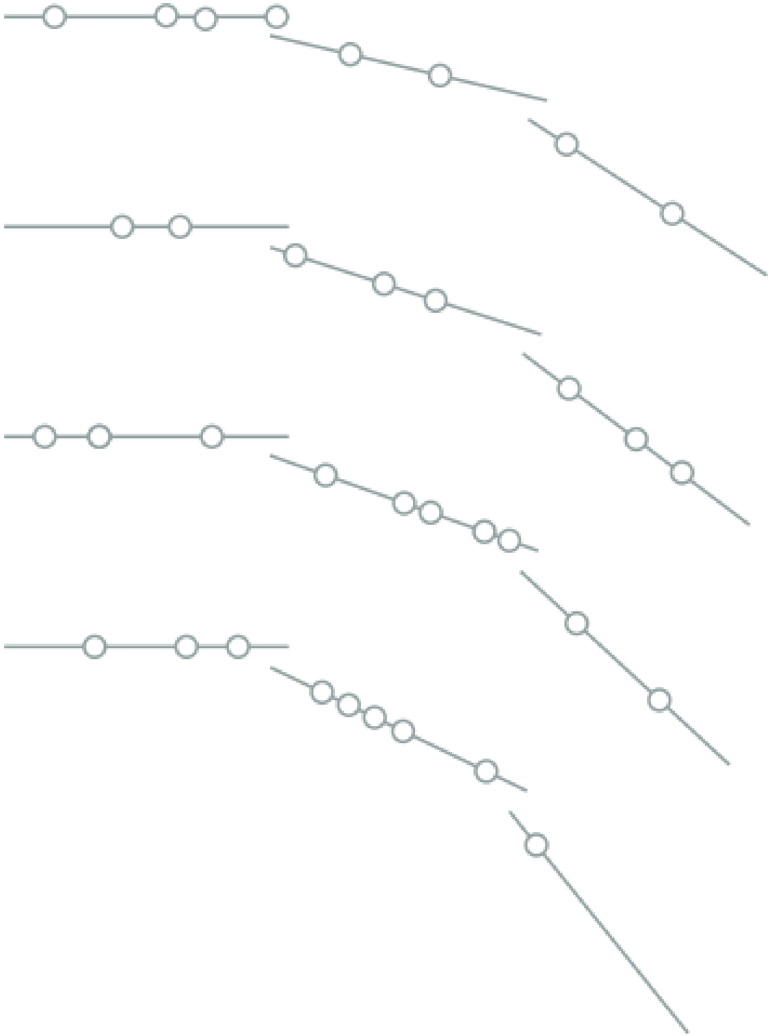
The TrackML challenge

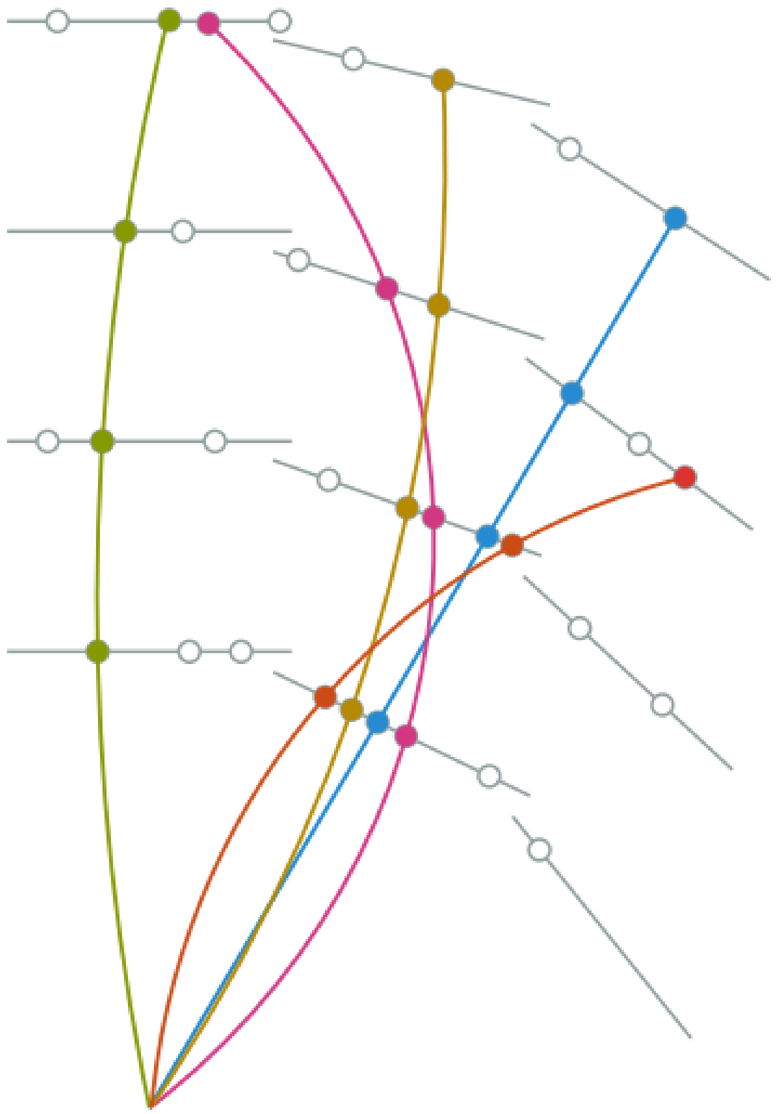
a retrospective

Moritz Kiehn (for the TrackML organizers)

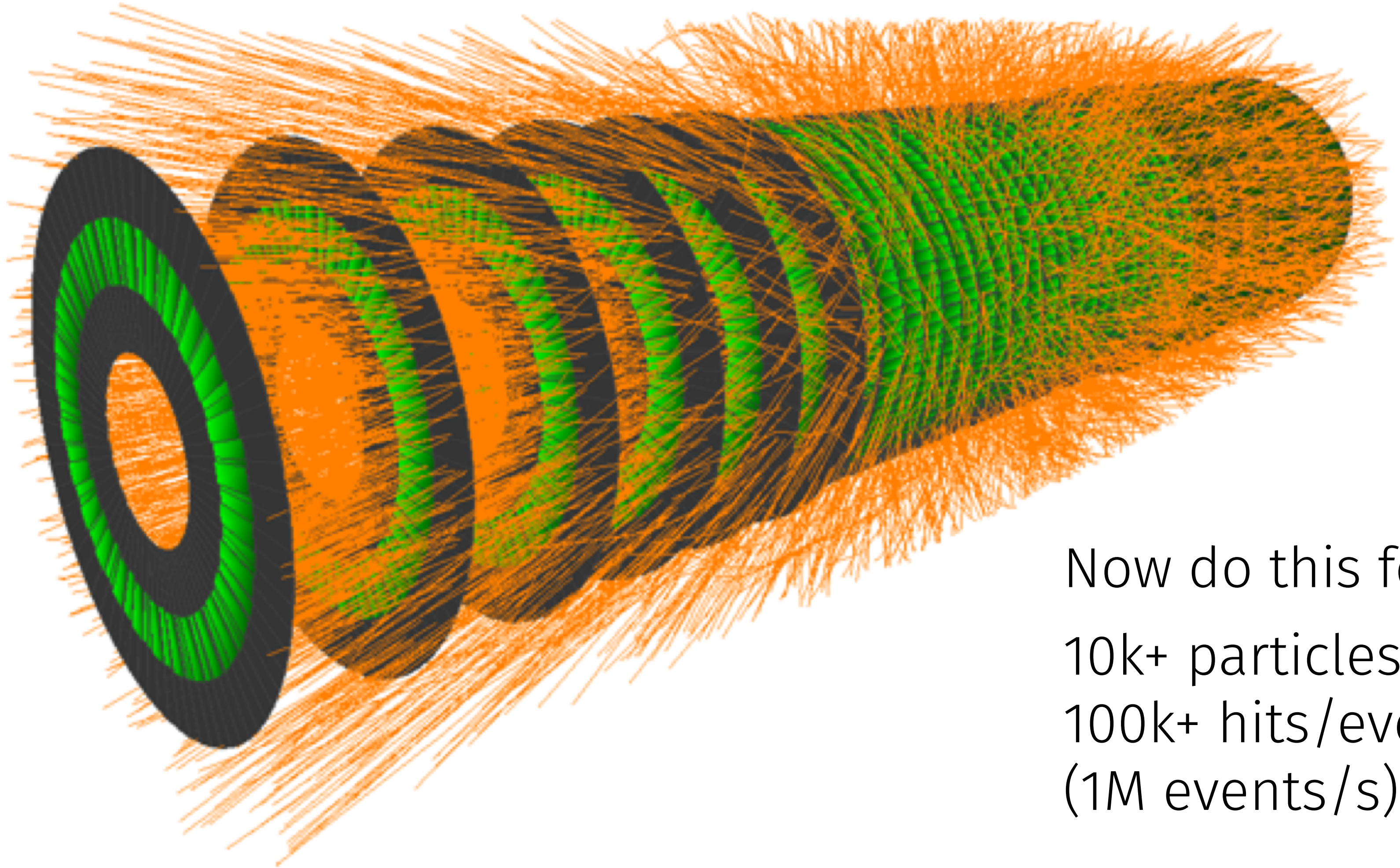
Université de Genève

Learning to Discover
Institute Pascal, Orsay, October 2019





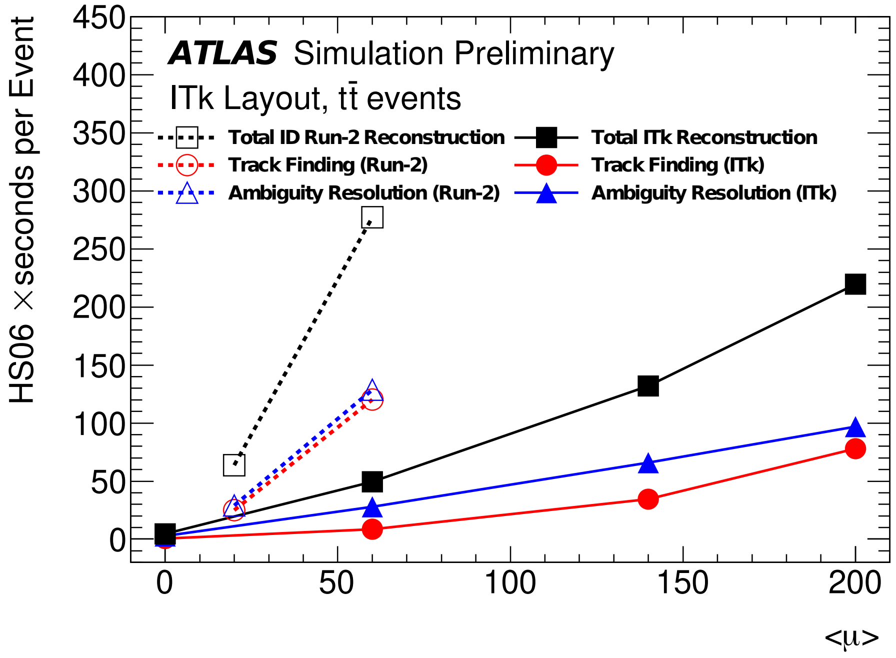
Obviously!

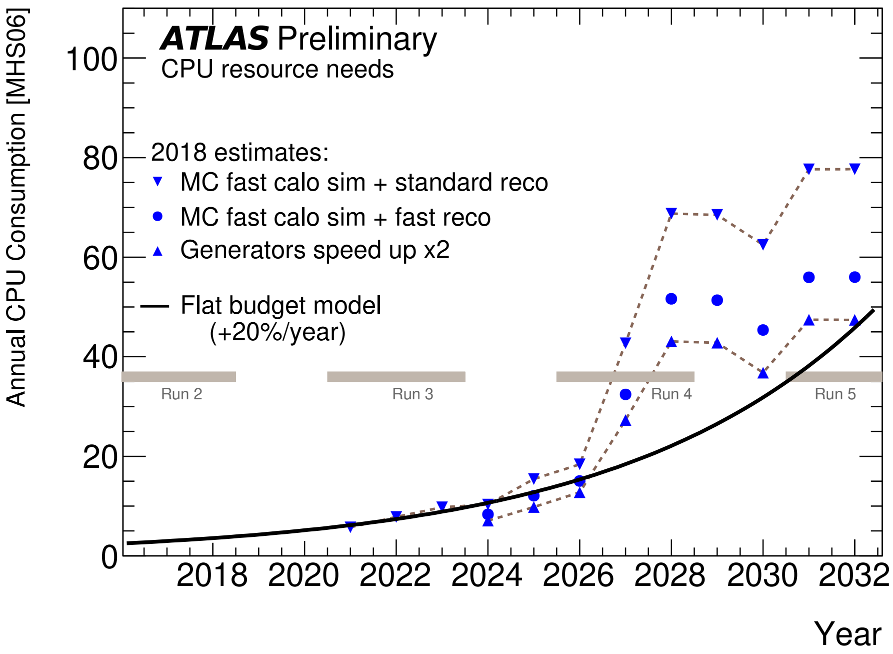


Now do this for

10k+ particles/event
100k+ hits/event
(1M events/s)

Current combinatorial approach





Aside: galactic algorithms

> A galactic algorithm is one that runs faster than any other algorithm for problems that are sufficiently large, but where "sufficiently large" is so big that the algorithm is never used in practice.
>
> Source: Wikipedia

Example: matrix multiplication
Coppersmith–Winograd $O(N^{2.374})$ vs. Strasser $O(N^{2.8074})$

Are there sub-galactic tracking algorithms, faster only for $\mu > 100$?

How can we find performant/faster/better scaling tracking algorithms?

(Pro-tip: let someone else do it)

The TrackML challenge

Modern algorithms
Non-obvious approaches

For charged particles in (high energy) colliders

A public machine learning challenge for tracking algorithms

Fresh eyes not just physicists

Cash prizes for motivated participants

Organized by

Paolo Calafiura, Steven Farrell, Heather Gray (LBNL Berkeley), Jean-Roch Vlimant (Caltech), Isabelle Guyon (ChaLearn, U Paris Saclay), Laurent Basara, Cécile Germain (LAL/LRI, U Paris Saclay), David Rousseau, Yetkin Yilnaz (LAL Orsay, U Paris Saclay), Vincenzo Innocente, Andreas Salzburger (CERN), Ilija Vukotic (U of Chicago), Tobias Golling, Moritz Kiehn, Sabrina Amrouche (U Genève), Edward Moyse (U of Massachusetts), Vava Gligorov (LPNHE Paris), Mikhail Hushchyn, Andrey Ustyuzhanin (Yandex)

TrackML sponsors

Sponsored by

CERN

kaggle

NVIDIA®

UNIVERSITÉ DE GENÈVE

Paris-Saclay Center for Data Science

INSTITUTE DATAIA
Data Science, Intelligence & Society

CHA LEARN

COMMON GROUND

CERN openlab

Inria
INVENTORS FOR THE DIGITAL WORLD

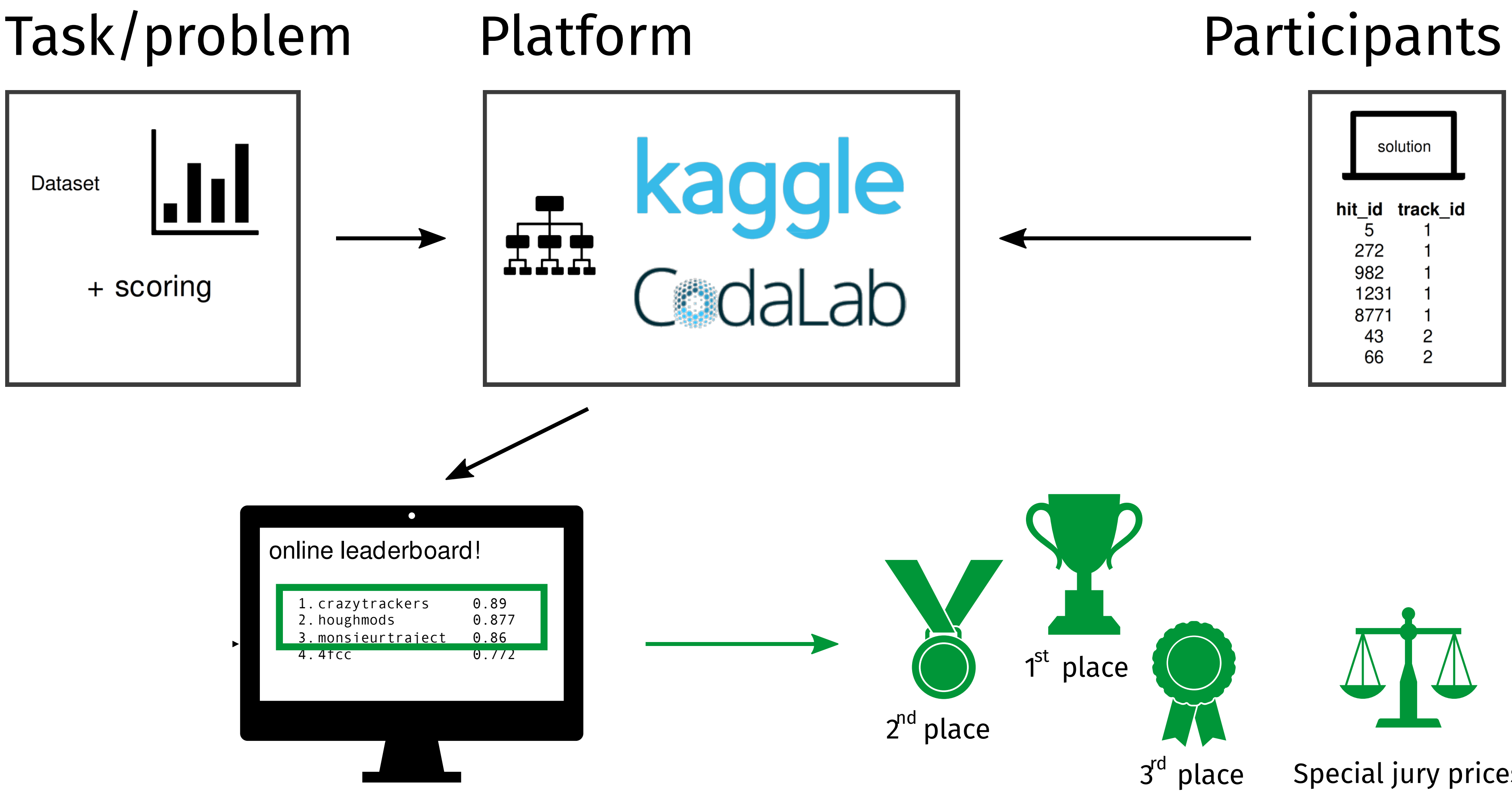
Caltech

Fermilab

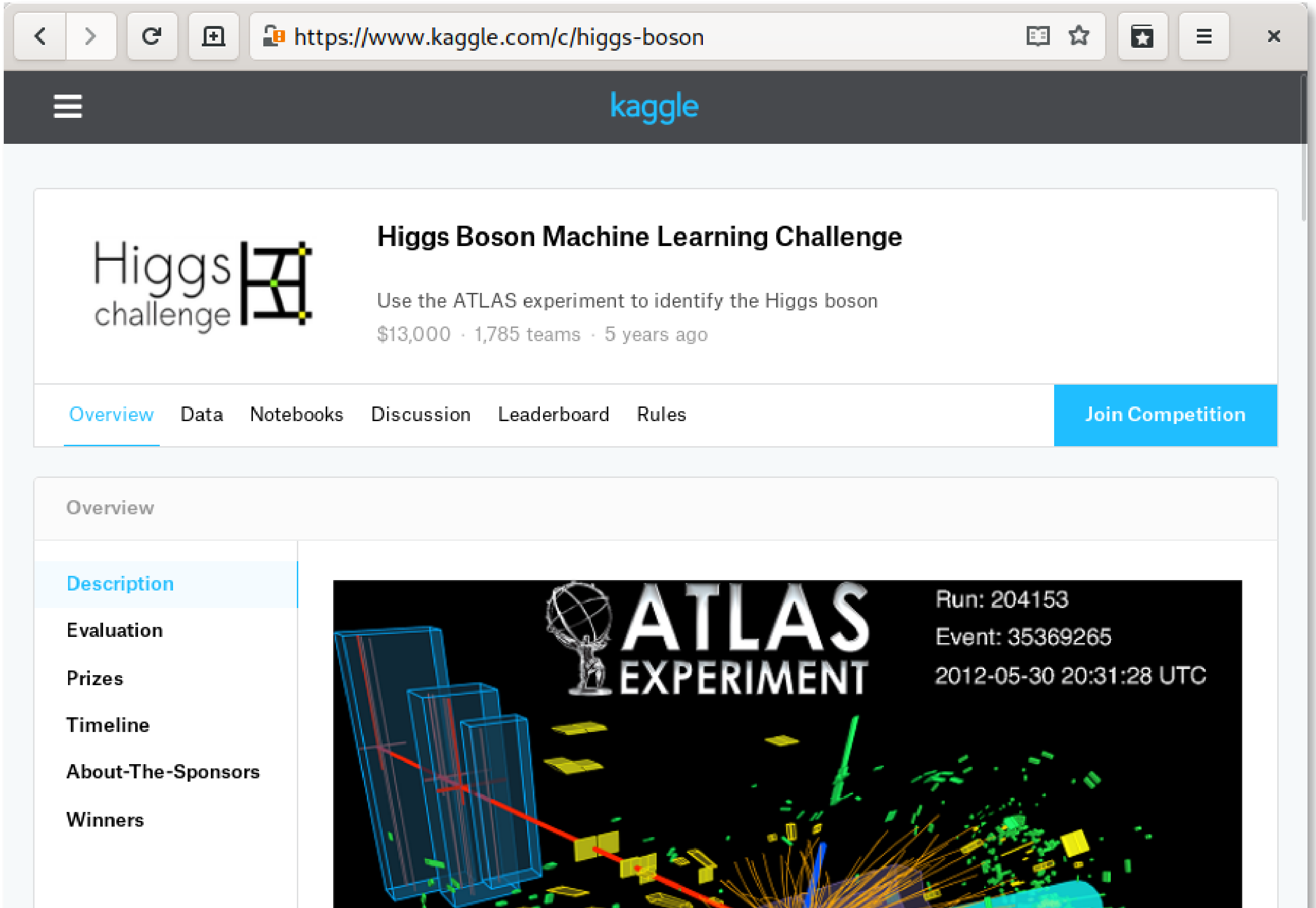
erc

UNIVERSITÉ PARIS SUD

université PARIS-SACLAY

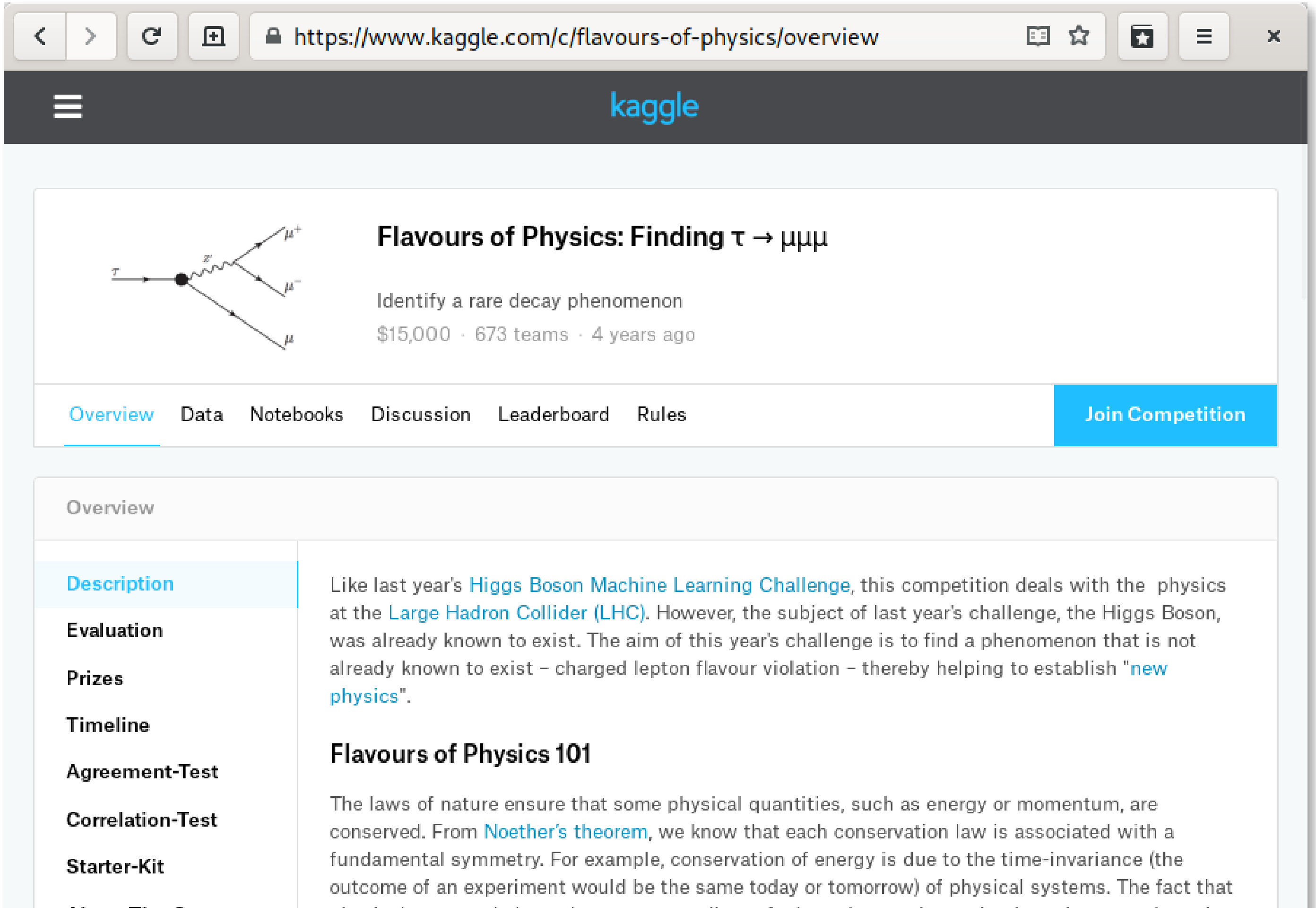

Task/problem
Platform
Participants
Dataset
+ scoring
kaggle
CodaLab
solution
hit_id track_id
5 1
272 1
982 1
1231 1
8771 1
43 2
66 2
online leaderboard!
1. crazytrackers 0.89
2. houghmods 0.877
3. monsieurtraject 0.86
4. 4fcc 0.772
1st place
2nd place
3rd place
Special jury prices

Prior art: Higgs boson challenge



Classification of Higgs events over background

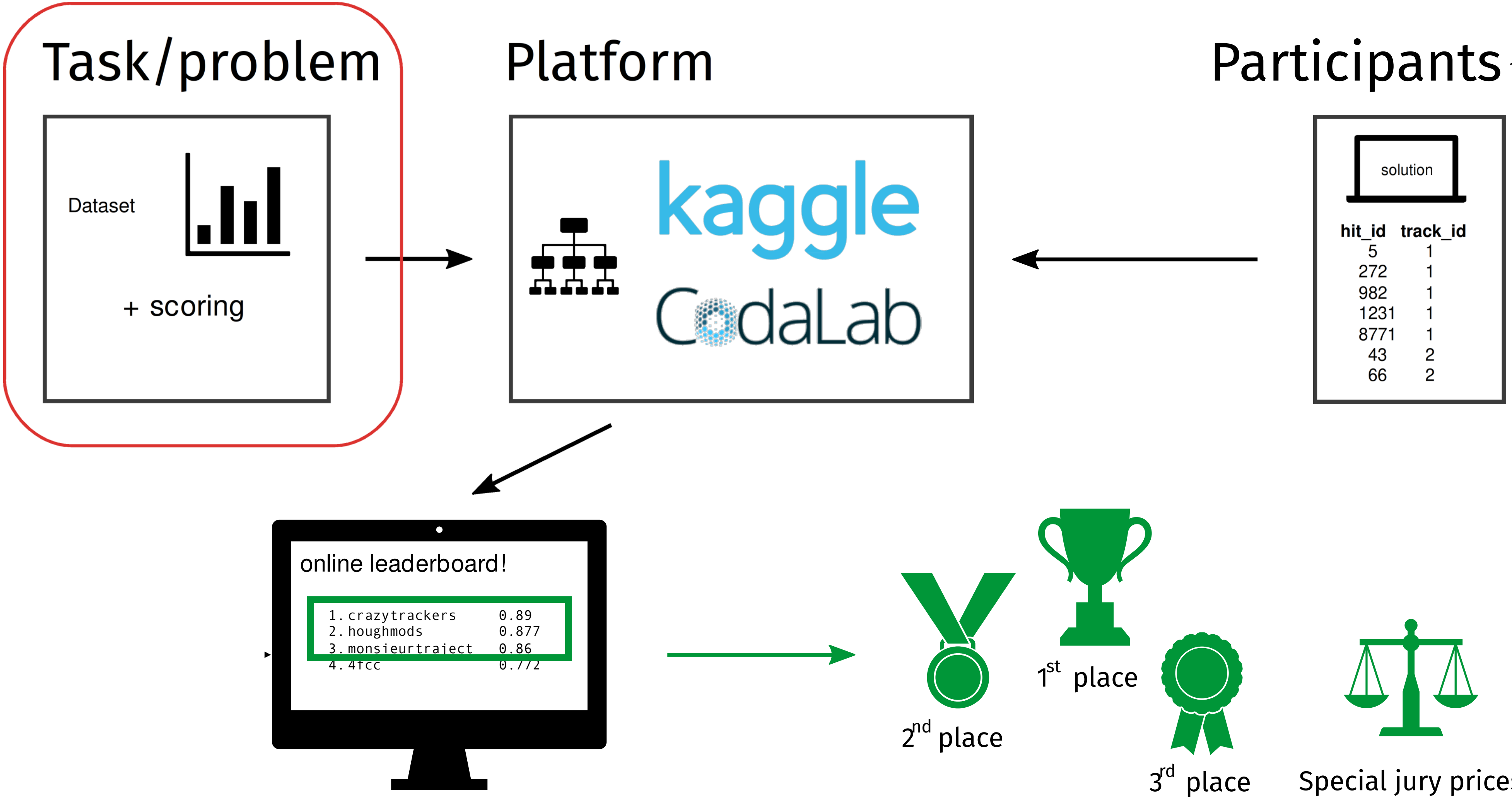
Prior art: Flavour of Physics



Identify new physics events $\tau \to \mu\mu\mu$

Includes particle-level variables

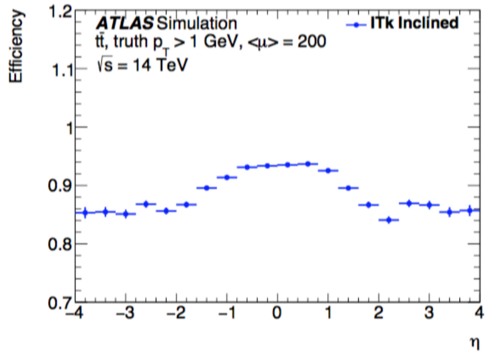
What did we want to achieve?

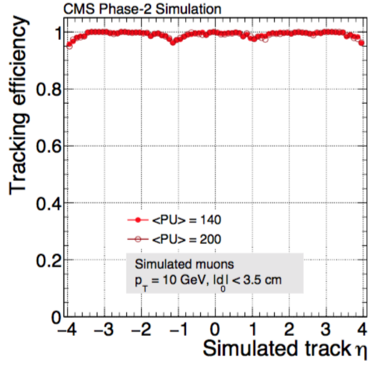

Task/problem
Platform
Participants
Dataset
+ scoring
kaggle
CodaLab
solution
hit_id track_id
5 1
272 1
982 1
1231 1
8771 1
43 2
66 2
online leaderboard!
1. crazytrackers 0.89
2. houghmods 0.877
3. monsieurtraject 0.86
4. 4fcc 0.772
1st place
2nd place
3rd place
Special jury prices

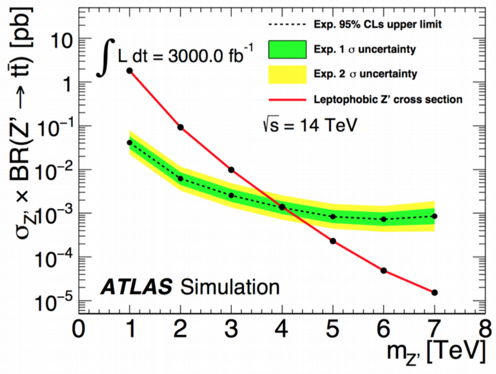
What is the problem?

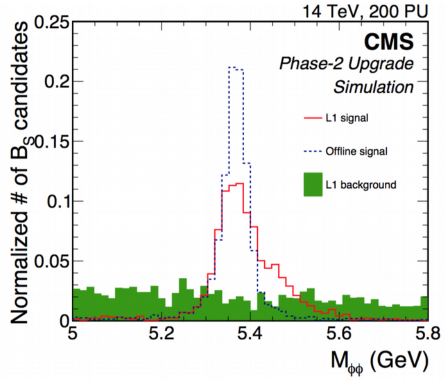
Tracking has many metrics

- Global efficiency
- Efficiency for certain classes
- Fake rate vs. purity
- Momentum resolution
- Impact resolution
- Physics impact
- …





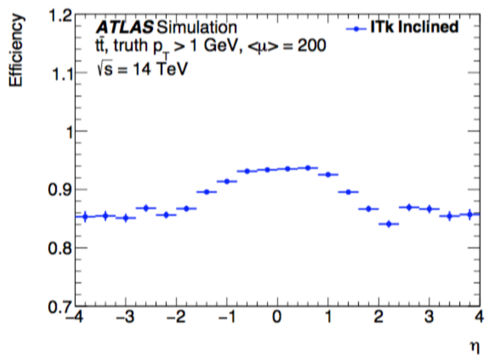


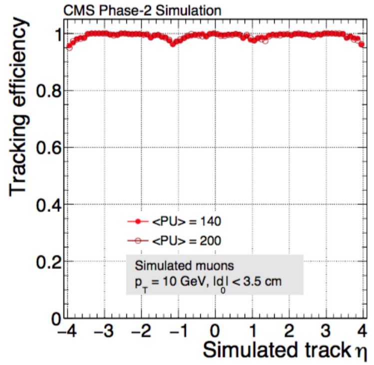


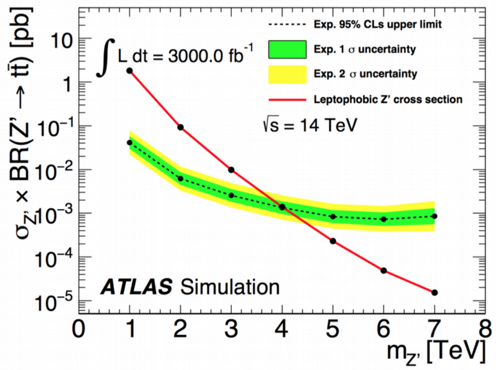
What is the problem?

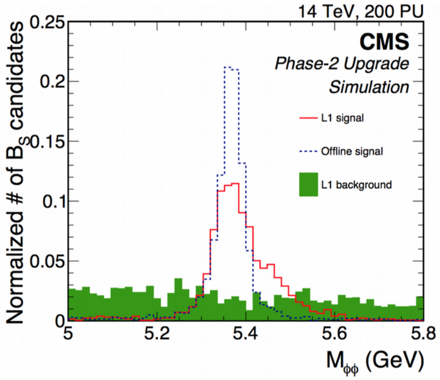
Tracking is multitudes

- Track seeding
- Track finding (extension)
- Track fitting
- Primary/secondary vertex finding
- …

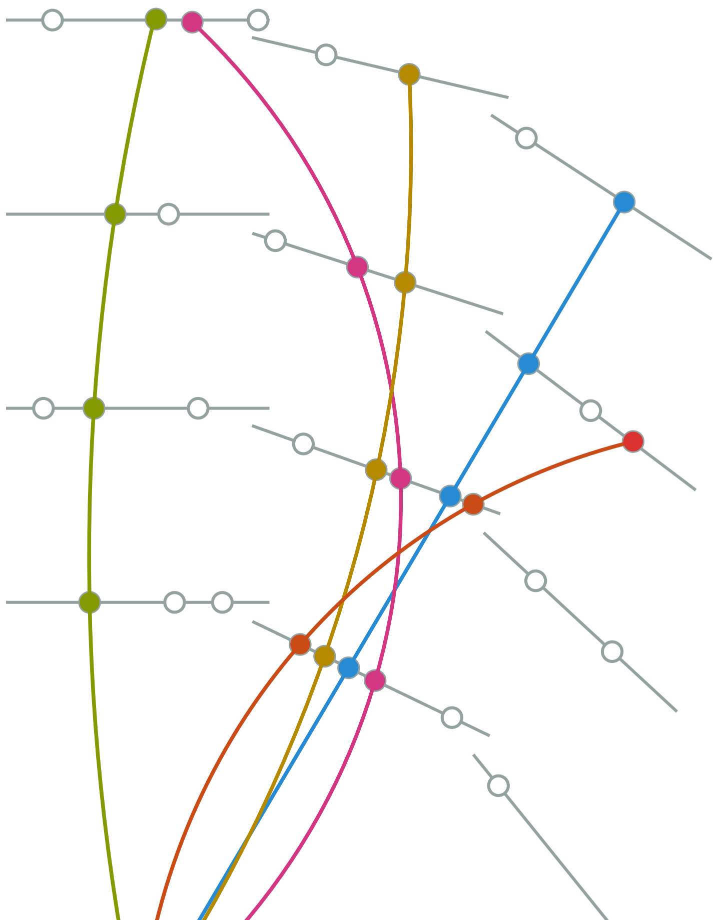








The problem is connecting the dots

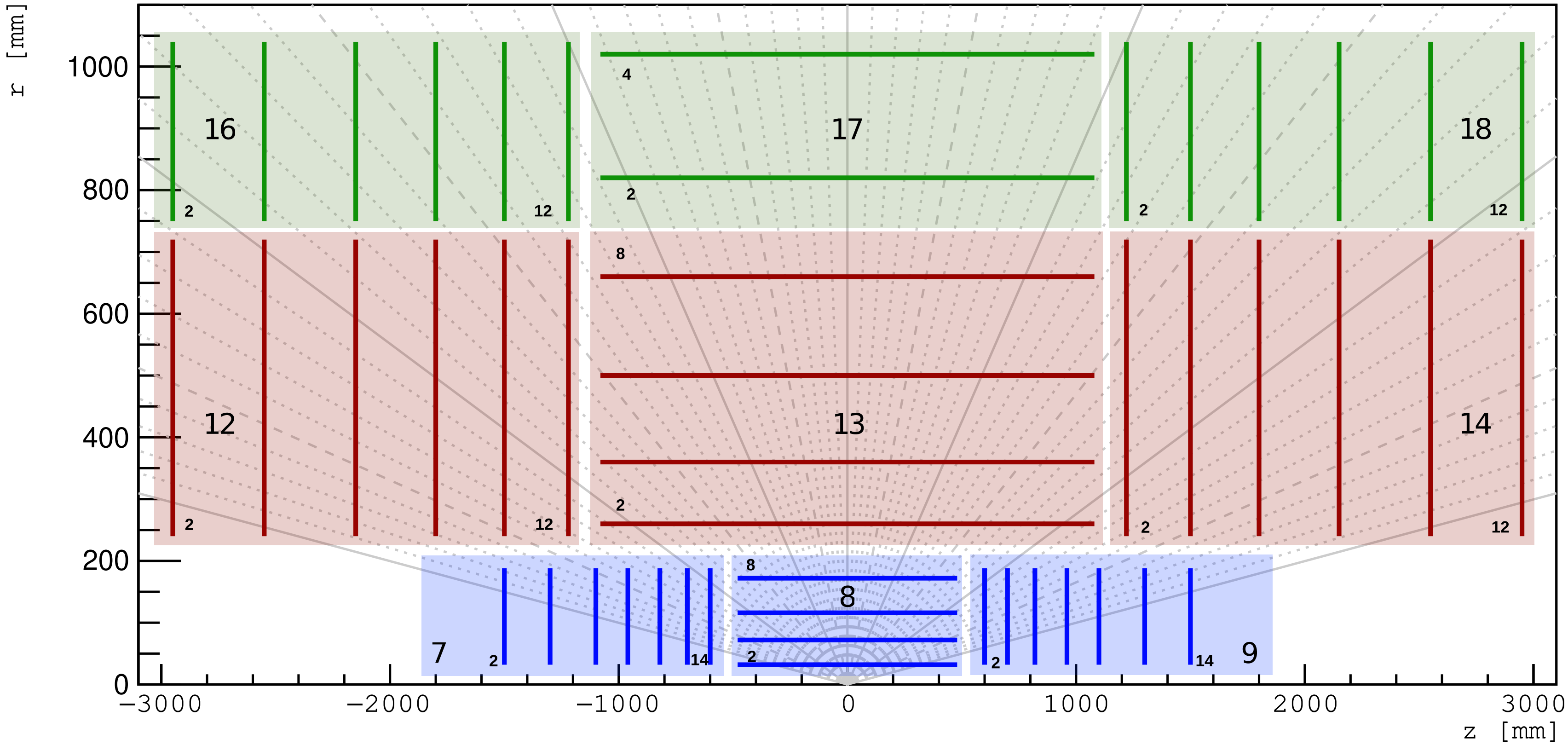


No parameter estimation
(Kalman filter works)

No hit merging/splitting
(NN mostly work)

The challenge setup

A virtual detector



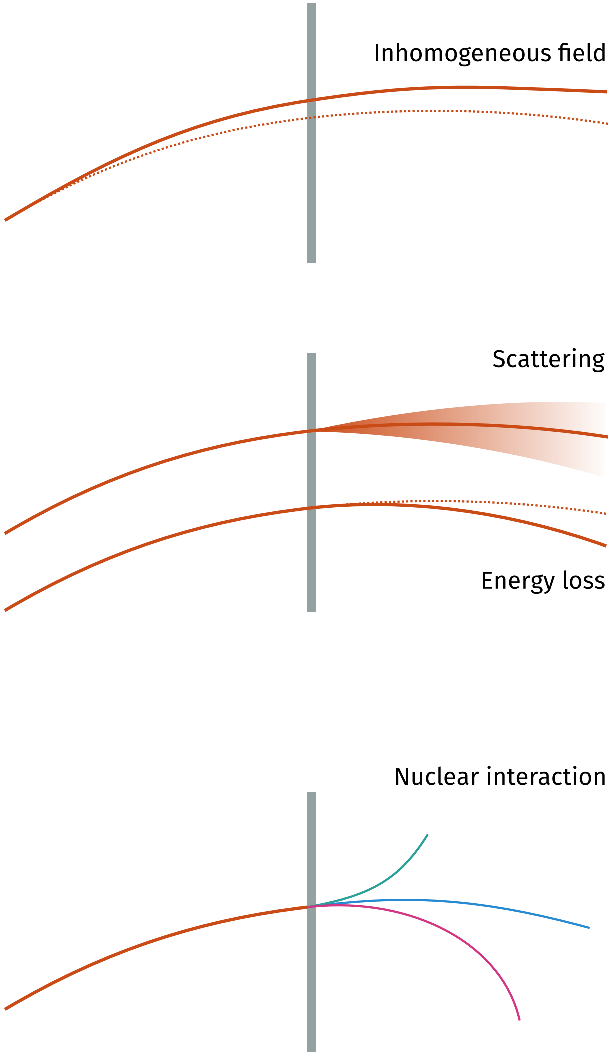
Dataset

$t\bar{t}$ + μ=200 soft QCD pileup

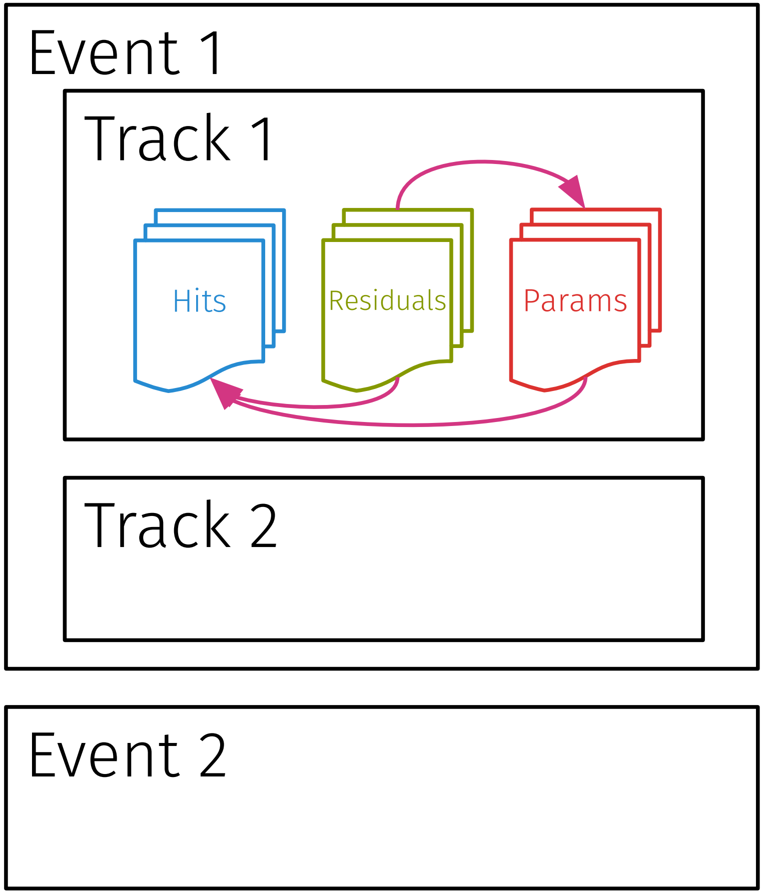
- Generated w/ Pythia8

Fast simulation based on ACTS

- Simplified geometry
- Parametric interactions
- Space points, no local info



Aside: HEP event data

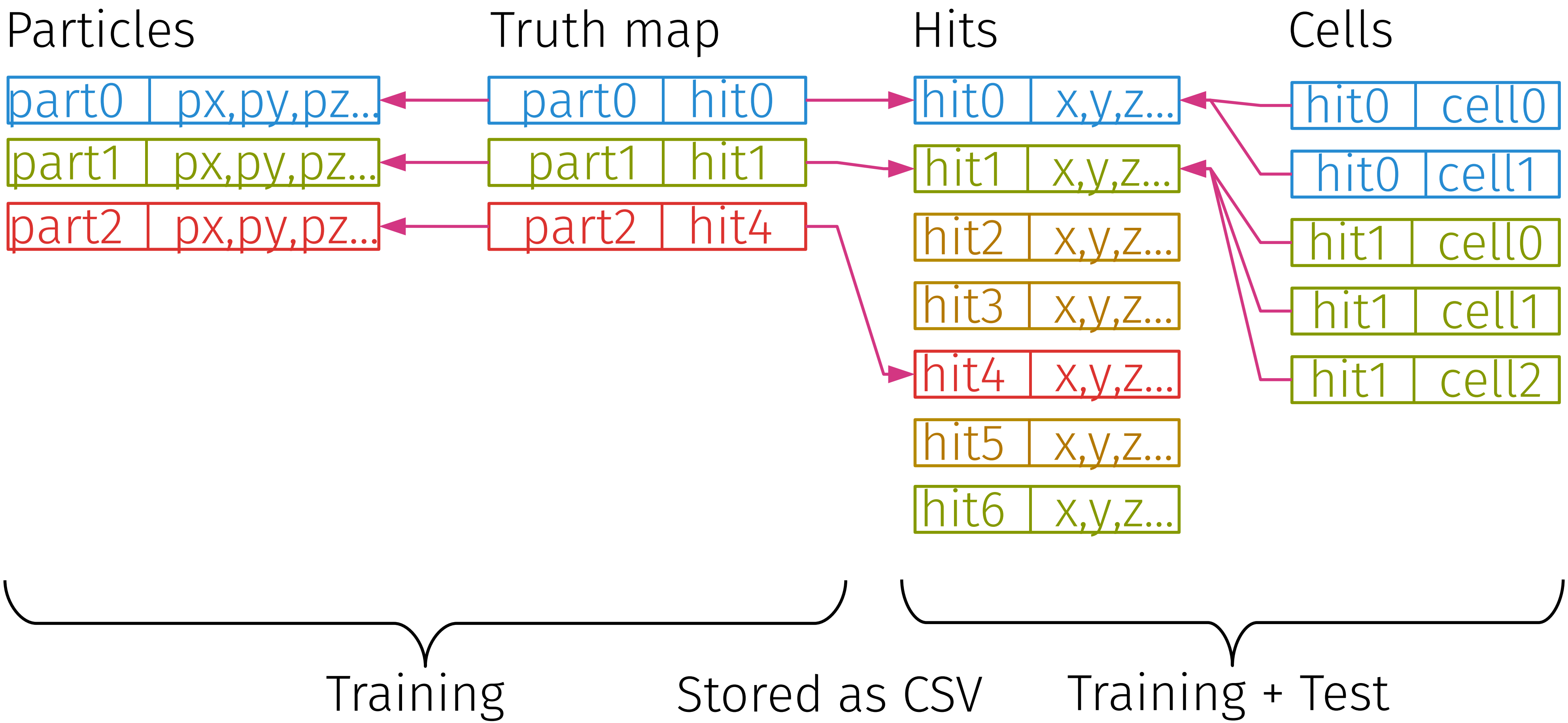


Does this look familiar to you?

```
std::vector<std::vector<double>> px;
std::map<int, std::vector<float>> something;
std::vector<std::vector<TObject*>> objects;
```

Custom, deeply-nested data structures

Everyone else likes flat data

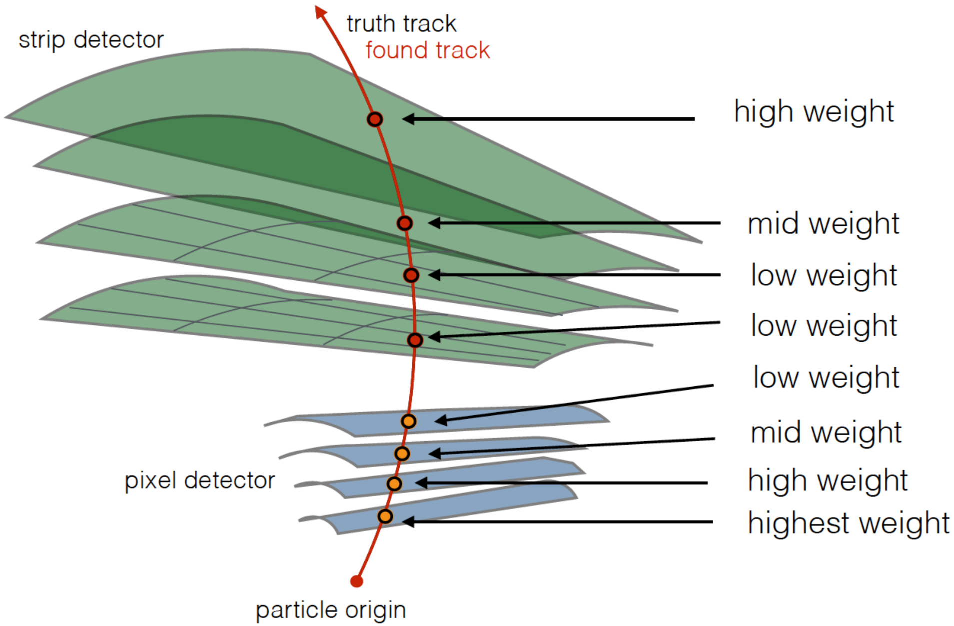


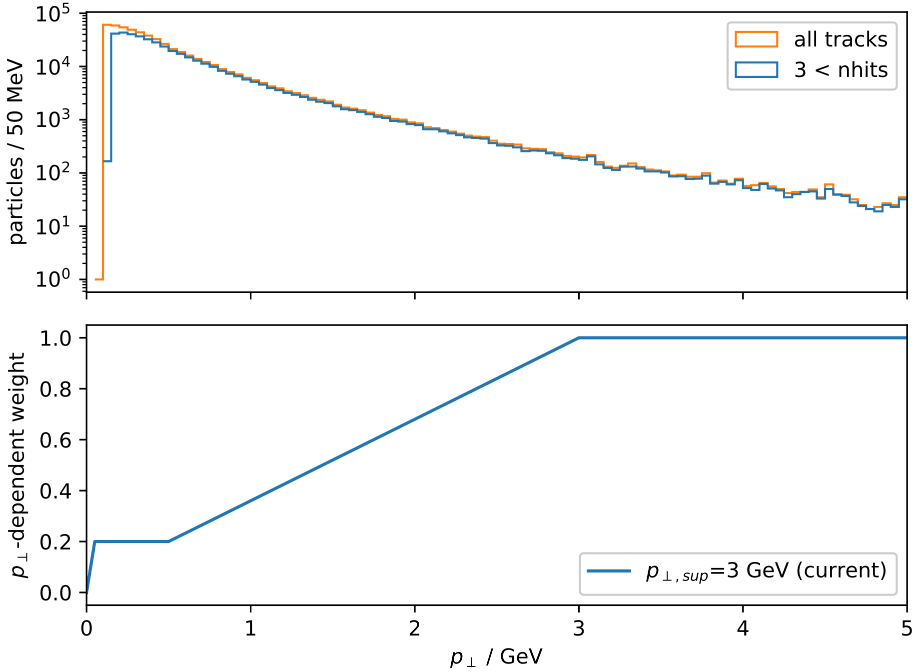
Accuracy metric

$$\text{track} = \{5, 23, 42, \ldots\}$$
$$\text{majority particle} = \{5, 17, 23, 42, \ldots\}$$
$$\text{good hits} = \text{track} \cap \text{majority particle}$$

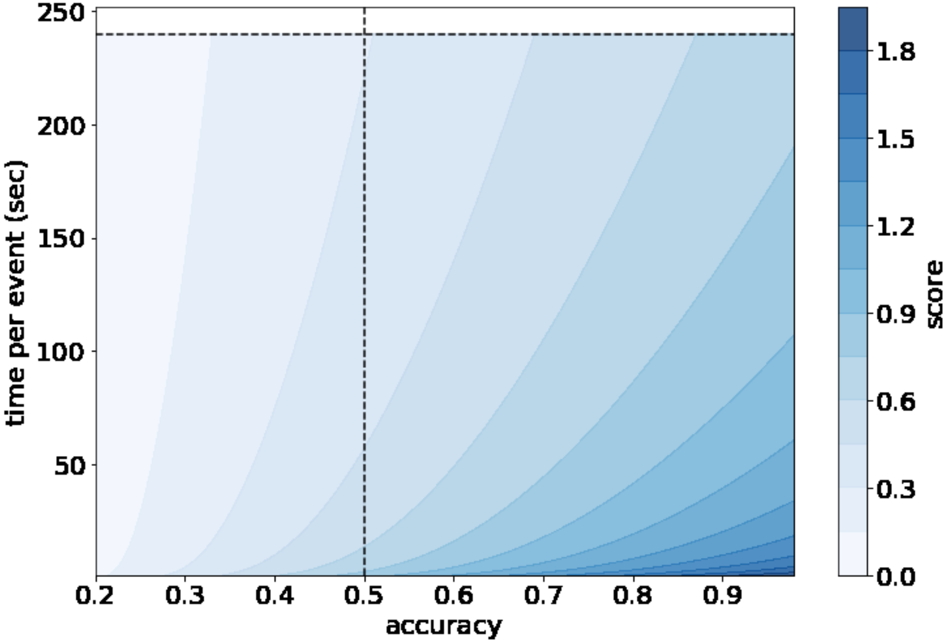
$$S \sim \sum_{\{\text{events}\}} \sum_{\{\text{tracks}\}} \begin{cases} 0 & \#\text{good hits} < 50\,\%, \#\text{hits} < 3 \\ \sum_{\{\text{good hits}\}} w_i & \text{else} \end{cases}$$
$$S_{perfect} = 1$$
$$w_i = w_i\,(\text{hit order}, \text{particle } p_\perp)$$

Accuracy metric (cont’d)



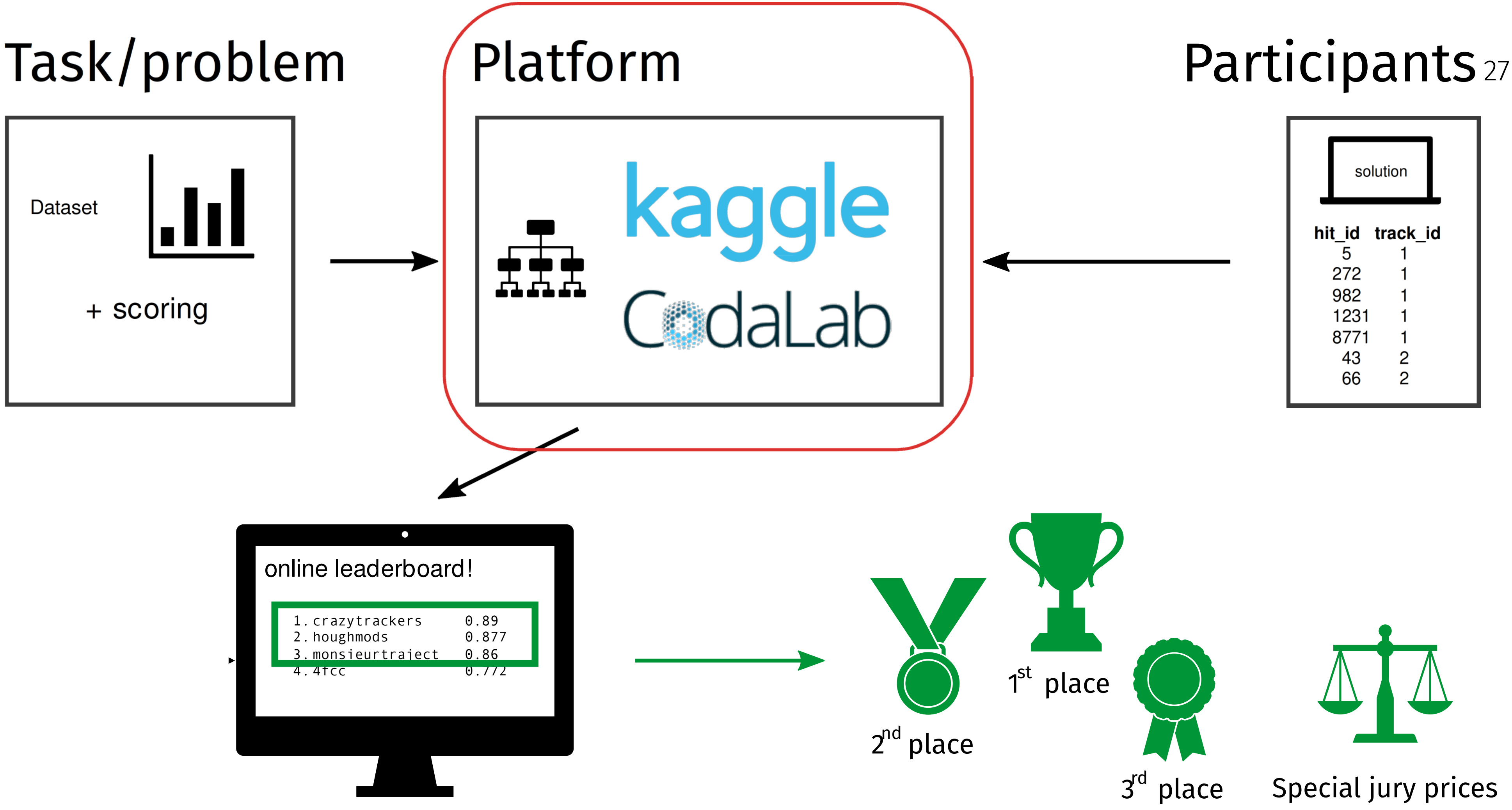


Throughput metric



Combine accuracy score and runtime

- 0 for $t > 600s$ or score < 0.5
- $\text{Log}(1 + (600s/\text{time})) \times (\text{score} - 0.5)^2$
- Score only primary particles


Task/problem
Platform
Participants
Dataset
+ scoring
kaggle
CodaLab
solution
hit_id track_id
5 1
272 1
982 1
1231 1
8771 1
43 2
66 2
online leaderboard!
1. crazytrackers 0.89
2. houghmods 0.877
3. monsieurtraject 0.86
4. 4fcc 0.772
1st place
2nd place
3rd place
Special jury prices

Accuracy phase on kaggle

Ran until August 2018

600+ participants

Submit results only

Only measure accuracy

12k€, 8k€, 5k€ prizes
+ NVIDIA V100 GPU

1	—	Top Quarks		0.92182	10
2	—	outrunner		0.90302	9
3	—	Sergey Gorbunov	HEP	0.89353	6
4	—	demelian	HEP	0.87079	35
5	—	Edwin Steiner		0.86395	5
6	—	Komaki		0.83127	22
7	—	Yuval & Trian		0.80414	56
8	—	bestfitting		0.80341	6
9	—	DBSCAN forever		0.80114	23
10	—	Zidmie & KhaVo		0.76320	26
11	—	Andrea Lonza		0.75845	15
12	—	Finnies		0.74827	56
13	—	Rei Matsuzaki		0.74035	12
14	—	Mickey		0.73217	10
15	—	Vicens Gaitan		0.70429	19
16	—	Robert		0.69955	3
17	—	Yuval-CPMP tribute band		0.69364	20
18	—	N. Hi. Bouzu		0.67573	9
19	—	Steins;Gate		0.66763	12
20	▲1	Victor Nedel'ko		0.66723	4

Throughput phase on CodaLab

Ran until March 2019

Only 10+ active participants

Submit results only

Measure accuracy and speed

7k€, 5k€, 3k€ prizes
+ NVIDIA V100 GPU

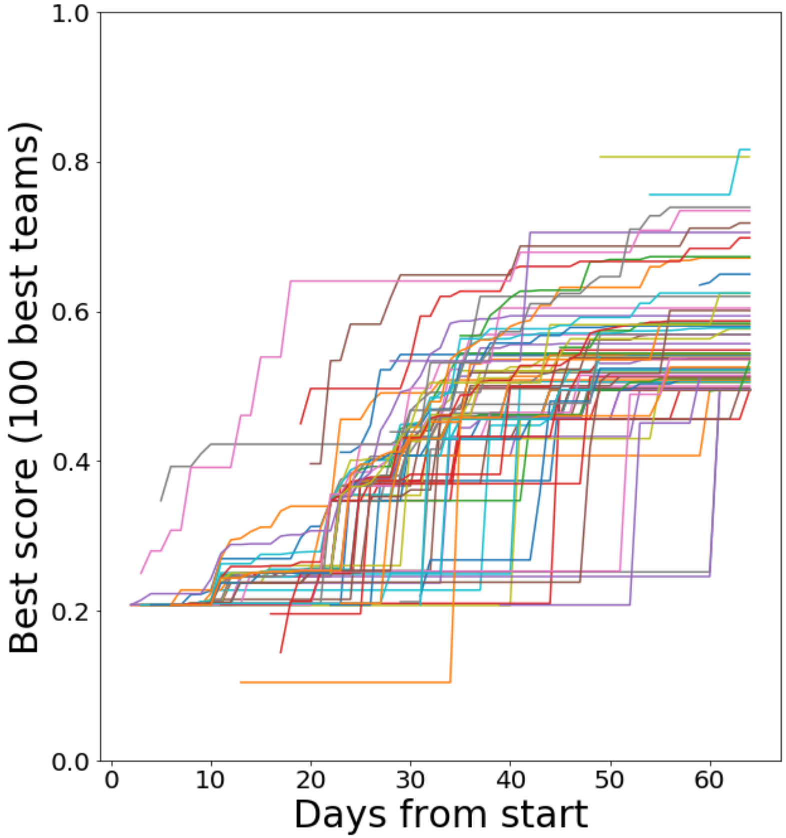
RESULTS

#	User	Entries	Date of Last Entry	score ▲	accuracy_mean ▲	accuracy_std ▲	com (sec
1	sgorbuno HEP	9	03/12/19	1.1727 (1)	0.944 (2)	0.00 (14)	28.
2	fastrack HEP	53	03/12/19	1.1145 (2)	0.944 (1)	0.00 (15)	55.
3	cloudkitchen (HEP)	73	03/12/19	0.9007 (3)	0.928 (3)	0.00 (13)	364
4	cubus	8	09/13/18	0.7719 (4)	0.895 (4)	0.01 (9)	675
5	Taka	11	01/13/19	0.5930 (5)	0.875 (5)	0.01 (12)	266
6	Vicennial	27	02/24/19	0.5634 (6)	0.815 (6)	0.01 (10)	127
7	Sharad	57	03/10/19	0.2918 (7)	0.674 (7)	0.02 (4)	190
8	WeizmannAI	5	03/12/19	0.0000 (8)	0.133 (11)	0.01 (11)	88.
9	harshakoundinya	2	03/12/19	0.0000 (8)	0.085 (13)	0.01 (6)	49.
10	iWit	6	03/10/19	0.0000 (8)	0.082 (15)	0.01 (8)	48.

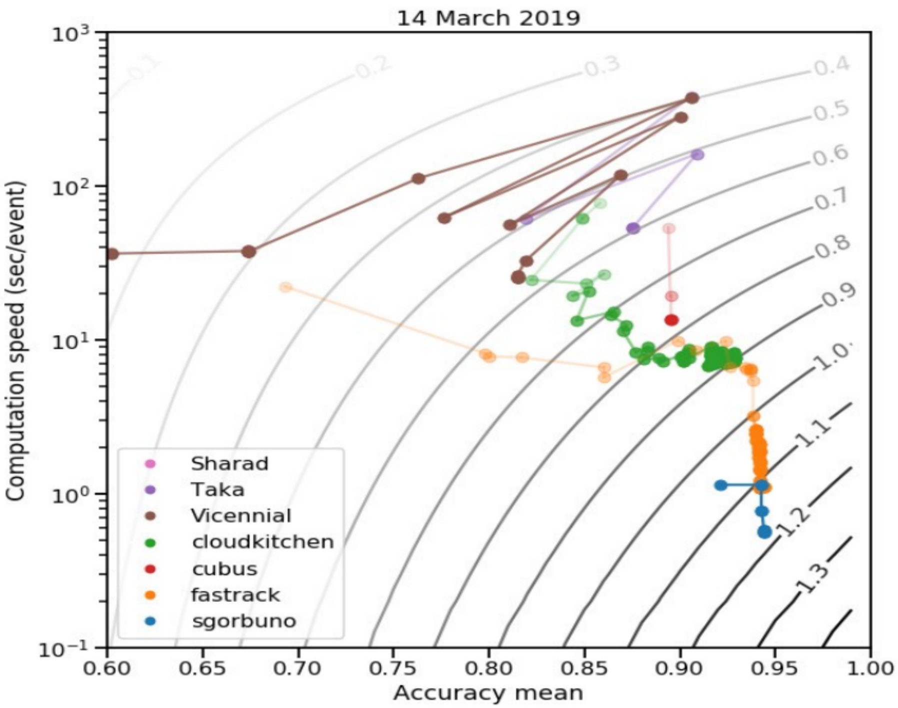
What worked well

Clear score progression

Accuracy

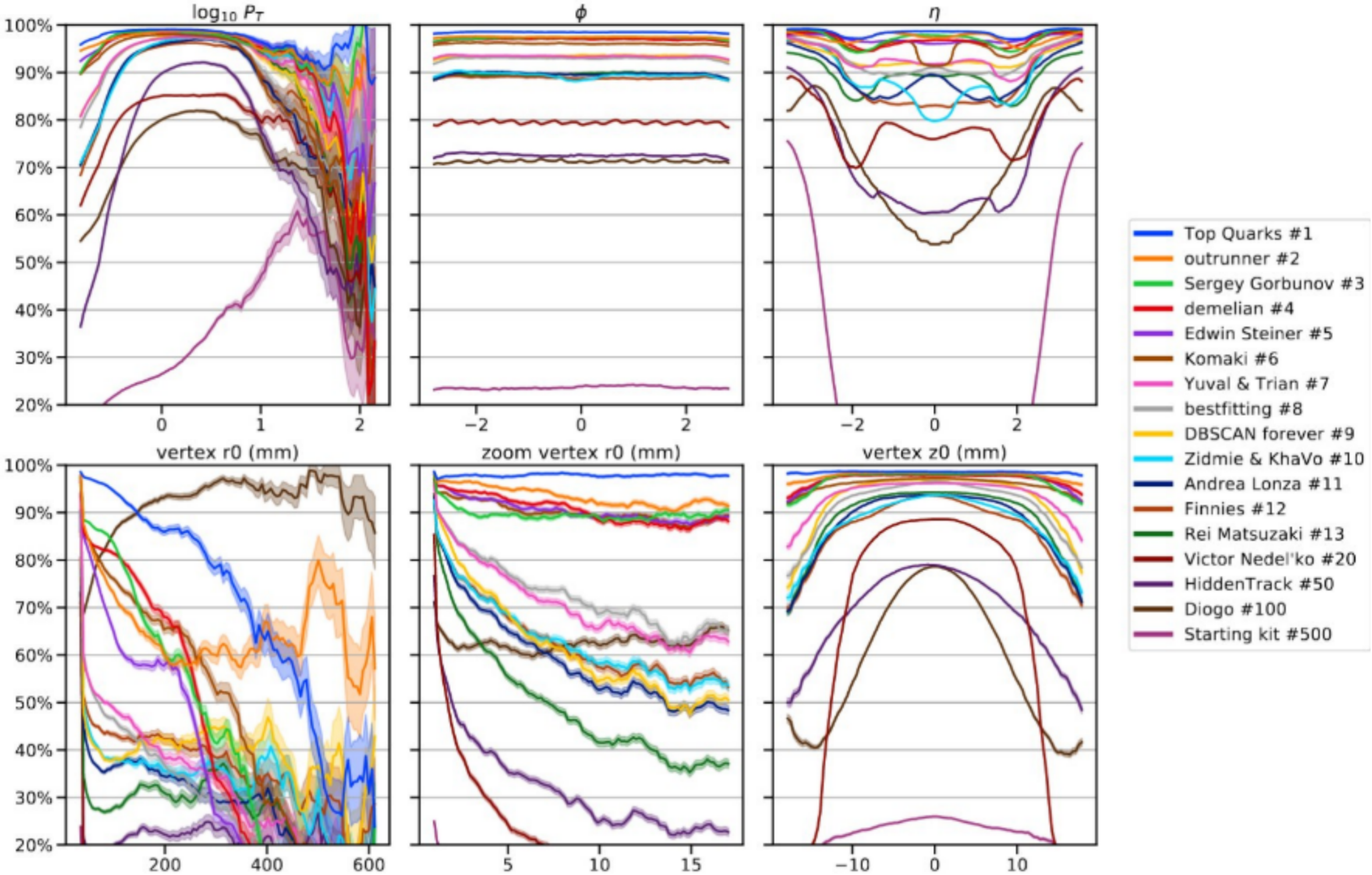


Throughput



Plots from Laurent Basara

Good score = good physics



In no particular order

Simple file format

Participants discussions

What did not work well or What we should have done

(but did not or could not)

Again, no particular order

Full simulation (Geant4?) vs. fast simulation

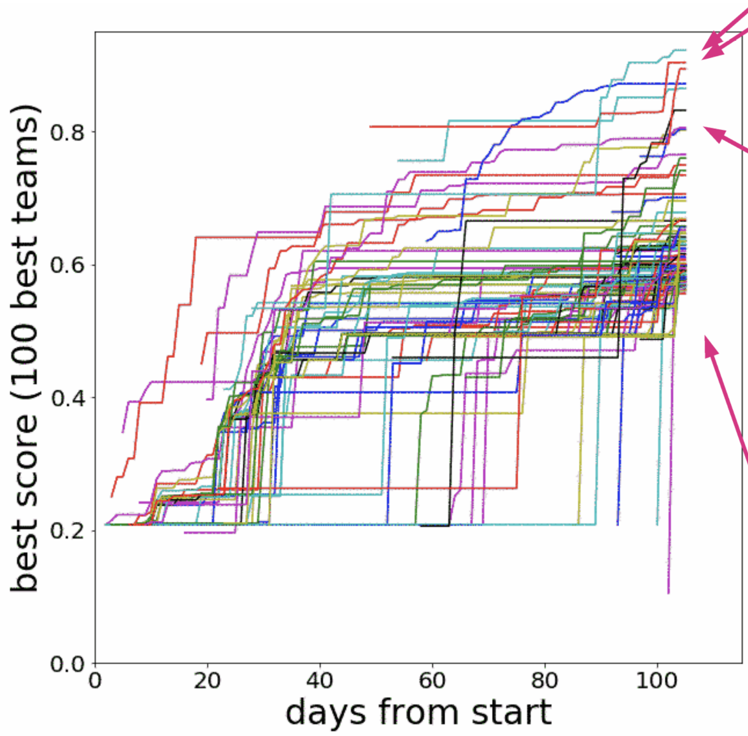
Signal type and detector maybe to optimistic

Less (true) ML solutions than expected

No classical solution as comparison

(Winning) Solutions

Accuracy phase



In the money   Gold   Silver   Bronze

#	△pub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	Top Quarks			0.92182	10	1y
2	—	outrunner			0.90302	9	10mo
3	—	Sergey Gorbunov	Dedicated talk		0.89353	6	10mo
4	—	demelian			0.87079	35	1y
5	—	Edwin Steiner			0.86395	5	10mo
6	—	Komaki			0.83127	22	10mo
7	—	Yuval & Trian			0.80414	56	10mo
8	—	bestfitting			0.80341	6	10mo
9	—	DBSCAN forever			0.80114	23	10mo
10	—	Zidmie & KhaVo			0.76320	26	1y
11	—	Andrea Lonza			0.75845	15	10mo
12	—	Finnies			0.74827	56	10mo
13	—	Rei Matsuzaki			0.74035	12	10mo
14	—	Mickey			0.73217	10	1y
15	—	Vicens Gaitan			0.70429	19	1y
16	—	Robert			0.69955	3	1y
17	—	Yuval-CPMP tribute band			0.69364	20	1y
18	—	N. Hi. Bouzu			0.67573	9	1y
19	—	Steins;Gate			0.66763	12	1y
20	▲1	Victor Nedel'ko			0.66723	4	1y
21	▼1	atom1231 & Kent AI Lab			0.66320	42	10mo
22	▲1	Nerdiholic			0.65420	12	1y
23	▼1	Sergey Zlobin			0.65352	23	1y

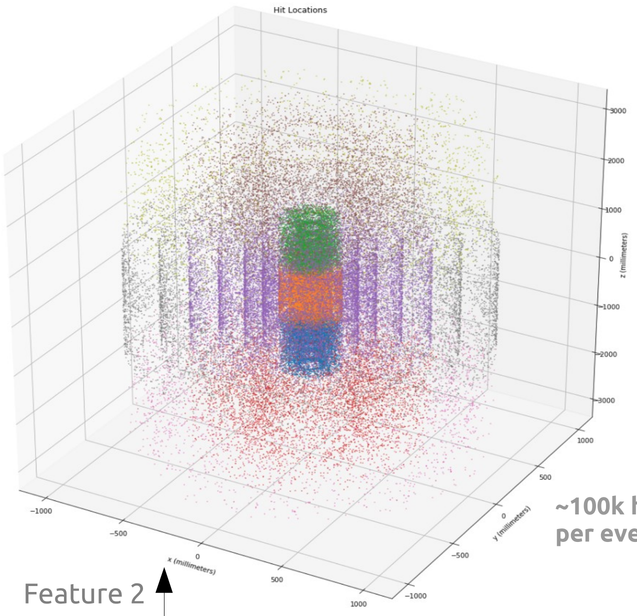
100

Accuracy #12: Finnies (Jury Deep Learning Prize)

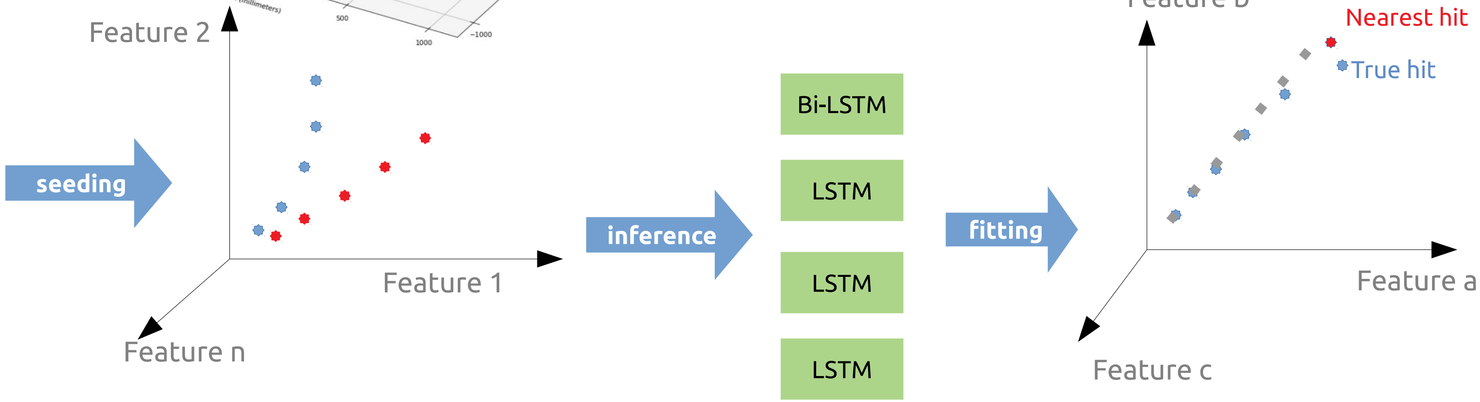
Liam Finnie & Nicole Finnie
IBM Germany R&D
Bosch Centre for AI

https://github.com/jliamfinnie/kaggle-trackml

Solution Pipeline



~100k hits (~10k tracks) per event

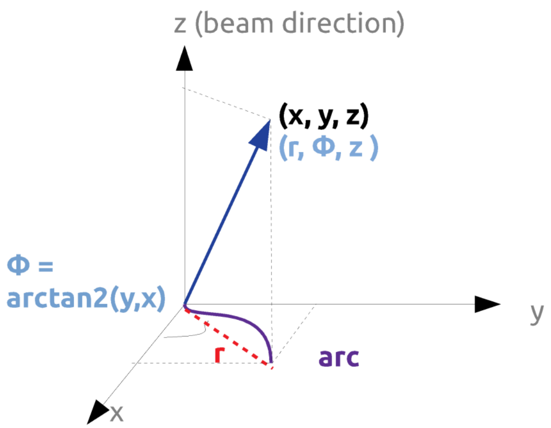


Track seeding
(clustering)

Inference & Ensembling

Track fitting
(k-nearest neighbour)

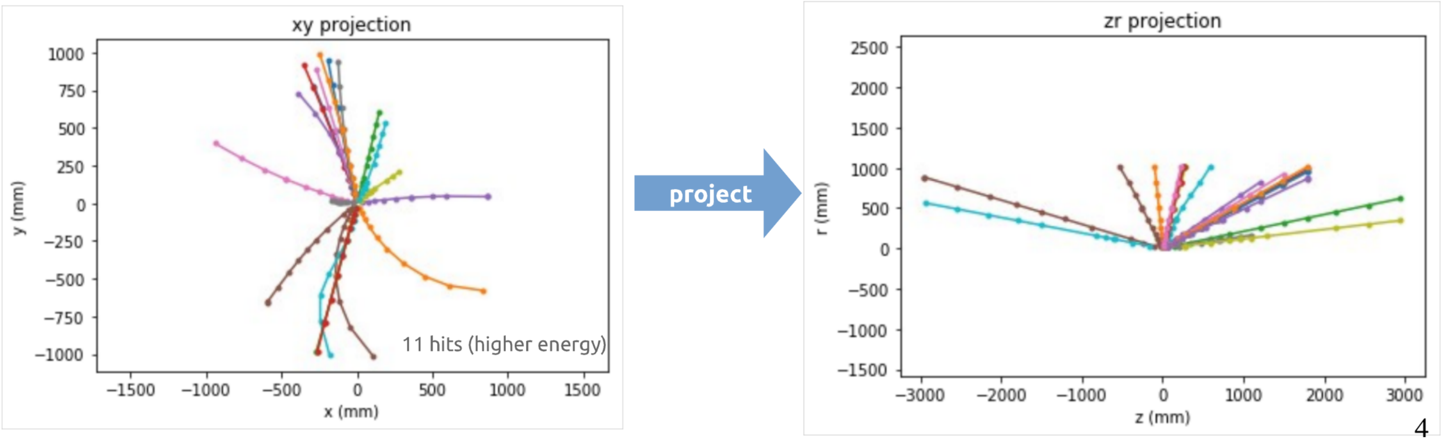
Feature Engineering... for people who don't know physics :D



Data we use: (x, y, z) coordinates of hits

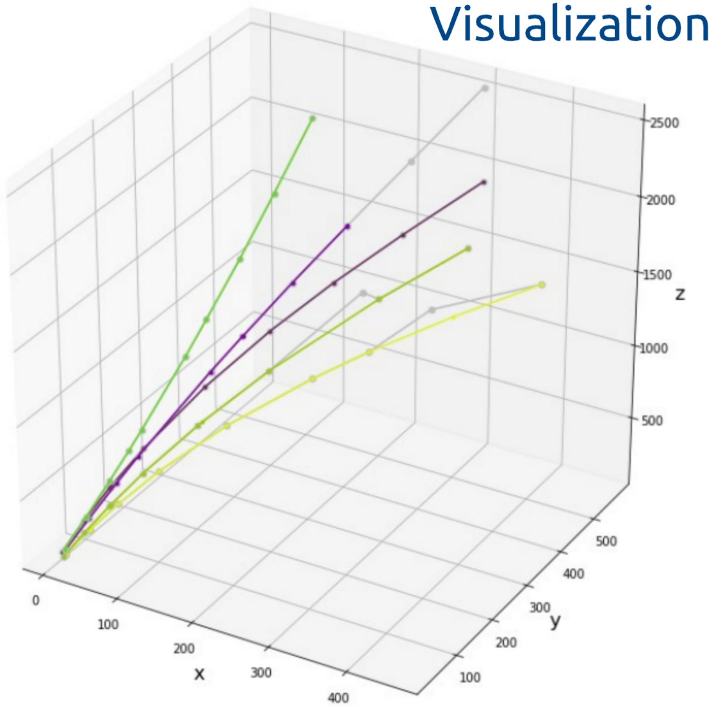
For clustering: sin(Φ), cos(Φ), z/arc
(new feature: generate possible arcs using train data)

For LSTM: Φ, r, z, z/r



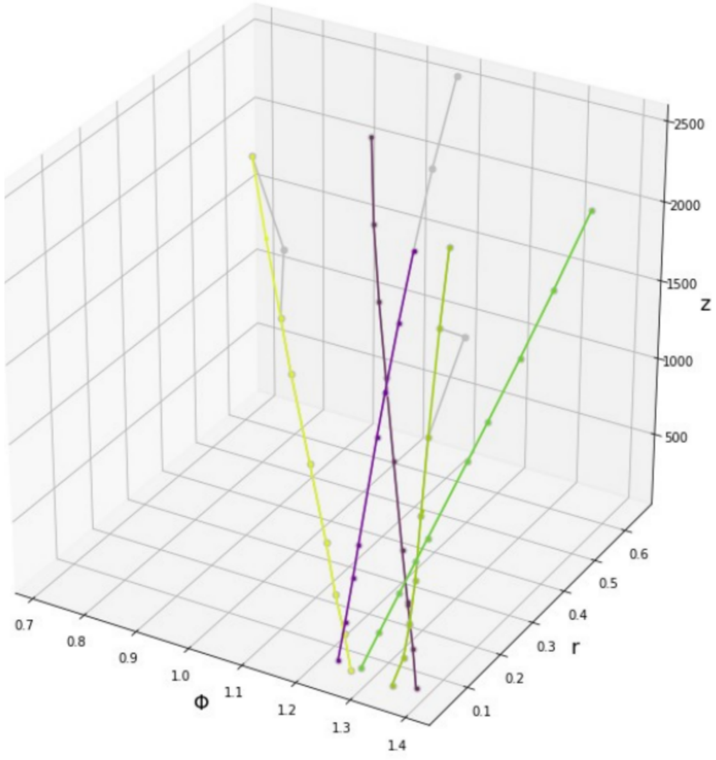
Cartesian -> Polar coordinates: **easier for LSTM to learn**

Visualization after fitting



Cartesian coordinates

--- predicted (grey)
--- ground truth (colour)



Polar coordinates

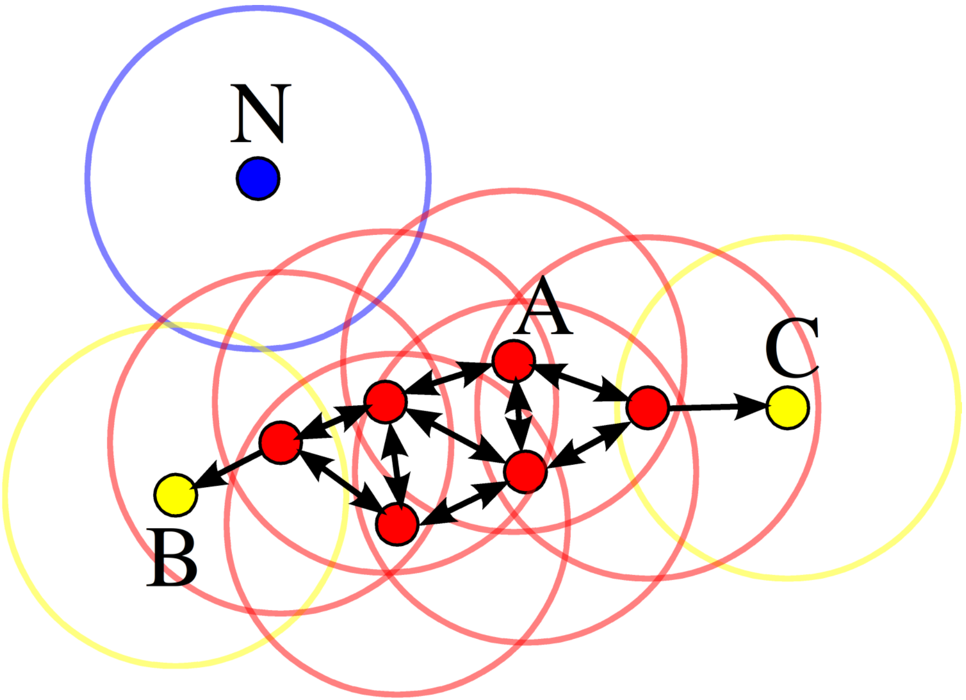
Accuracy #9: DBSCAN forever (Jury Clustering Prize)

Jean-Francois Puget “CPMP”
Software engineer at IBM in France
https://github.com/jfpuget/Kaggle_TrackML

DBSCAN?

Density-based clustering

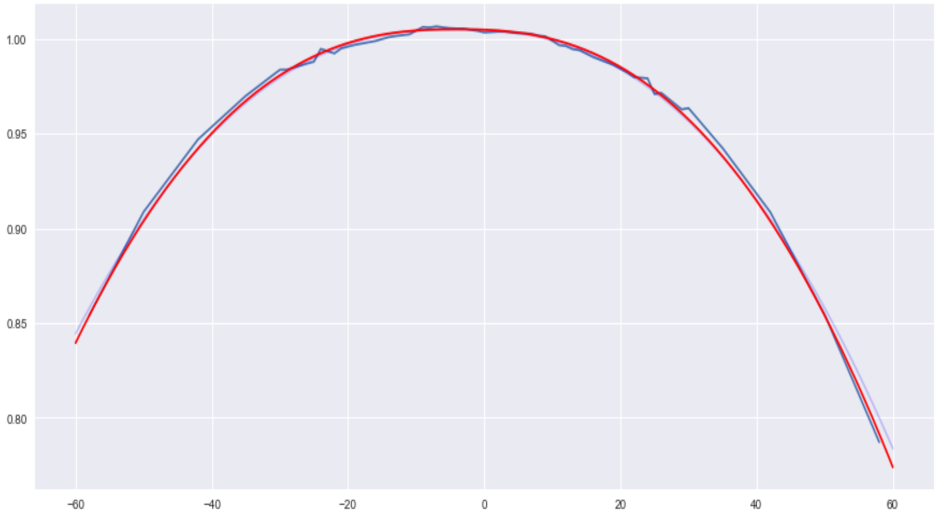
- Few parameters: distance, min #, (metric)
- Simple and available
- Used in starting kit score ≈ 0.2



wikipedia.org/wiki/DBSCAN

DBSCAN forever – Improvements

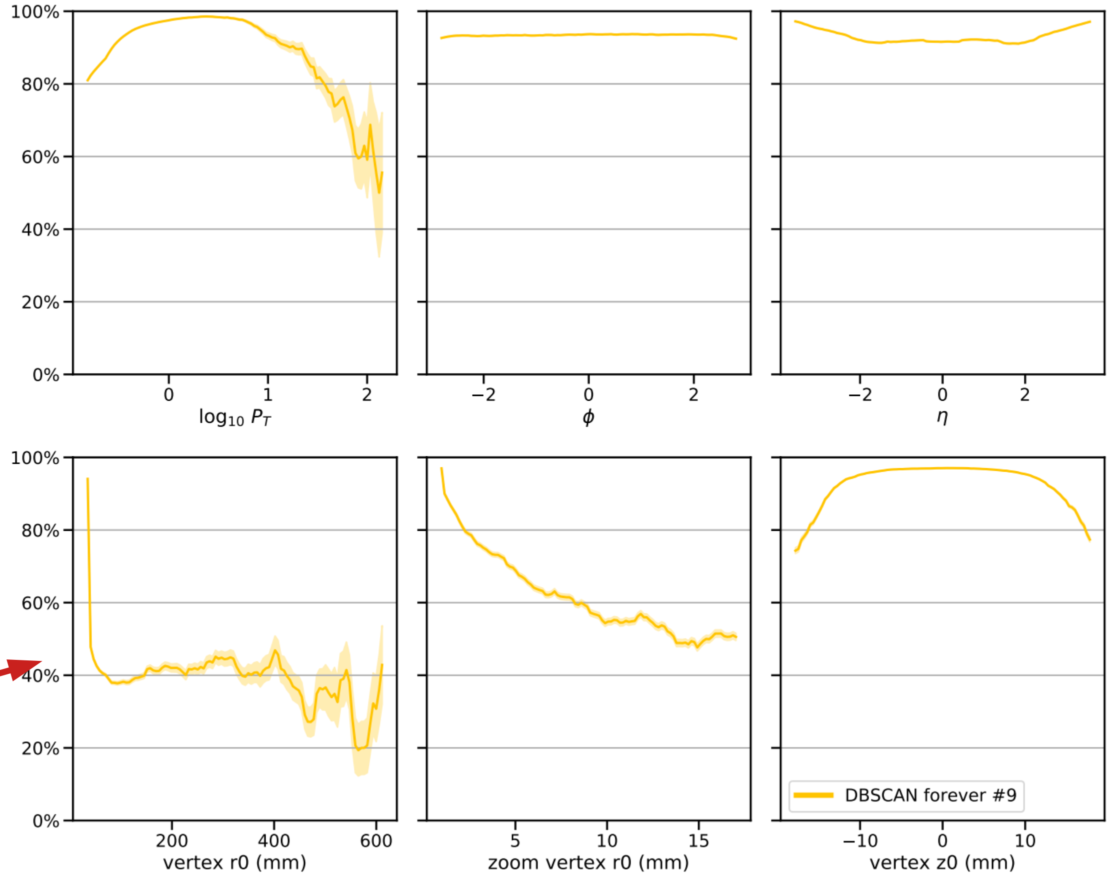
- Hough-transform-like unfolding for helix model
  - Pick a (r_0, z_0) pair
  - Compute ρ, ϕ, η-like for each hit
  - Assumes d0 = 0
- Run for many (r_0, z_0) pairs
- Different parameters for inner/outer detectors



Magnetic field extracted from data

From CPMP Kaggle post

DBSCAN forever – Efficiencies



Probably:

$d_0 = 0$ assumption in helix unrolling

DBSCAN forever – Take away

Manually tuned, classical algorithm with smart preprocessing

Implementation

- Pure python
- DBSCAN from scikit-learn

Runtime

- 3Gb per worker
- Timing unknown

Accuracy #4: demelian

Dmitry Emeliyanov

https://github.com/demelian/fastrack

FASTTrack: Graphs, CA, Kalman filter

From D. Emeliyanov

Accuracy #2: outrunner

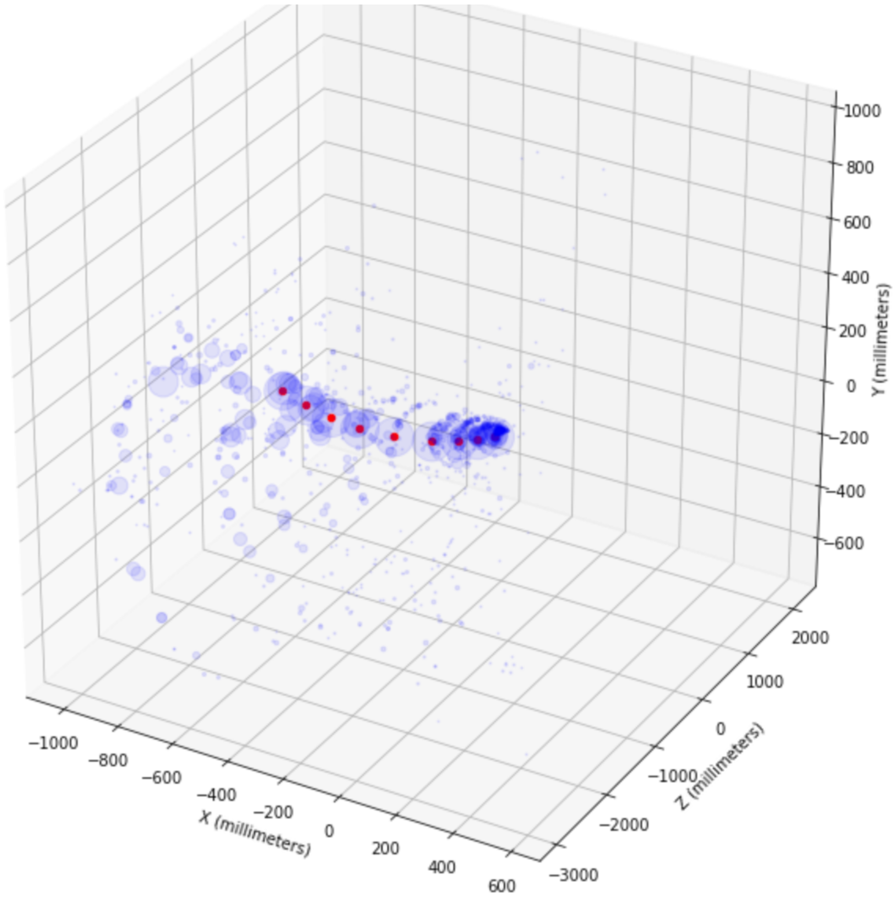
Pei-Lien Chou

Software engineer image-based deep learning in Taiwan.

Kaggle Notebook

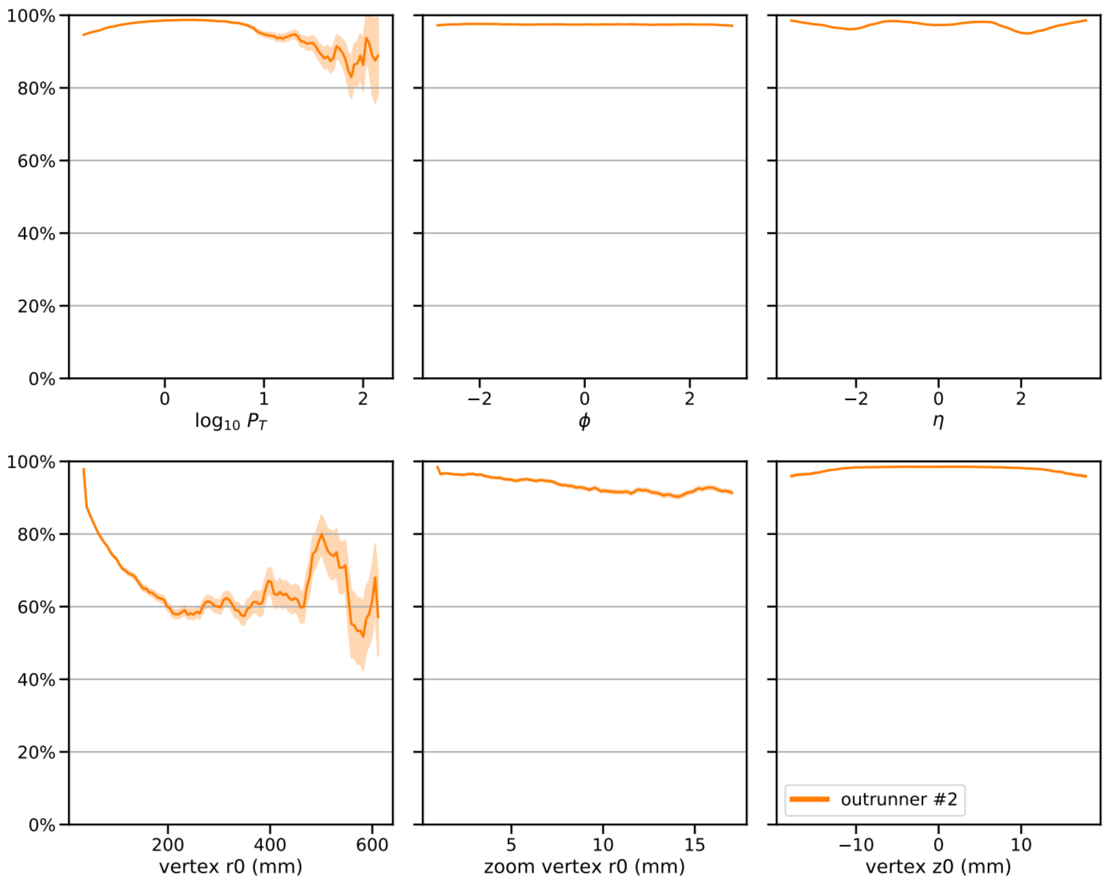
outrunner – Setup

- Train DNN on hit pairs
  - 27 inputs (x,y,z,cells,...)
  - 4k-2k-2k-2k-1k hidden layers
- Compute full hit adjacency matrix: probability P(i,j) that 2 hits match
- Pick high probability comb.
- Helix-like fit for cleaning



Graphics from outrunner

outruner – Efficiencies



outrunner – Take away

- True Deep Learning Solution
  - No track following
  - No geometric modelling
- But: slow execution

- Implementation
  - Pure python
  - Keras for ML
- Runtime
  - multiple hours / event

Accuracy #1: Top Quarks

Johan Sokrates Wind “icecuber”

Industrial Mathematics Master student in Norway (main contributor)

Erling Solberg “erlinsol”

https://github.com/top-quarks/top-quarks

Top Quarks – Overview

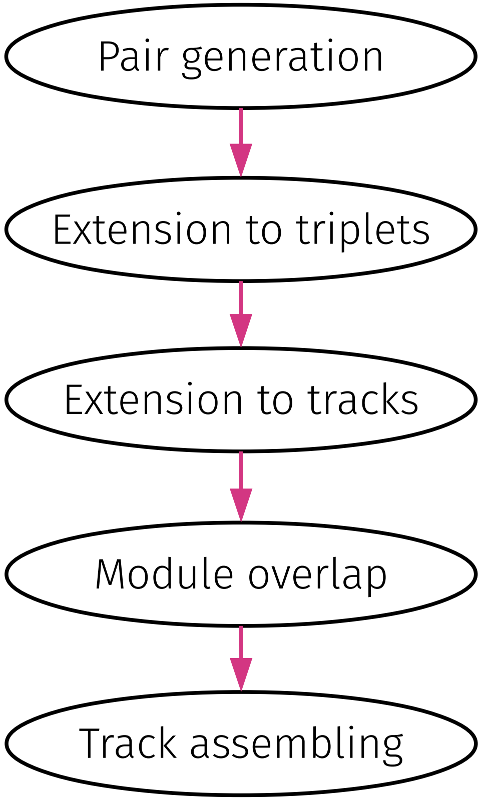


Illustration from J-R. Vlimant

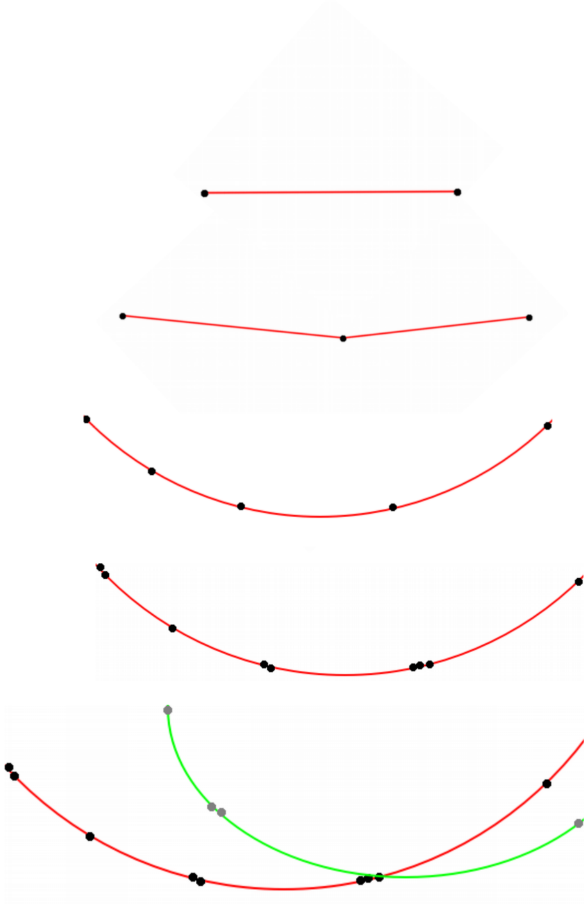
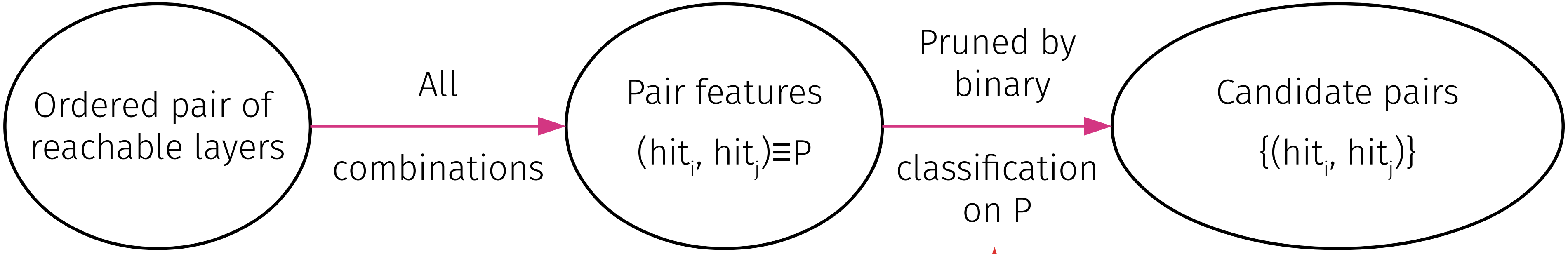


Illustration from J.S. Wind

Top Quarks – Pair generation



Logistic regression on test data

Illustration from J-R. Vlimant

Top Quarks – Extension to triplets

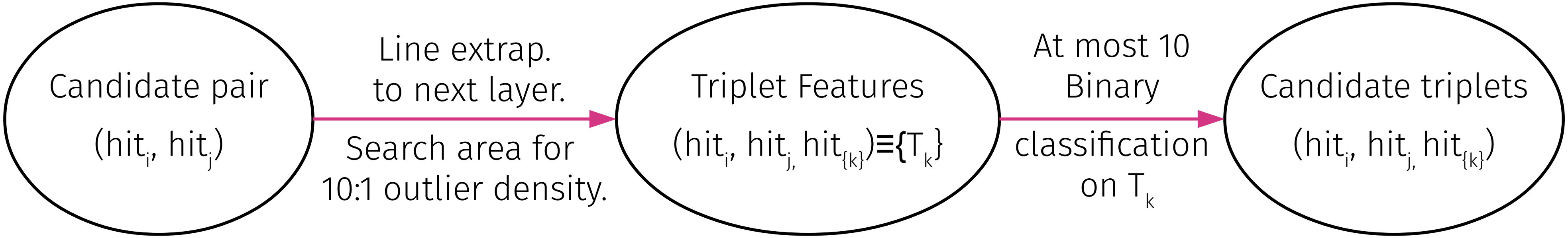
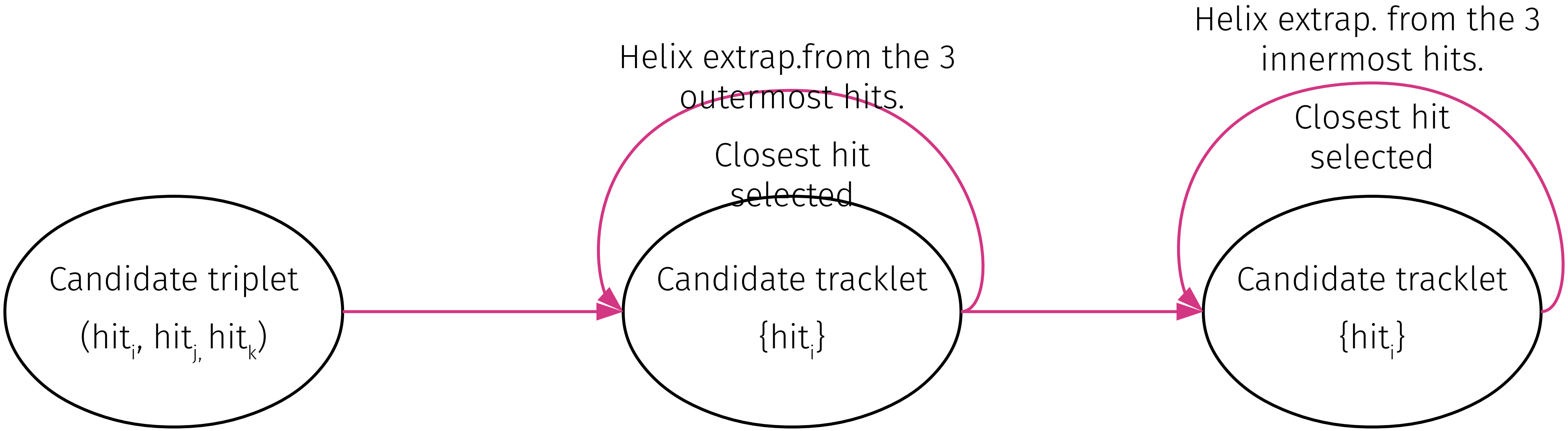


Illustration from J-R. Vlimant

Top Quarks – Extension to tracklets



Extrapolation w/ 2nd order circle approximation

Magnetic field from data

Illustration from J-R. Vlimant

Top Quarks – Module overlap

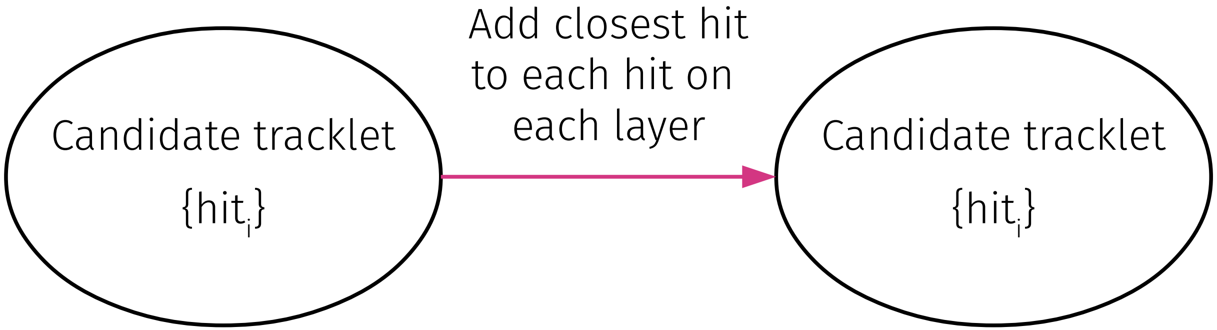
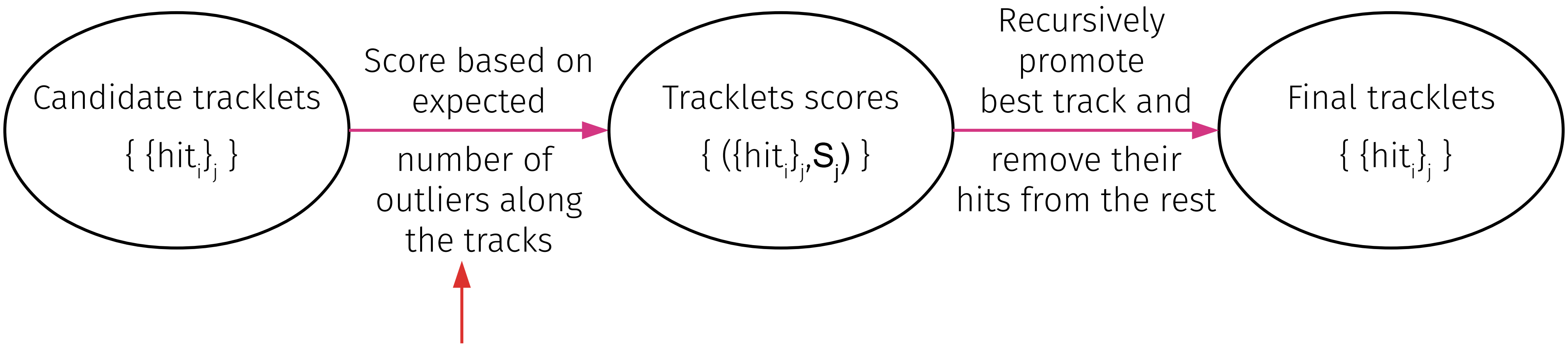


Illustration from J-R. Vlimant

Top Quarks – Track assembly



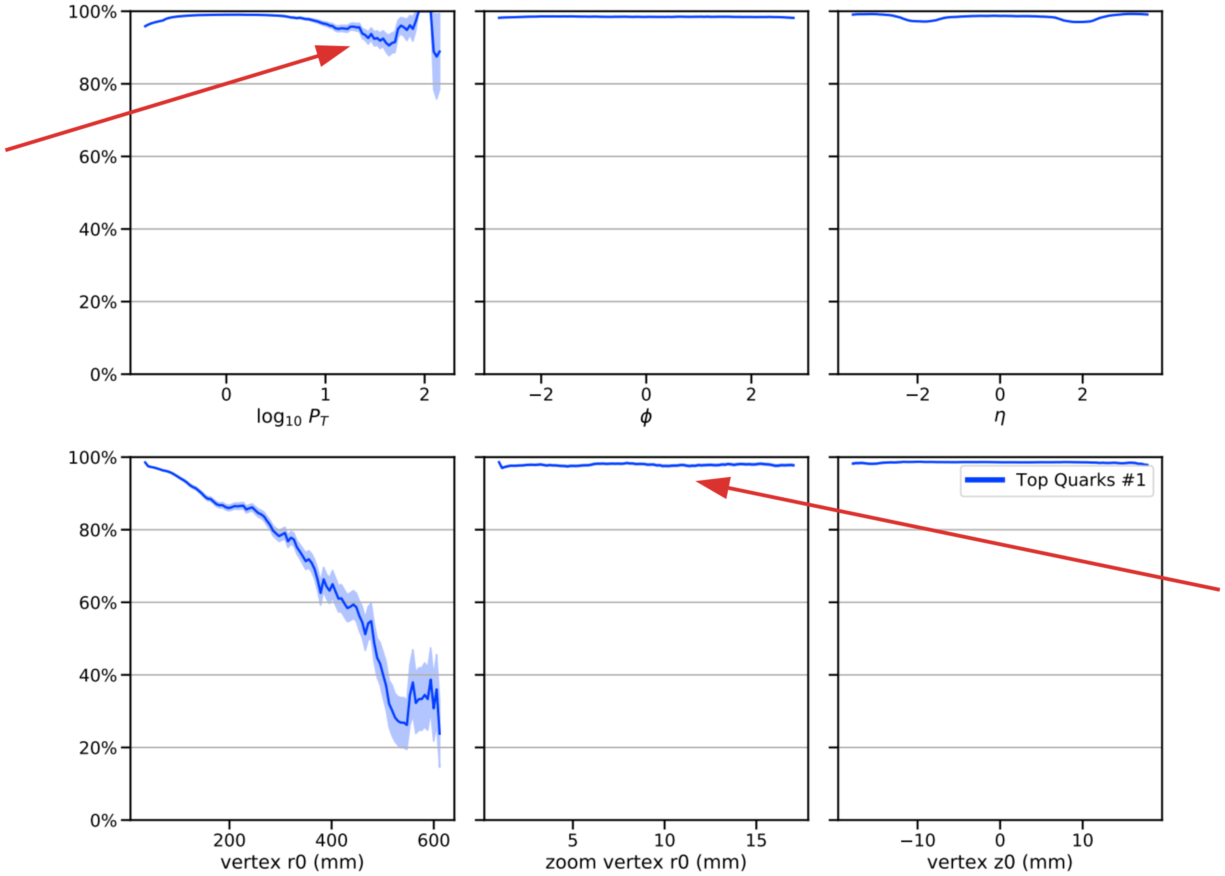
Interesting idea:

Model noise instead of signal

Illustration from J-R. Vlimant

Top Quarks – Efficiencies

A bit strange, but exists in almost every submission



Good

Top Quarks – Take away

- Custom algorithm:
  - Track following with ML sprinkles on top
- Custom implementation w/ fast runtime enables fast experimentation
- Served as inspiration for throughput phase, e.g. #3 Marcel Kunze

- Implementation
  - Custom C++ code
  - Custom quad-tree based hit lookup
  - Python/scikit-learn for training
- Runtime
  - 8min / event
  - Memory 2.8Gb avg, 4Gb max

Accuracy #100: diogo (Organizer’s pick)

Diogo R. Ferreira

Researcher at the University of Lisbon, focusing on data science and nuclear fusion

https://github.com/diogoff/trackml-100

diogo – Routes

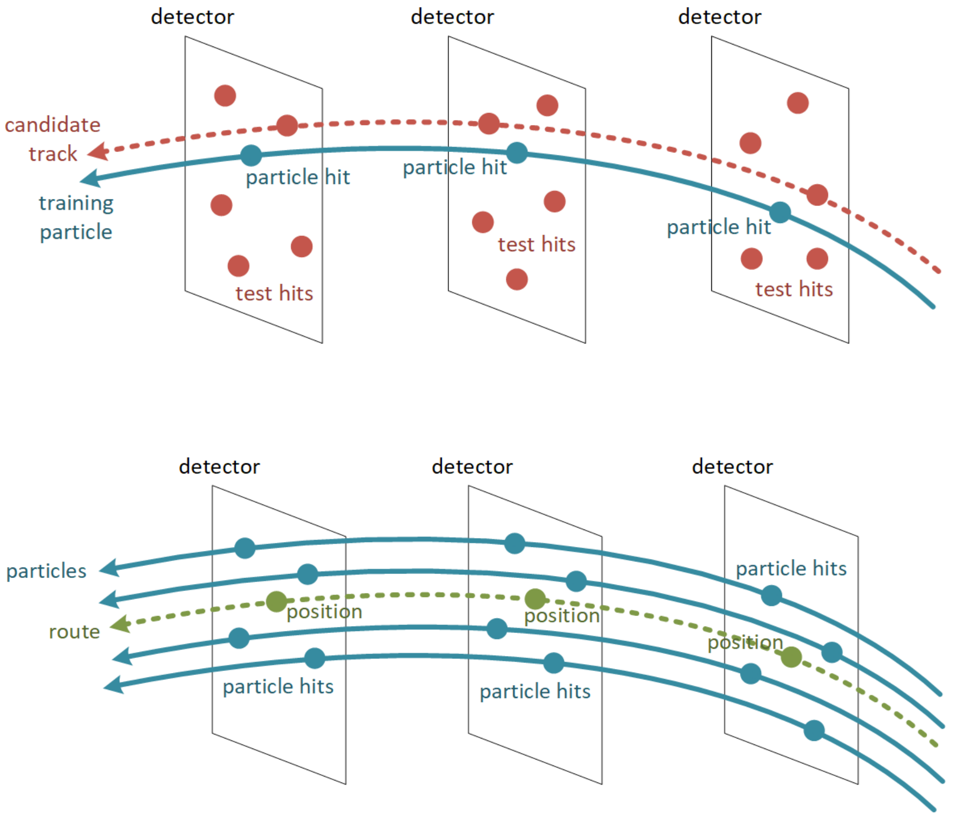
Build routes from truth

- All seen sequences of traversed modules
- Average estimates for shared sequences

On reconstruction

- Pick closest route(s) to hit
- Select route by distance

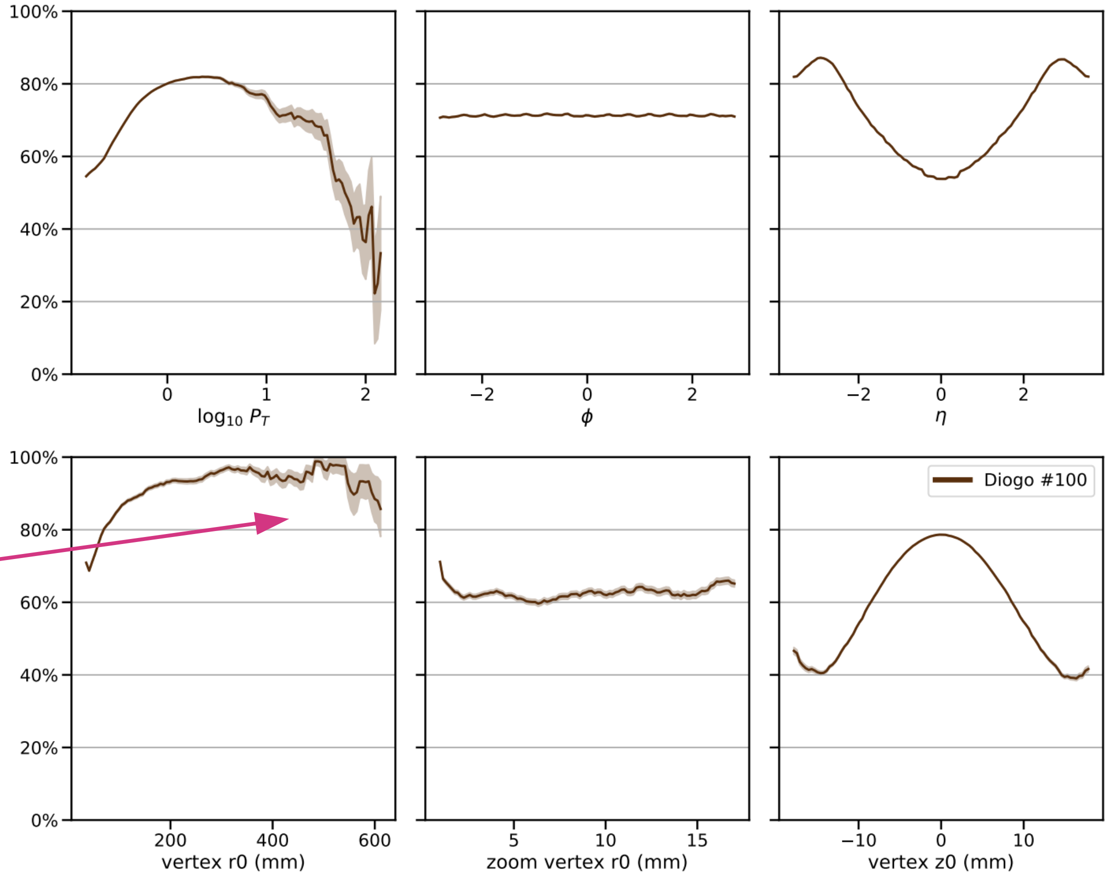
Similar to LHC triggers



Graphics from github.com/diogoff/trackml-100

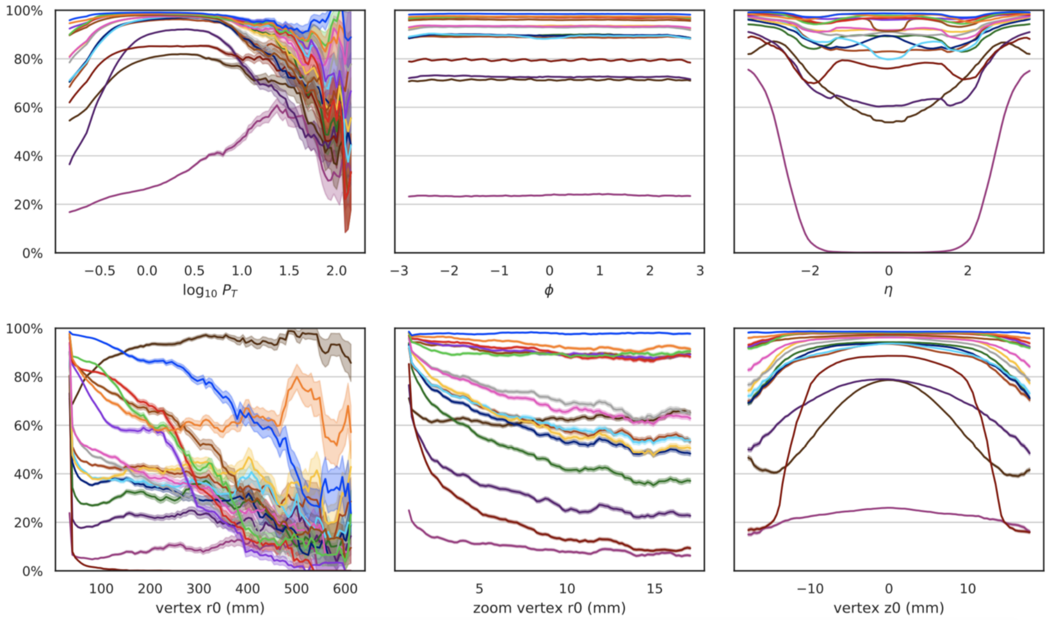
diogo – Efficiencies

High performance for displaced vertices



Summary

- Interesting solutions from non-domain experts
- Simple algorithms can be quite powerful
- But, this is a complex problem that sometimes requires complex solutions



Details e.g. in arXiv:1904.06778