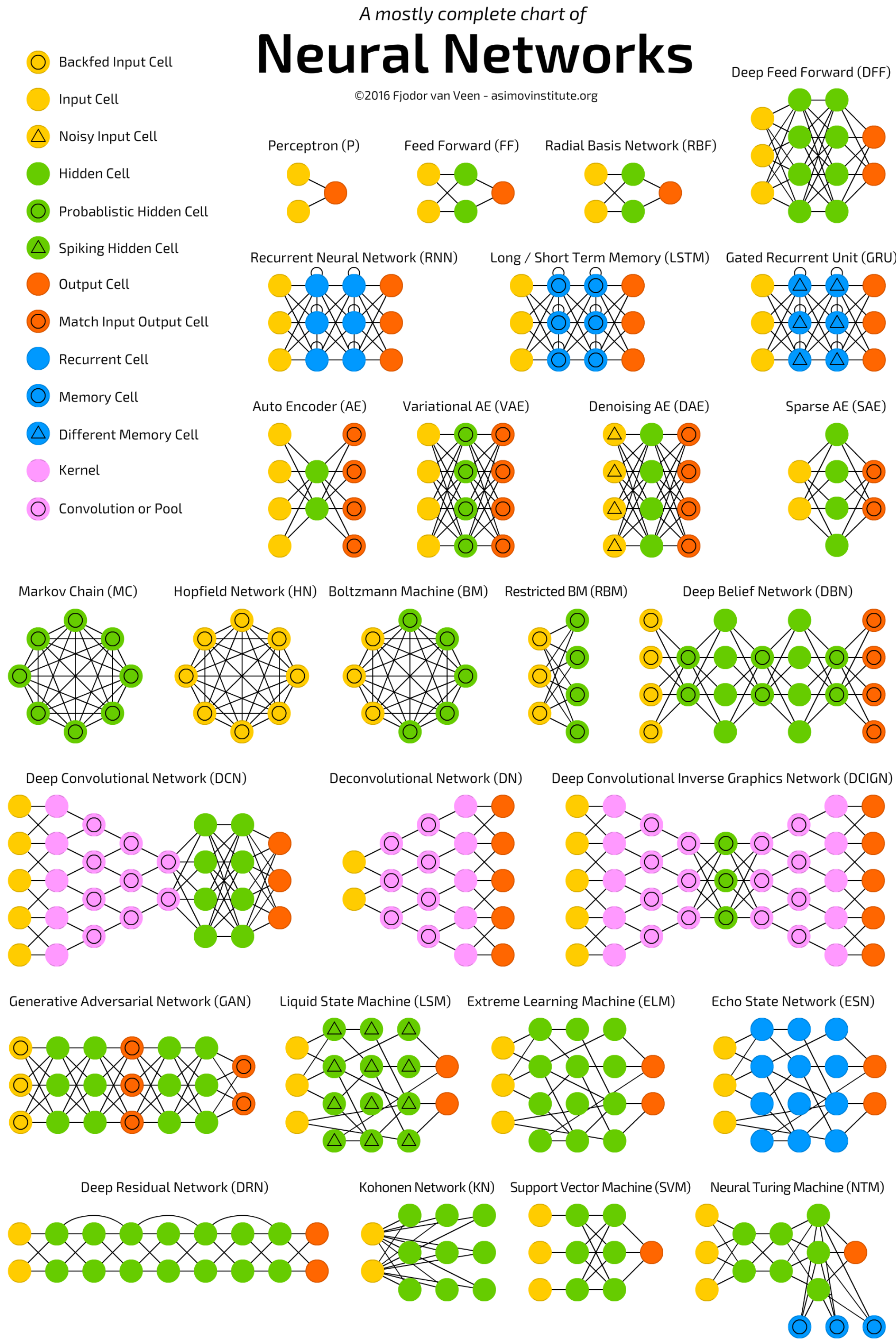
σ(x) = sigmoid function
is the *Activation Function*

Decision Tree:



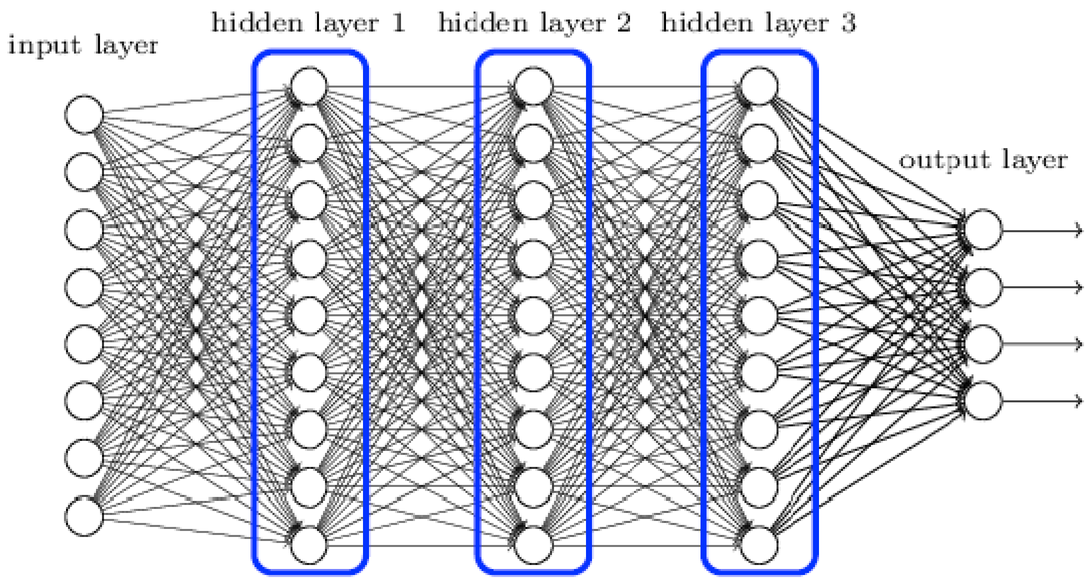


- Some particle physicists started using ML techniques in the 90's, typically facing resistance by old-schoolers who were afraid of delegating physics intuition to „black boxes“
- Nowadays, Neural Networks (NN) and Boosted Decision Trees (BDT) are very standard tools, widely used in LHC analyses
- Probably because most „low hanging fruits“ have been reaped already, and what remains are the toughest cases



http://www.asimovinstitute.org/neural-network-zoo/

Deep Learning



- Basic idea: learn multiple levels of representations that correspond to different levels of abstraction
- Computationally intensive (which is why it became a thing only recently), but **suitable for parallelization** ($\Rightarrow$ GPUs)
- It is now making its way into the LHC experiments, and probably going to replace traditional NN and BDT

A sad story

Scuola Normale Superiore

Tesi di Perfezionamento in Fisica

A measurement of the Gluon Splitting Rate into $c\bar{c}$ Pairs in Hadronic Z^0 Decays with the ALEPH detector

Candidate
Andrea Giammanco

Supervisor
Prof. Lorenzo Foà

Ph.D. Thesis
Pisa, April 2003

Chapter 5

Neural Networks

Neural networks [75, 76, 77, 78] (NN) are a powerful tool for pattern recognition, and are widely applied in several fields, ranging from financial predictions to weather forecasts to character recognition. In particle physics they are mostly used for event or particle classification tasks.

The analysis described in this dissertation makes use of a neural network to separate the $g \to c\bar{c}$ events from the backgrounds.

5.1 What is a Neural Network

Several definitions exist of what is meant by "neural network". One of the most agreed on is the following [76]:

A neural network is an interconnected assembly of simple processing elements, **units** *or* **nodes***, whose functionality is loosely based on the animal neuron. The processing ability of the network is stored in the inter-unit connection strengths, or weights, obtained by a process of adaptation to, or* **learning from***, a set of training patterns.*

The "neuron" is the fundamental processing element of a neural network. It takes its name from the analogy with the cells responsible of signal transmission in the brain.

The information processing performed by the real (biological) neuron may be crudely summarized as follows: it receives inputs from other neurons (or from the surrounding environment), combines them in some way, performs a generally non-linear operation on the result, and then outputs the final result. The artificial neuron mimics the real one by multiplying the input values by some number or weight to indicate the strength of the link (the "synapse", following the biological similitude); the weighted signals are then summed to produce an overall unit activation. If this activation exceeds a certain threshold the

83

- The data analysis in my PhD thesis was based on a NN
- At that time, NN was exotic: an entire chapter was needed
- That chapter ended with a demonstration that *More than two hidden layers are (...) unnecessary*. It didn't age well...

Supervised vs unsupervised learning

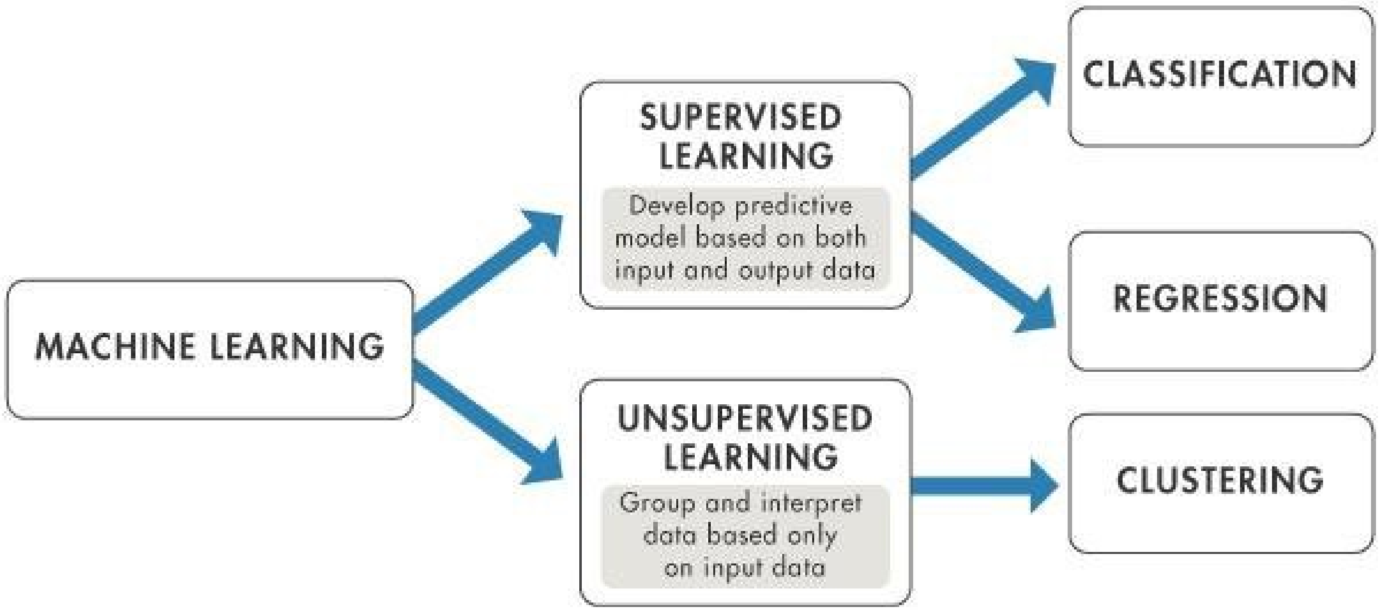
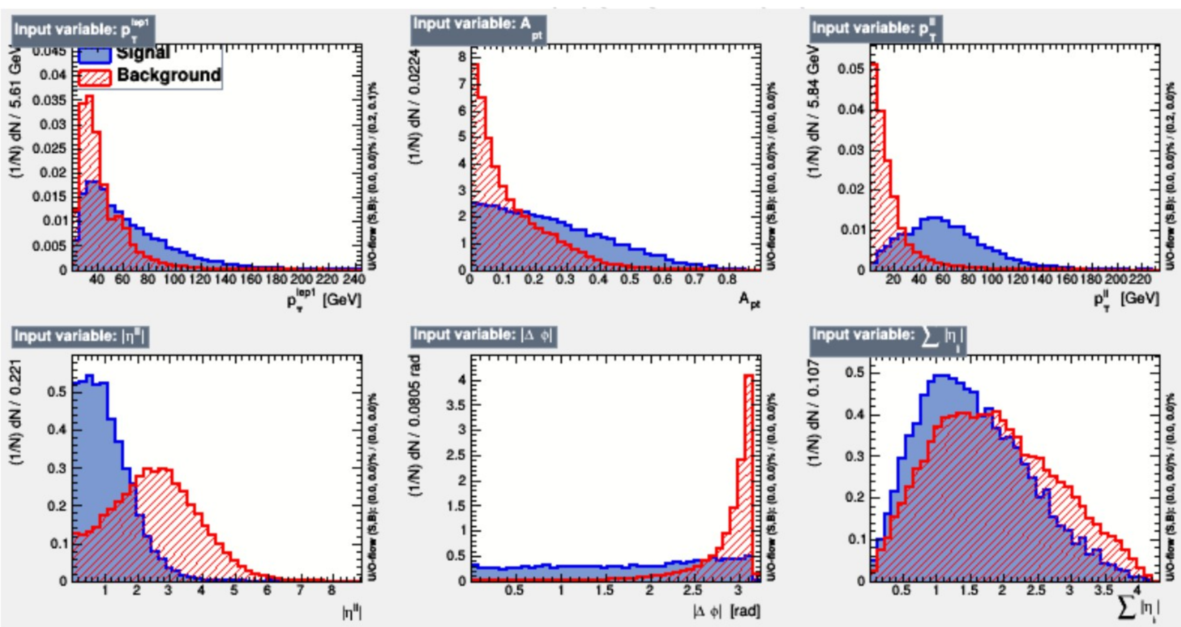


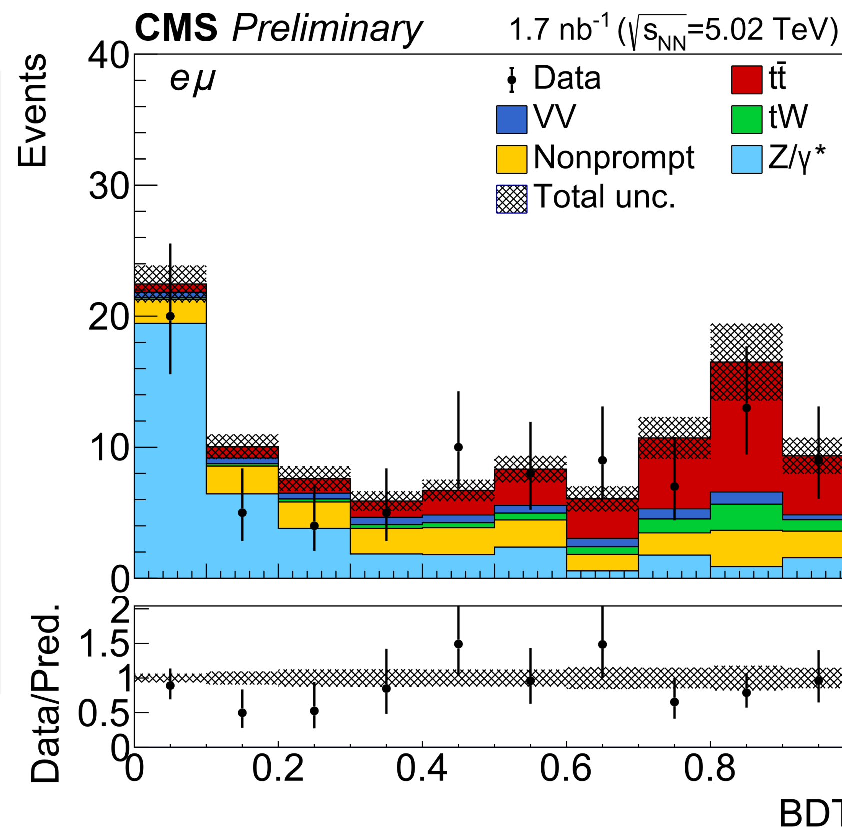
Image from Rory Bunker

- **Supervised learning**: iteratively present the algorithm with simulated data, for which you already know the truth (target value); minimize distance between *output* and *target* values
  - *Regression*: target is a continuous distribution
  - *Classification*: target is a finite set of categories; most commonly just two: signal vs background
- **Unsupervised learning**: look for patterns in data

Most common use case in HEP: signal-vs-background classification



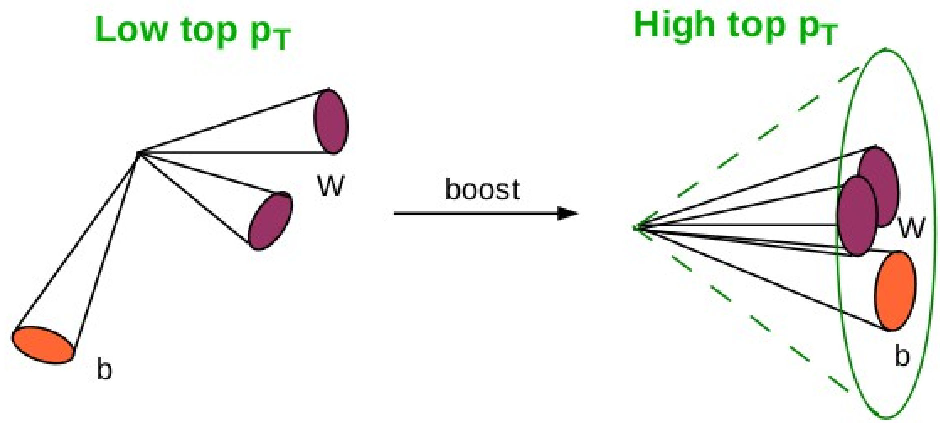
Input: some features of the data that discriminate between the signal (here the top quark) and the sum of all backgrounds



Output: a combined discriminant (signal is red)

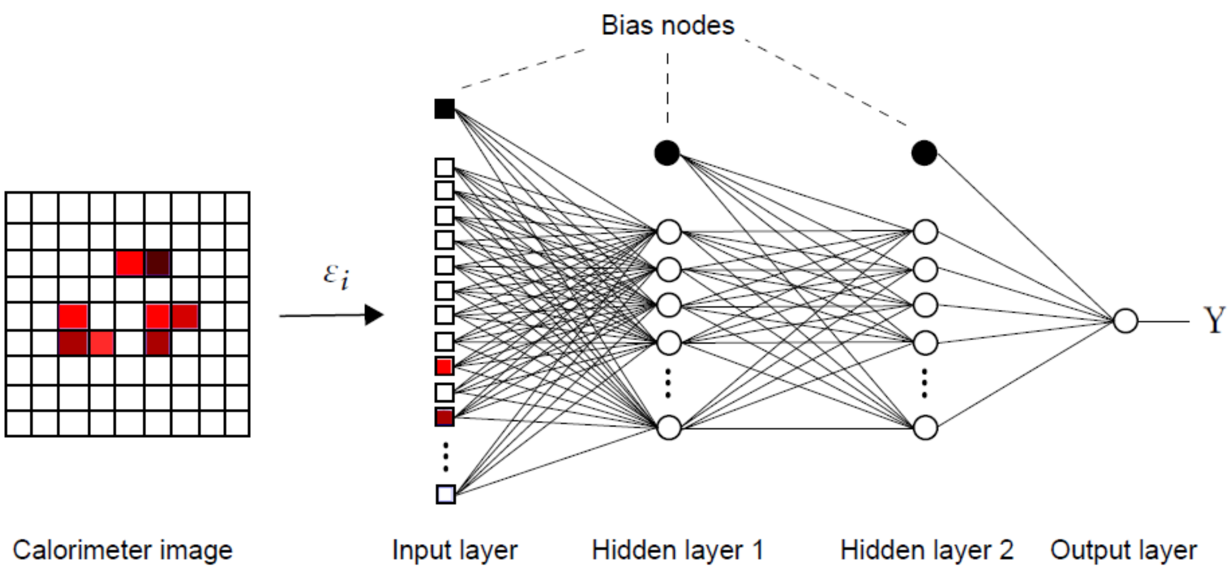
CMS collaboration, *Evidence for top quark production in nucleus-nucleus collisions*, CMS-HIN-19-001

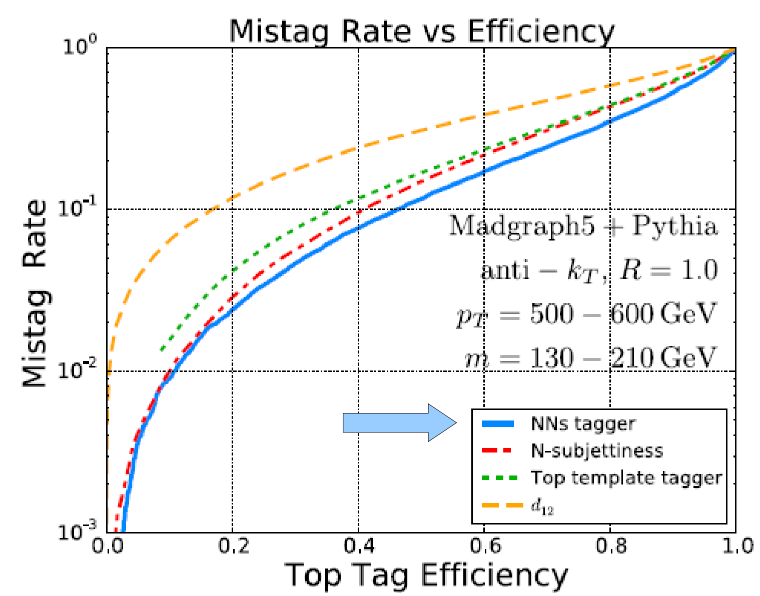
Classification via image recognition



From http://www.quantumdiaries.org/2012/08/05/boost/

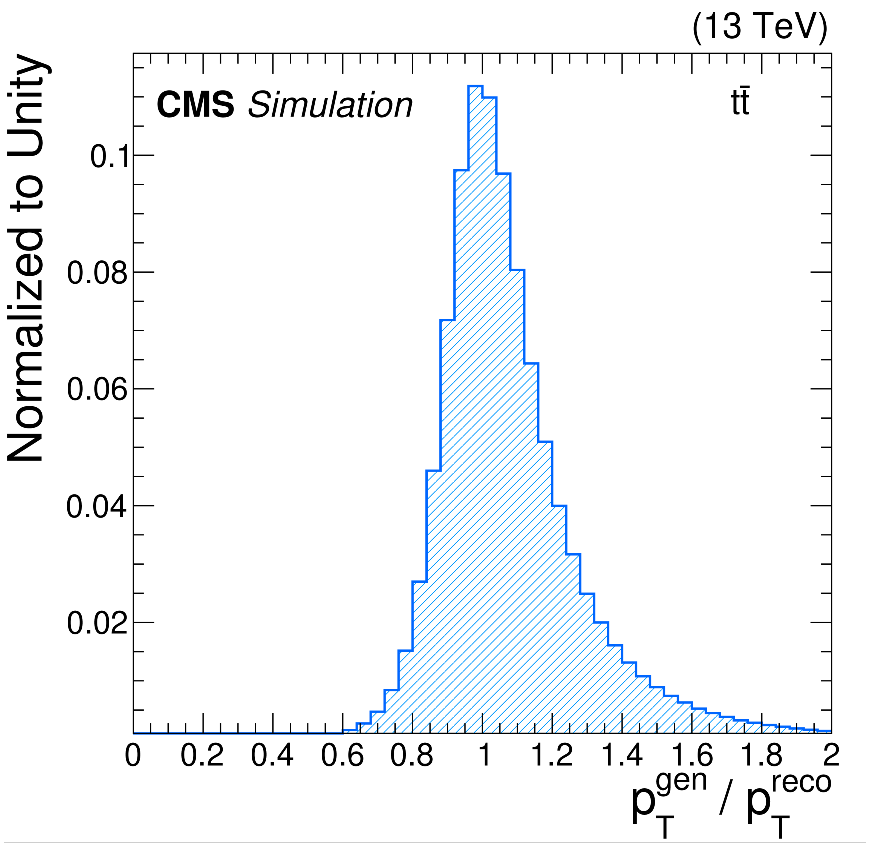
- At very high momentum, top quark decay products form a single jet
- This *top-jet* tends to have three distinct sub-clusters, while normal jets tend to be uniform
- What about using *digital image recognition* methods, e.g.:



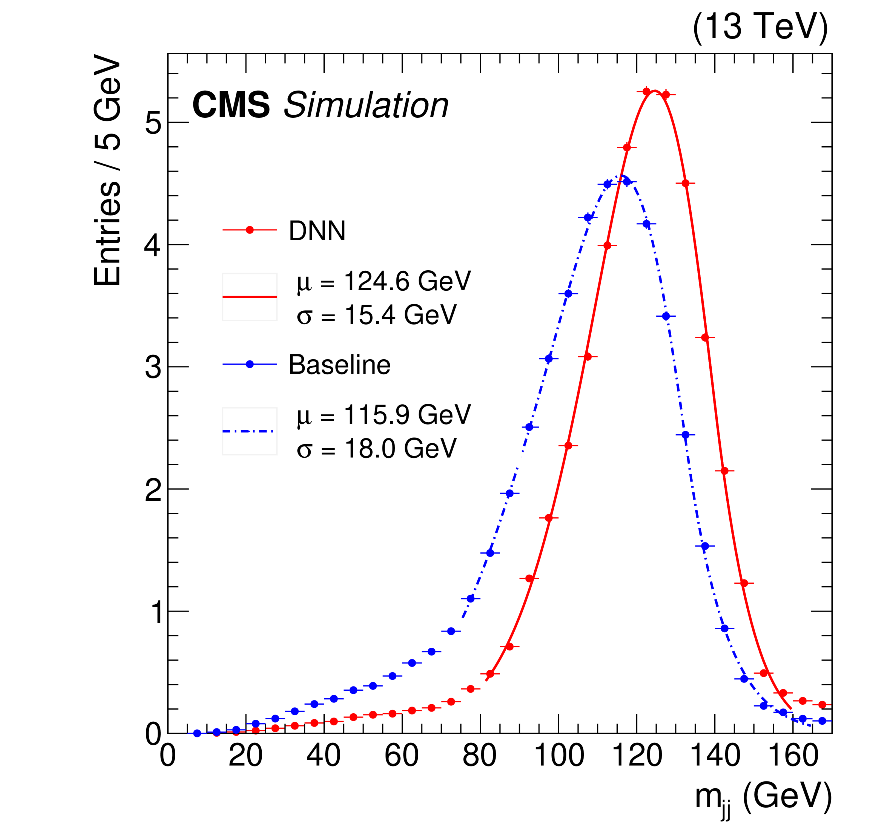


Mihailo Backovic et al, *Playing tag with ANN: boosted top identification with pattern recognition*, JHEP 1507 (2015) 086

A regression problem



Target distribution used for the training of the DNN (in simulated events without Higgs)

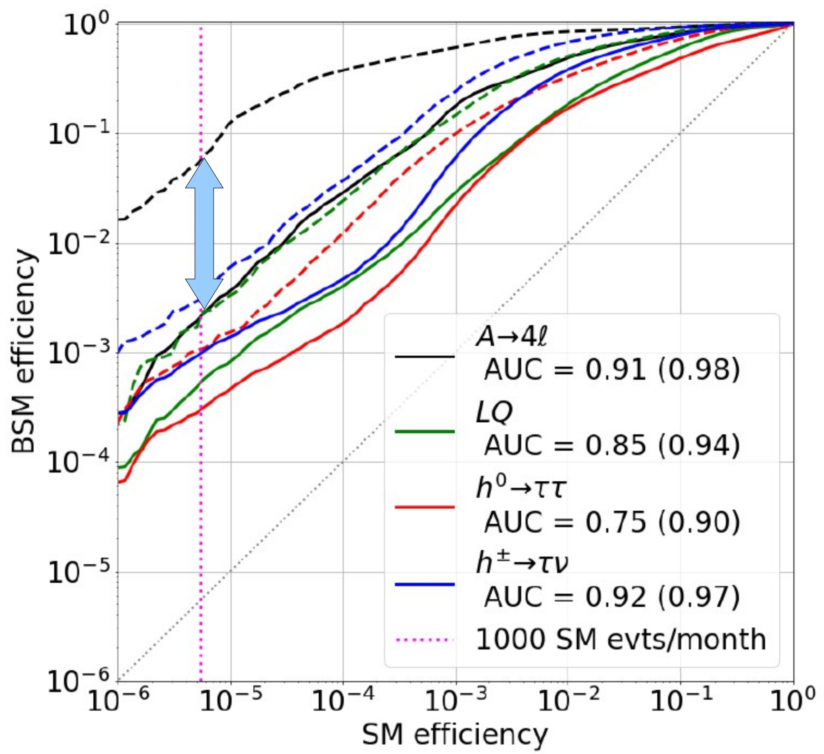


Result on Higgs mass:
less bias, better resolution

CMS collaboration, *A deep neural network for simultaneous estimation of b jet energy and resolution*, arXiv:1912.06046 [hep-ex]

Anomaly detection with Machine Learning (ML)

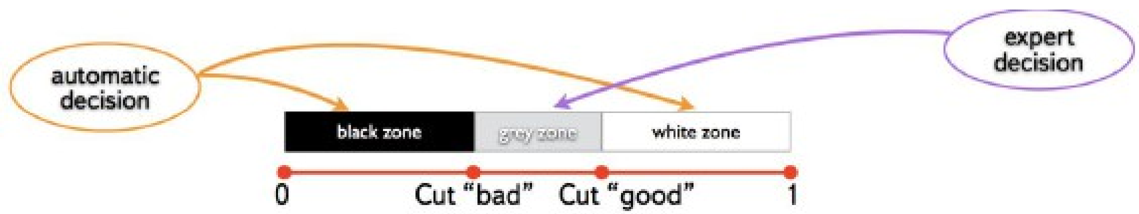
- A recent paper proposes *autoencoders* (AE) to search for new physics in LHC data in a model-independent way
- Training only on background, i.e. simulated Standard Model events
- Unsurprisingly, for any specific model a traditional BDT (dashed curves) is more efficient than AE (solid curves)
- Authors suggest a two-steps strategy: catch anomalies *at trigger level* with fast AE, to then characterise with dedicated analysis



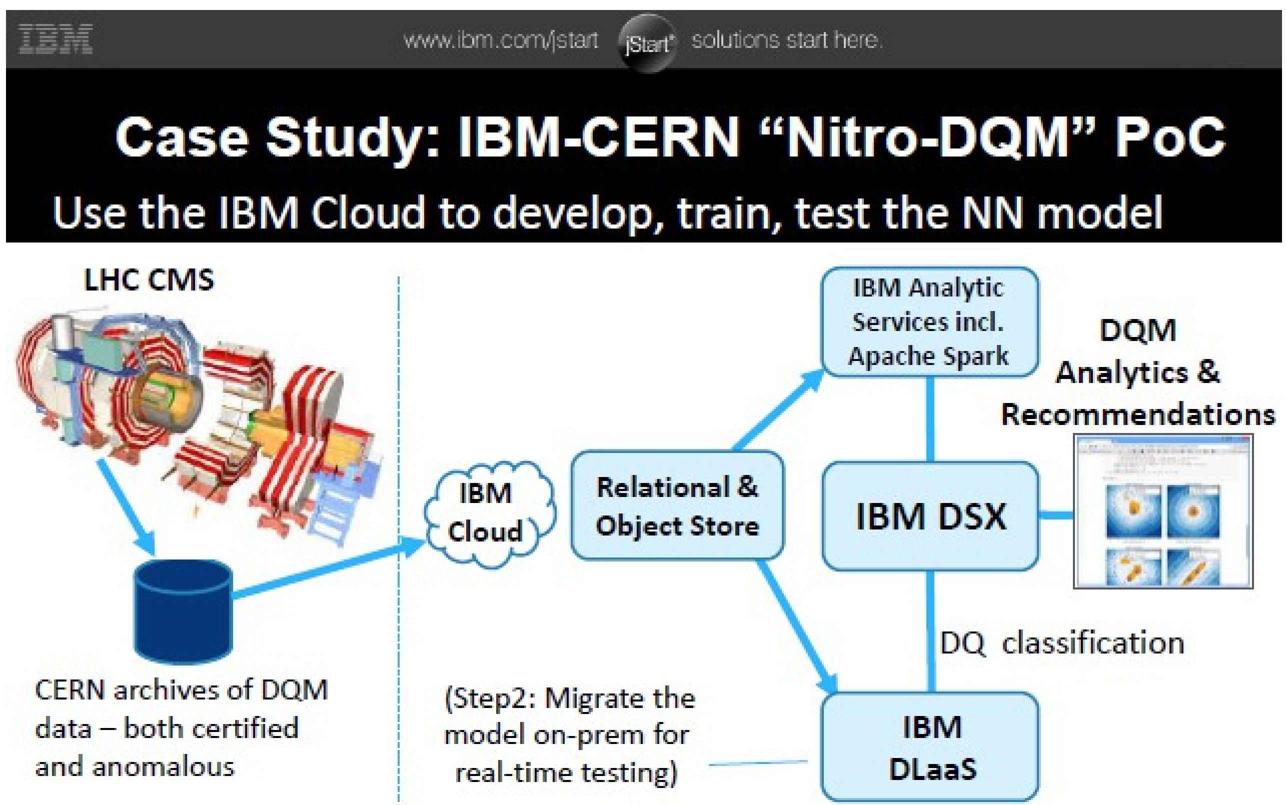
O. Cerri et al., *Variational Autoencoders for New Physics Mining at the Large Hadron Collider*, arXiv:1811.10276 [hep-ex], JHEP (2019) 36

Another kind of anomaly detection:

Data Quality Monitoring (DQM)



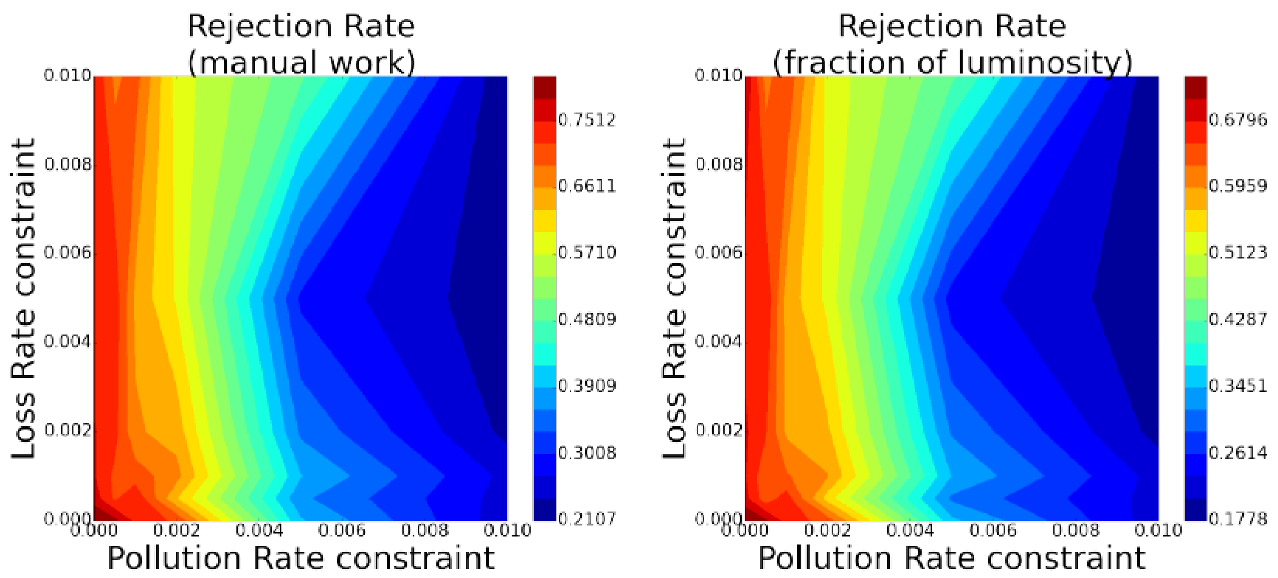
Goal: minimize grey zone, save time of humans



Pictures from Yandex and IBM speakers at the Inter-experimental Machine Learning Working Group Workshop on Machine Learning

Another kind of anomaly detection:
Data Quality Monitoring (DQM)

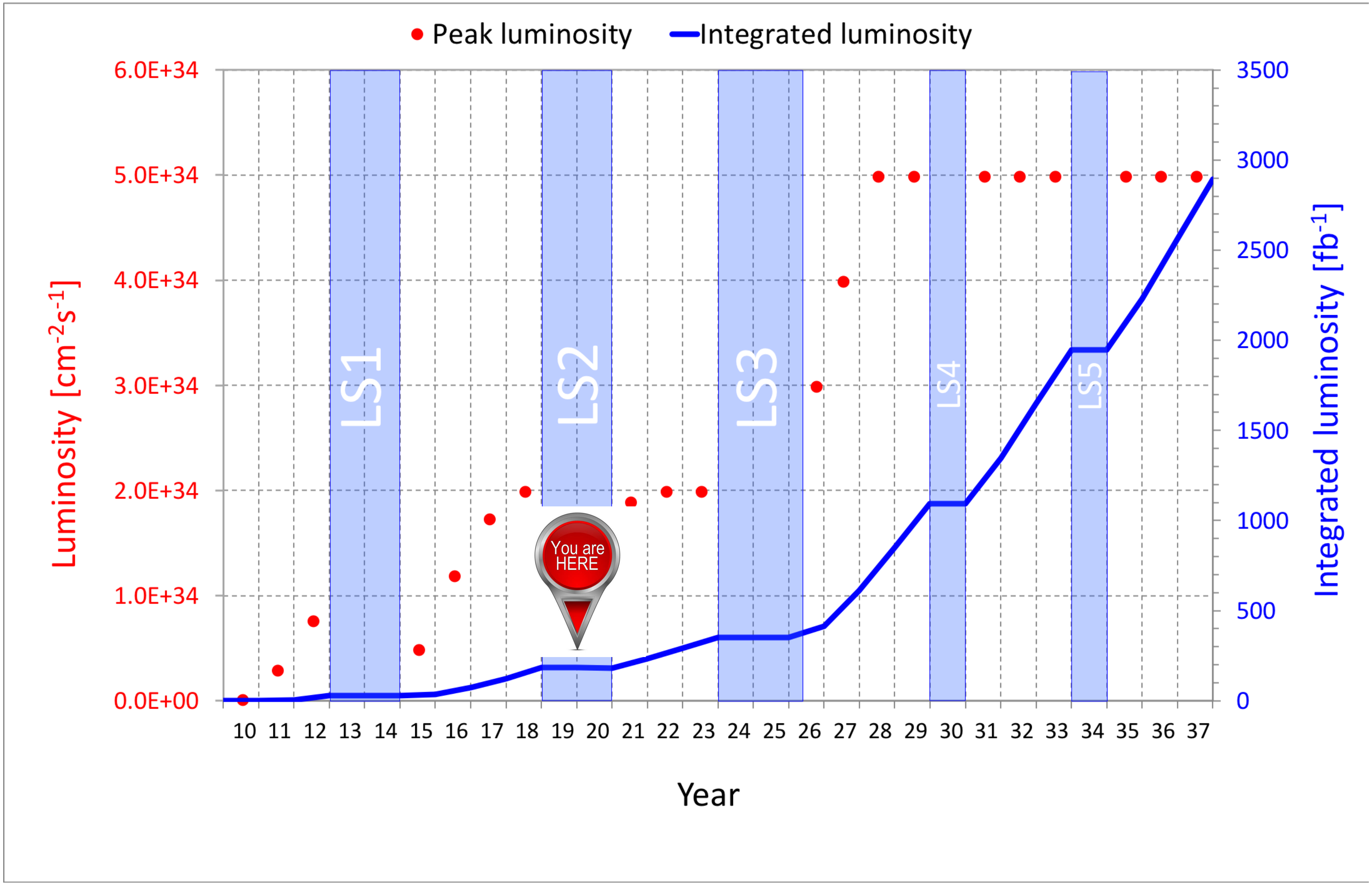
Goal: minimize grey zone, save time of humans



M. Borisyak et al., *Towards automation of data quality system for CERN CMS experiment*, arXiv:1709.08607 [physics.data-an]

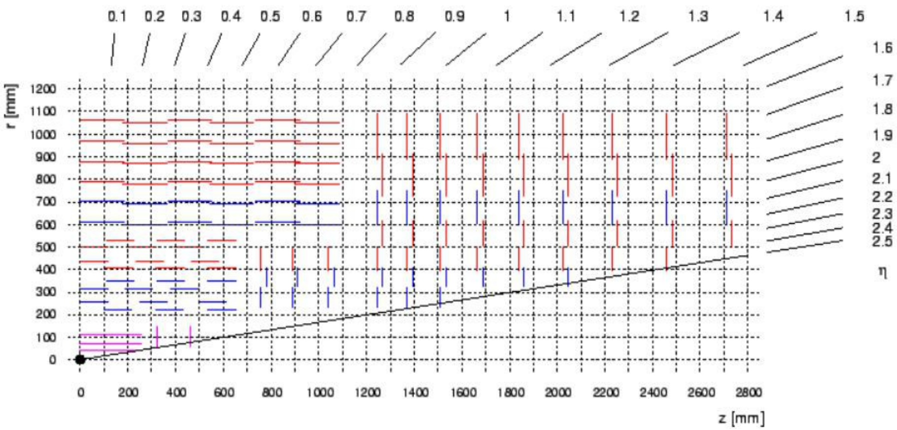
Challenges facing CMS data analysis in the 20's

Higher collision rate, more pileup, larger accumulated samples...

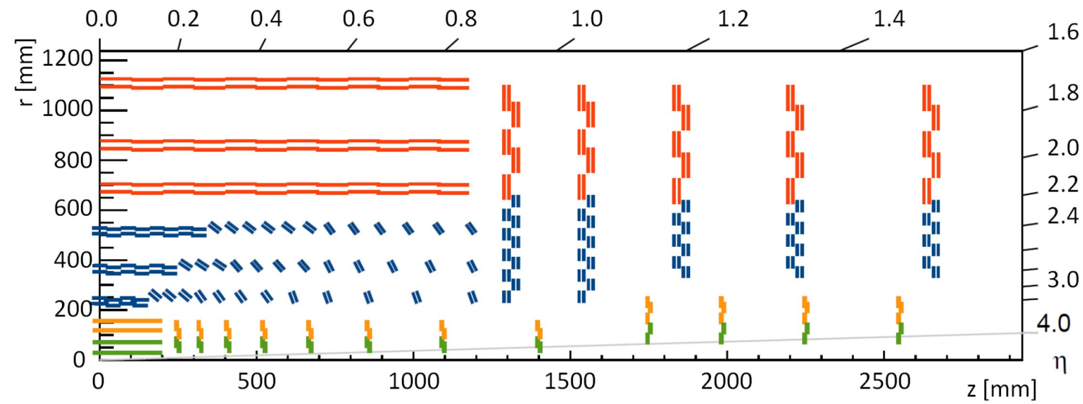


...and detectors with many more channels

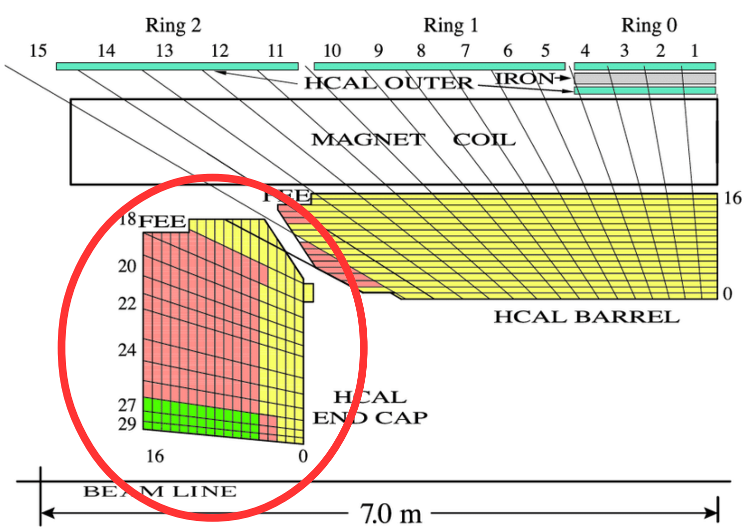
Tracker:



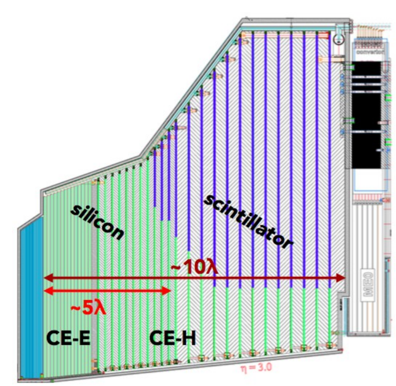
Its upgrade for Run 4:



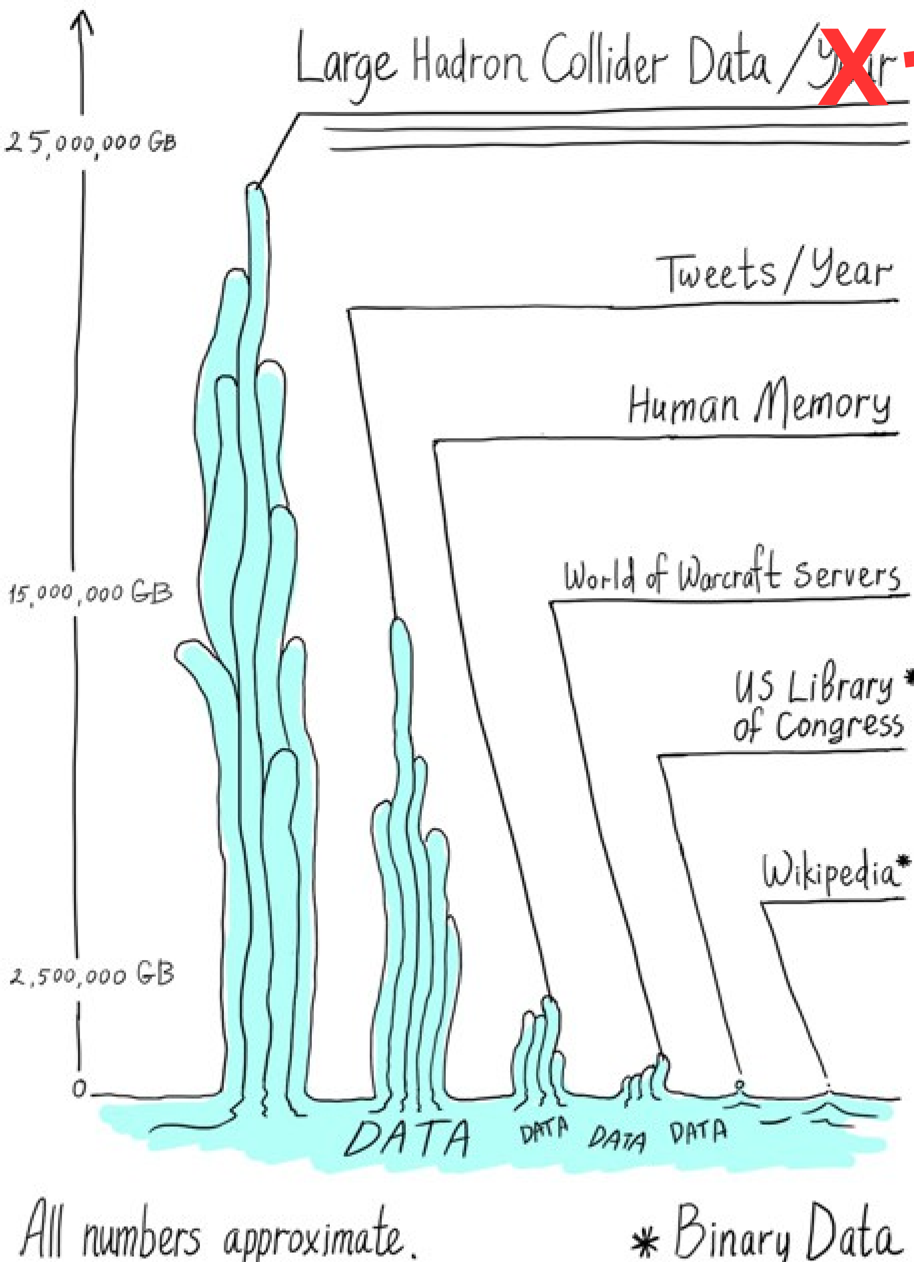
Hadron calorimeter:

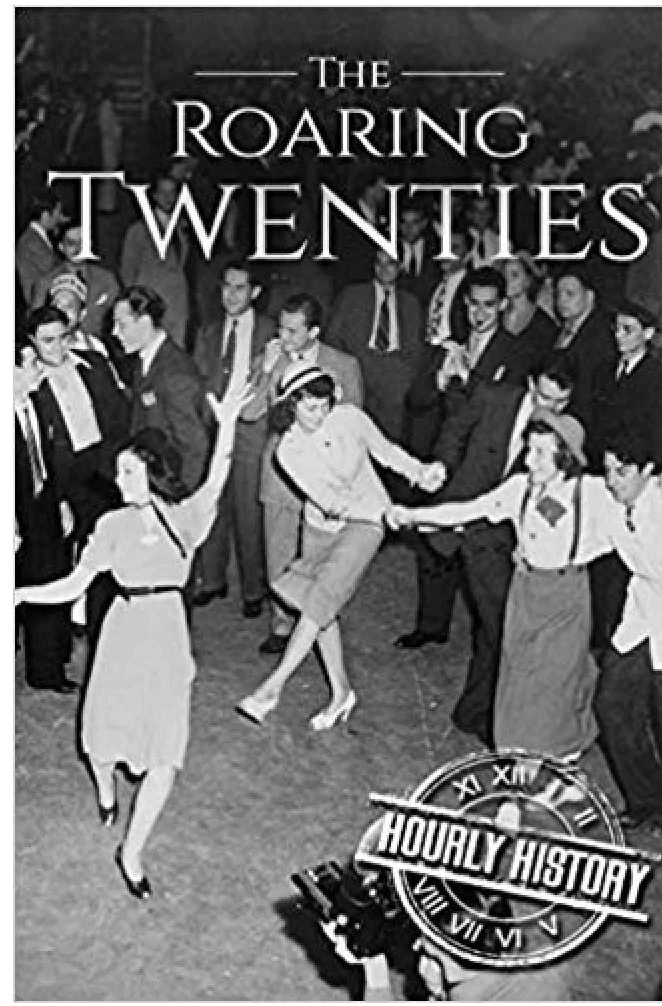


Its endcap's upgrade in Run 4:



Conclusion


Large Hadron Collider Data / Year
month
25,000,000 GB
15,000,000 GB
2,500,000 GB
0
Tweets / Year
Human Memory
World of Warcraft Servers
US Library of Congress *
Wikipedia*
DATA
DATA
DATA
DATA
All numbers approximate.
* Binary Data


THE
ROARING
TWENTIES
HOURLY HISTORY

Some of my sources:

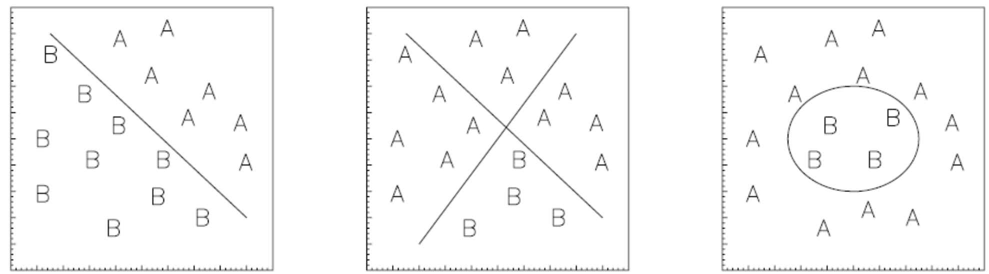
- Marco Rovere
  - Who clarified the role of CA in CMS tracking and pointed me to the *Connecting The Dots* talks
- Boris Mangano
  - Who made the pedagogical cartoons on CMS tracking and explained me several practical tracking issues
- Fosco Loregian and Michael Weiss
  - Who gave me "math feedback" on an early draft
- I also stole material from the sources acknowledged in my slides, plus F. Ragusa, F. Pantaleo, S. Neuhaus, G. Salam, M.Kagan

Statistical significance

- Given some data X and a suitable test statistic T one starts with the p-value, i.e. the probability of obtaining a value of T at least as extreme as the one observed, if H0 is true.
- p can be converted into the corresponding number of "sigma," i.e. standard deviation units from a Gaussian mean. This is done by finding x such that the integral from x to infinity of a unit Gaussian N(0,1) equals p:

$$\frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-\frac{t^2}{2}}dt = p$$

Slide partially copied from T.Dorigo



(a) (b) (c)

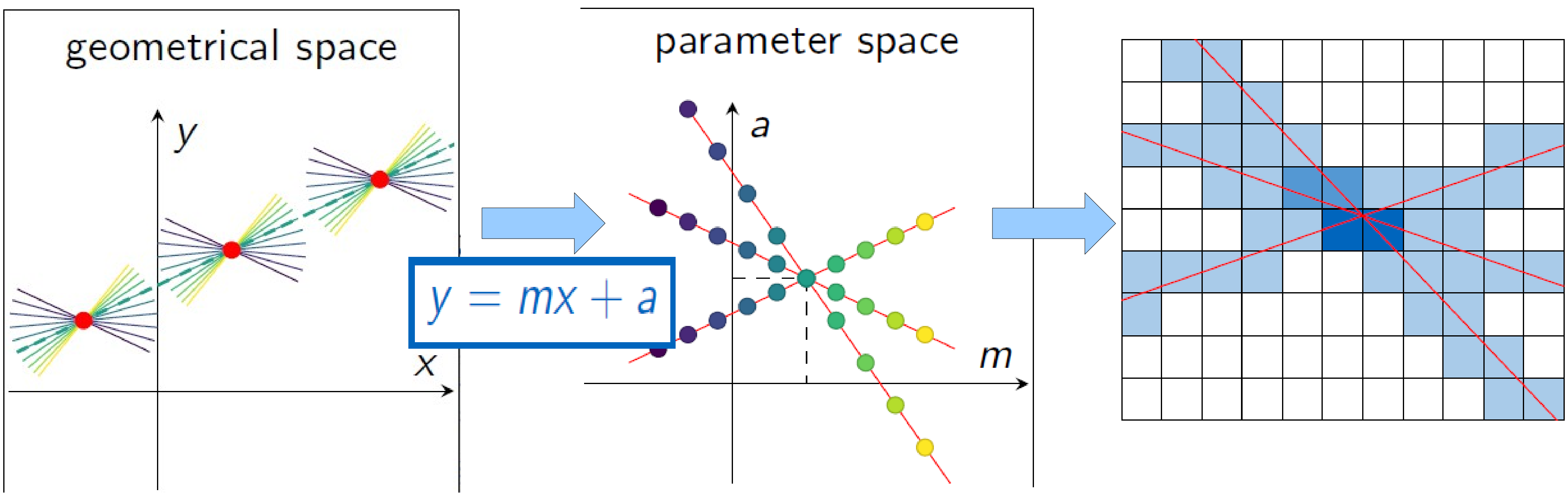
Figure 5.3: (a) Linearly separable problem, (b) non-linearly separable problem, (c) non-linearly separable problem with a closed contour.

Fig. 5.3b shows an example of "non-linearly separable problem": no straight line can properly separate the two categories of events. The introduction of a hidden layer overcomes the problem [83]. Intuitively, dividing the neurons in the hidden layer in two independent sub-groups, each one can be used by the net to find a straight line which confines all of the "B" events on one side, even if this means to have also a lot of "A" events on the same side. Working independently, the two groups will find, in general, different lines which obtain the same result in different ways; the neuron on the output layer can then make use of both the results. Other decision surfaces having more complex shapes can need more sub-groups, and so more units in the hidden layers, to be approximated. Problems with higher complexity (like the closed contour in Fig. 5.3c) can be better approximated with the help of a second hidden layer [84]. More than two hidden layers are shown to be unnecessary [77].

Myself in 2003. This paragraph didn't age well.

Hough Transform for tracking

- The methods seen so far are all *local*: a global fit would be better but too slow: very large combinatorics of hits, and very large error matrices because errors are correlated across the whole trajectory
- Novel idea: use Hugh Transforms in tracking, following example from digital image processing



Points are hits;
Lines are possible trajectories;
Dashed line is true one

Points are trajectories;
Point where the lines intersect gives the best-fit trajectory

Reduce dimensionality by binning and choose by majority vote

40 years ago

Image credit: CERN archive