SWAN usage at COMPASS

SWAN workshop 2019

Brian Ventura

SWAN usage at COMPASS *Summary*

1. Study of the DVCS process at COMPASS
2. The current analysis softwares
3. Post analysis using SWAN

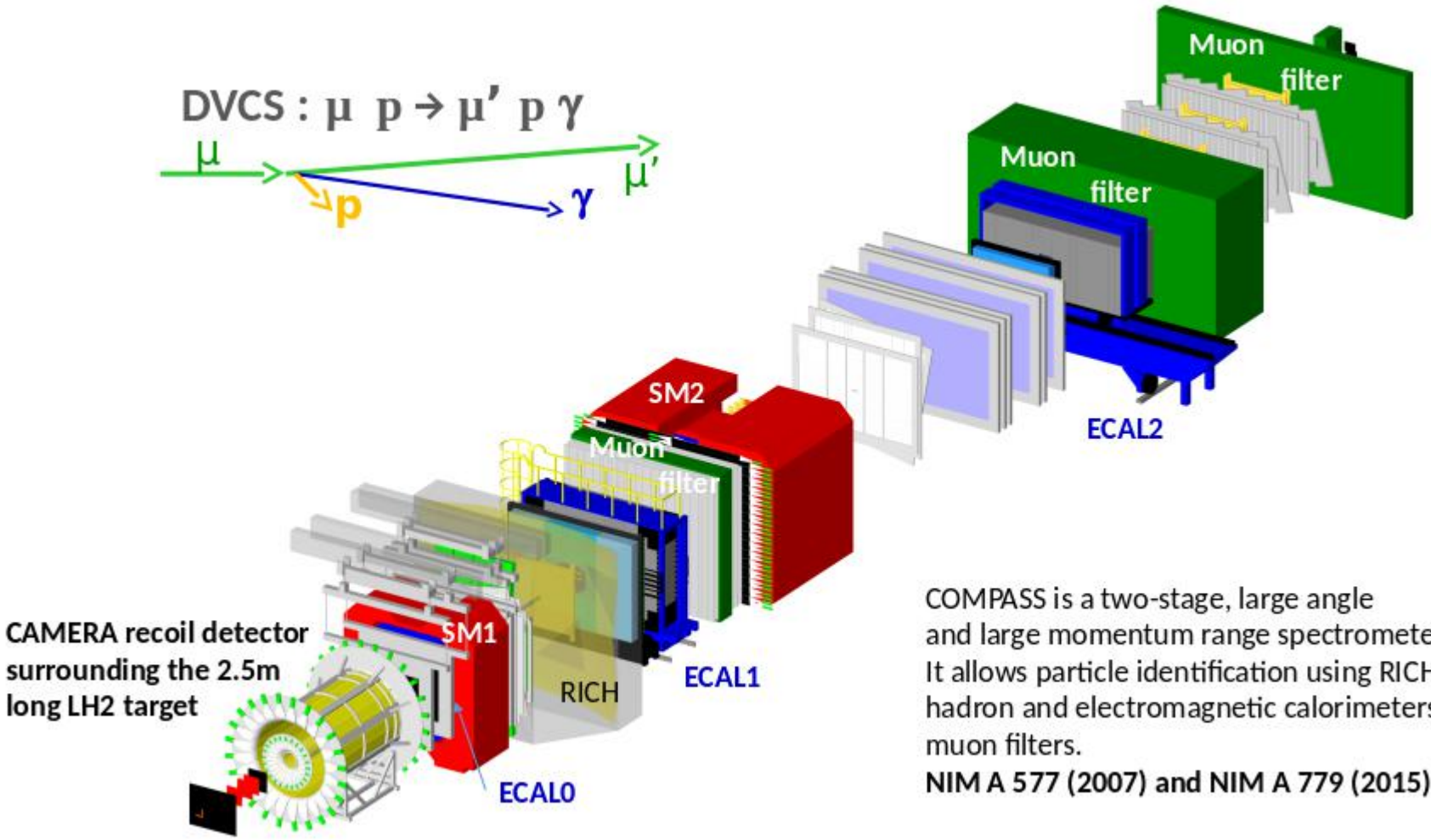
The COMPASS experiment & the DVCS process

μp→μpγ

SWAN usage at COMPASS

Study of the DVCS process at COMPASS

- Exclusive process with small cross section compared with large background
- Detection of the complete initial and final state.
- Remove background using the overdetermined kinematics



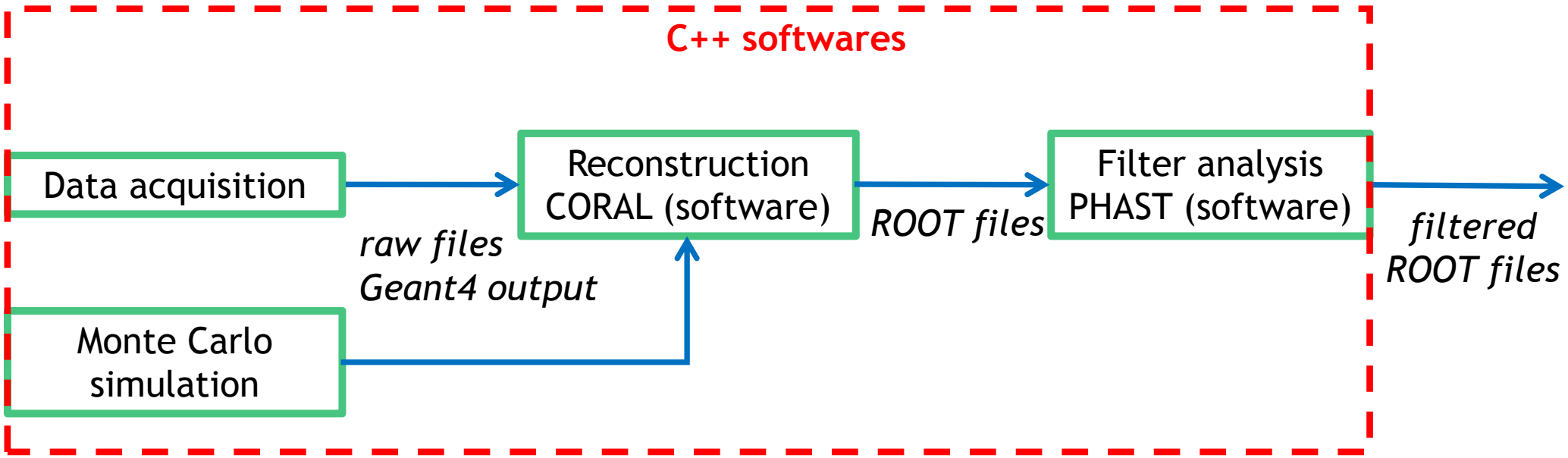
COMPASS is a two-stage, large angle and large momentum range spectrometer. It allows particle identification using RICH, hadron and electromagnetic calorimeters and muon filters.

NIM A 577 (2007) and NIM A 779 (2015) 69

The current chain of analysis softwares

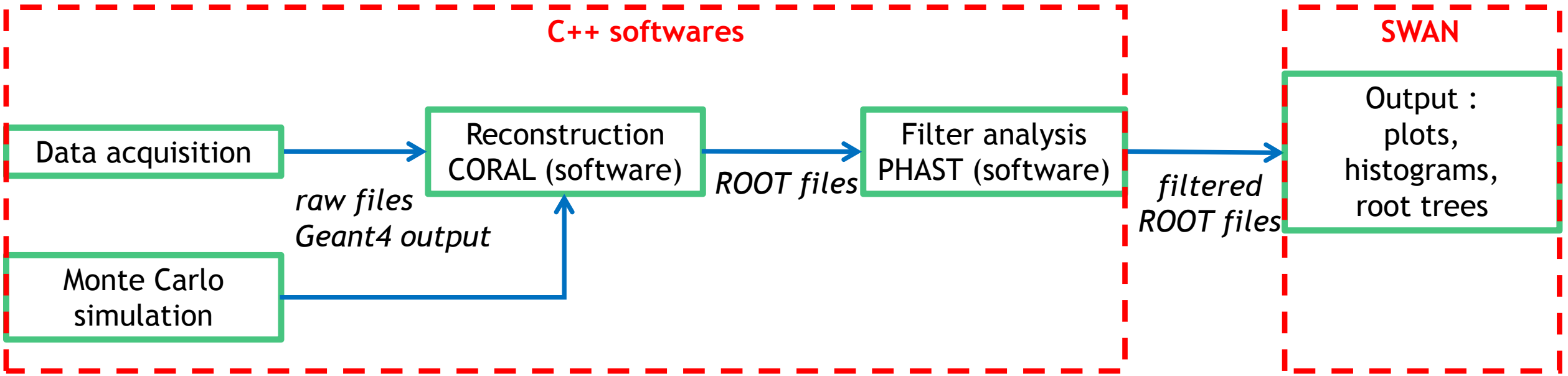
SWAN usage at COMPASS

Current chain of analysis softwares



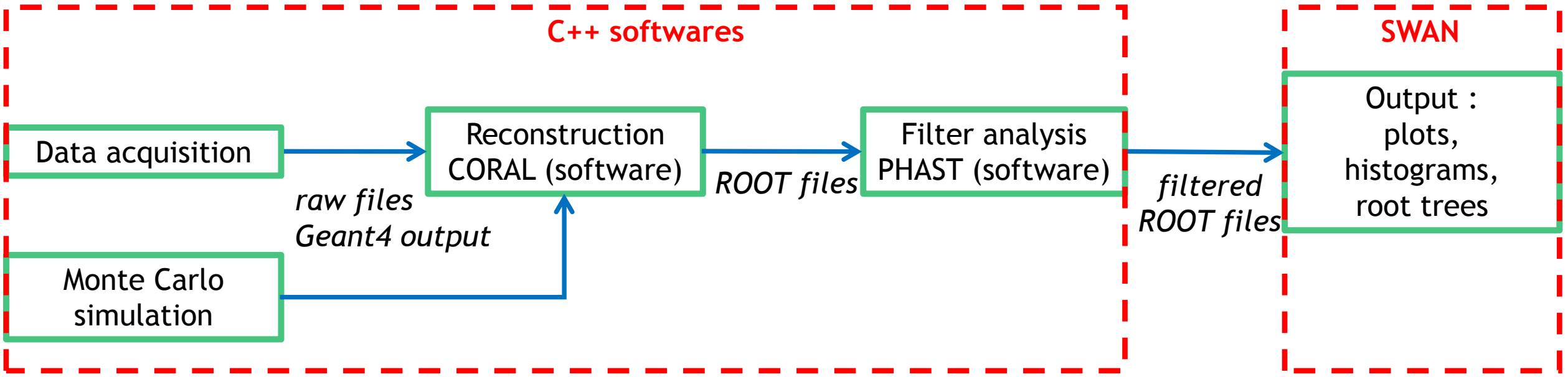
SWAN usage at COMPASS

Current chain of analysis softwares



SWAN usage at COMPASS

Current chain of analysis softwares



At the end of the C++ part : DVCS events candidates + background using first well determined cuts on target, triggers, radiation lengths, etc.

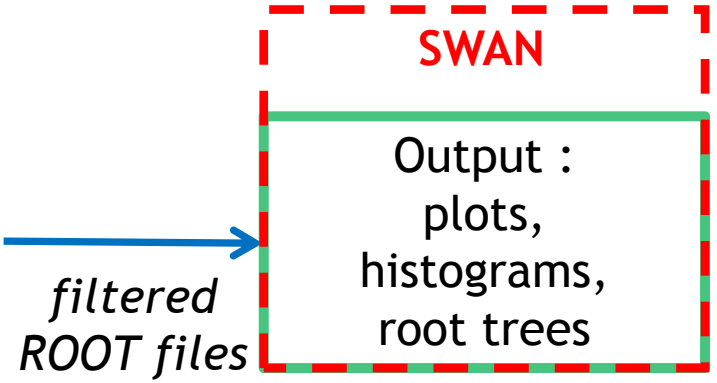
$$\text{DVCS} : \mu\ p \rightarrow \mu'\ p\ \gamma$$

Second cuts performed to get the DVCS exact topology (exclusivity) and remove the background

Post analysis using SWAN

SWAN usage at COMPASS

Study of the DVCS process at COMPASS



For all the events candidates, extract the DVCS exact topology

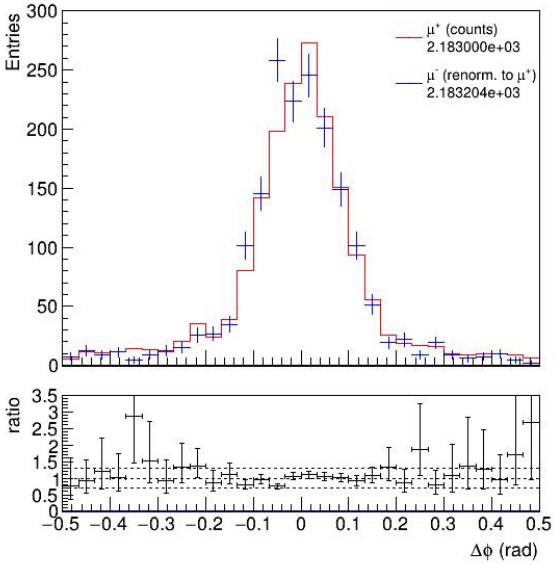
$$DVCS : \mu\ p \rightarrow \mu'\ p\ \gamma$$

+ plot many other physics observables

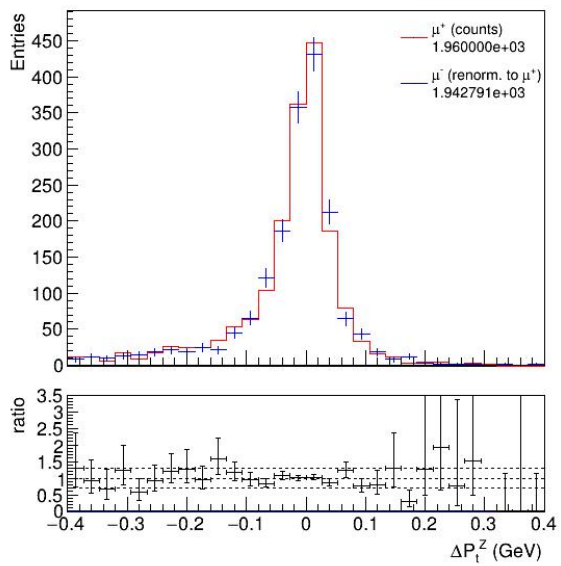
Pros & cons of SWAN

- Well organised analysis in notebooks using sections (use of markdown, latex, etc)
- Flexible enough to make systematic studies: use SWAN to adjust some cuts
- Drawback of interactivity: can be slow with too many events

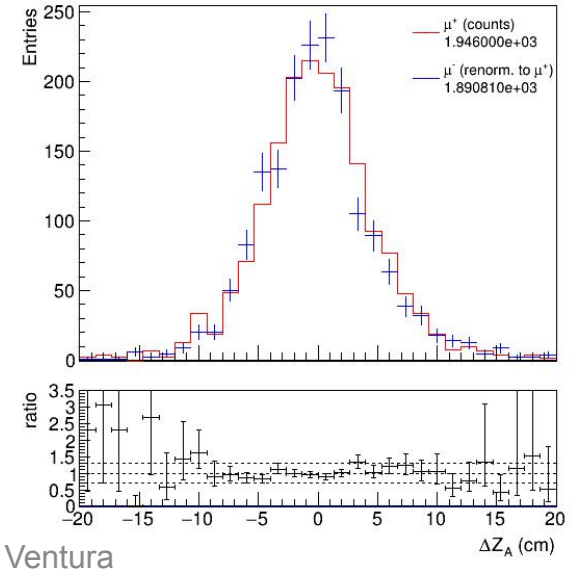
$\Delta\varphi = \varphi^{cam} - \varphi^{spec}$



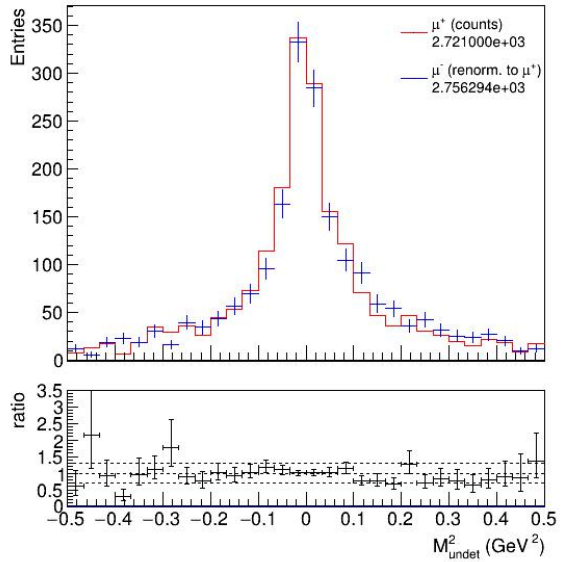
$\Delta p_T = p_T^{cam} - p_T^{spec}$



$\Delta z_A = z_A^{cam} - z_A^{Z_B}$ and vertex



$M^2_{Undet} = (k+p-k'-q'-p')^2$



Pros & cons of SWAN

- Drawback of interactivity: can be slow with too many events

SWAN usage at COMPASS Study of the DVCS process at COMPASS

Pros & cons of SWAN

- Drawback of interactivity: can be slow with too many events → solution: using spark clusters !

But How to use ROOT & spark clusters in SWAN ?

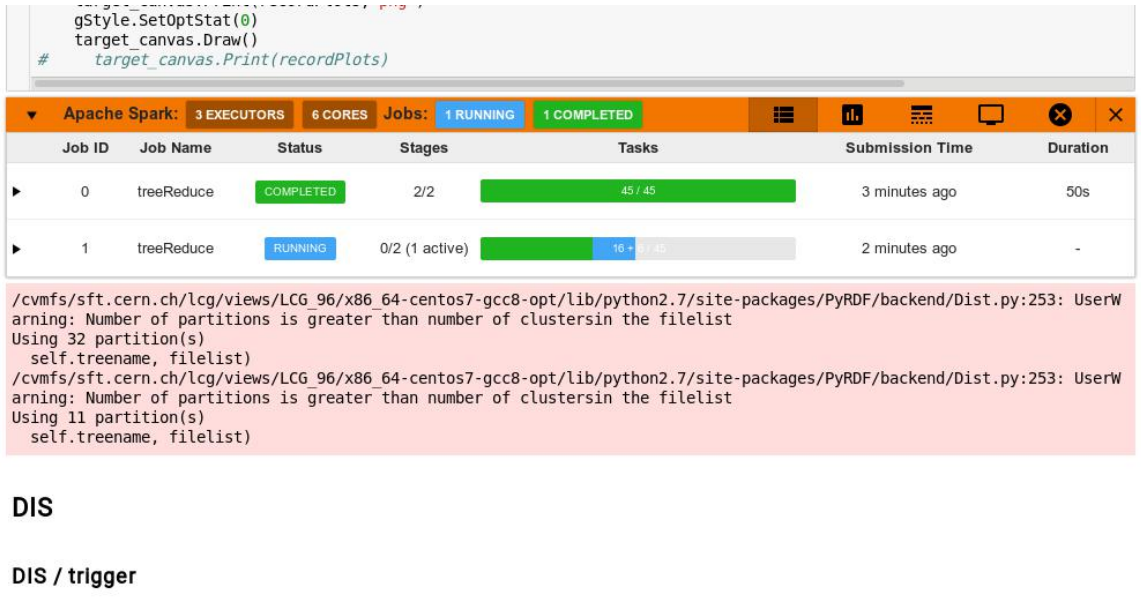
SWAN usage at COMPASS Study of the DVCS process at COMPASS

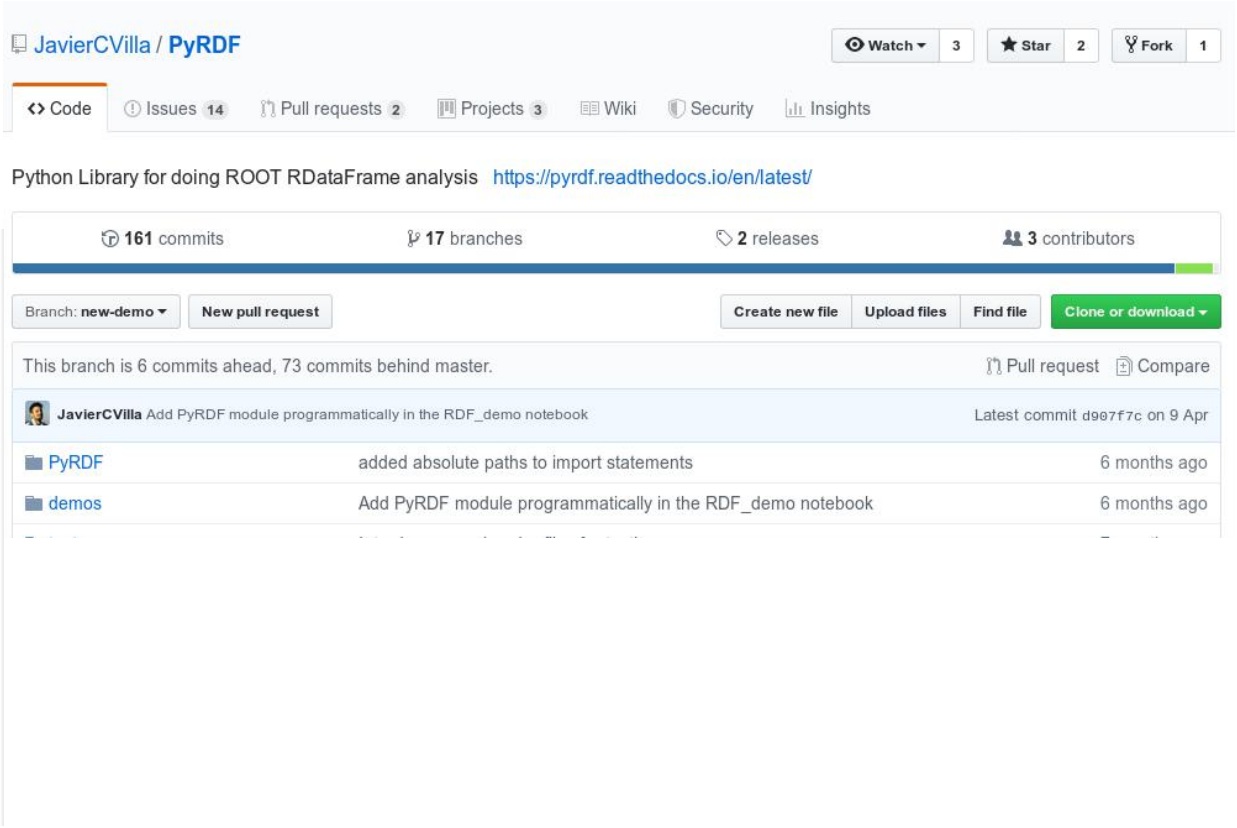
Pros & cons of SWAN

- Drawback of interactivity: can be slow with too many events → solution: using spark clusters !

But How to use ROOT & spark clusters in SWAN ?

https://github.com/JavierCVilla/PyRDF/tree/new-demo





SWAN usage at COMPASS Study of the DVCS process at COMPASS

Pros & cons of SWAN

- Drawback of interactivity: can be slow with too many events → solution: using spark clusters !

But How to use ROOT & spark clusters in SWAN ?

https://github.com/JavierCVilla/PyRDF/tree/new-demo

```
    gStyle.SetOptStat(0)
    target_canvas.Draw()
#     target_canvas.Print(recordPlots)
```

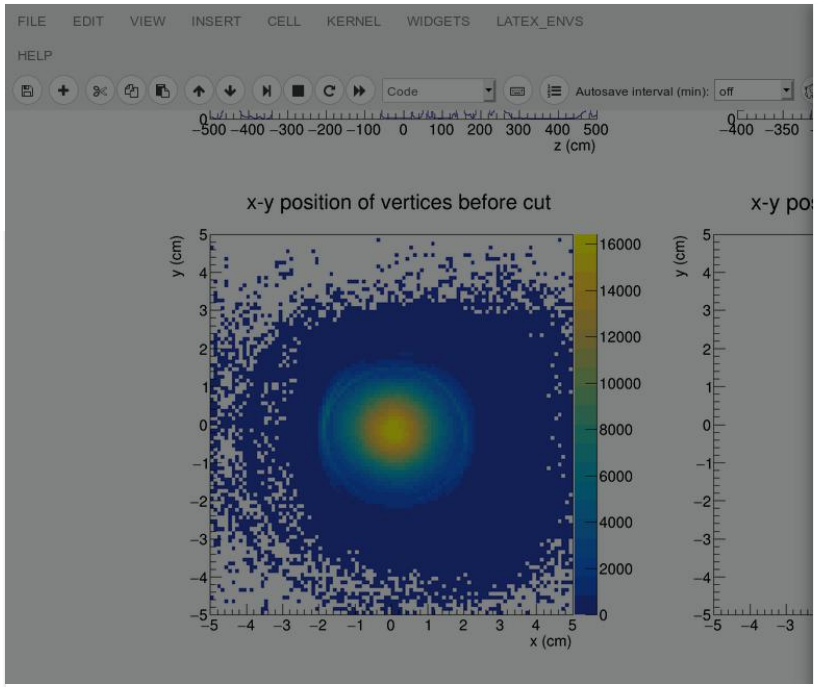
Apache Spark: 3 EXECUTORS 6 CORES Jobs: 1 RUNNING 1 COMPLETED

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
0	treeReduce	COMPLETED	2/2	45 / 45	3 minutes ago	50s
1	treeReduce	RUNNING	0/2 (1 active)	16 + 7 / 45	2 minutes ago	-

```
/cvmfs/sft.cern.ch/lcg/views/LCG_96/x86_64-centos7-gcc8-opt/lib/python2.7/site-packages/PyRDF/backend/Dist.py:253: UserW
arning: Number of partitions is greater than number of clustersin the filelist
Using 32 partition(s)
  self.treename, filelist)
/cvmfs/sft.cern.ch/lcg/views/LCG_96/x86_64-centos7-gcc8-opt/lib/python2.7/site-packages/PyRDF/backend/Dist.py:253: UserW
arning: Number of partitions is greater than number of clustersin the filelist
Using 11 partition(s)
  self.treename, filelist)
```

DIS

DIS / trigger



Spark clusters connection

You are connected to **k8s**

The following variables were instantiated

- sc = SparkContext
- spark = SparkSession

Connection details

- Spark version is 2.4.3
- Spark History Server is available here
- Spark Metrics are available here
- Spark driver logs of the running application show/hide

Go to the notebook | Restart Spark session

SWAN usage at COMPASS Study of the DVCS process at COMPASS

The **PyRDF** module provides a pythonic wrapper of the existing ROOT **RDataFrame** class

```
import PyRDF

# Initialize RDataFrame object
df = PyRDF.RDataFrame(dataset)

# Define operations
df2 = df.Filter("x > 0")
        .Define("r2", "x*x + y*y")
rHist = df2.Histo1D("r2")

# Display histogram
rHist.Draw()
```

J. Cervantes and E. Tejedor, "ROOT RDataFrame and Spark", HSF Software Forum

SWAN usage at COMPASS

Study of the DVCS process at COMPASS

The **PyRDF** module provides a pythonic wrapper of the existing ROOT **RDataFrame** class

```
import PyRDF

# Initialize RDataFrame object
df = PyRDF.RDataFrame(dataset)

# Define operations
df2 = df.Filter("x > 0")
        .Define("r2", "x*x + y*y")
rHist = df2.Histo1D("r2")

# Display histogram
rHist.Draw()
```

J. Cervantes and E. Tejedor, "ROOT RDataFrame and Spark", HSF Software Forum

```
ROOT::EnableImplicitMT();                    Run a parallel analysis
ROOT::RDataFrame df(dataset);                on this (ROOT, CSV, ...) dataset
auto df2 = df.Filter("x > 0")                only accept events for which x > 0
             .Define("r2", "x*x + y*y");     define r2 = x² + y²
auto rHist = df2.Histo1D("r2");              plot r2 for events that pass the cut
df2.Snapshot("newtree", "out.root");         write the skimmed data and r2
                                             to a new ROOT file
```

Lazy execution guarantees that all operations are performed in **one event loop**

E. Guiraud, "RDataFrame", CHEP 2018

SWAN usage at COMPASS

Study of the DVCS process at COMPASS

The **PyRDF** module provides a pythonic wrapper of the existing ROOT **RDataFrame** class

```
import PyRDF

# Initialize RDataFrame object
df = PyRDF.RDataFrame(dataset)

# Define operations
df2 = df.Filter("x > 0")
        .Define("r2", "x*x + y*y")
rHist = df2.Histo1D("r2")

# Display histogram
rHist.Draw()
```

J. Cervantes and E. Tejedor, "ROOT RDataFrame and Spark", HSF Software Forum

```
ROOT::EnableImplicitMT();  ..... Run a parallel analysis
ROOT::RDataFrame df(dataset);  ..... on this (ROOT, CSV, ...) dataset
auto df2 = df.Filter("x > 0")  ..... only accept events for which x > 0
             .Define("r2", "x*x + y*y");  ..... define r2 = x² + y²
auto rHist = df2.Histo1D("r2");  ..... plot r2 for events that pass the cut
df2.Snapshot("newtree", "out.root");  ..... write the skimmed data and r2
                                           to a new ROOT file
```

Lazy execution guarantees that all operations are performed in **one event loop**

E. Guiraud, "RDataFrame", CHEP 2018

Thanks a lot, for RDataFrames and PyRDF !

Thanks to Javier Cervantes Villanueva for his reactivity regarding PyRDF !

SWAN usage at COMPASS Conclusion

Pros & cons of SWAN

- Well organised analysis in notebooks using sections (use of markdown, latex, etc), and different kernel availables
- Possibility to share notebooks with others
- Flexible enough to make systematic studies: use SWAN to quantify the influence of parameters and cuts
- Access to spark clusters ! Interfaced with ROOT thanks to PyRDF

SWAN usage at COMPASS Conclusion

Pros & cons of SWAN

- Well organised analysis in notebooks using sections (use of markdown, latex, etc), and different kernel availables
- Possibility to share notebooks with others
- Flexible enough to make systematic studies: use SWAN to quantify the influence of parameters and cuts
- Access to spark clusters ! Interfaced with ROOT thanks to PyRDF

Propositions of improvements

- RDataFrames: always have to define classes members as columns before using them.

  df.Define("rho0_mass", "rho0.M()")
  for rho0 a TLorentzVector

- SWAN: sometimes really slow connection or kernel not responding. Stability ?