Investigation of beam-dust interactions in the LHC using displaced bunches

Johannes Sadler

01/09/2023

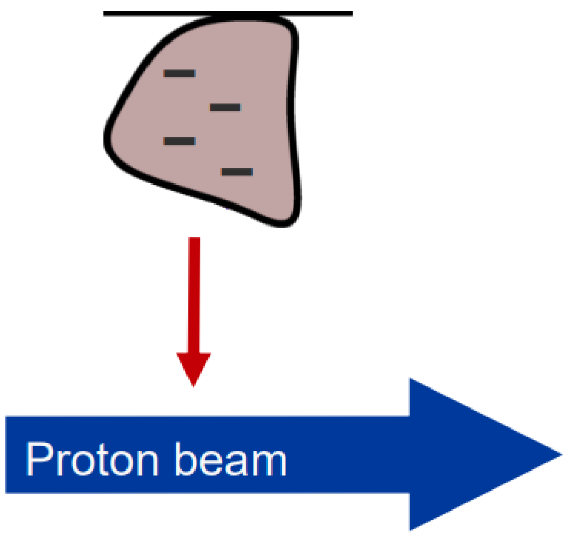
“UFOs” at the LHC

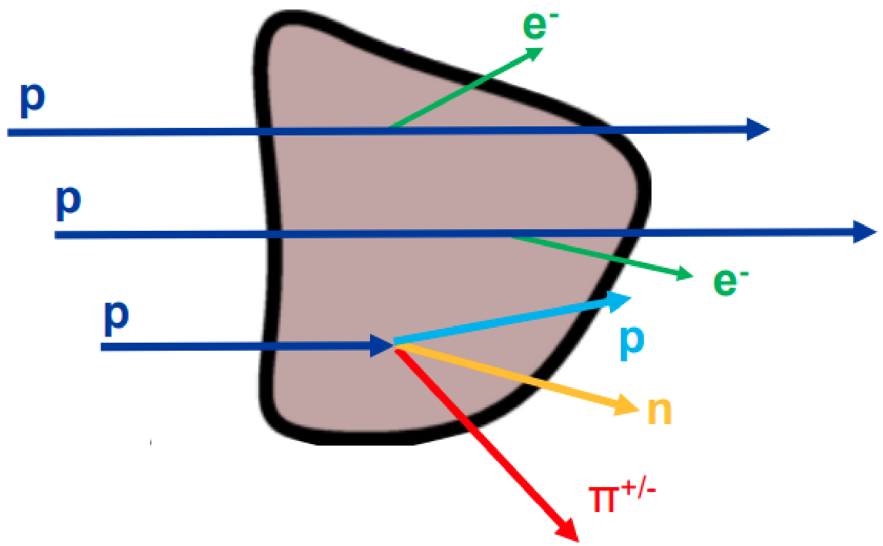
- **UFO – unidentified falling object**
- **Refers to a transient beam loss event, believed to be caused by dust particles tens of micrometers in size**
- **Standard UFOs have maximal duration of 1-2ms**
- **Present since the start of the LHC**
- **Detrimental to machine availability:**
  - During Run II (2015-2018):
    - 139 beam dumps – lose a few hours
    - 12 magnet quenches – lose half a day

The UFO hypothesis

1 Attraction by the beam:

Dust grains can acquire a *negative* charge and get attracted by the field of the circulating proton beam



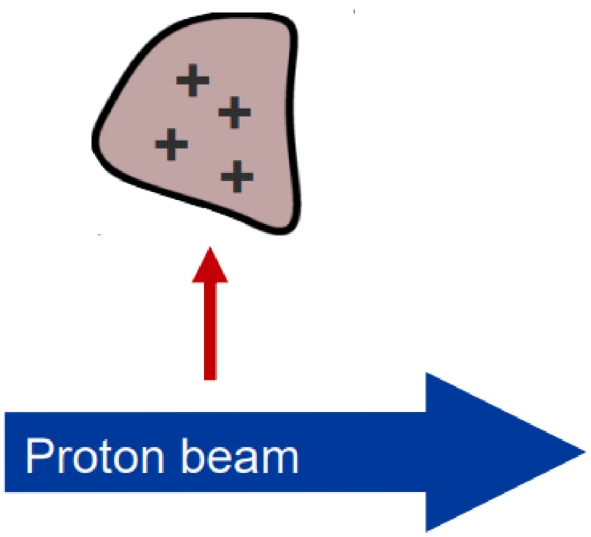


2 Proton-matter interactions:

- **When the dust grain enters the beam, it gets ionized by the traversing protons**
- **A small fraction of the protons has a nuclear collision → *beam losses***

3 Repulsion from the beam:

The now *positively* charged dust grain gets repelled from the circulating beam



Slide from A. Lechner, Dust workshop

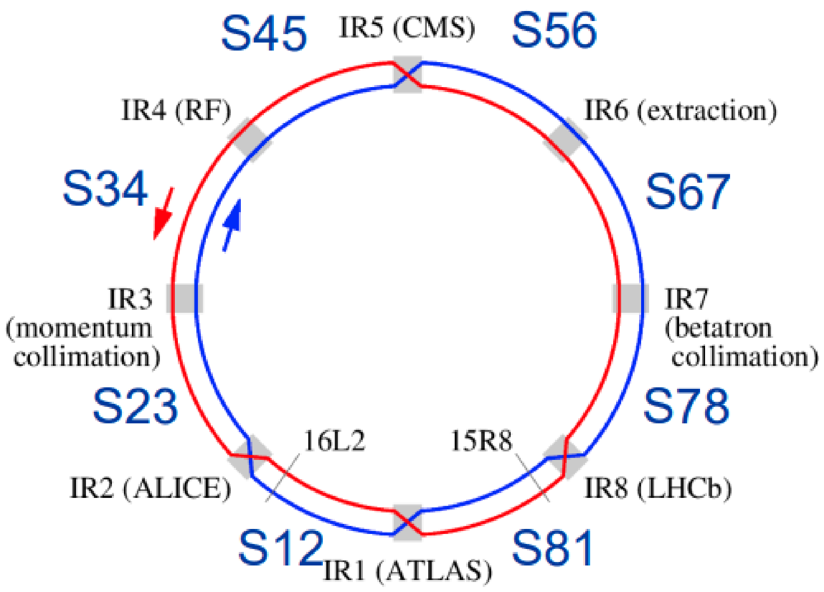
Detecting UFOs

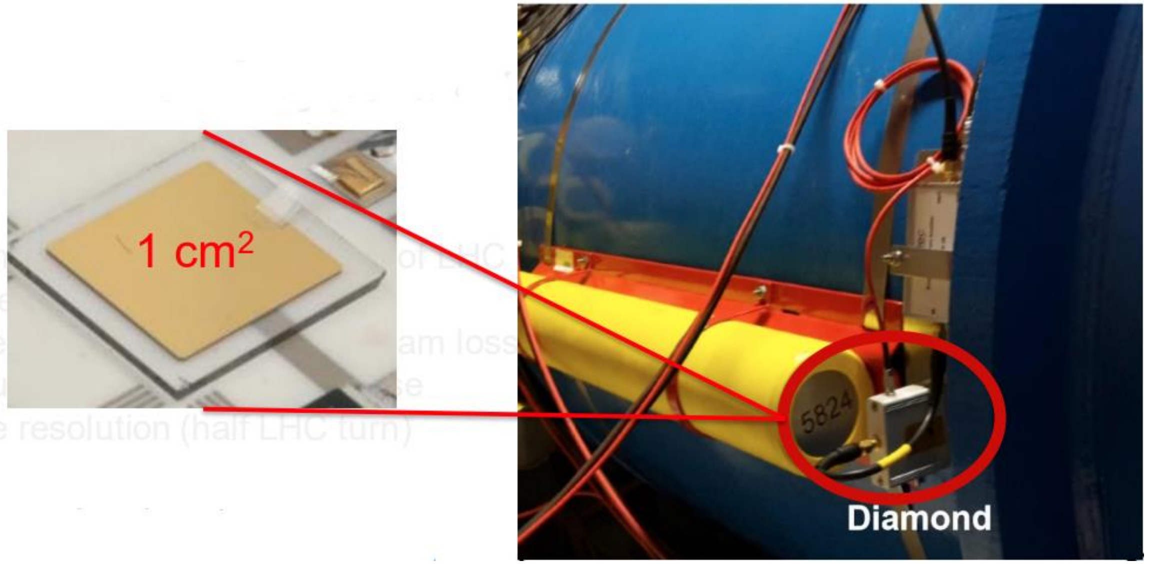
Ionization chamber BLMs

- **Main beam loss monitoring system of the LHC, 40µs time resolution**
- **Not considered further in this project**

Diamond BLMs

- **6 devices, 3 per beam, installed downstream of the primary collimators in the betatron collimation region.**
- **Detect losses from protons that are scattered by small angles and hit the collimator.**
- **Sample rate 650MHz, so 1.54ns time resolution.**
- **Can distinguish bunch-by-bunch losses.**





Displaced bunches

Aim: map the trajectory of the dust particle to understand more about the release mechanism

- Building on experiments with varying transverse bunch sizes
- Suggested in the PhD thesis of Bjorn Lindstrom
- Assumptions:
  - UFO treated as a point particle
  - UFO treated as stationary when bunches pass
  - Bunches have Gaussian proton density profile
  - Losses are proportional to the proton flux through the dust particle

$$\text{signal} \propto \frac{N_i}{\sigma_{xi}\sigma_{yi}\sqrt{2\pi}} \exp -\frac{1}{2}\left(\frac{x_i^2}{\sigma_{xi}^2} + \frac{y_i^2}{\sigma_{yi}^2}\right)$$

Define m_i to be the measured losses normalised by the bunch intensity.

$$\frac{m_i}{m_j} = \exp\left(\frac{x_j \Delta x}{\sigma_x^2} + \frac{y_j \Delta y}{\sigma_y^2} - \frac{\Delta x^2}{2\sigma_x^2} - \frac{\Delta y^2}{2\sigma_y^2}\right)$$

$$\Delta x = x_j - x_i \text{ and } \Delta y = y_j - y_i.$$

Scaling:

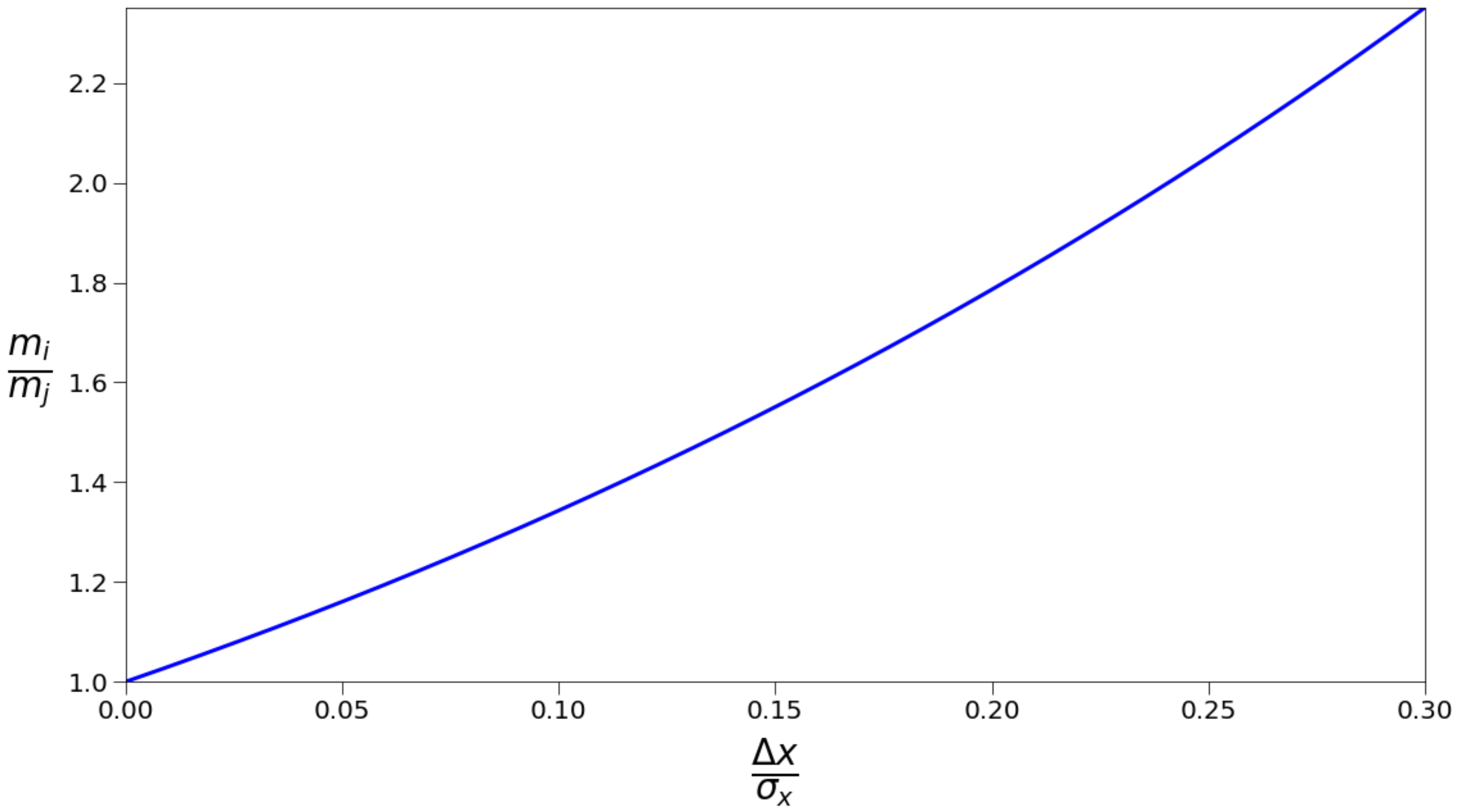
$$\frac{m_i}{m_j} = \exp\left(\tilde{x}_j \Delta\tilde{x} + \tilde{y}_j \Delta\tilde{y} - \frac{\Delta\tilde{x}^2}{2} - \frac{\Delta\tilde{y}^2}{2}\right)$$

Full derivation available in my project report.

Which ratios do we get for realistic parameters?

Ratio of losses

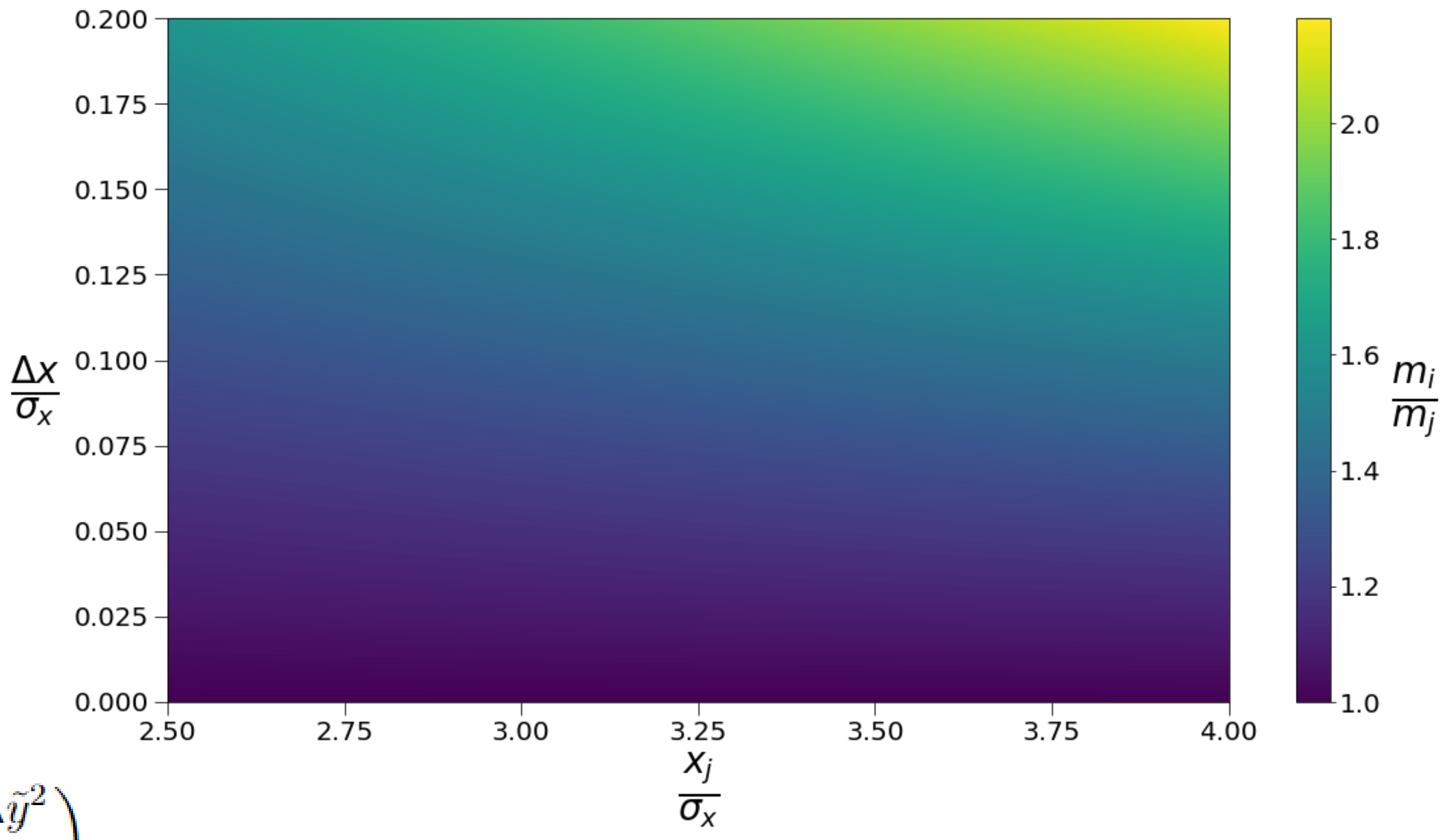
- Consider bunch displaced in x only
- Fix UFO position
- Plot ratio of normalized losses against bunch displacement
- Assume UFO at $x_j = 3\sigma_x$
- For a displacement Δx = 0.1σ, we expect a ratio of losses 1.34



$$\frac{m_i}{m_j} = \exp\left(\tilde{x}_j \Delta\tilde{x} + \tilde{y}_j \Delta\tilde{y} - \frac{\Delta\tilde{x}^2}{2} - \frac{\Delta\tilde{y}^2}{2}\right)$$

Ratio of losses

- Now vary also the UFO position
- Plot ratio of normalised losses against UFO position and bunch displacement
- Higher ratio expected for UFOs further from the beam center
- Caution – lower absolute losses



$$\frac{m_i}{m_j} = \exp\left(\tilde{x}_j \Delta\tilde{x} + \tilde{y}_j \Delta\tilde{y} - \frac{\Delta\tilde{x}^2}{2} - \frac{\Delta\tilde{y}^2}{2}\right)$$

Calculation of UFO position

- **Taking the natural logarithm:** $\log\left(\frac{m_i}{m_j}\right) = x_j \Delta x + y_j \Delta y - \frac{\Delta x^2}{2} - \frac{\Delta y^2}{2}$
- **Fix origin, let (X_j, Y_j) be the position coordinates of bunch j, let (χ, ψ) be the position coordinates of the UFO**

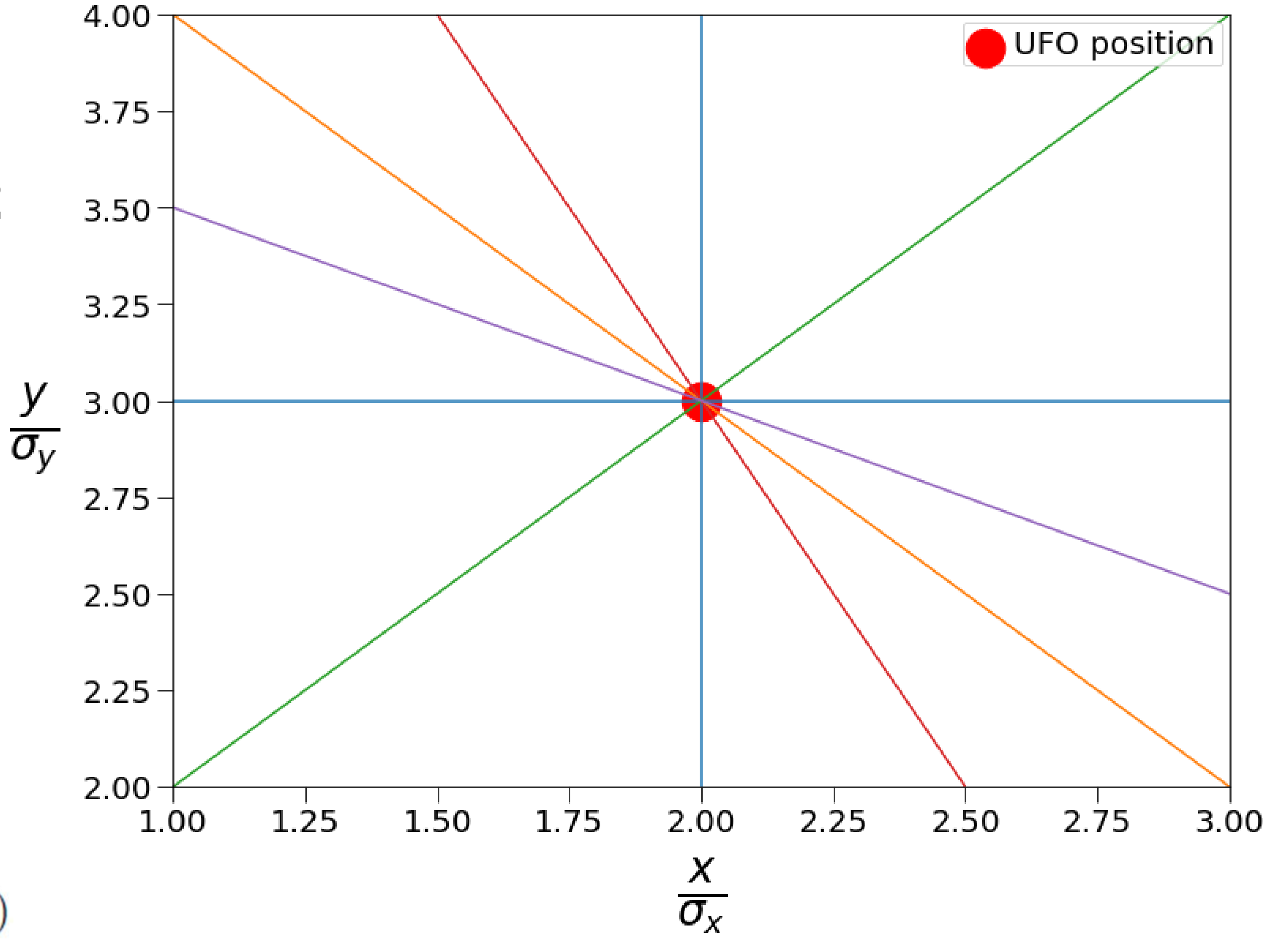
$$\chi(X_i - X_j) + \psi(Y_i - Y_j) = \log\left(\frac{m_i}{m_j}\right) + \frac{1}{2}(X_i^2 + Y_i^2 - X_j^2 - Y_j^2)$$

- **Only unknowns are (χ, ψ)**
- **For every 2 bunches, you have a line of possible UFO positions**
- **N bunches → N choose 2 lines**

$$_nC_r = \binom{n}{r} = \frac{n!}{r!(n-r)!}$$

Visualisation of UFO position information

- **In the ideal case, all lines will intersect exactly at the UFO position**
- **Plot the case of 4 bunches → 4 choose 2 = 6 lines**
- **Bunch positions chosen: (0,0), (0.1,0), (0,0.1), (0.2,0.2)**
- **UFO position chosen: (2,3)**
- **Loss signals calculated from the model**

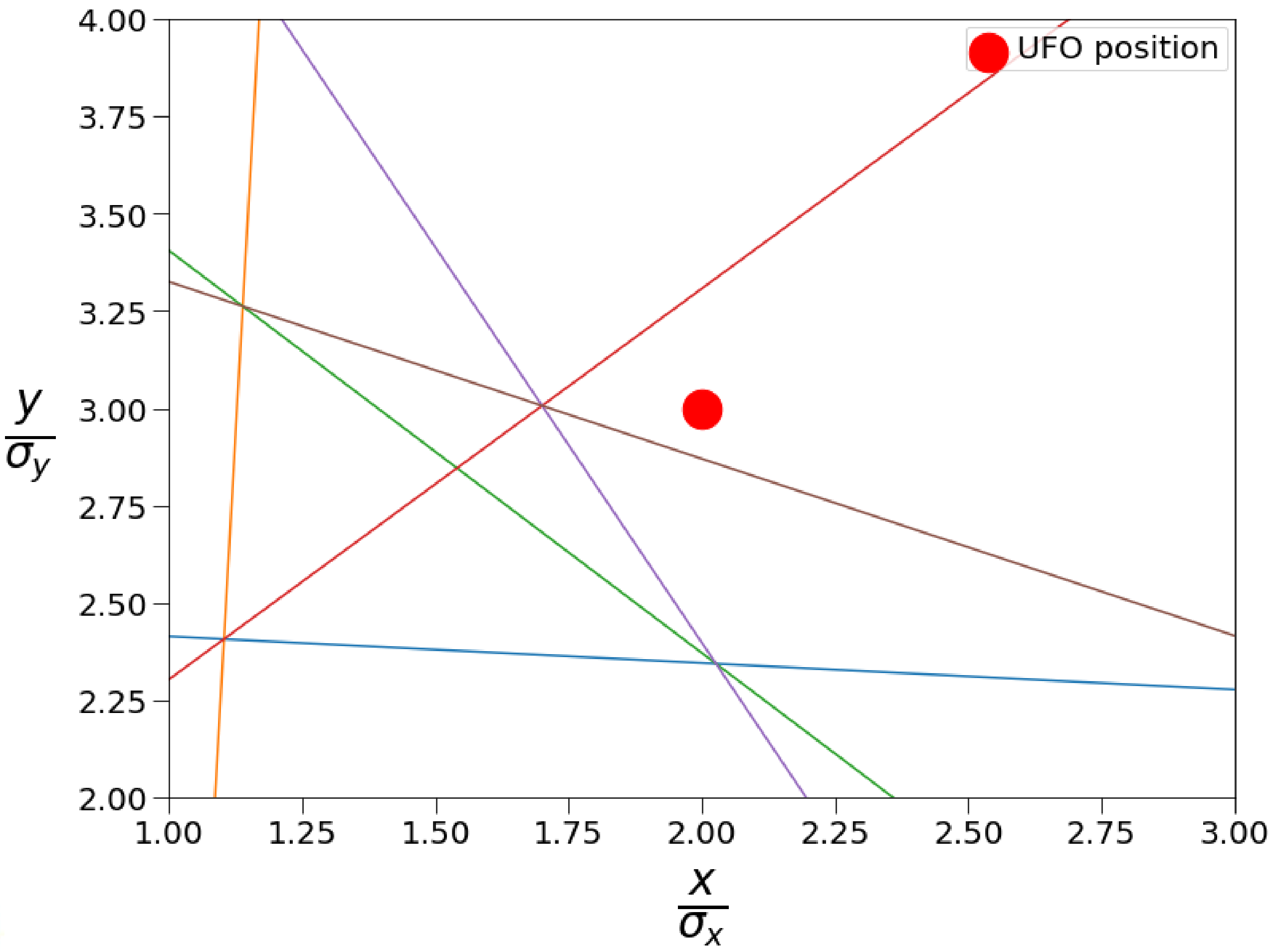


$$\chi(X_i - X_j) + \psi(Y_i - Y_j) = \log\left(\frac{m_i}{m_j}\right) + \frac{1}{2}(X_i^2 + Y_i^2 - X_j^2 - Y_j^2)$$

Visualisation of UFO position information

- **In the realistic case, lines will be skewed by noise in the position and loss signal measurements.**
- **Plot the case of 4 bunches → 4 choose 2 = 6 lines**
- **Bunch positions chosen: (0,0), (0.1,0), (0,0.1), (0.2,0.2)**
- **UFO position chosen: (2,3)**
- **Loss signals calculated from the model**
- **Random Gaussian noise added to each position and loss signal**

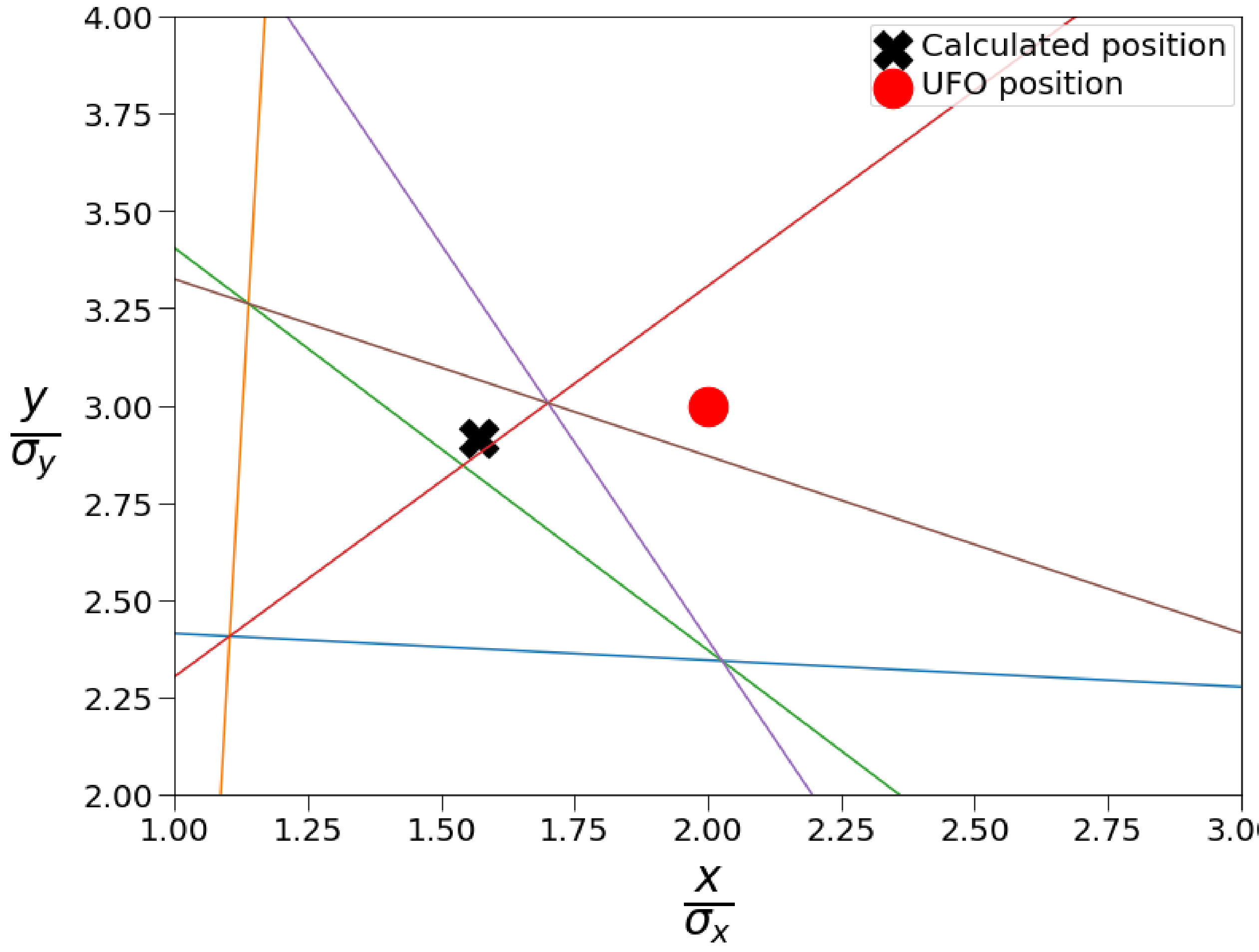
$$\chi(X_i - X_j) + \psi(Y_i - Y_j) = \log\left(\frac{m_i}{m_j}\right) + \frac{1}{2}(X_i^2 + Y_i^2 - X_j^2 - Y_j^2)$$



Calculation of UFO position from N bunches

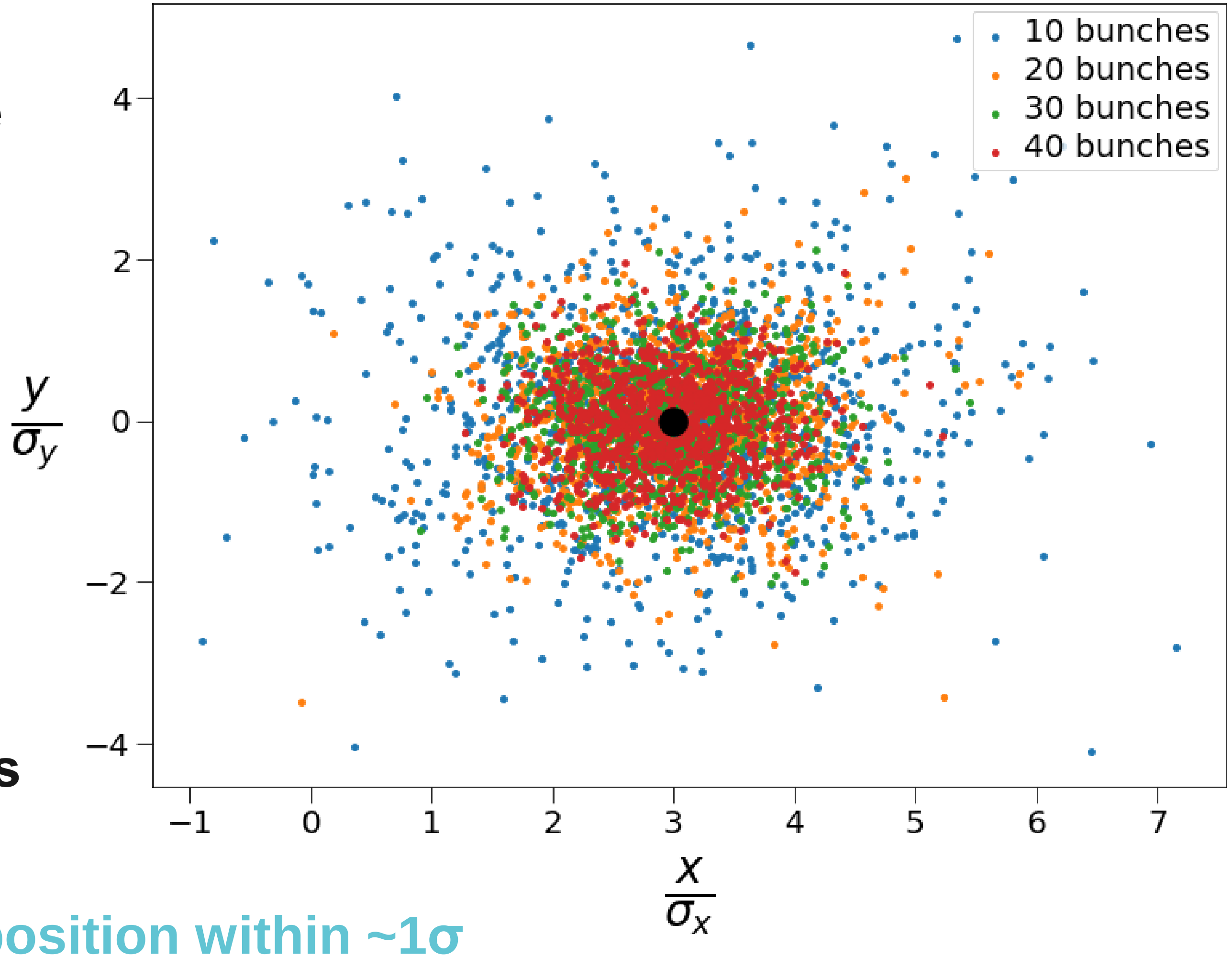
Python code written to implement the following algorithm:

1. **Find the N choose 2 pairs of bunches**
2. **Set up the linear equation for each pair**
3. **Least squares fit to the N choose 2 lines, weighted by the squared distance between the bunches**



Testing

- **Generate N bunch positions with a random uniform distribution in the range [-0.05,0.05]x[-0.05,0.05]**
- **Choose UFO position: (3,0)**
- **Calculate loss ratios from model**
- **Add random Gaussian noise to position and loss signals**
  - Noise of standard deviation 0.005 for position coordinates
  - Noise standard deviation 10% for loss signals
- **Do this for 3≤N≤48, and repeat 1000 times for each N**



For 40 bunches, we measure the true UFO position within ~1σ

Analysis of Diamond BLM data

Searching for UFOs

Context

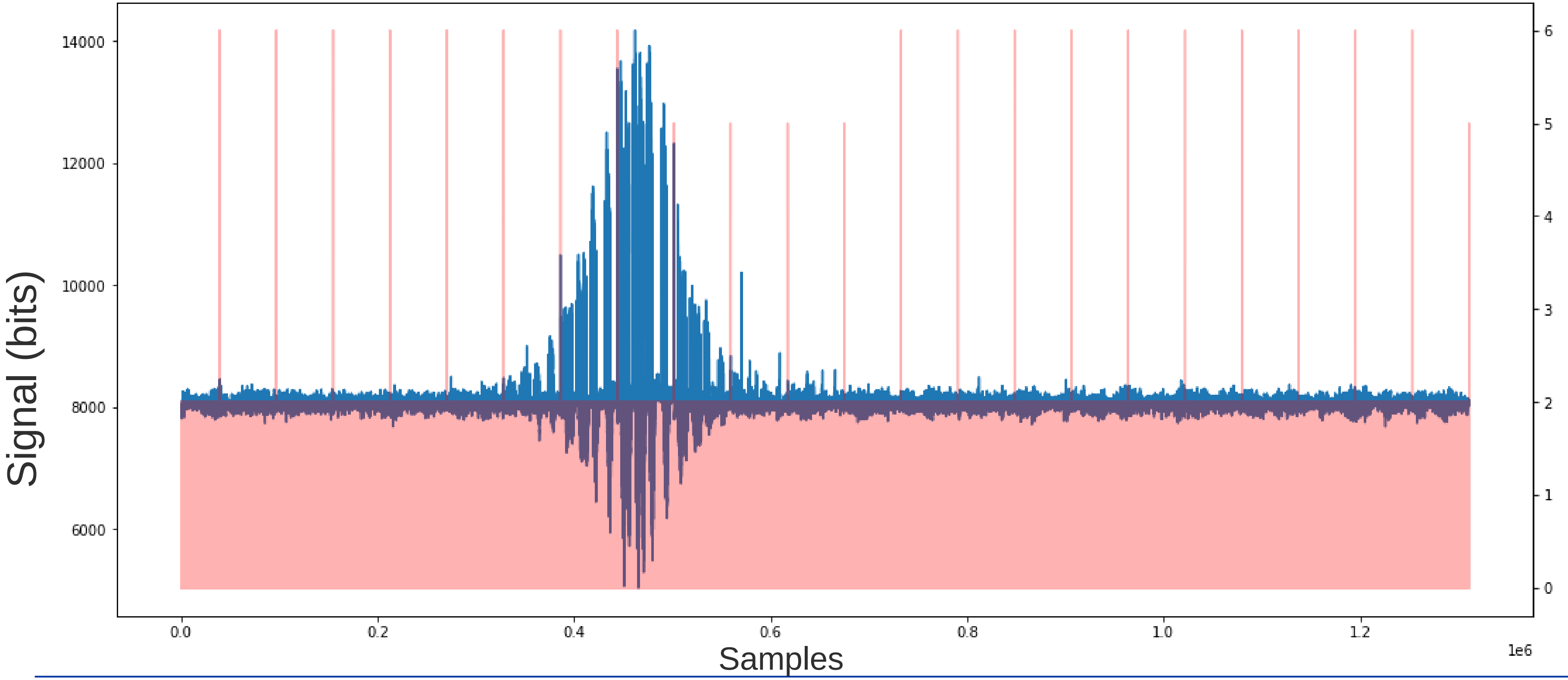
- We need bunch-by-bunch loss data to use the displaced bunches idea
- Diamond BLMs provide the bunch-by-bunch loss data
- Real-time UFO detection method established in LHC Run II to trigger read-out during beam operation (UFO auto-trigger)
- Hardware change during LS2
- Auto-trigger reactivated this year to continue with UFO studies
- Diamond BLM readout also triggered at every injection and every beam dump (timing event)

First task: look at the data

Searching the data manually… find some events

- **Query NXCALS using Timber for time windows when we had high intensity in the machine (i.e. not during injection)**
- **5 events found**

Example of typical UFO signal:



Blue – loss signal
Red – flags indicating bunches and turns

We can transform these into a 2d matrix, where we group the signal in turns and bunches:

For the losses on the 3rd turn, 2000th bunch:

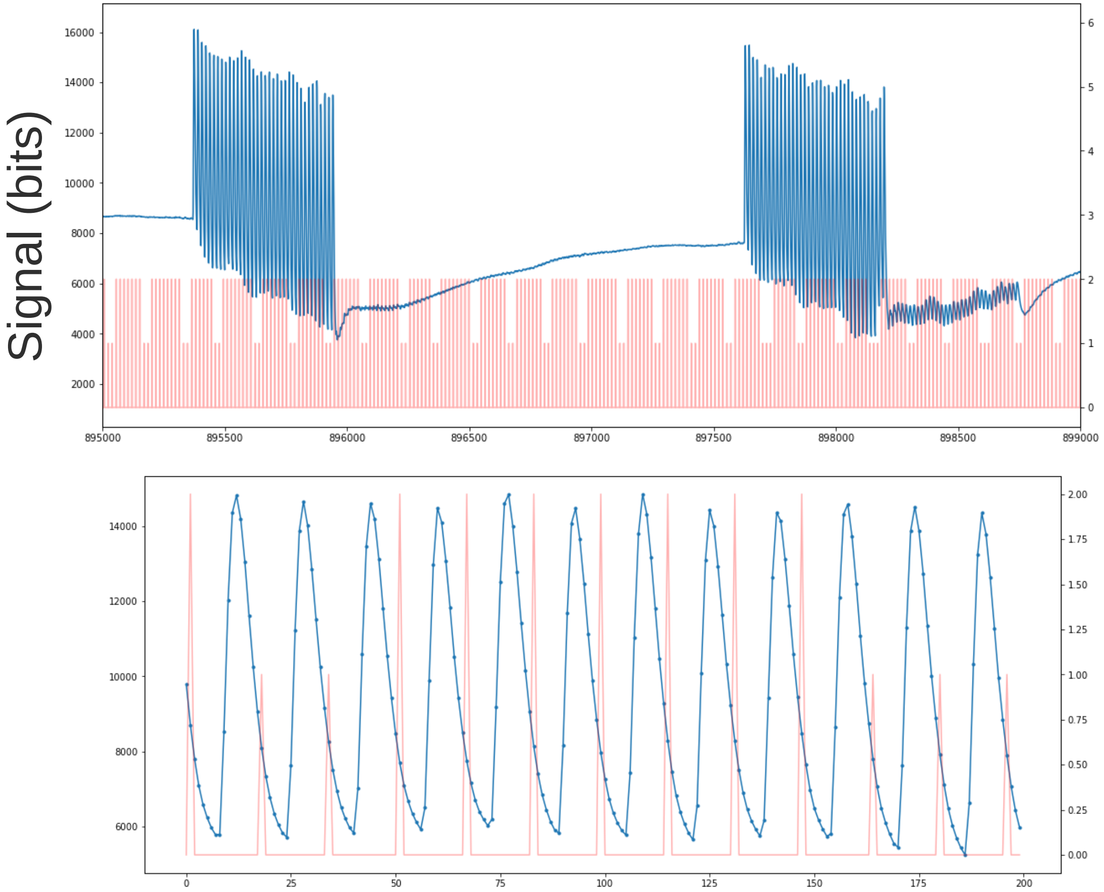
matrix[2][1999]

Attempting to characterize UFOs

Aim: look at bunch-by-bunch losses for UFO events, normalised by the bunch intensity.

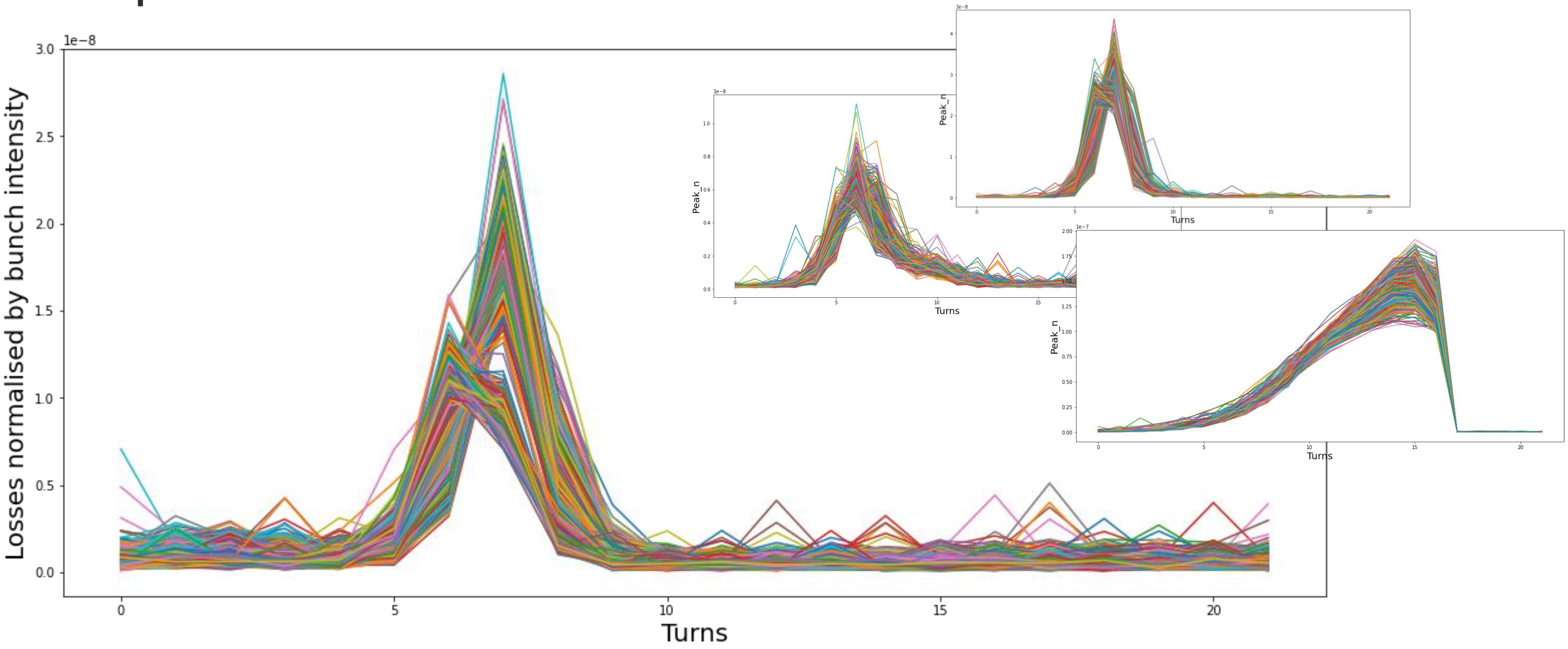
- How to remove the baseline?
- Rigorous approach: fit decaying exponential to bunch tails, extrapolate, numerically integrate…
- We choose to take the max-min over a bunch as a first approximation
- Then query the bunch intensities and filling pattern from NXCALs and normalise – set unfilled bunch slots to zero

Zooming in on the UFO signal



UFO signature? Filled bunches over turns

- **All filled bunches see the UFO and have losses on the same turn. This is unique compared to other events that were seen.**



Data processing - pydust

Challenges:

- **650MHz sample rate, 25ns bunch slots → either 16 or 17 samples per bunch. Use Awkward Array, a library for nested, variable-sized data.**
- **230,000 total events to process. Use Apache Spark – a distributed computing framework designed for large volumes of data.**
- **Combine dBLM readout with data coming from other devices – beam mode, bunch intensities, energy. Must align the timestamps.**
- **Inconsistencies in the data:**
  - Bunch flag inconsistencies – for some events, we have a missing bunch flag, which results in a bunch that has 32 bunches
  - Bunch count inconsistencies – for some events, we found turns with less than the normal 3564 bunch slots.

A python library, pydust, was created to process the dBLM data. All 230,000 events from this year processed and saved to file in parquet format (~400 Gb).

https://gitlab.cern.ch/machine-protection/ufo-studies

Filtering algorithm and results:

The following filtering algorithm was implemented to identify UFO events:

1. **Remove the events with inconsistencies**
2. **Count the number of turns that see losses, based on the following thresholds:**
   - At least 20 bunch slots must be filled
   - 80% of the filled bunches must have normalised losses above 0.1e-8.

   Choose events with losses for 1-20 turns – removes continuous losses

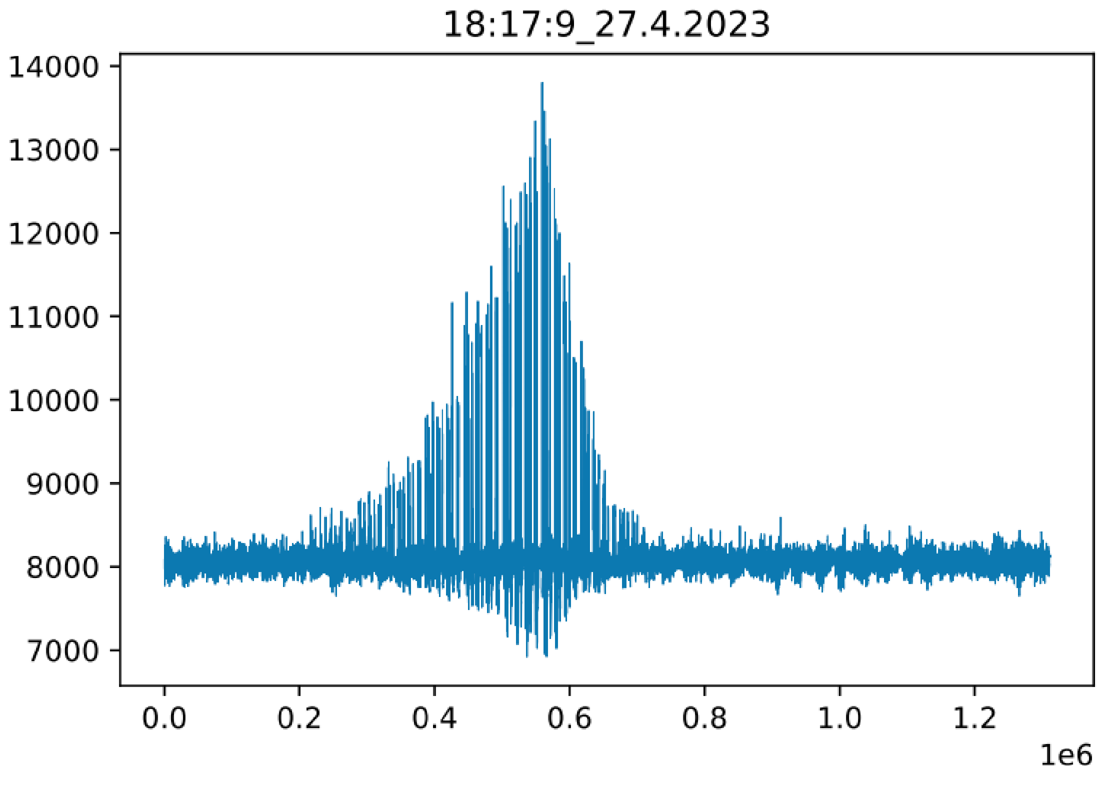
.3 – beam 1
.5 – beam 2

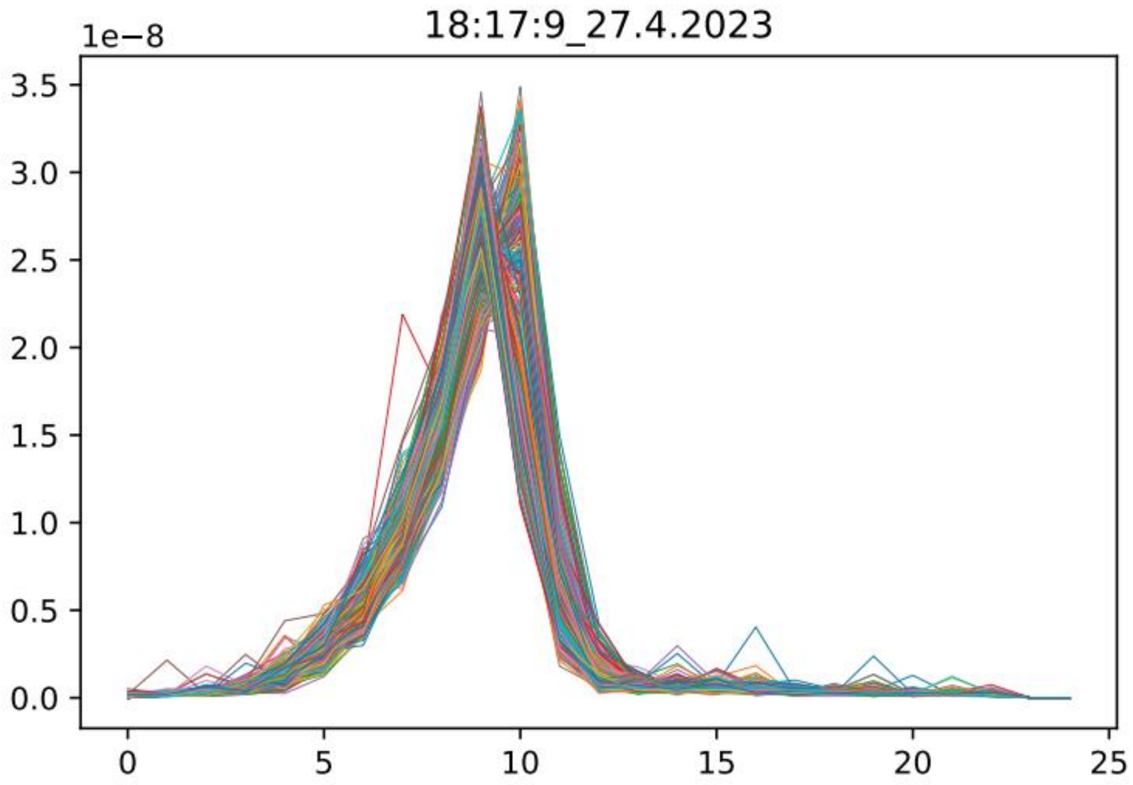
Device	All events	Filter 1	Filter 2	Manually labelled as UFO
HC.TZ76.BLMDIAMOND3.3	62,745	9294	27	13
HC.TZ76.BLMDIAMOND3.5	67,420	12,958	30	10
HC.TZ76.BLMDIAMOND2.3	10,050	7,405	4	1
HC.TZ76.BLMDIAMOND2.5	17,870	1,207	2	1
HC.TZ76.BLMDIAMOND.3	16,186	15,021	4	0
HC.TZ76.BLMDIAMOND.5	62,576	62,075	4	1
Total	**236,847**	**107,960**	**71**	**26**

All but one UFO events found occurred during the RAMP

An example picked out by the algorithm

device: HC.TZ76.BLMDIAMOND3.3
beam_mode: RAMP
energy: 6005.4
total_intensity: 6.00E+13
loss_turns: 7

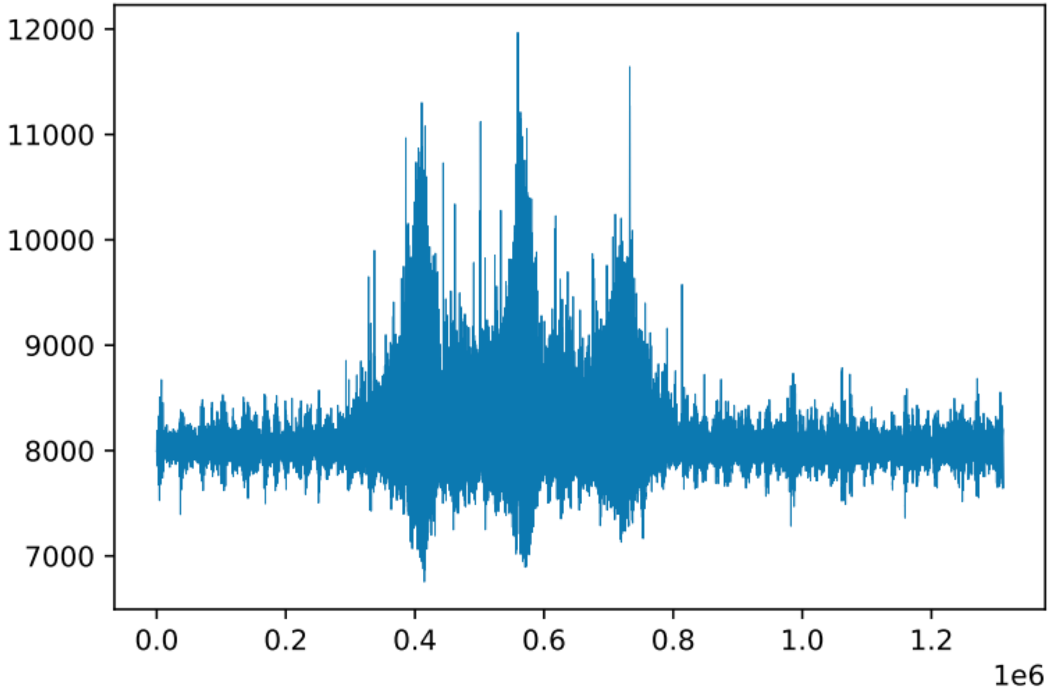


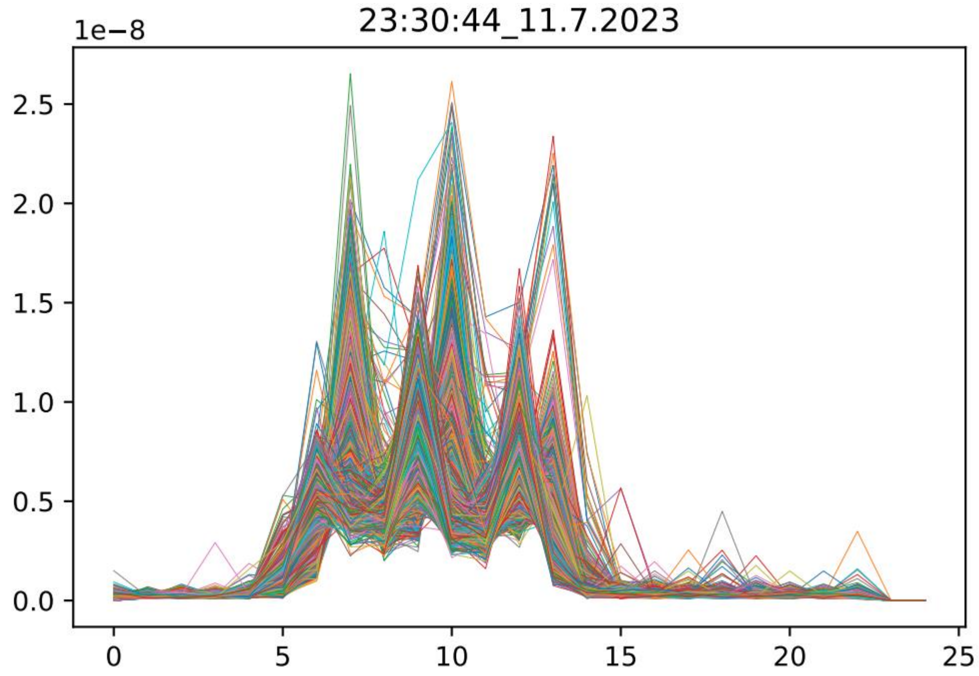


A few interesting events…

Multiple peaks – orbiting UFO?

device: HC.TZ76.BLMDIAMOND3.5
beam_mode: RAMP
energy: 5859.7197
total_intensity: 3.85E+14
loss_turns: 8





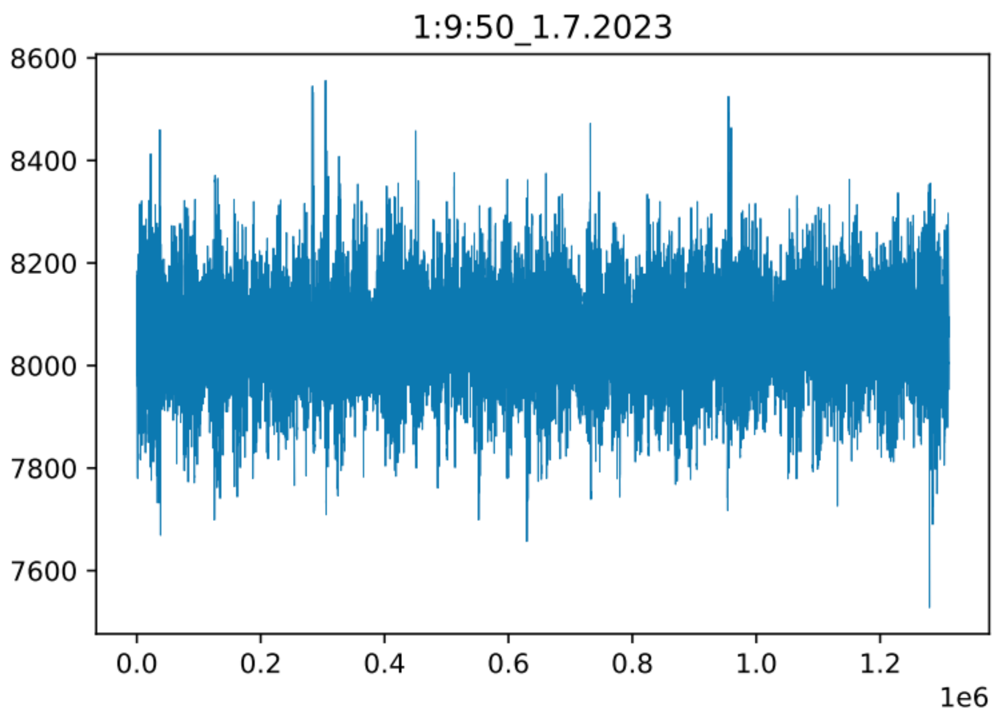
A few interesting events…

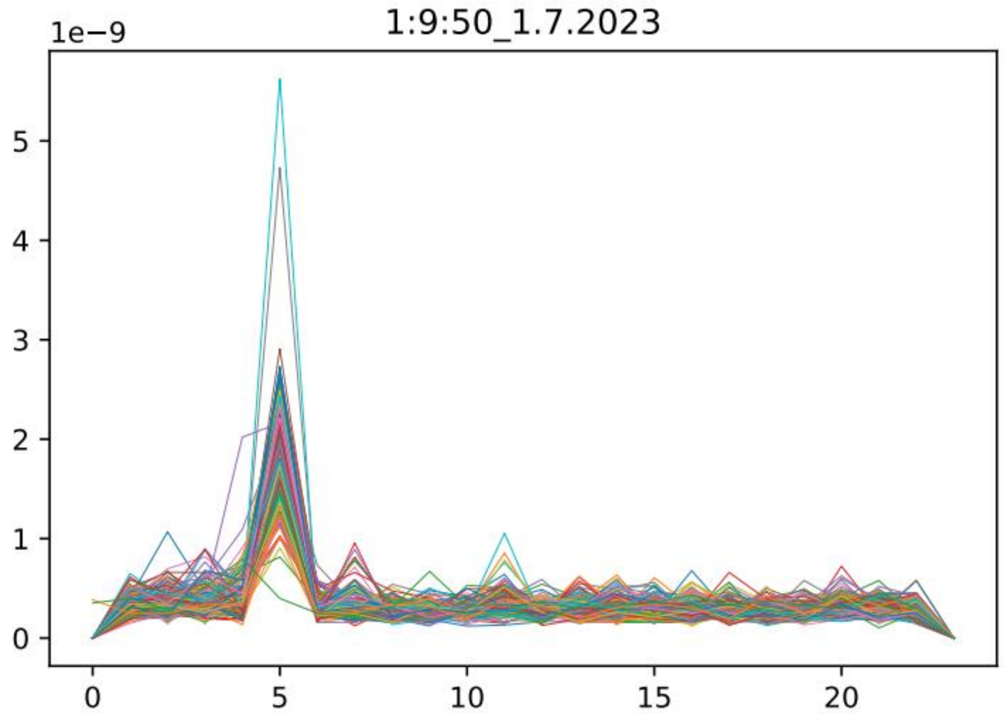
Can we use this approach to detect UFOs that aren't visible from the raw signal (low number of bunches)?

UFO buster triggered in same second

To be checked in more detail

device: HC.TZ76.BLMDIAMOND3.3
beam_mode: RAMP
energy: 5213.52
total_intensity: 1.27E+13
loss_turns: 1





Summary

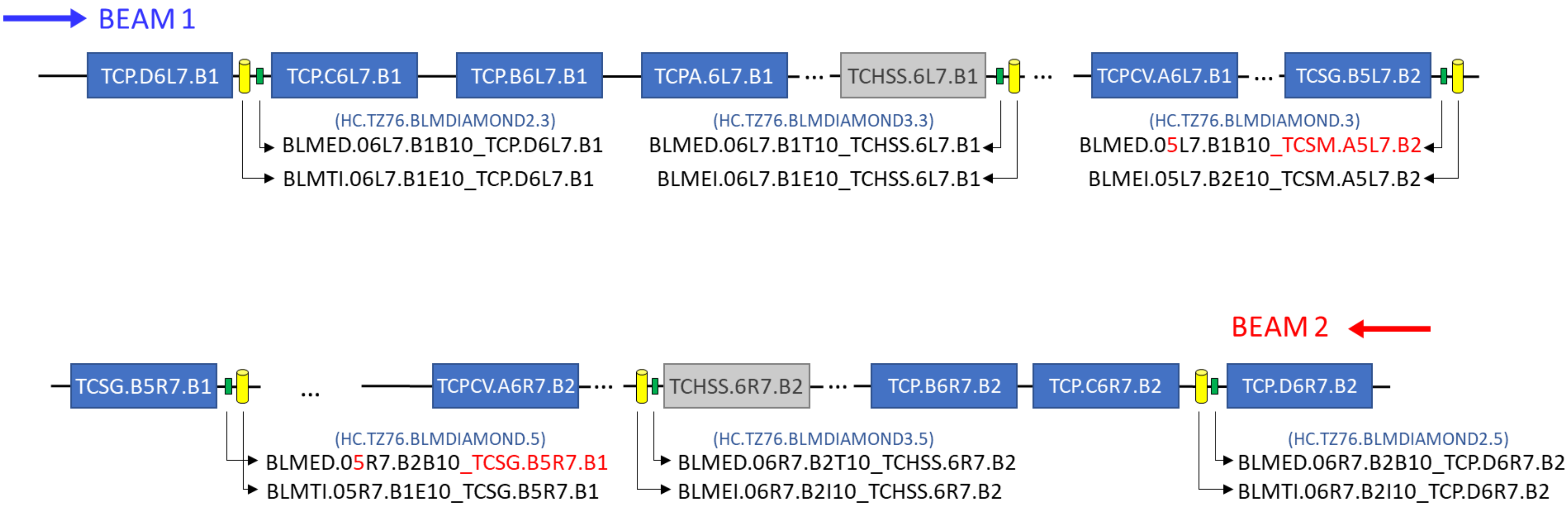
- Shown how bunch-by-bunch positions and normalised losses for N bunches can be used to calculate the UFO position
- Systematic analysis of UFO readout data from this year, finding 26 UFO events from a dataset of 230,000 events
- Dataframe for filtered and manually labelled UFOs saved to be studied further
- Post-processed data allows easy querying and filtering, i.e. distribution of trigger events over time, by device, beam mode, etc. This can be used to better understand the UFO auto-trigger algorithm.

Questions:

- Why do we have so many events in the ramp? Finding is consistent with studies from 2018. Detection bias or increased UFO rate?
- Why do we trigger primarily on 2 of the 6 devices?

home.cern

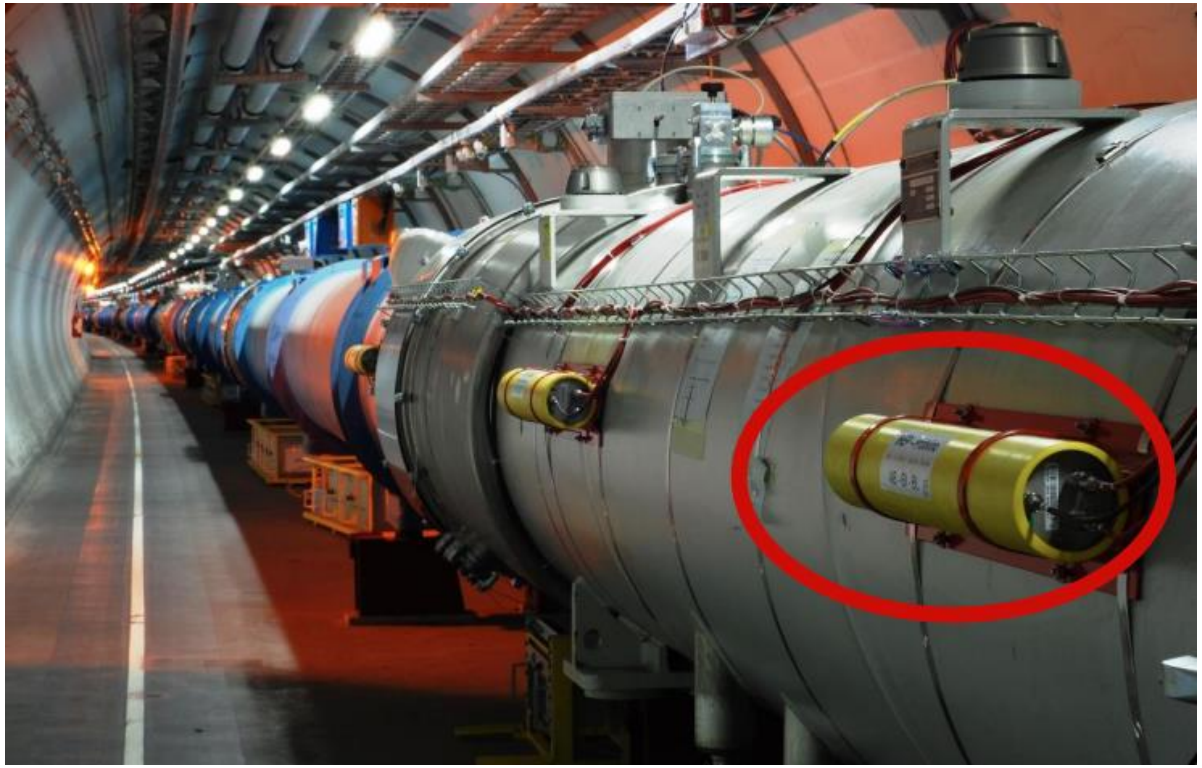
dBLM locations



Detecting UFOs

Ionization chamber BLMs

- Main beam loss monitoring system of the LHC
- 4000 detectors, covers all 27km
- 40μs time resolution
- Record local beam losses due to inelastic collisions between beam protons and dust particle nuclei
- Beam dumps when anomalous beam losses detected



Results of N bunch calculations