

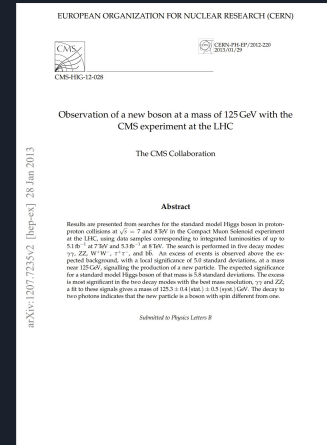
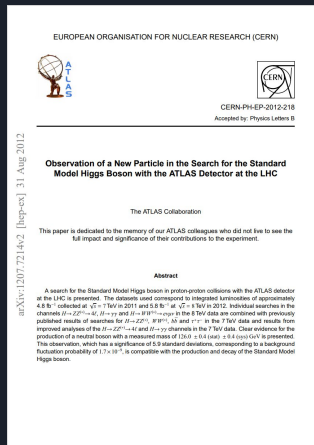


Using Python, coffea, and ServiceX to Rediscover the Higgs. Twice.

Baidyanath Kundu (Manipal Institute of Technology)

Gordon Watts (University of Washington)

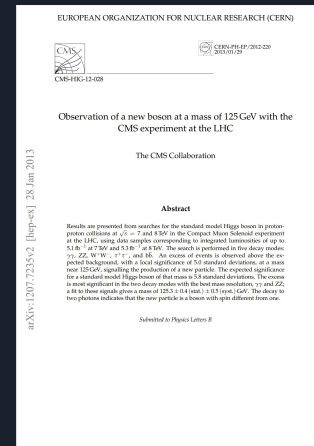
What Higgs, Exactly?



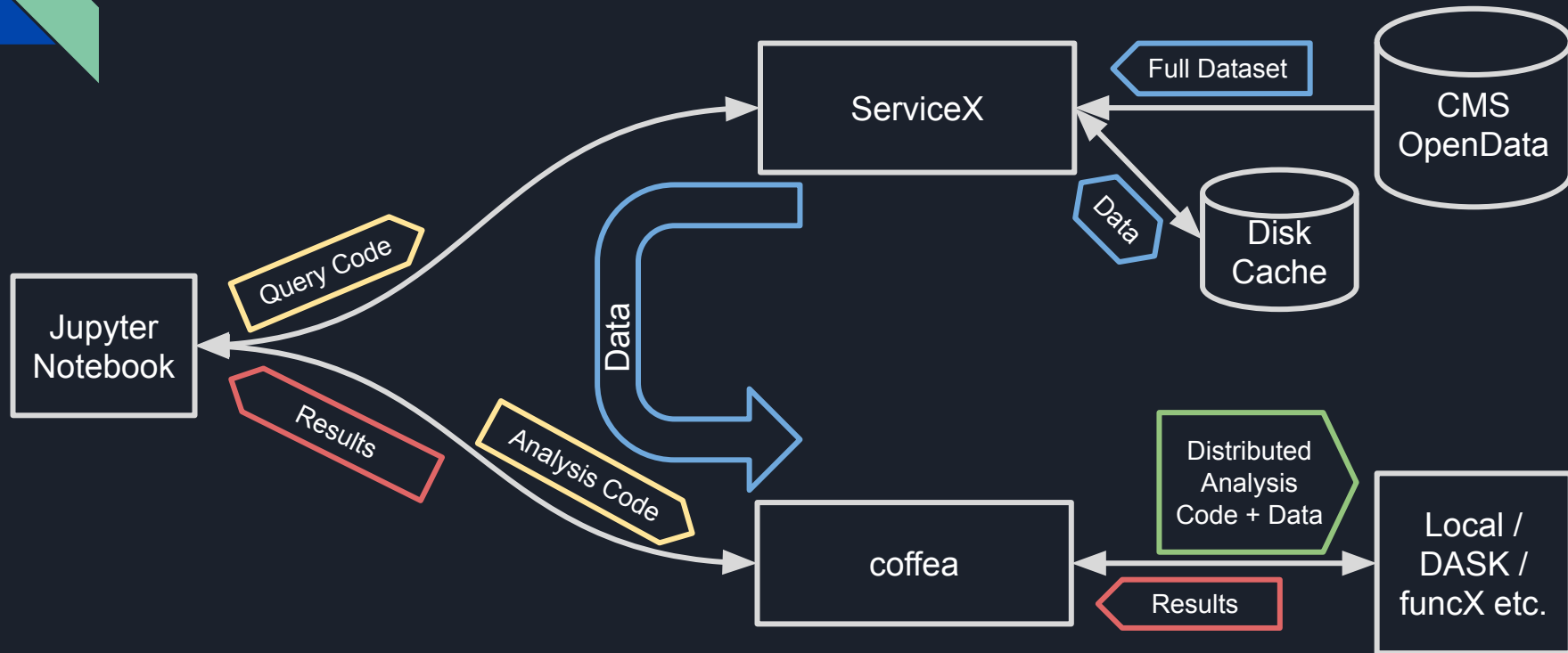
Both experiments have released their Run 1 Higgs Discovery Data on CERN Open Data Including Source Code To Reproduce Analyses!

What Higgs, Exactly?

Can we reproduce their
results using the ServiceX,
func_adl, coffea tool chain?



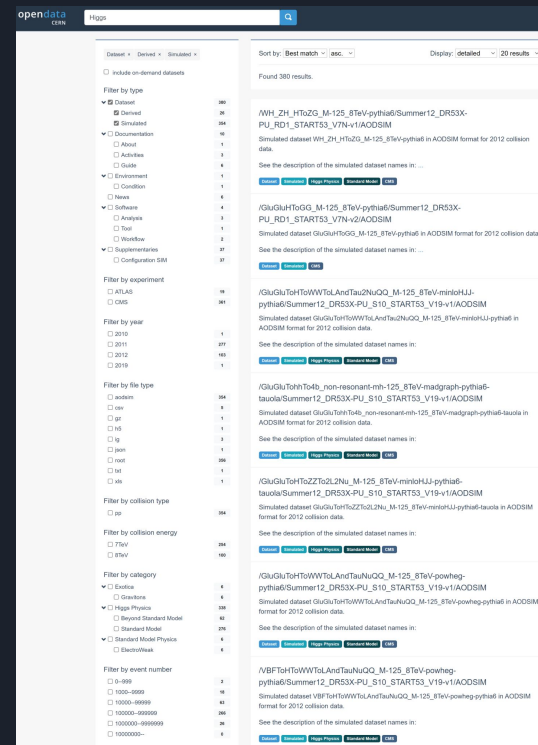
Analysis In Python



CERN opendata

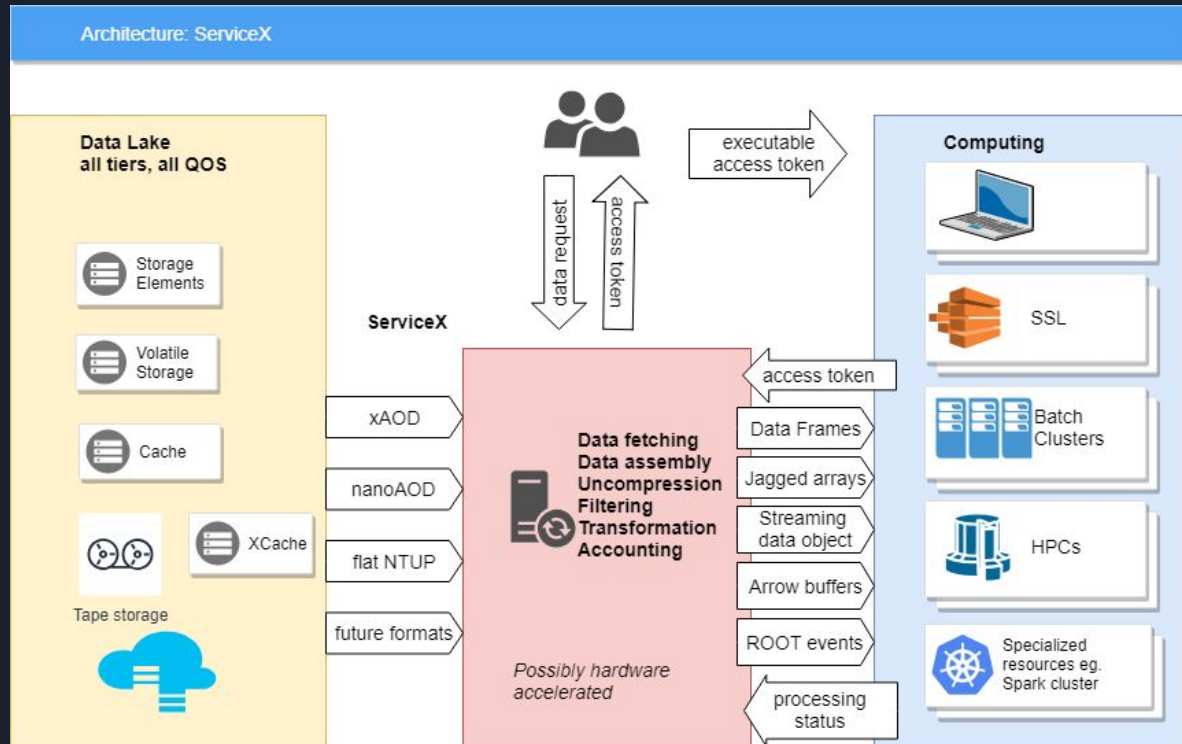


Over 2 PB of data, 1000's of datasets



The screenshot shows the CERN opendata website interface. The top navigation bar includes 'opendata' and 'CERN'. A search bar is present. The main content area displays a list of datasets. The first dataset is '13TeV-pythia6Summer12_DR53X-PU_RD1_STARTS3_V7N-v1AODSIM'. The interface includes filters for 'Filter by type' (Dataset, Simulated), 'Filter by experiment' (ATLAS, CMS), 'Filter by year' (2010, 2011, 2012, 2019), 'Filter by file type' (adex, ev, gl, h5, lg, json, root, nt, nt2), 'Filter by collision type' (pp, ppb), 'Filter by collision energy' (13TeV, 8TeV), 'Filter by category' (Evolve, Higgs Physics, Beyond Standard Model, Standard Model Physics, Electroweak), and 'Filter by event number' (0-999, 1000-9999, 10000-99999, 100000-999999, 1000000-9999999, 10000000-99999999). The dataset list shows details for each entry, including the dataset name, description, and links to download or view the dataset.

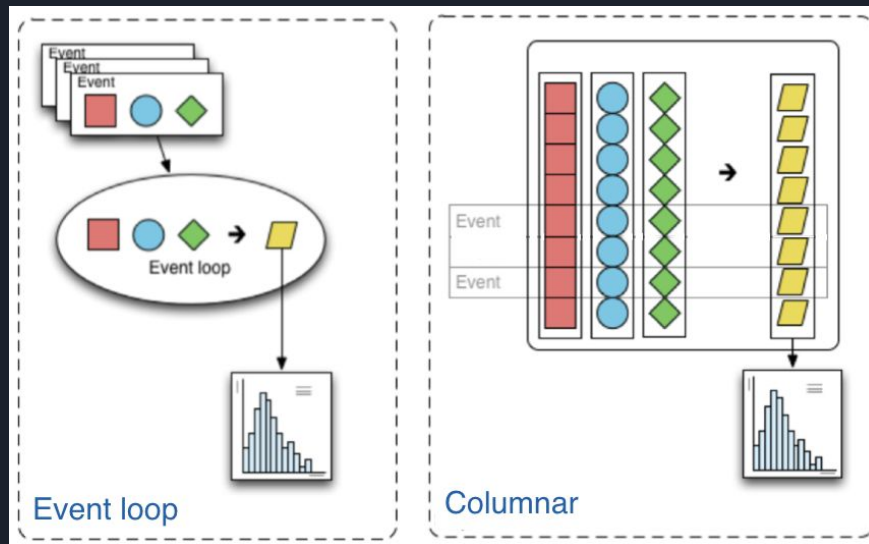
ServiceX: Overview



coffea

A light-weight processing framework for processing event columnar data.

- Processing files
- Distributed processing in a farm (DASK, slurm, Spark, etc.)
- Flexible data model that makes column-wise data look like row-wise.



In the end, many parts of coffea should disappear into dedicated packages... until then, missing gaps in functionality have a home here.

[documentation](#)

Notebooks

We have 3 notebooks to work through!

- Introduction to func_adl, ServiceX, and coffea
- The ATLAS Higgs Discovery Dataset
- The CMS Higgs Discovery Dataset

Introduction

Lets load our environment first

```
[1]: from func_adl_serviceX import ServiceXSourceUpROOT, ServiceXSourceCMSRun1A00
from hist import Hist
import awkward as ak
```

Flat ROOT Files

ATLAS has distributed it's open data as flat ROOT files.

- On CERNOpenData they are a single zip file
- But they have been distributed as files available via EOS from CERN Open Data's EOS instance.

```
[2]: ggH125_ZZ4lep = 'root://eospublic.cern.ch/eos/opendata/atlas/OurestartreachDatasets/2020-01-22/4lep/MC/mc_345060.ggH125_ZZ4lep.4lep.root'
ggH125_ZZ4lep = 'https://atlas-opendata.web.cern.ch/atlas-opendata/samples/2020/4lep/MC/mc_345060.ggH125_ZZ4lep.4lep.root'
ggH125_ZZ4lep_source = ServiceXSourceUpROOT([ggH125_ZZ4lep], 'mini', backend='open_uproot')
```

- We use the `root://` address instead of `http://` due to efficiency and caching.
- `mini` is the tree name in the file
- `backend` basically describes the type of file - this is a flat root file that can be opened by the `uproot` python package.

Now that we have a reference to the datasource, lets pick out a single column and bring its contents back to our local instance:

```
[3]: r = (ggH125_ZZ4lep_source
        .Select(lambda e: {'lep_pt': e['lep_pt']}))
        .AsAwkwardArray()
        .value()
        )

[3]: <Array [{"lep_pt": [5.19e+04, ... 1.11e+04]}] type='164716 * {"lep_pt": var * floa...>
```

[Tutorial GitHub Repo](#)

Team Effort

- Data and MC: ATLAS & CMS Collaborations for all the data, and making it public!
- ServiceX: Ben Galewsky and Andrew Eckart and Suchandra Thapa and everyone else on the team
- CMS Run 1 AOD Transformer: Baidyanath Kundu
- Uproot/Flat ROOT backend: Mason Proffit
- IRIS-HEP for supporting a large fraction of these people in one way or another.
- CMS Awkward Code: Brian Cruz

Running This Yourself...

Tutorial GitHub Repo

- ➡ Instructions are on the repo
- ➡ We are happy to give out the ServiceX end-points (unofficial atm)
- ➡ Some integration with coffea is in beta
- ➡ Not yet obvious that CERN Open Data portal can handle the data load!
- ➡ Binder is not supported

Using

You can find the final notebooks used in the talk in the `talks` directory. The `notebooks` directory contains practice notebooks used to develop concepts for the talk. They are not necessarily well documented.

ServiceX for the demo

☰ README.md

```
api_endpoints:  
- endpoint: http://xxx.org  
  type: open_uproot  
- endpoint: http://yyy.org  
  type: cms_run1_aod
```

```
backend_types:  
- type: open_uproot  
  return_data: parquet  
- type: cms_run1_aod  
  return_data: root
```

Please get in touch with us to get the address of the open instances running `ServiceX`.

Setting up the environment

Setup your environment:

1. This has been run under python 3.9.6. It should work with anything that is 3.7 or greater.
2. Check out this repository locally, and check out the `coffea` patched repository locally.
3. For the `coffea` repository, check out the branch `pr_servicex_flat_root_files`. For this package use the head.
4. `python -m venv .venv`, and activate the new environment.

5. `pip install -r requirements.txt`

6. In the root directory of the checked out `coffea` package, run `pip install -e[servicex]`.

From there you can start `jupyter-lab`.

Running on binder

It is not currently possible to run on `binder` as `ServiceX` uses a non-standard port to download data.