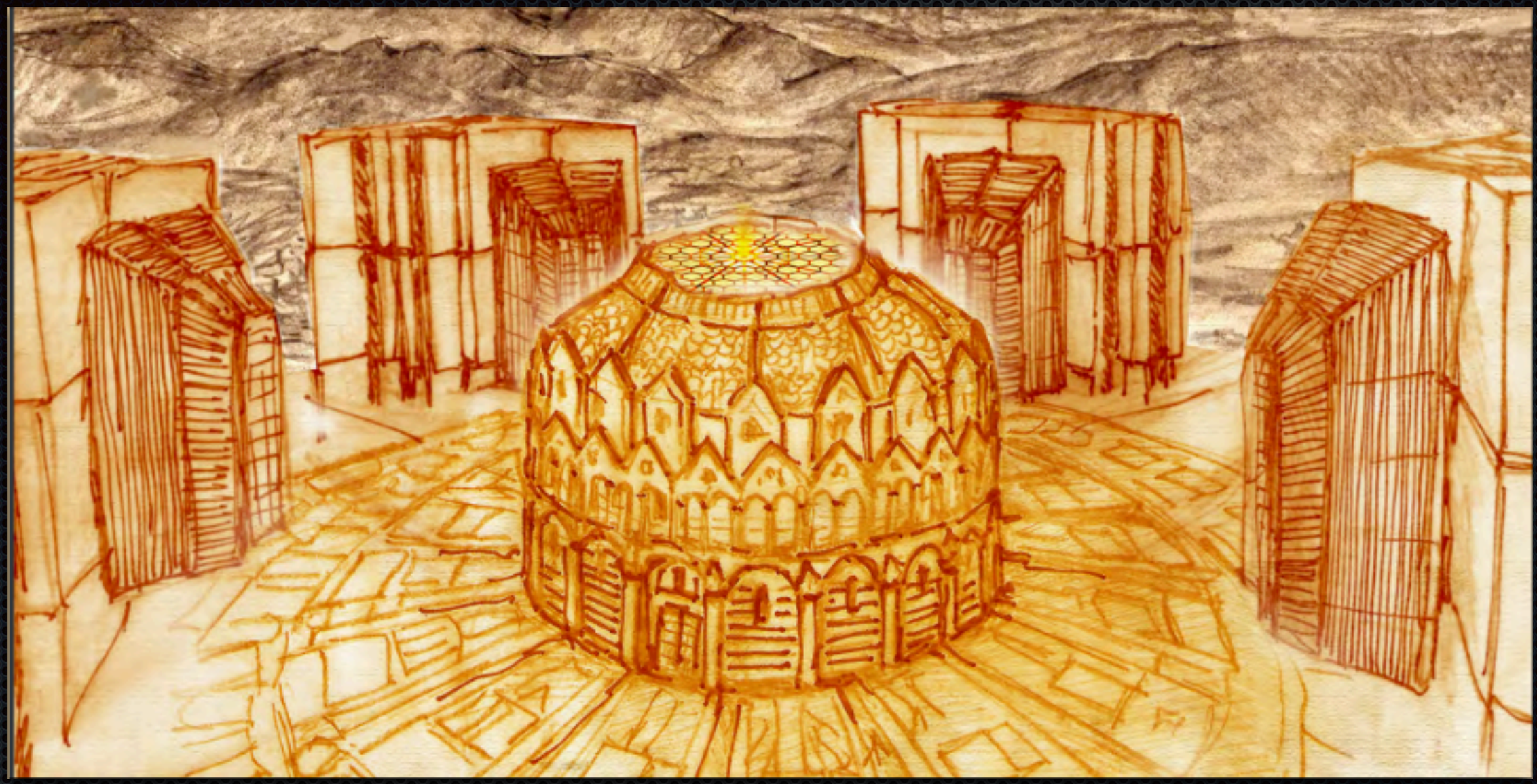
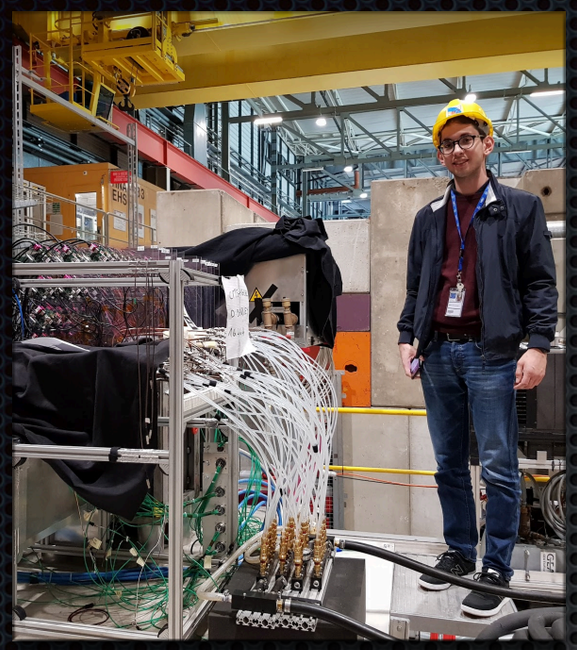


SWAN for HGCAL Beam Tests Analysis

Matteo Bonanomi
CMS
CNRS, LLR-Ecole Polytechnique
@matteobonanomi
@mbonanom

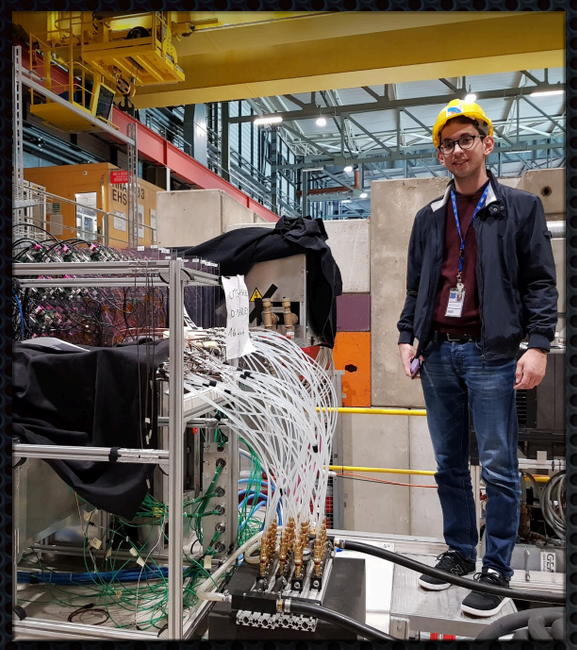
1st Users' Workshop
CERN
11/10/2019

Who am I?



- Matteo Bonanomi, 24 yo;
- Particle physics PhD student at École Polytechnique with the CMS esperiment;
- Working on the CMS High Granularity Calorimeter (HCGAL) beam tests;
- ROOT User/addicted for my analyses …

Who am I?

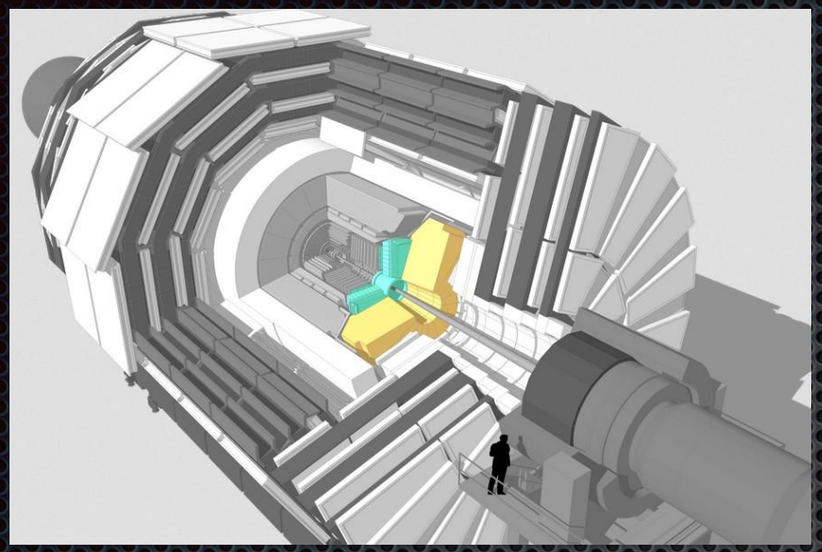


- Matteo Bonanomi, 24 yo;
- Particle physics PhD student at École Polytechnique with the CMS esperiment;
- Working on the CMS High Granularity Calorimeter (HCGAL) beam tests;
- ROOT User/addicted for my analyses … until my PhD started (😉🐍);
- SWAN User since ~1 year now.

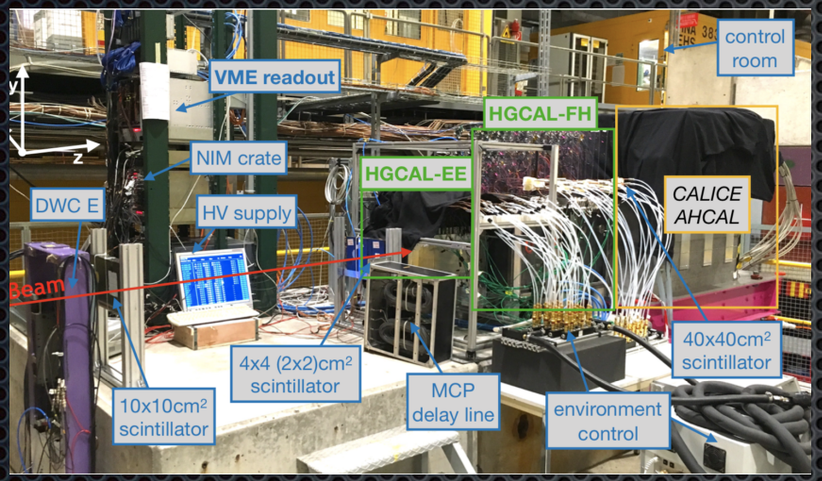
In this talk

- The HGCAL beam test analysis;
- From `ROOT` to `pandas` **using** `uproot`;
- SWAN in my everyday life;
- Feedbacks and conclusions.

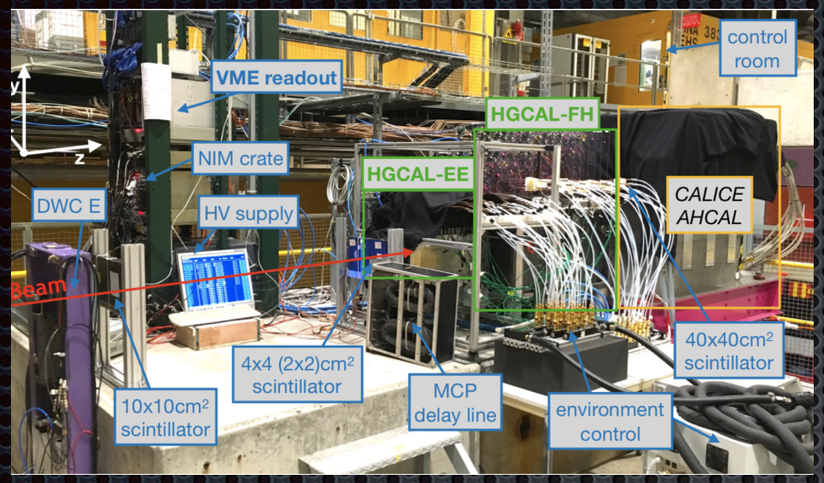
HGCAL in a nutshell



- **High Granularity Calorimeter** (HGCAL);
- Replacement of **CMS endcap calorimeters** for HL-LHC programme;
- First large scale test beam in October 2018: **O(1e6) data** to be analysed;
- **Analysis performed ~ entirely in SWAN** using the Jupiter notebook integration.



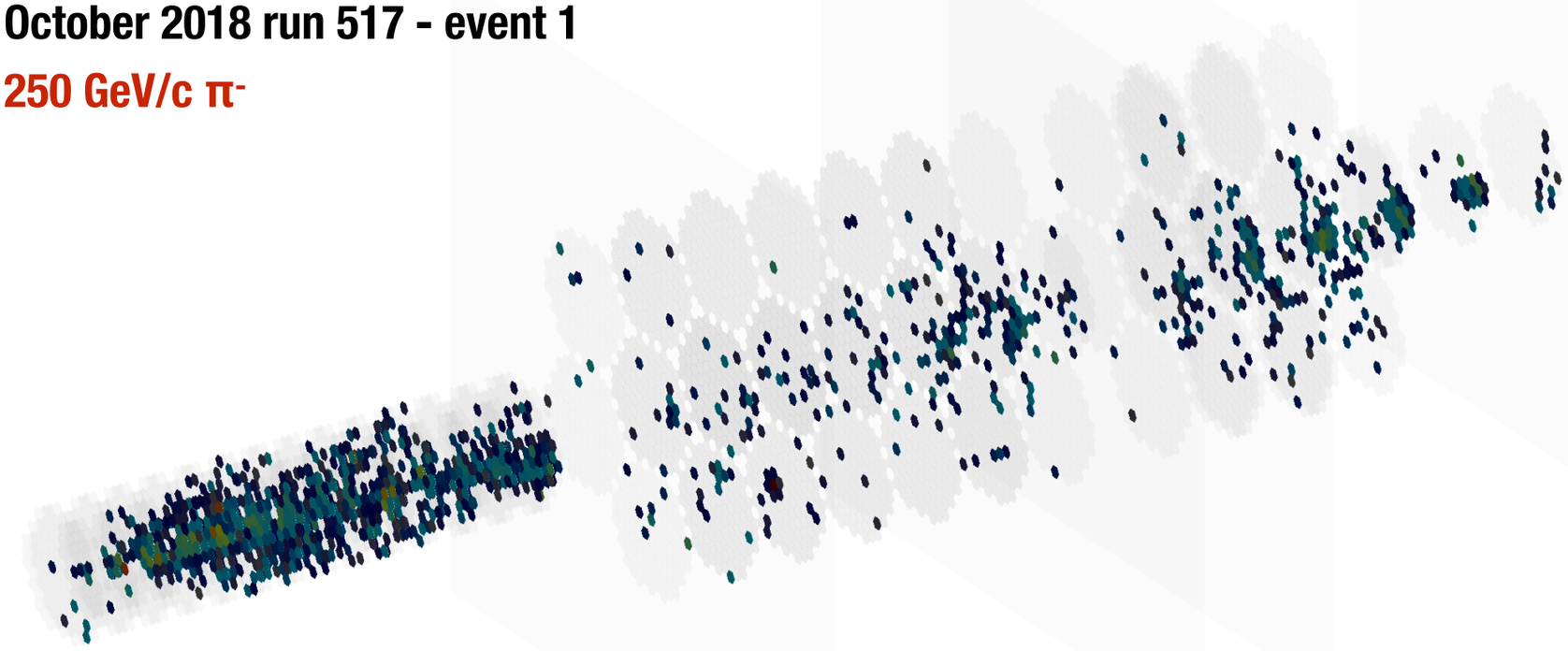
HGCAL in a nutshell



- **Tabular data: 1 row:1 event;**
- **O(1e6) data** to be analysed;

October 2018 run 517 - event 1

250 GeV/c π^-



Columnar data: from ROOT to data frames

```
*******************************************************************************
*Tree    :hits      : HGC rechits                                            *
*Entries :     6488 : Total =       591763506 bytes  File  Size =  106092451 *
*        :          : Tree compression factor =   5.58                       *
*******************************************************************************
*Br    0 :event     : event/i                                                *
*Entries :     6488 : Total  Size=      26725 bytes  File Size  =      13787 *
*Baskets :        4 : Basket Size=      44991 bytes  Compression=   1.90     *
*.............................................................................*
*Br    1 :trigger_timestamp : trigger_timestamp/l                            *
*Entries :     6488 : Total  Size=      52781 bytes  File Size  =       3970 *
*Baskets :        4 : Basket Size=      44991 bytes  Compression=  13.16     *
*.............................................................................*
*Br    2 :run       : run/i                                                  *
*Entries :     6488 : Total  Size=      26709 bytes  File Size  =        584 *
*Baskets :        4 : Basket Size=      44991 bytes  Compression=  44.92     *
*.............................................................................*
```

- ROOT TTrees with arrays per event;
- Difficult to handle analysis cuts and to combine TTrees;
- Easily converted into **flattened pandas df** using **uproot**

```
df = uproot.open(fname)[key].pandas.df(branches)
```

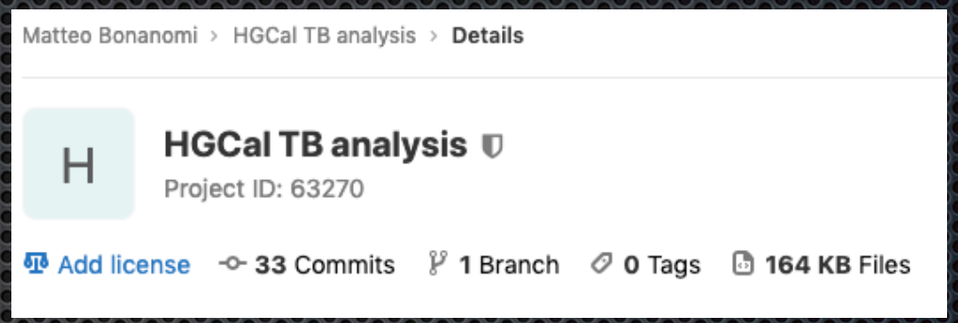
```
In [10]: %%time
         df_sim = do_df.do_df(energy, location_sim, isMC=True)

         CPU times: user 13.3 s, sys: 6.93 s, total: 20.2 s
         Wall time: 20.3 s

In [11]: df_sim.head()

Out[11]:
```

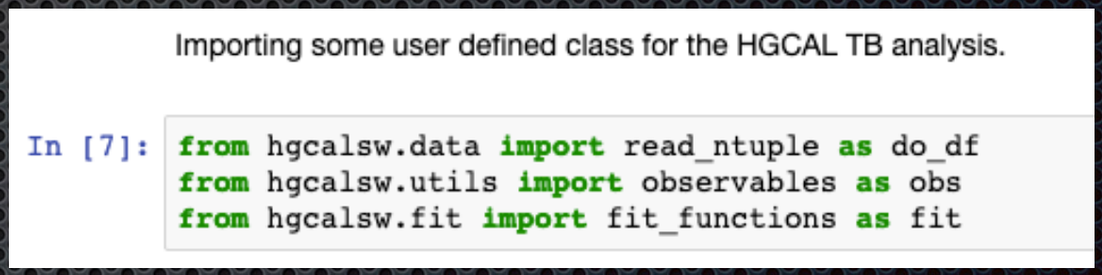
	rechit_chip	rechit_module	rechit_channel	rechit_energy	rechit_layer	ntracks	dwcReferenceType	b_x	b_y
event									
0	2.0	78.0	34.0	19.437500	1.0	1.0	13.0	-2.835938	1.137695
0	2.0	78.0	38.0	1.046875	1.0	1.0	13.0	-2.835938	1.137695
0	2.0	78.0	28.0	7.843750	1.0	1.0	13.0	-2.835938	1.137695
0	2.0	90.0	36.0	1.625977	2.0	1.0	13.0	-2.835938	1.137695
0	2.0	90.0	34.0	73.125000	2.0	1.0	13.0	-2.835938	1.137695

The power of 🐍 and SWAN (I)

My SWAN configuration

- Software Stack: 96 **python 3**
- Platform: CentOS 7
- # cores: 4
- Memory: 10 GB
  - 16 GB not available on all instances





The power of 🐍 and SWAN (II)

- ✓ SWAN offers a direct **access to eos** areas, along with a synchronisation with **CERN-Box**;
- ✓ Most of the python packages are already installed, but it is easy to add user defined or missing libraries;
- ✓ **Typical of a beam test analysis**: we are interested in **quick plots** for comparisons and to decide next analysis steps.

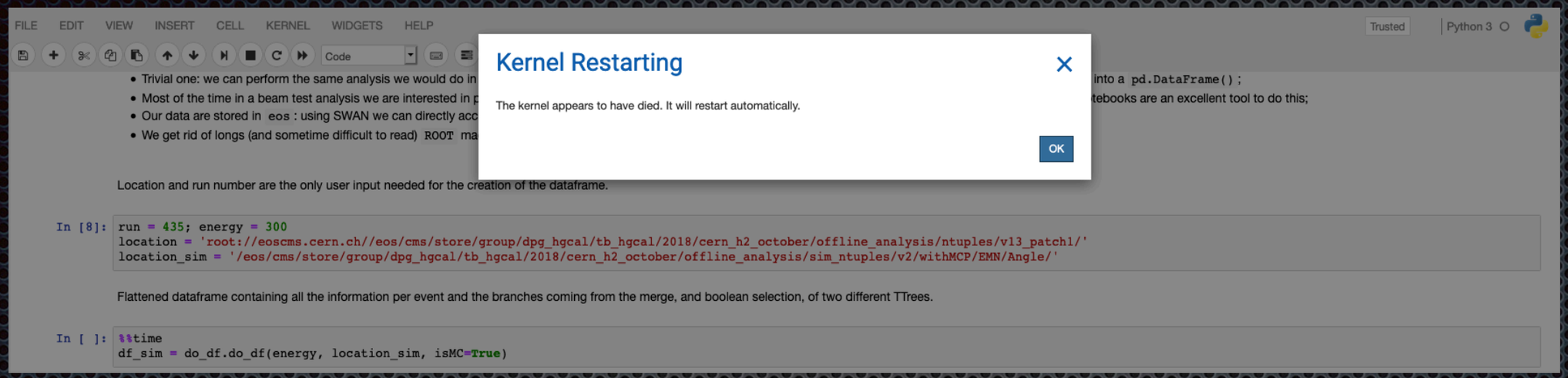
☹️ : Verbose **ROOT macros** difficult to modify and to keep under control when many plots are needed;

🙂 : **Jupyter notebooks** to exploit python (`pandas df`) quickness in handling data and visualization.

A day in life: beam test analysis

Let's have a look **`at this notebook`** to better understand my use-case of SWAN.

The downside of SWAN




```
df = uproot.open(fname)[key].pandas.df(branches)
```

When accessing large files (~3GB) the conversion to pandas df causes crashes of the kernel:

One solution is to work only with **chunks of files**, but this would impact on the statistics.

Why this problem occurs? How the `.pandas.df( )` command tweaks the memory?

: What would be nice to have

- Could more memory help to solve the kernel dying issue?
- Notebook extensions for an improved usability;
- It would be nice to have *collaborative notebooks*, shared among analysers;
- Same performance in accessing files directly from `eos` and from `xrootd`;

Let's wrap-up…

- SWAN + 🐍 in Jupyter notebooks are an extremely useful tool for analysers;
- SWAN Has many pros: synch with CERN-box, direct access to eos and "ready-to-go" environment ;
- SWAN Presents some downsides but none of them is limiting for the analysis.

Let's wrap-up…

- SWAN + 🐍 in Jupyter notebooks are an extremely useful tool for analysers;
- SWAN Has many pros: synch with CERN-box, direct access to eos and "ready-to-go" environment ;
- SWAN Presents some downsides but none of them is limiting for the analysis.

Let's wrap-up…

- SWAN + 🐍 in Jupyter notebooks are an extremely useful tool for analysers;
- SWAN Has many pros: synch with CERN-box, direct access to eos and "ready-to-go" environment ;
- SWAN Presents some downsides but none of them is limiting for the analysis.

Looking forward to see next SWAN steps and enjoy its improvements!