Making your analysis more efficient with ROOT

A basic introductory course

ROOT
Data Analysis Framework

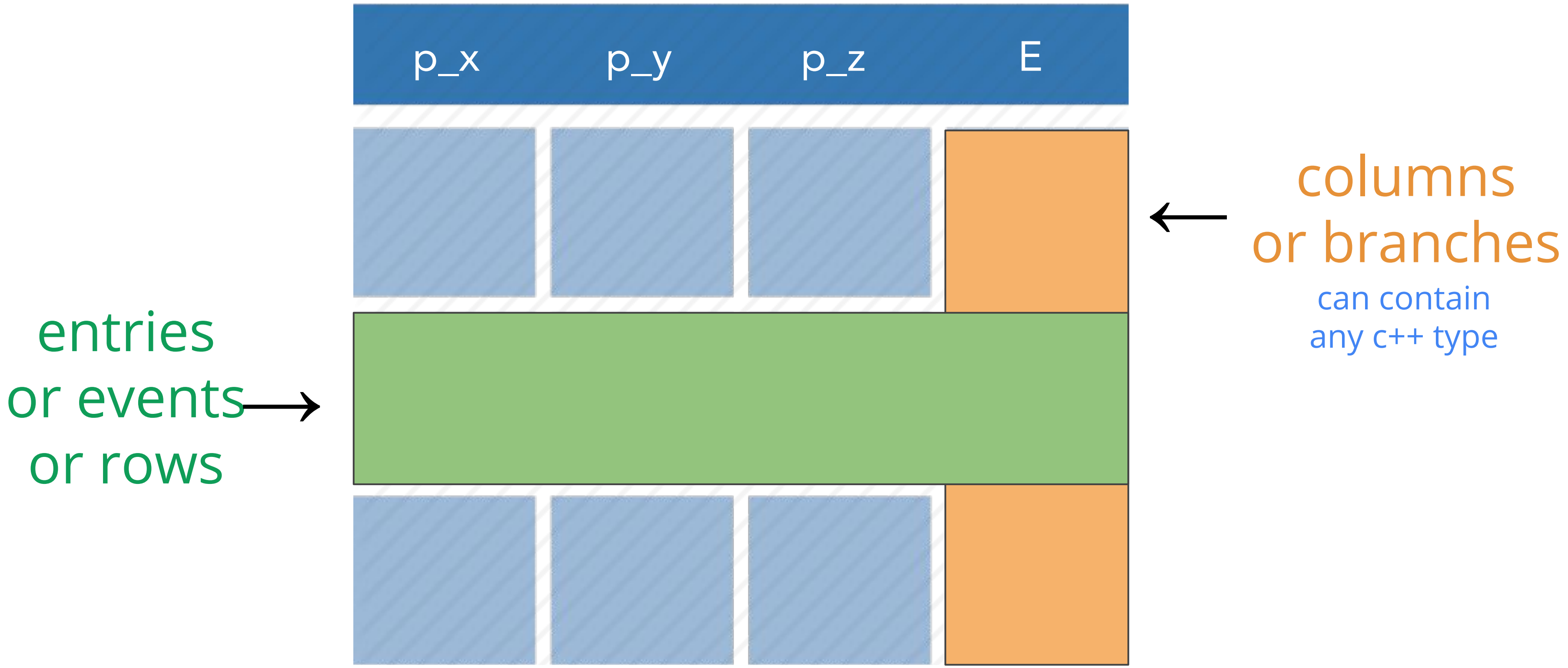
https://root.cern

This Tutorial

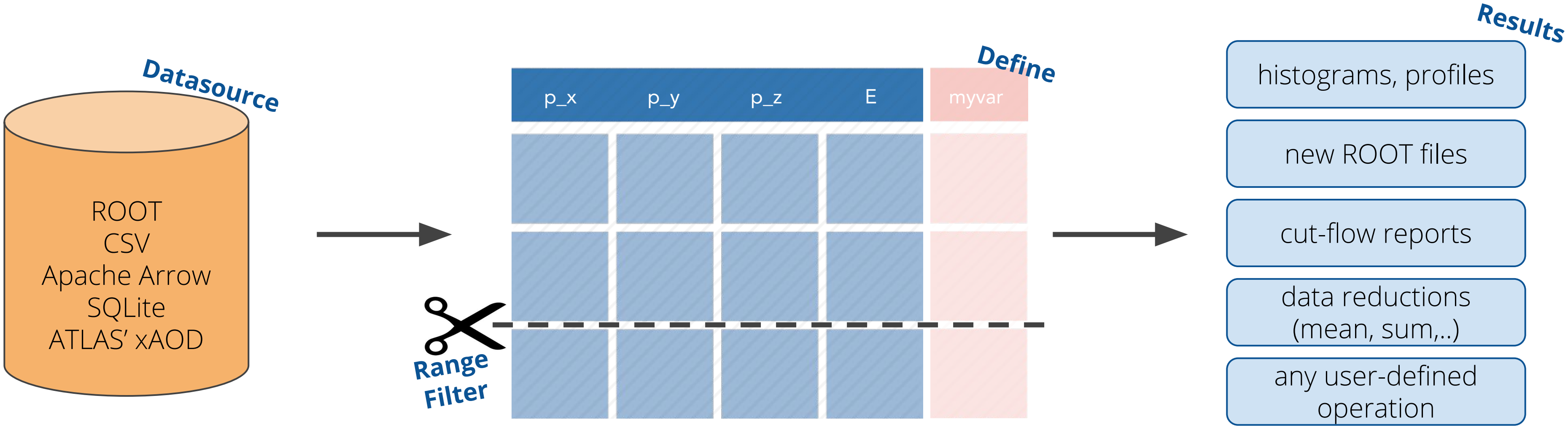
- **ROOT functionalities to get you to your results faster**
  - A very limited selection, a sort of starter kit for analysis
- **Focus on the treatment of datasets with ROOT dataframes**

Start with a brief presentation, then dive into a hands-on session
(based on SWAN - use ROOT on the web)

Columnar representation of data



ROOT dataframes in a nutshell



An ergonomic, fast C++ dataframe

```
ROOT::RDataFrame df("tree", "file.root");
```

on this (ROOT, CSV, ...) dataset

An ergonomic, fast C++ dataframe

`ROOT::RDataFrame df("tree", "file.root");` on this (ROOT, CSV, ...) dataset

`auto df2 = df.Filter("pt > 0")` only accept events for which `pt > 0`

An ergonomic, fast C++ dataframe

```
ROOT::RDataFrame df("tree", "file.root");          on this (ROOT, CSV, ...) dataset
auto df2 = df.Filter("pt > 0")                       only accept events for which pt > 0
             .Define("r", "sqrt(eta*eta + phi*phi)"); define r = sqrt(eta² + phi²)
```

An ergonomic, fast C++ dataframe

```
ROOT::RDataFrame df("tree", "file.root");
auto df2 = df.Filter("pt > 0")
             .Define("r", "sqrt(eta*eta + phi*phi)");
auto rHist = df2.Histo1D("r");
```

- `ROOT::RDataFrame df("tree", "file.root");` — on this (ROOT, CSV, ...) dataset
- `auto df2 = df.Filter("pt > 0")` — only accept events for which pt > 0
- `.Define("r", "sqrt(eta*eta + phi*phi)");` — define r = sqrt(eta² + phi²)
- `auto rHist = df2.Histo1D("r");` — plot r for events that pass the cut

An ergonomic, fast C++ dataframe

```
ROOT::EnableImplicitMT();                                   Run a parallel analysis
ROOT::RDataFrame df(“tree”, “file.root”);                    on this (ROOT, CSV, ...) dataset
auto df2 = df.Filter("pt > 0")                              only accept events for which pt > 0
             .Define("r", "sqrt(eta*eta + phi*phi)");       define r = sqrt(eta² + phi²)
auto rHist = df2.Histo1D("r");                              plot r for events that pass the cut
```

An ergonomic, fast C++ dataframe

```
ROOT::EnableImplicitMT();                       ······ Run a parallel analysis
ROOT::RDataFrame df("tree", "file.root");       ······ on this (ROOT, CSV, ...) dataset
auto df2 = df.Filter("pt > 0")                  ······ only accept events for which pt > 0
             .Define("r", "sqrt(eta*eta + phi*phi)"); ······ define r = sqrt(eta² + phi²)
auto rHist = df2.Histo1D("r");                  ······ plot r for events that pass the cut
```

✅ no boilerplate
✅ common tasks are already implemented
✅ implicit parallelisation

No templates: C++ → JIT → Python

C++

```
d.Filter([](double t) { return t > 0.; }, {"theta"})
  .Snapshot<vector<float>>("tree","file.root",{"pt_x"});
```

C++ with cling's just-in-time compilation

```
d.Filter("theta > 0").Snapshot("tree","file.root","pt_x");
```

PyROOT, automatically generated Python bindings

```
d.Filter("theta > 0").Snapshot("tree","file.root","pt_x")
```

RDataFrame: Design goals

simple and powerful interface

provide **high level features**, e.g.
less typing, better expressivity, abstraction of complex operations

allow **transparent optimisations**, e.g.
multi-thread parallelisation, lazy evaluation and caching

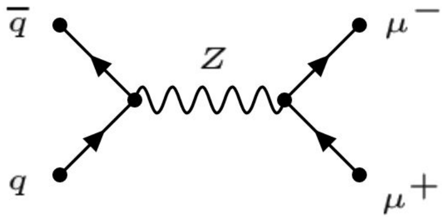
RDF docs **RDF tutorials**

Questions? Talk and work with us!

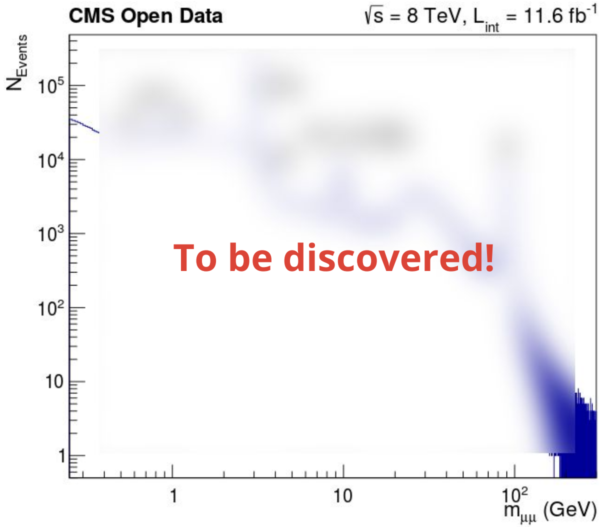
- **Mattermost: https://mattermost.web.cern.ch/root**
- Have a question about ROOT? https://root-forum.cern.ch
- Have an idea about evolving ROOT? https://root-forum.cern.ch/c/my-root-app-and-ideas
- Have a bug to report? https://root.cern/guidelines-submitting-bug
- Have some code ready to go in the next ROOT release? https://github.com/root-project/root/pulls
  - *Github pull requests are always welcome*: simple (and not so simple) bug fixes, typos, missing documentation, tutorials...

Introduction to the hands on

- Analysis of the di-muon spectrum using data from the CMS detector taken in 2012
- Rediscover particle resonances in a wide energy range from the η meson (548 MeV) up to the Z boson (91 GeV)



Physics interaction being subject of the analysis



Variable	Type	Description
nMuon	*unsigned int*	Number of muons in this event
Muon_pt	*float[nMuon]*	Transverse momentum of the muons (stored as an array of size nMuon)
Muon_eta	*float[nMuon]*	Pseudorapidity of the muons
Muon_phi	*float[nMuon]*	Azimuth of the muons
Muon_mass	*float[nMuon]*	Mass of the muons
Muon_charge	*int[nMuon]*	Charge of the muons (either 1 or -1)

Data format of the dataset

Important preliminary step

If you don't have a **CERNBox**, the CERN "DropBox-like" service, connect now to

cernbox.cern.ch

This is needed to carry out the hands-on on SWAN

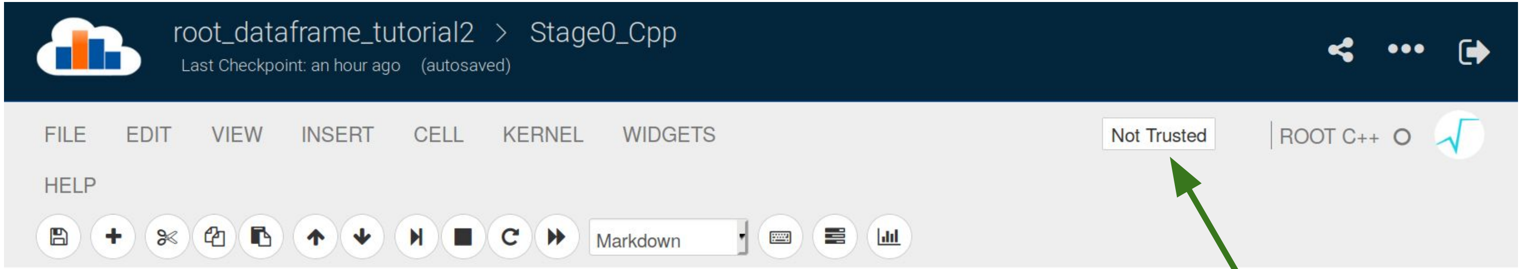
Hands on time!

Go here if you have a CERN account:

Open in SWAN

Go here otherwise:

launch binder



click this button to
execute a cell

click this button to
restart the notebook

click this button to
“trust” the notebook