Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Benchmark interactive analyses ongoing at INFN Napoli

Adelina D'Onofrio, Elvira Rossi, Francesco Cirotto, Francesco Conventi, Orso Iorio, Antimo Cagnotta, Antonio D'Avanzo, Gianluca Sabella, Bernardino Spisso, Francesco Gravili

Spoke 2 - WP2 weekly Meeting, 5th December 2023

Outline

- Brief introduction to the infrastructure used
- Use cases tested:
  - FCCee: simple test on Zee samples
  - CMS: top quark+MET analysis
    - ➡ collaboration with INFN Perugia
  - ATLAS: stop to 4-body SUSY analysis
    - ➡ collaboration with INFN Lecce

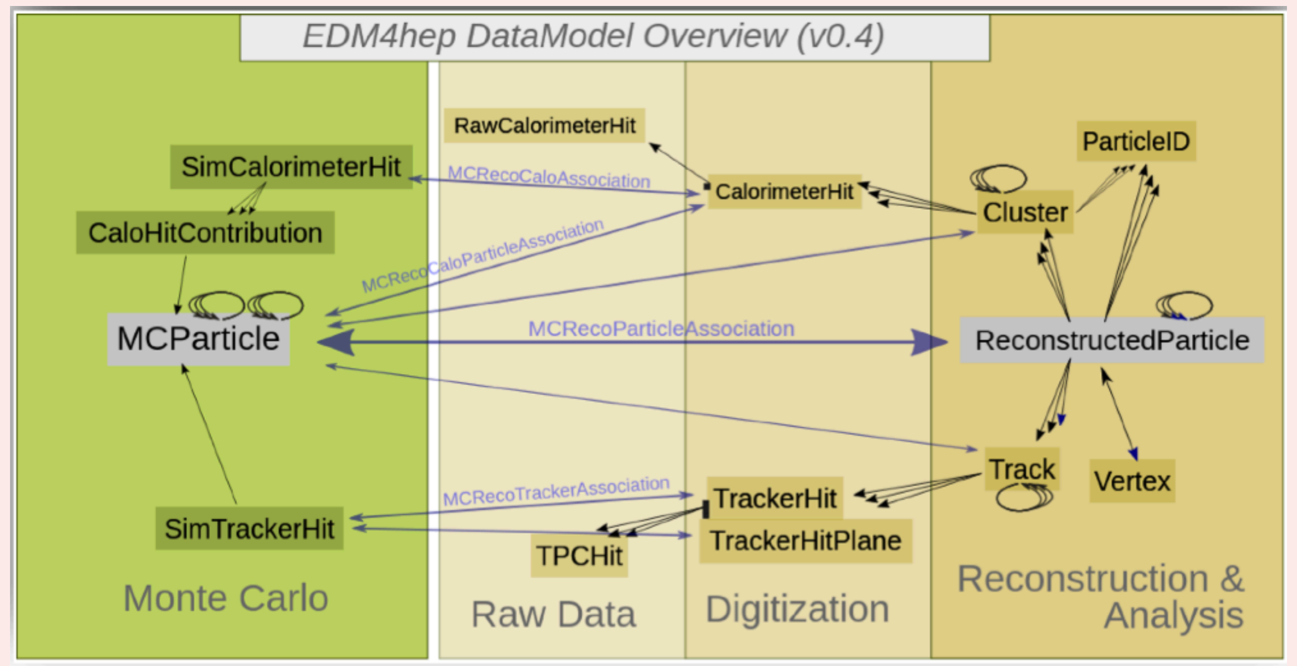
INFN Napoli infrastructure

- The local deployment is based on the *Open-Stack IaaS* paradigm
- Starting from the already existing *I.Bi.S.CO* installation, several updates were performed
- The cluster is made up of 2 identical virtual machines, each equipped with 1CPU quadCore and 8GB RAM, currently expanded up to 12 cores and 64GB
- Rocky Linux 8.6 is the operating system
- 2 nodes are equipped with **Docker** (20.10) for containerisation and **Kubernetes** (1.26.3) for the orchestration
  - One node plays as controlplane, etcf & worker; the other node acts as a plain worker
- The cluster is equipped with **JupyterHub** & **JupyterLAB** where the user can play with **Python, ROOT & Dask** libraries

13/10 WP2.5 presentation *link*

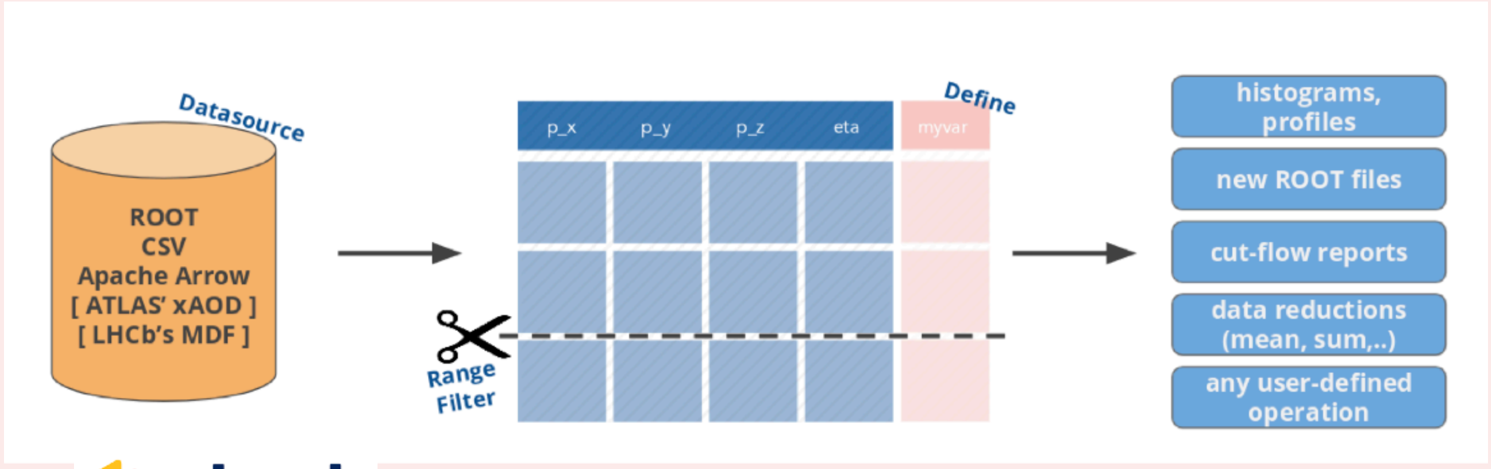
FCCee use-case Workflow

EDM4hep input data format

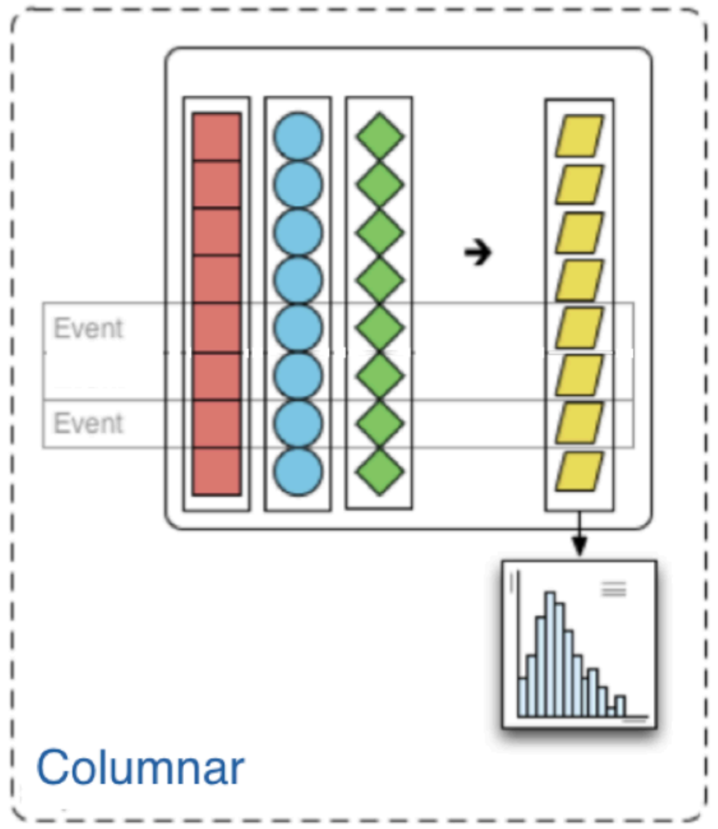


flat input ntuples

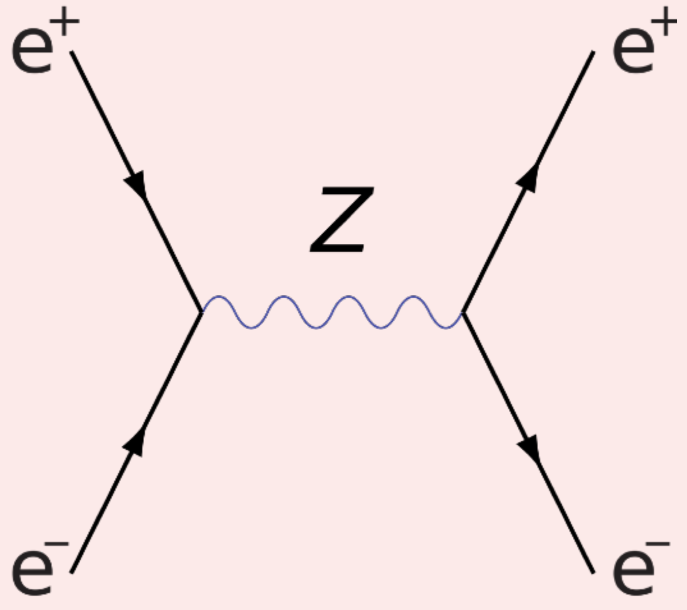
RDataFrame



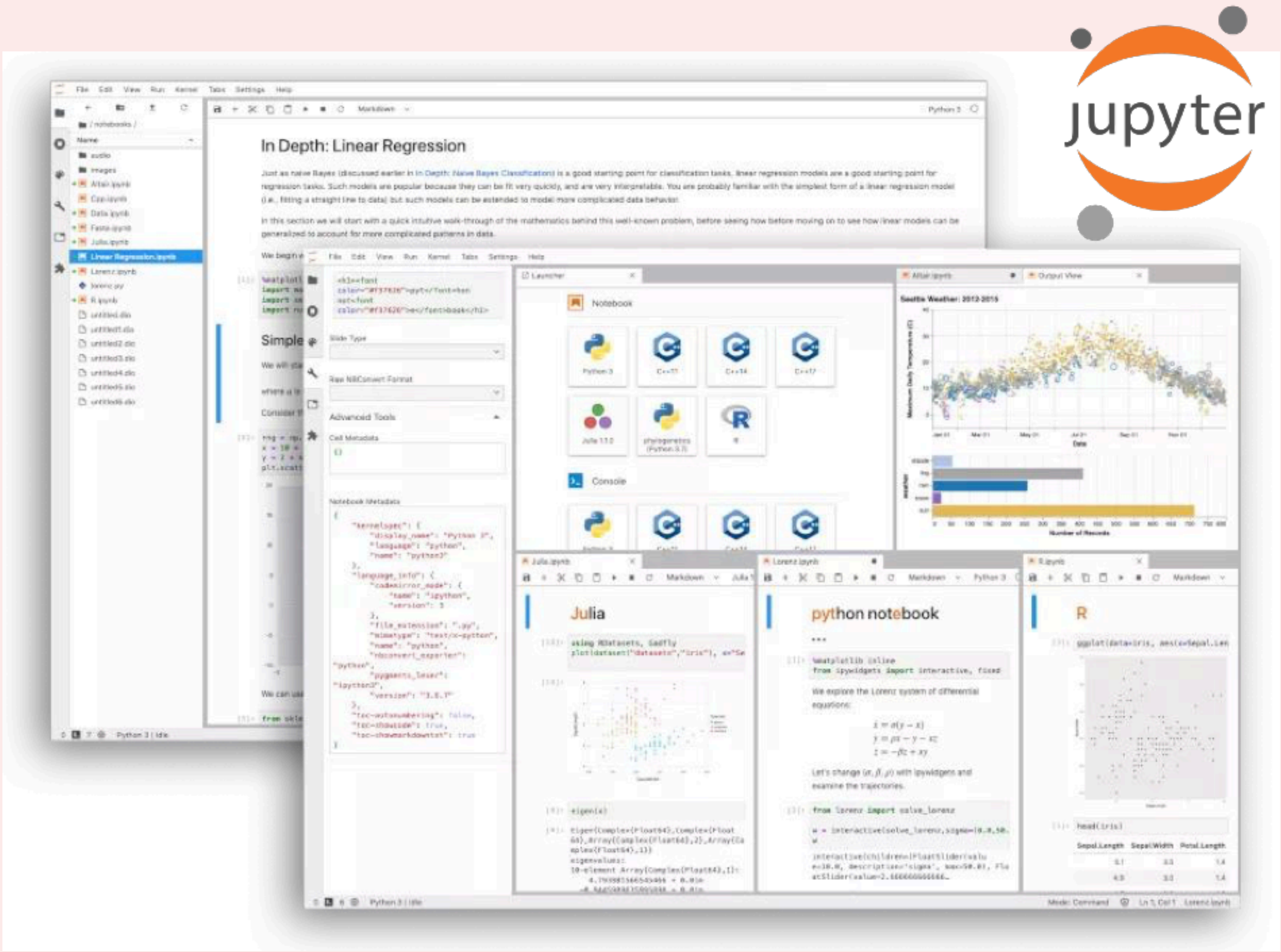
+ dask used as backend



Use case



New approach to data analysis

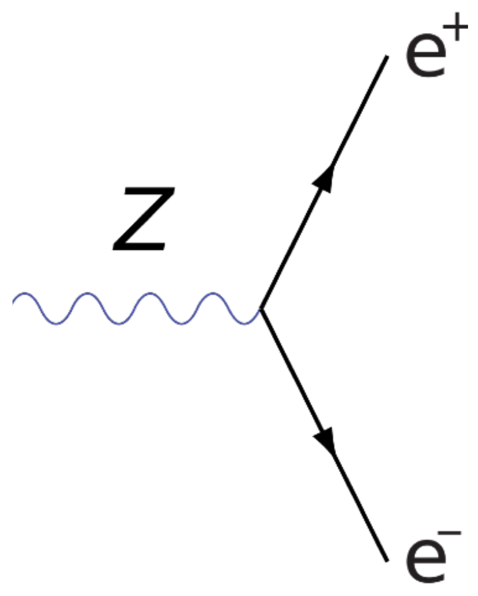


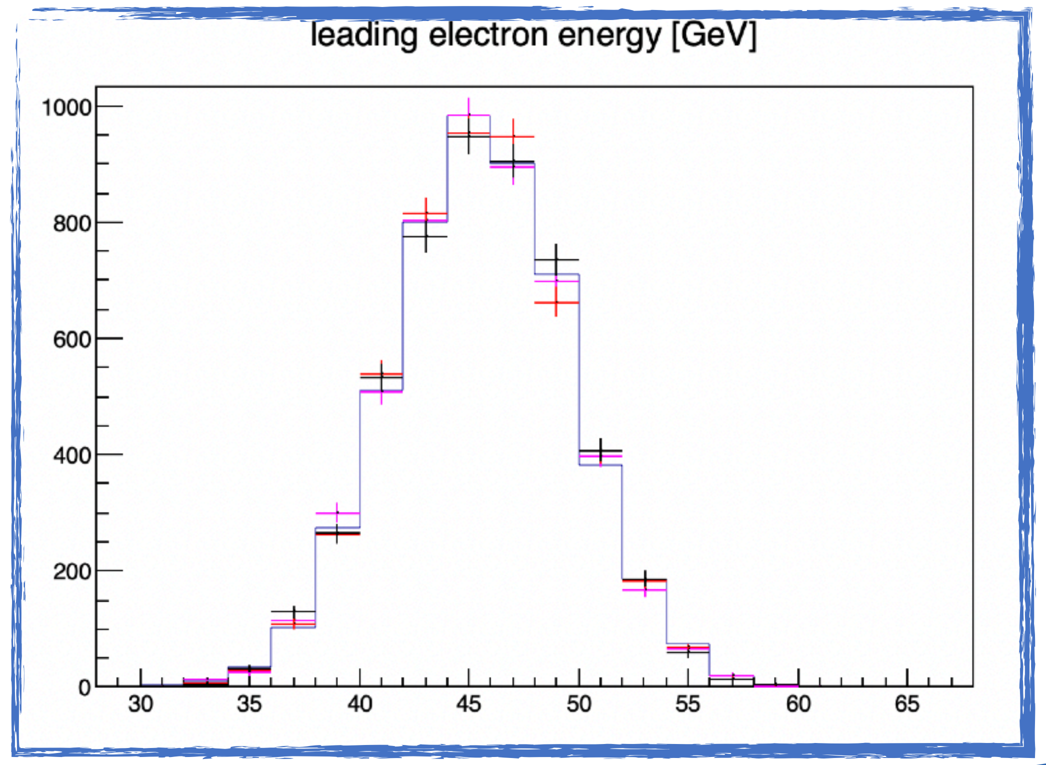
Feasibility study & Preliminary performance evaluation

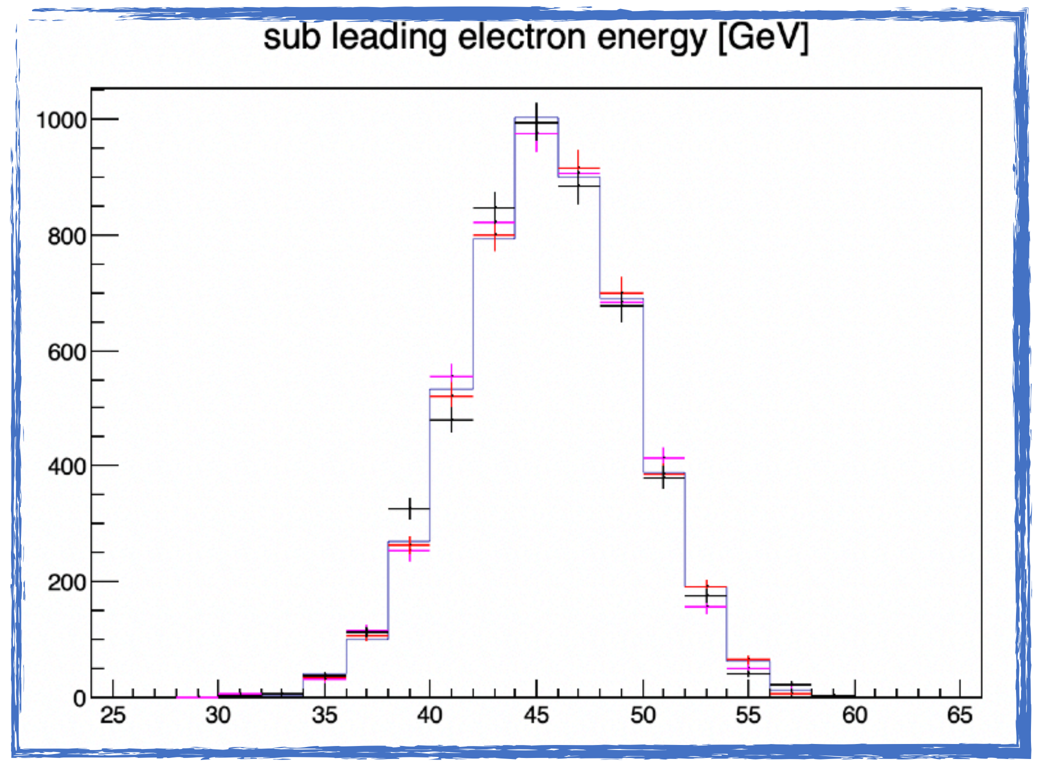
13/10 WP2.5 presentation *link*

Simple test

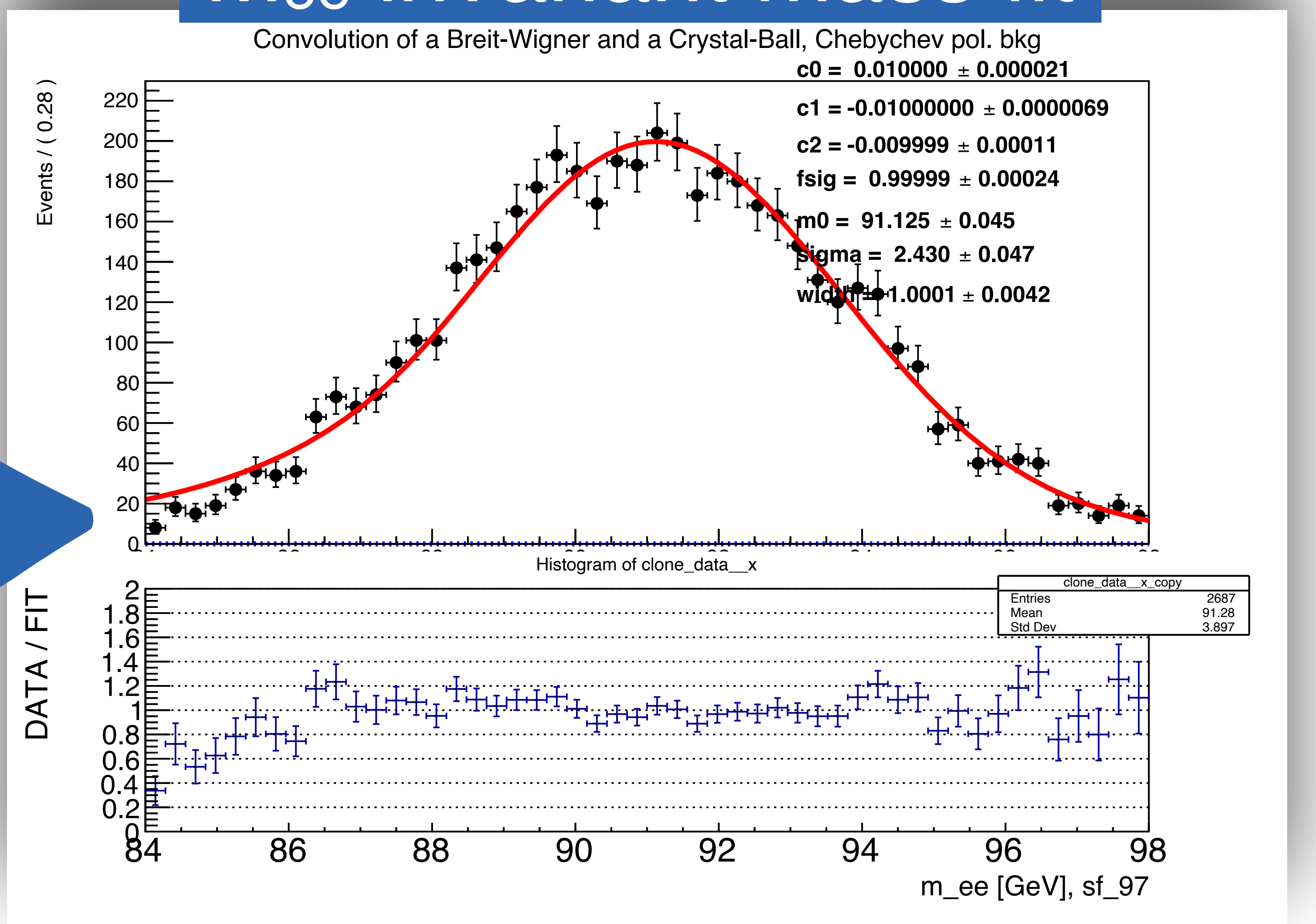
- FCCee simulation: /eos/experiment/fcc/ee/tmp/ee_Z_ee_EDM4Hep.root
  - 5k events, scaled to 1M events replicating the available dataset
  - Mimic systematic variations, gaussian smearing the electrons energy to compute M_{ee} resolution





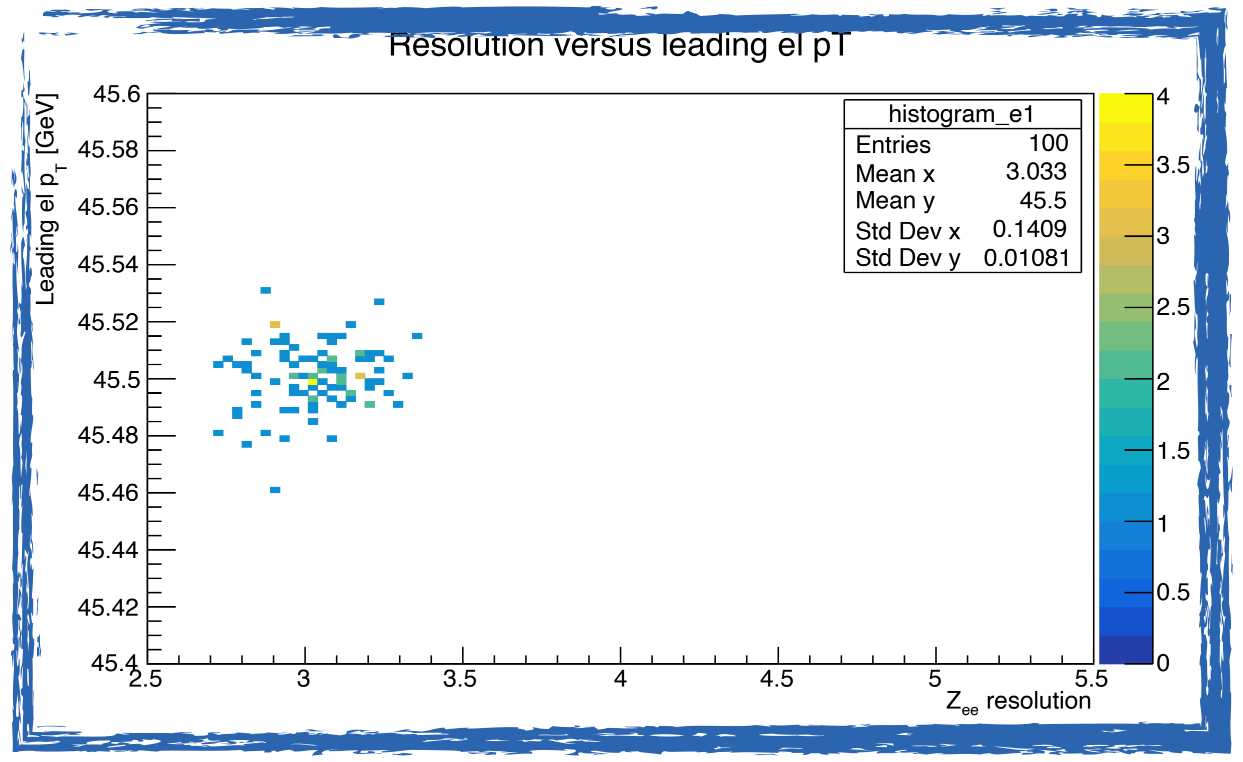


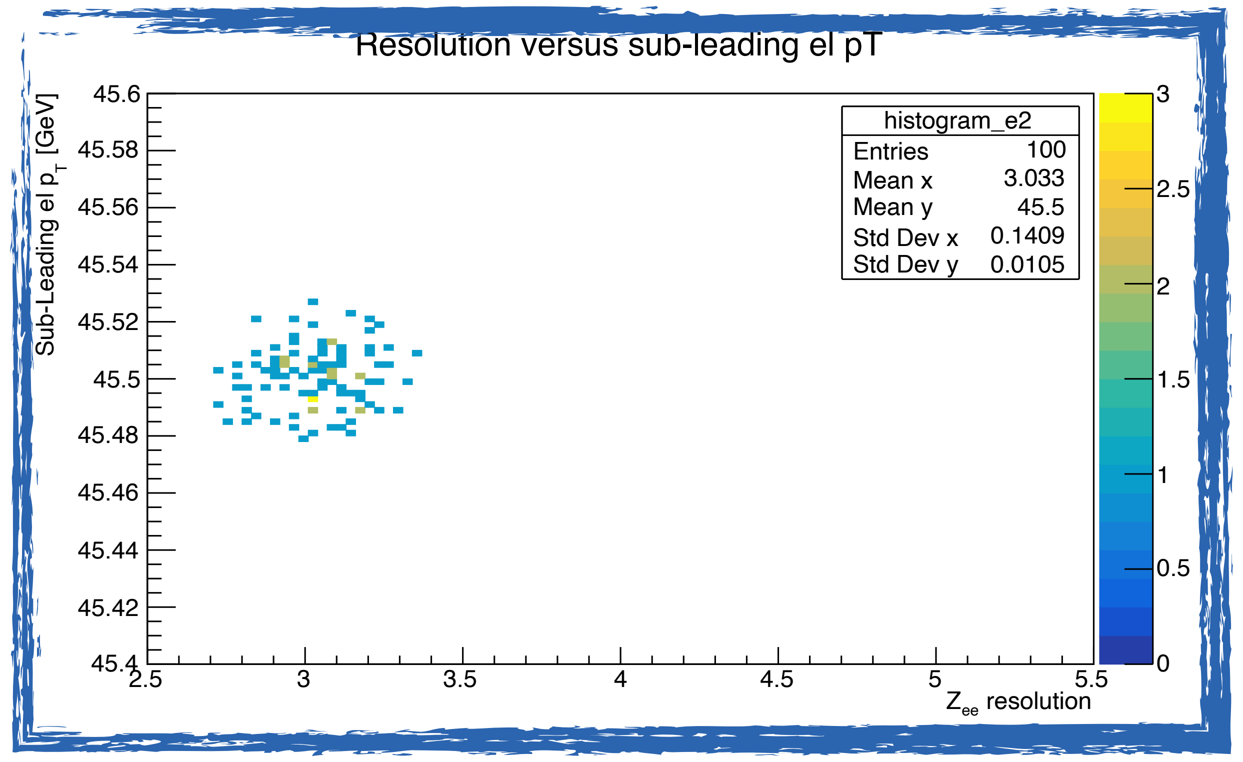
M_{ee} invariant mass fit



Selection and histogramming interactively via RDataFrame on JupyterHub

[github link to the code]





Preliminary results

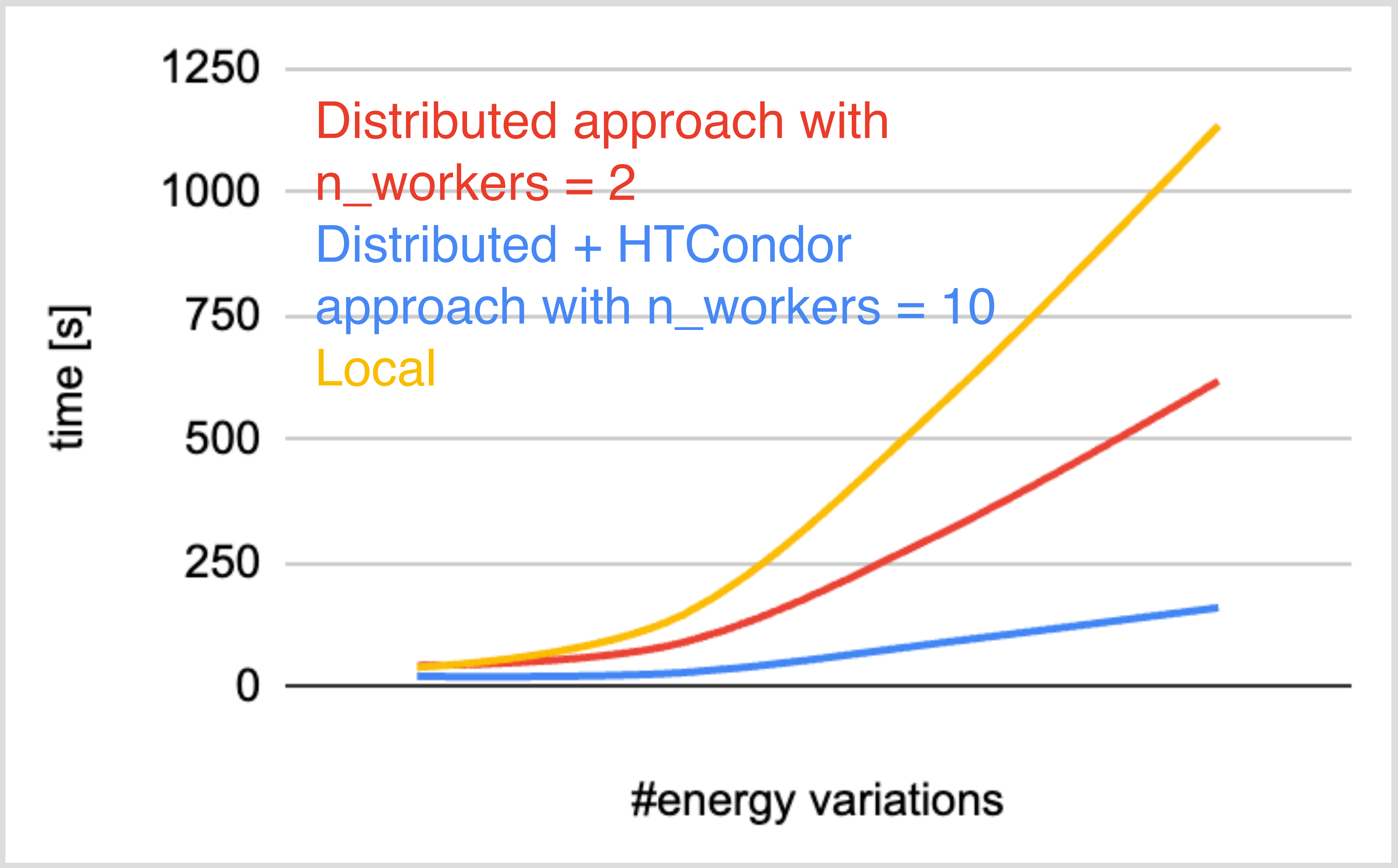
Scaling without changing your code

```
from dask.distributed import LocalCluster, Client
if distributed == True:
    RDataFrame = ROOT.RDF.Experimental.Distributed.Dask.RDataFrame
    ROOT.RDF.Experimental.Distributed.initialize(my_initialization_function)
else:
    RDataFrame = ROOT.RDataFrame
    my_initialization_function()
```

} Local

⋮ No changes required to the rest of the code

```
df = df.Define('w_nominal', '1')
df = df.Define("m_e","0.0005124") #GeV
df_ge = df.Define("goodelectrons", "Particle.charge[0]*Particle.charge[1] < 0.").Filter("goodelectrons > 1")
```

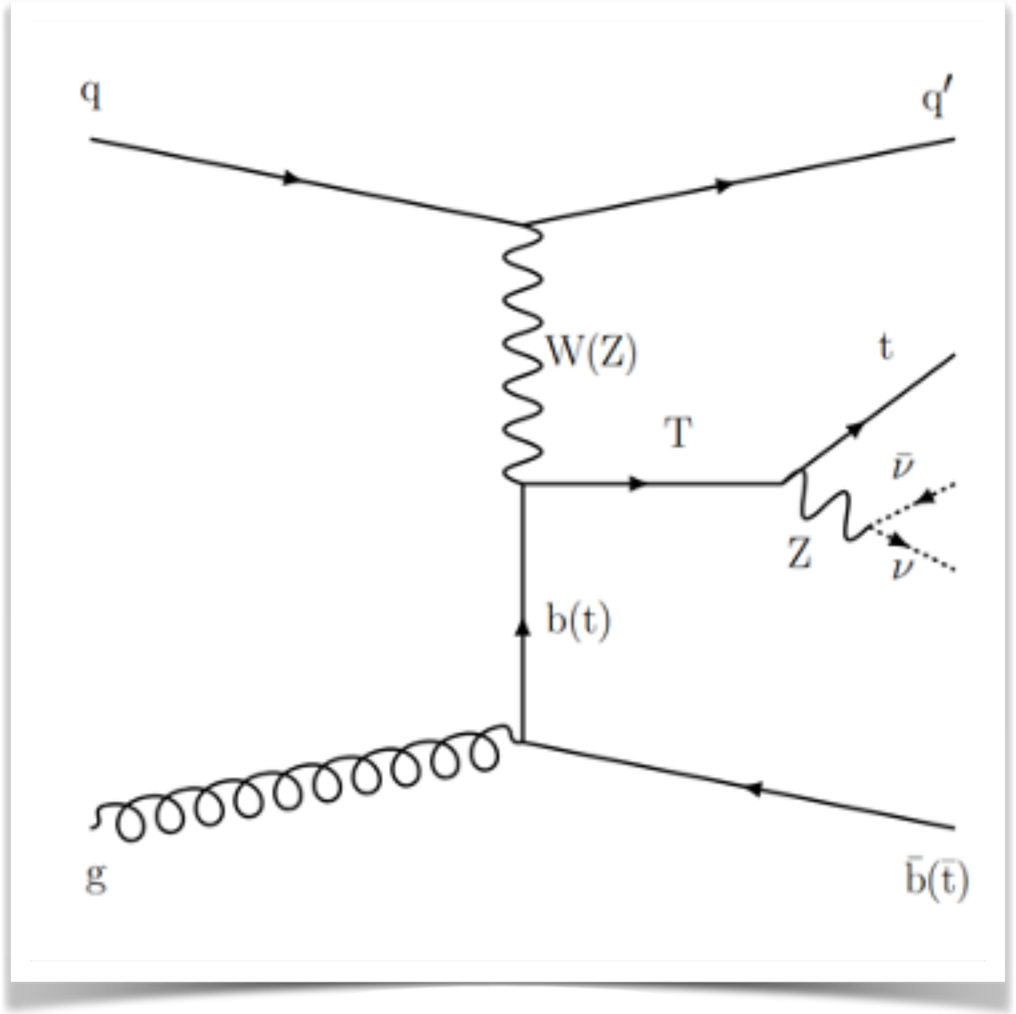


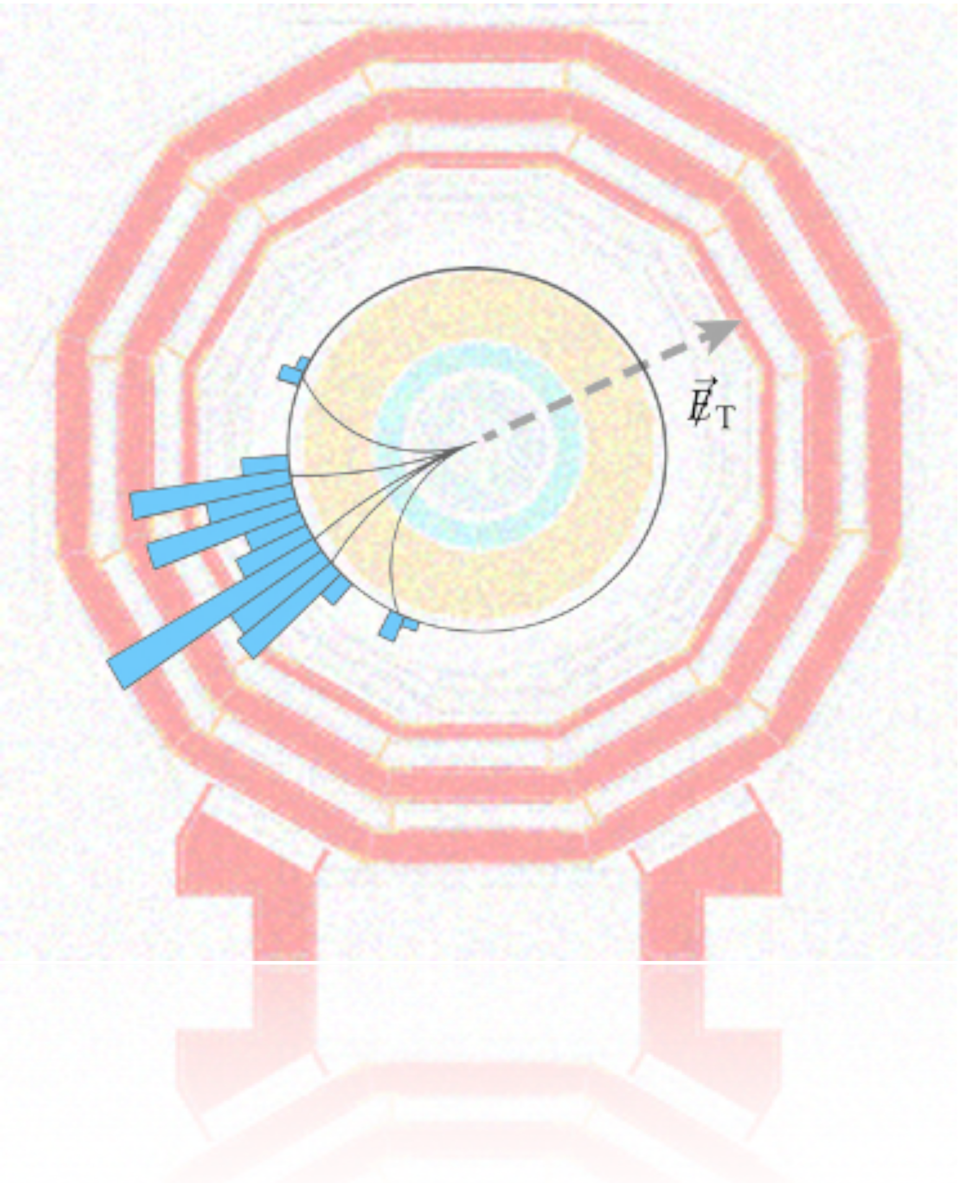
- Exploiting the distributed approach, the execution time halves wrt the local approach
- Moving to a Dask+HTCondor model, we gain up to another factor 2
  - Increasing the number of workers, the execution time further improves
- Advantage: use this use case as simple test for who wants to benefit from the WP5 infrastructure

CMS use-case

- Early Run 3 analysis (2022-2023 data taking)
- Beyond Standard Model searches
- Vector-Like Quark T in T→ tZ channel
- Final state: hadronic Top quark and Z (νν)
- Development of the already published full Run 2 analysis *JHEP05(2022)093*, with the idea to extend the results interpretation to more models predicting the same final state
  - Dark Matter production in association with a Top quark
  - Technicolor models *The Radiative Flavor Template at the LHC*

24/05 WP2.5 presentation *link*





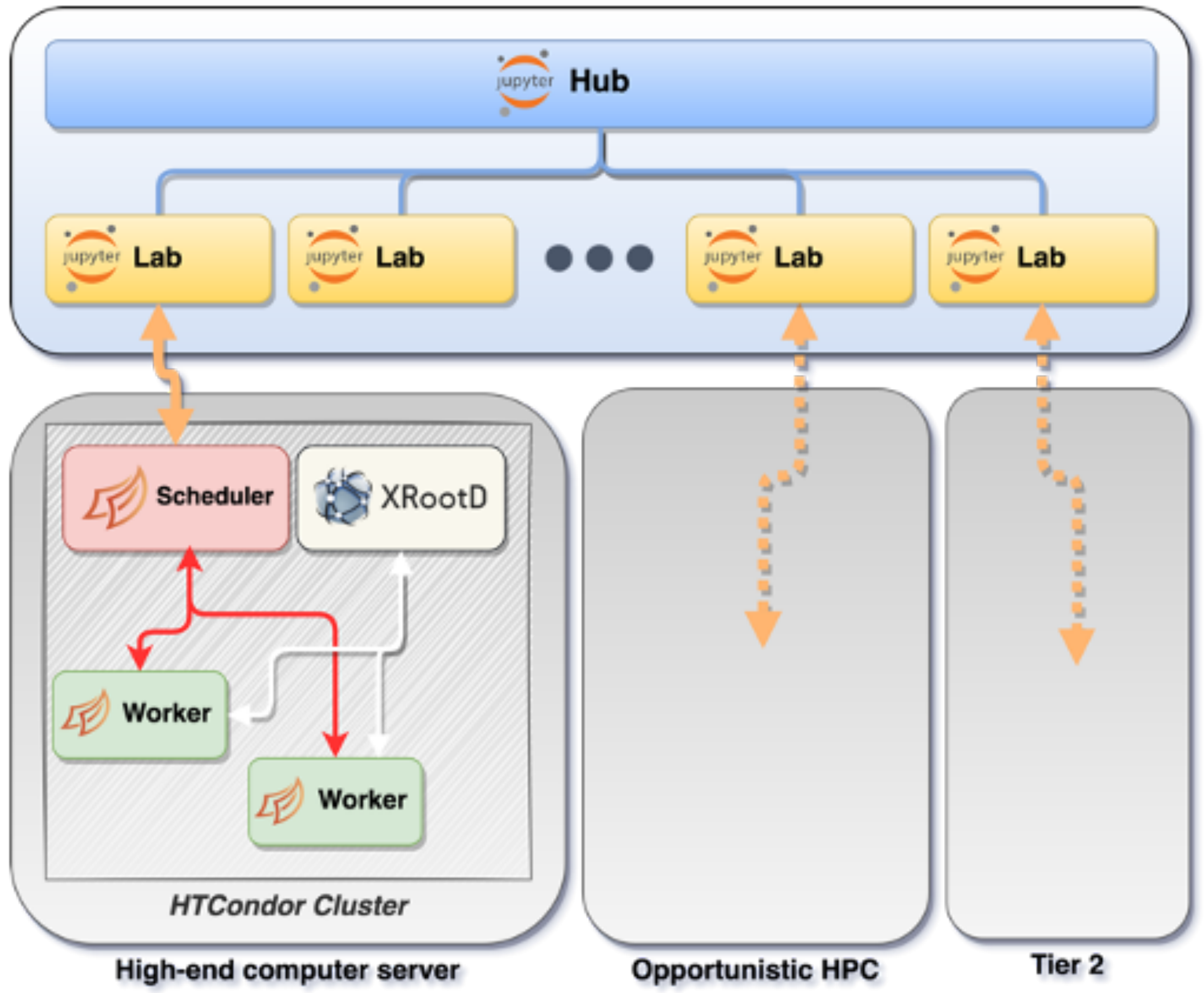
State of the art

CRAB
Cms Remote Analysis Builder

Workflow in 2 steps:

- Data preprocessing, evaluation through ML model.
  Using CMS NanoAOD tools (pyROOT-based) and CRAB.
- Skimming and selection using Interactive Analysis
  - Input: ntuple from the 1st step
  - Selection + variables calculation through RDataFrame
  - Distribution of the process using Dask
  - Output: TH1D easy to manage, also possible to store snapshot using remote storages

Working on Perugia's analysis facility

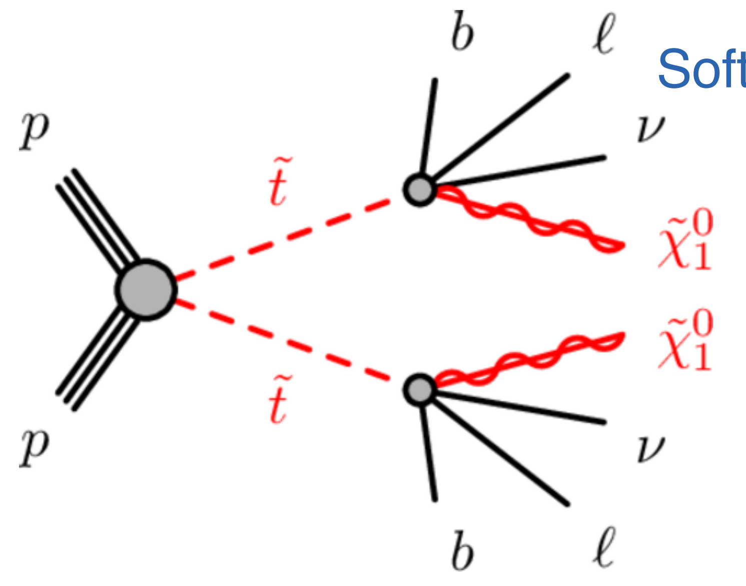


- Analysis still far from the end, more processes will have to be added that will slow it down
- Currently the results are very promising
  Time reduced from ~1d to ~3h and there is still room for development

ATLAS use-case

SUperSYmmetry: Beyond Standard Model theory

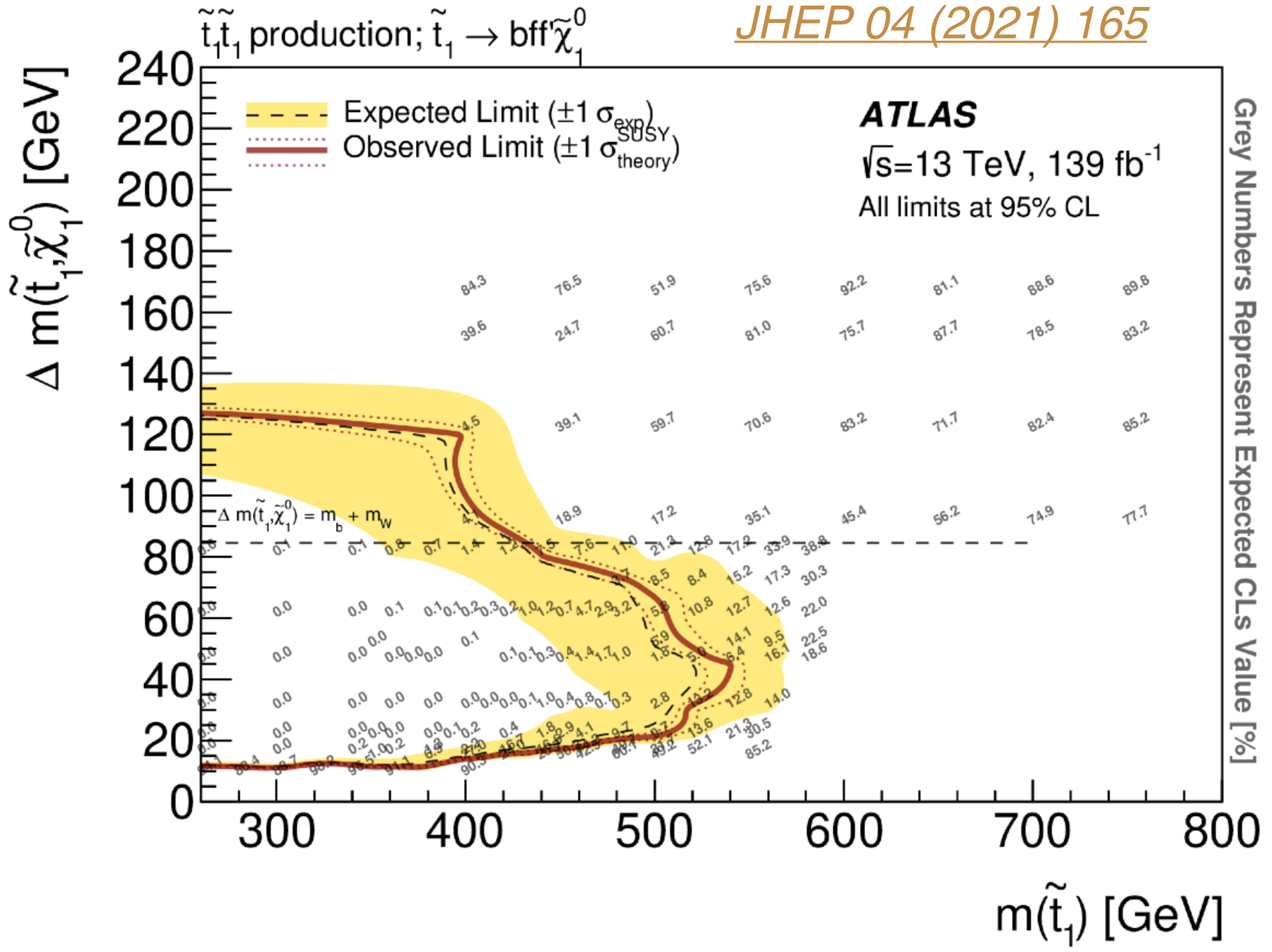
- Three different analysis in the Run 2 paper, already published, according to mass splitting between *stop* ($\tilde{t}_1$) and *neutralino* ($\tilde{\chi}^0_1$), allowing different decay modes:
  - 2 body → $\Delta m > m_t$
  - 3 body → $m_W + m_b < \Delta m < m_t$
  - 4 body, the one picked up → $\Delta m < m_W + m_b$
- Common final state signature: 2 OS leptons (electrons/muons), jets and missing transverse energy
- Cut & Count based approach
- Final, i.e. starting from flat ntuples, event selection done with *ROOT RDataFrame* and 3 helper classes, 100% python based
  - List of cuts, in dictionaries
  - I/O, mainly to define and store output structures/yields
- Main workflow, to extract nominal yields and systematic variations, starting from single *TTree(s)* and/or *TChain(s)*



Soft leptons coming from a virtual W* boson decay

Compressed mass spectra:
$\Delta m < m_W + m_b$

JHEP 04 (2021) 165



Preliminary results: execution time halved

Conclusions & Next Steps

- Three use cases tested, in different scenarios: different experiments and analyses
- Interactive analyses feasibility studies on the INFN Naples infrastructure succeeded
- Towards an INFN national cloud infrastructure with a datalake model to facilitate future analyses (hopefully starting from LHC Run 3)
- Very productive collaboration with other INFN divisions

➡ **Short term goals:**

- Deploy of the code & relative instructions to allow other users to test it when the AF will be released
  - Benchmark studies with local performance evaluation
- Porting finalisation of all the previously mentioned analysis and evaluate the performance obtained using dask on the local cluster, dask on kubernetes or distributed, wrt the original implementation

➡ **Medium-long term goals:**

- Move the analyses to the national AF, using ICSC resources
- Evaluate scalability and simultaneous performance with increasing number of workers

Thank you!

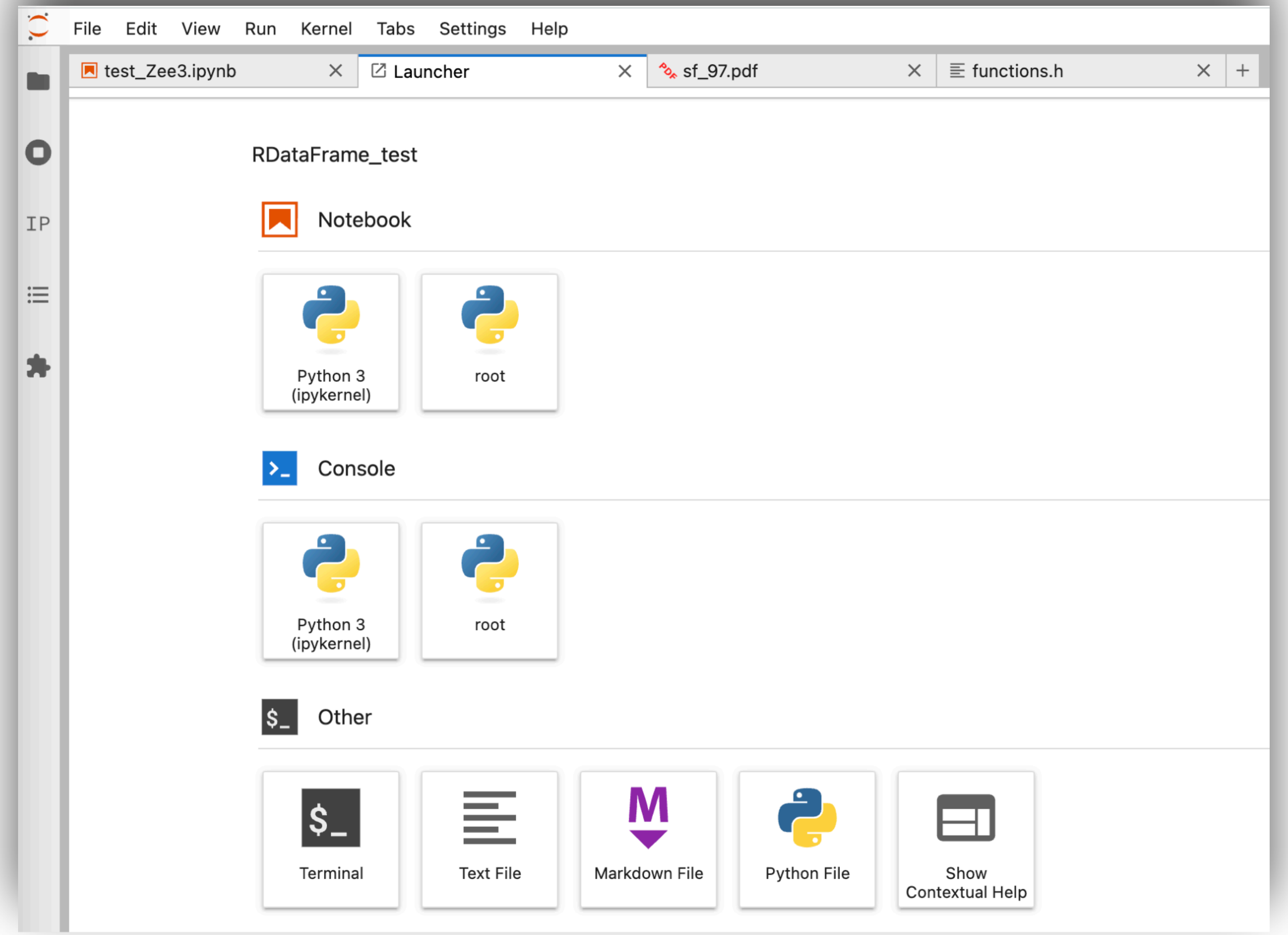
Back-up

Motivations

- Most of the LHC searches/measurements rely on locally developed scripts that process the datasets, with parallel tasks and on an asynchronous batch system
- **Challenges of the future** e+e− colliders are pushing to **re-think the HEP computing models**
  - Impact on several aspects, from software to the computing infrastructure
- From the software perspective, **interactive/quasi interactive analysis** is a promising paradigm
  - User-friendly environment
  - The implementation is simplified by adopting open-source industry standards: *Dask, Jupyter Notebooks* and *HTCondor*
  - Validating new frameworks (e.g. *ROOT RDataFrame* with multi-threading)
- **Preliminary feasibility studies** have been pursued on **FCCee pseudo-data,** exploiting **INFN Napoli** analysis Facilities (**AFs**)
  - Distributed infrastructure which leverages *Dask*

Efficient & user friendly infrastructure

- Python & ROOT (v 6.28) kernels available
- Terminal
- Notebook implementation
  - Completely exportable and replicable



Local *vs* distributed approach


```
from dask.distributed import LocalCluster, Client
if distributed == True:
    RDataFrame = ROOT.RDF.Experimental.Distributed.Dask.RDataFrame
    ROOT.RDF.Experimental.Distributed.initialize(my_initialization_function)
else:
    RDataFrame = ROOT.RDataFrame
    my_initialization_function()
```

Distributed }

Local }

⋮

No changes required to the rest of the code

```
df = df.Define('w_nominal', '1')
df = df.Define("m_e","0.0005124") #GeV
df_ge = df.Define("goodelectrons", "Particle.charge[0]*Particle.charge[1] < 0.").Filter("goodelectrons > 1")
```

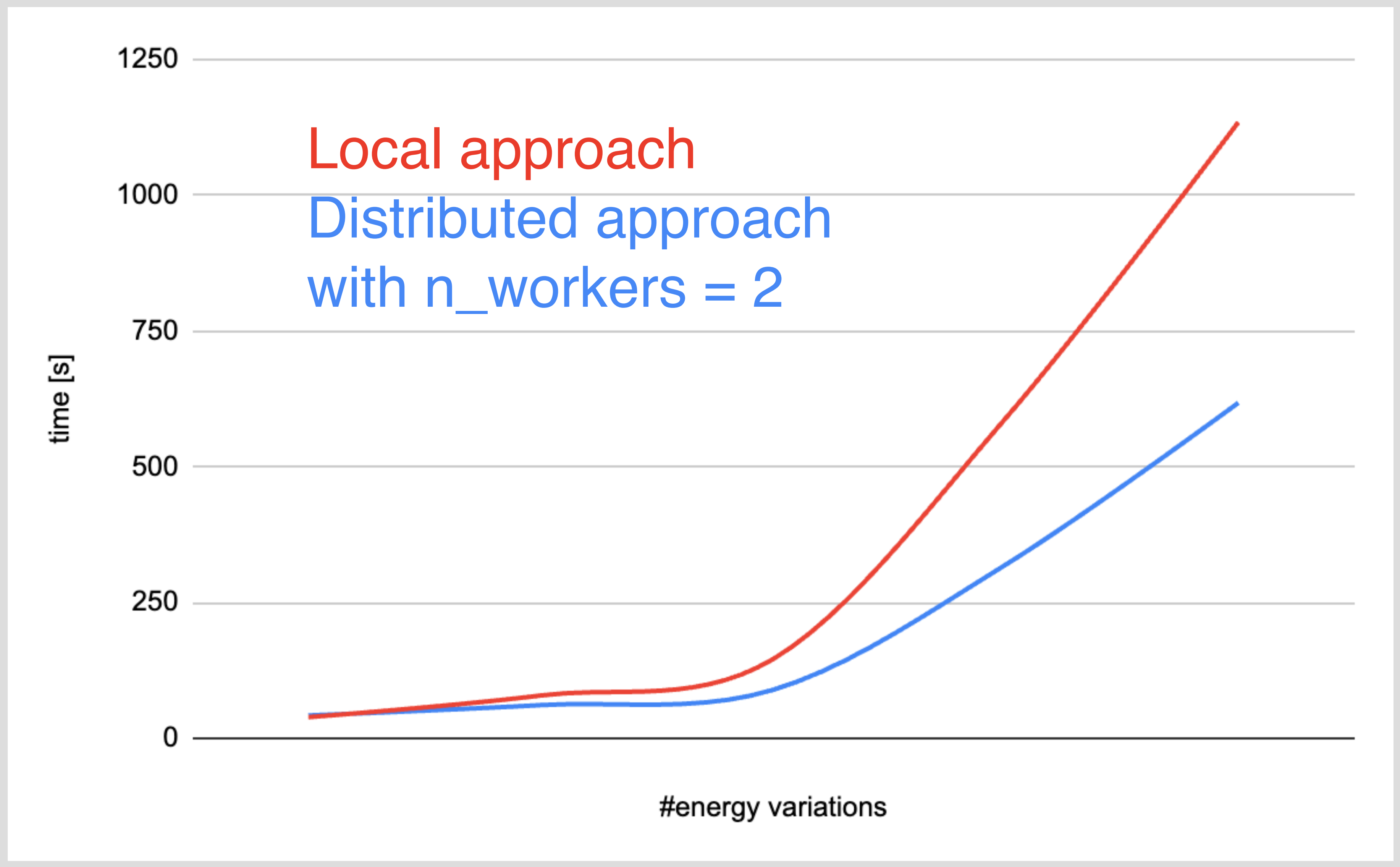
How to compare the performance?

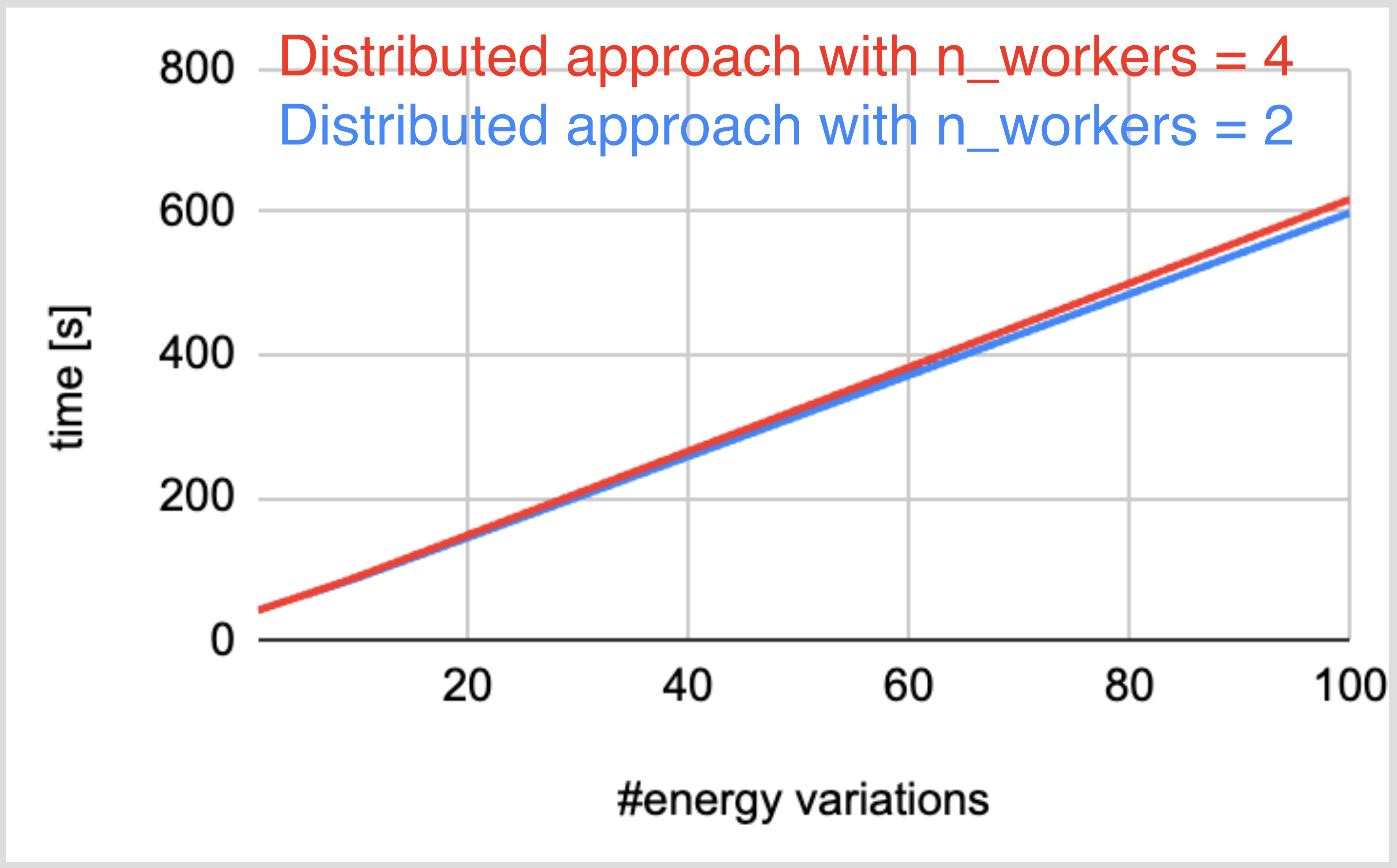
Defined Metric	
Overall execution time	Time elapsed from the start of the execution (execution triggered) to the end of execution

Local *vs* distributed approach

FCCee use-case

```
cluster = LocalCluster(n_workers=2, threads_per_worker=1, processes=True)
```

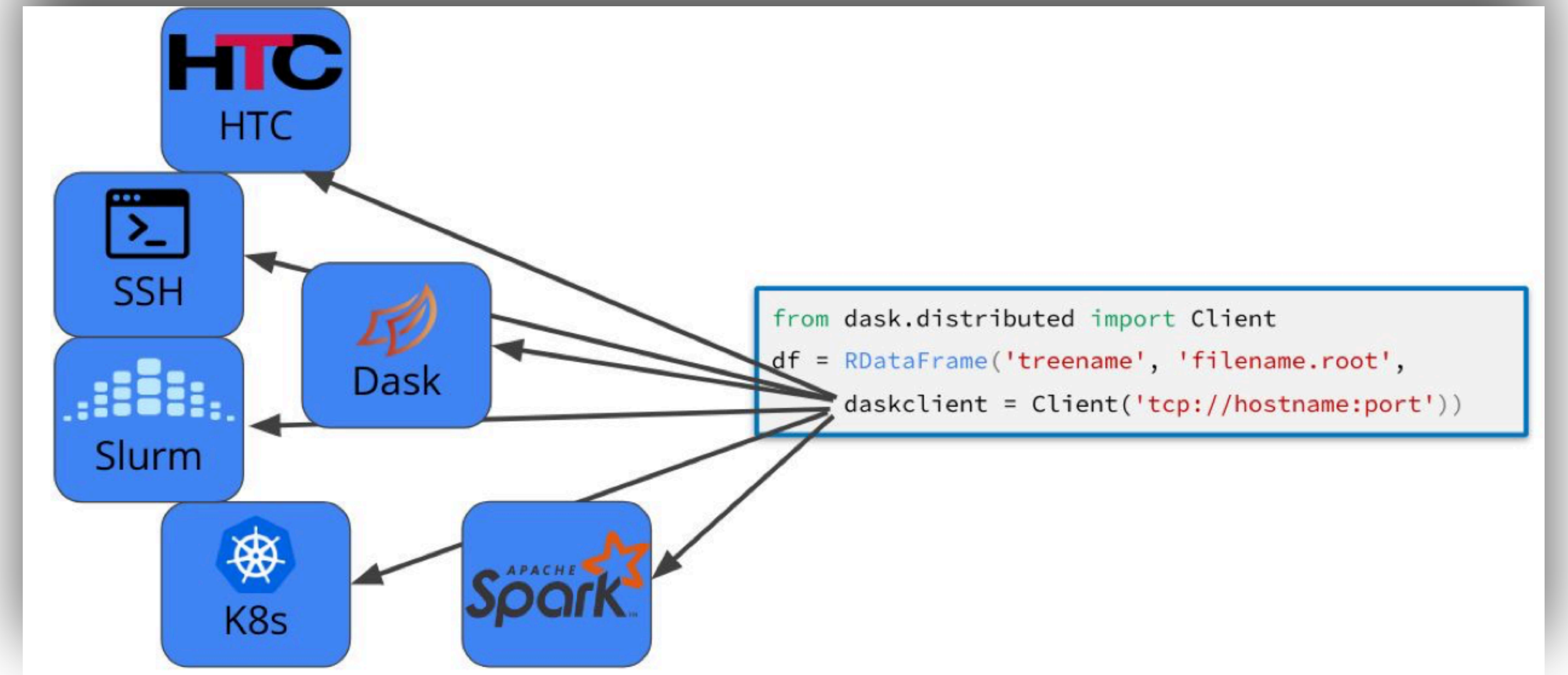




- Exploiting the distributed approach, the execution time halves wrt the local approach if we iterate over a significative number of energy variations (> 10)
- Changing the number of workers from 2 to 4, the execution time is stable

Towards a Dask + HTCondor model

- Based on INFN Perugia *analysis facility*
- Introducing HTCondor queues, the performance improves by a factor 2
- Increasing the number of workers is beneficial when running on many iterations



FCCee use-case

n_workers = 2

# iterations	Dask+HTCondor	Dask
1	22.96 s	42.02 s
50	258.35 s	320 s
100	497.71 s	618 s

Dask + HTCondor

# iterations	n_workers = 2	n_workers = 10
1	22.96 s	20.36 s
50	258.35 s	90.89 s
100	497.71 s	159.26 s