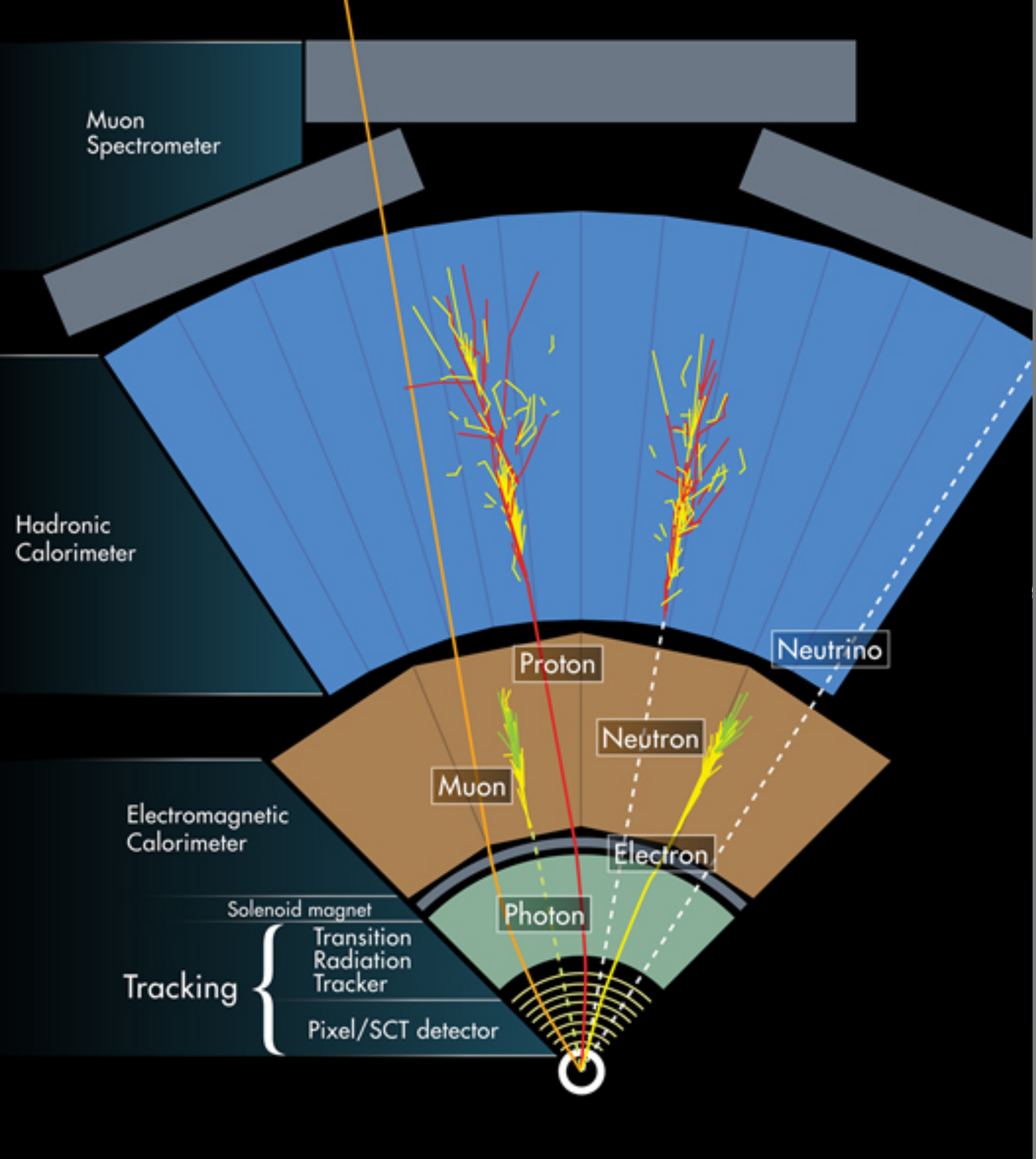


Simplified-Fast Detector Simulation

with MadAnalysis 5

arXiv:2006.09387 [hep-ph]

Goals of this tutorial...

How to...

- ...define a Jet clustering algorithm?
- ...define Reconstruction efficiencies?
- ...define a Smearing functions?
- ...define (mis)tagging functions?

Prerequisite?

- Python 2.7 & C++ compiler on your machine (you don't need to know neither)
- A sample showered in your favourite program i.e. Pythia, Herwig, Sherpa
- FastJet (simply type > install fastjet)
- Beginner level English knowledge.

How to define a jet clustering algorithm?

FastJet basics

1.> set main.fastsim.package = fastjet

2.> set main.fastsim.algorithm = XXX

3.> set main.fastsim.radius = XXX

4.> set main.fastsim.ptmin = XXX

1. Initialize FastJet
2. Set clustering algorithm e.g. antikt
3. Set clustering radius
4. Set minimum minimum transvers momentum for reconstructed jet

Set reconstruction mode for SFS:

> set main.fastsim.jetrecomode = XXX

jets

apply smearing to the reconstructed jet object

constituents

apply smearing before reconstructing jet objects

Which one constituents
jets should I use?

Jet smearing is designed to act on reconstructed jet objects after clustering. Most of the experimental fitting has been done using relatively stable jet objects. Thus you can use the "jets" option for most of the cases. Constituent smearing is applied to hadrons which will go into the clustered jet. This method is essential for jet-substructure studies.

How to define reconstruction efficiencies?

Reconstruction Efficiencies

> define reco_efficiency <object> <function> [<domain>]

- Jets (j)
- Hadronic taus (ta)
- Electrons (e)
- Muons (mu)
- Photons (a)

Any function in LaTeX or Python format. It can depend on PT, E, ETA, PX, PY, PZ, PHI of your object and it can include any functional form.

Where does this function live?

Use AND/OR to separate multi- domain input!

Reconstruction Efficiencies

```
> define reco_efficiency e 0.0 [PT < 10 or ABSETA>2.5]
> define reco_efficiency e 0.85 [ABSETA < 1.5 and PT >= 10 and PT < 100]
> define reco_efficiency e 0.95 [ABSETA >= 1.5 and ABSETA <= 2.5 and PT >= 100]
```

Probability to reconstruct each electron →

$$\varepsilon_e(p_T,\eta) = \begin{cases} 0\,\% & p_T < 10 \text{ GeV or } |\eta| > 2.5 \\ 85\,\% & |\eta| < 1.5 \;\&\; 10 \le p_T < 100 \text{ GeV} \\ 95\,\% & 1.5 \le |\eta| \le 2.5 \;\&\; p_T \ge 100 \text{ GeV} \end{cases}$$

What is happening in the background?

- Your input is going to be parsed into a C++ code to construct a piecewise function.
- The value calculated in the function per domain is the probability to accept the given particle.
- The particle of your choosing will be either reconstructed or thrown away according to this probability calculated via given function per domain.

How to define a smearing function?

Particle Smearing

```
> define smearer <object> with <obs> <function> [<domain>]
```

- Jets (j)
- Hadronic taus (ta)
- Electrons (e)
- Muons (mu)
- Photons (a)

Observable to be smeared, such as E, PT, PX, PY, PZ, ETA, PHI.

Any function in LaTeX or Python format. It can depend on PT, E, ETA, PX, PY, PZ, PHI of your object and it can include any functional form.

Where does this function live? Use AND/OR to separate multi-domain input!

Smearer

```
> define smearer j with E sqrt(E^2*0.05^2 + E*1.5^2)  [ABSETA <= 3]
> define smearer j with E sqrt(E^2*0.13^2 + E*2.7^2)  [ABSETA > 3 and ABSETA <= 5]
```

Standard deviation for jet energy smearing →

$$\sigma_j(E) = \begin{cases} 0.05 \oplus 1.5\sqrt{E} & \text{for } |\eta| \leq 3 \,, \\ 0.13 \oplus 2.7\sqrt{E} & \text{for } 3 < |\eta| \leq 5 \end{cases}$$

arXiv: 1405.6569

What is happening in the background?

- Your input is going to be parsed into a C++ code to construct a piecewise function as standard deviation of a Gaussian centred around the given energy value.
- The jet energy will be smeared within that Gaussian.
- The p_T and energy of the jet will be modified accordingly.

How to define (mis)tagging functions?

Particle Tagging

> define tagger <true> as <reco> <function> [<domain>]

- Jets (j)
- B-jets (b)
- C-jets (c)
- Hadronic taus (ta)
- Electrons (e)
- Muons (mu)
- Photons (a)

What would you like your true particle to be reconstructed as.

Any function in LaTeX or Python format. It can depend on PT, E, ETA, PX, PY, PZ, PHI of your object and it can include any functional form.

Where does this function live? Use AND/OR to separate multi-domain input!

Tagging options

Truth level		Reco level
• Jets (j)		➡ b/c/ta/e/a
• B-jets (b)		➡ j/b/c
• C-jets (c)		➡ j/b/c
• Hadronic taus (ta)	**as**	➡ j/ta
• Electrons (e)		➡ j/mu/a
• Muons (mu)		➡ j/e/a
• Photons (a)		➡ j/e/mu

What is happening in the background?

- Your input is going to be parsed into a C++ code to construct a piecewise function.
- The value calculated in the function per domain is the probability to (mis)tag the given particle.
- The particle of your choice will be either reconstructed as it is or reconstructed as the particle that its appointed according to this probability calculated via given function per domain.

What’s next?

To get a LHE-type output of the detector simulation, simly type

> set main.outputfile = “my_sample.lhe.gz”

> submit MY_FIRST_FASTSIM

Hands on!

Here we will generate a Drell-Yan process and try to apply some detector effects. Then we will plot the corresponding distributions to see its effects.

Data preparation in MadGraph:

Open Madgraph and generate the following sample,

```
> generate p p > l+ l-
> output DrellYan
> launch
```

Don’t forget to shower the events! We need the HEPMC file!

SFS setup in MadAnalysis 5

```
set main.fastsim.package = fastjet
set main.fastsim.algorithm = antikt
set main.fastsim.radius = 0.5
```

→ FastJet setup

```
define smearer mu with PT sqrt(0.01^2  + pt^2*1.0e-4^2) [abseta <= 0.5 and pt > 0.1]
define smearer mu with PT sqrt(0.015^2 + pt^2*1.5e-4^2) [abseta > 0.5 and abseta <= 1.5 and pt > 0.1]
define smearer mu with PT sqrt(0.025^2 + pt^2*3.5e-4^2) [abseta > 1.5 and abseta <= 2.5 and pt > 0.1]
```

→ p_T smearing for muon tracking

```
define reco_efficiency mu 0.0  [pt <= 10.0 or abseta > 2.4]
define reco_efficiency mu 0.891 [abseta <= 1.5 and pt > 10.0 and pt <= 1000.]
define reco_efficiency mu 0.891*exp(0.5 - pt*5.0e-4) [abseta <= 1.5 and pt > 1000.]
define reco_efficiency mu 0.882 [abseta > 1.5 and abseta <= 2.4 and pt > 10.0 and pt <= 1000.]
define reco_efficiency mu 0.882*exp(0.5 - pt*5.0e-4) [abseta > 1.5 and abseta <= 2.4 and pt > 1000.]
```

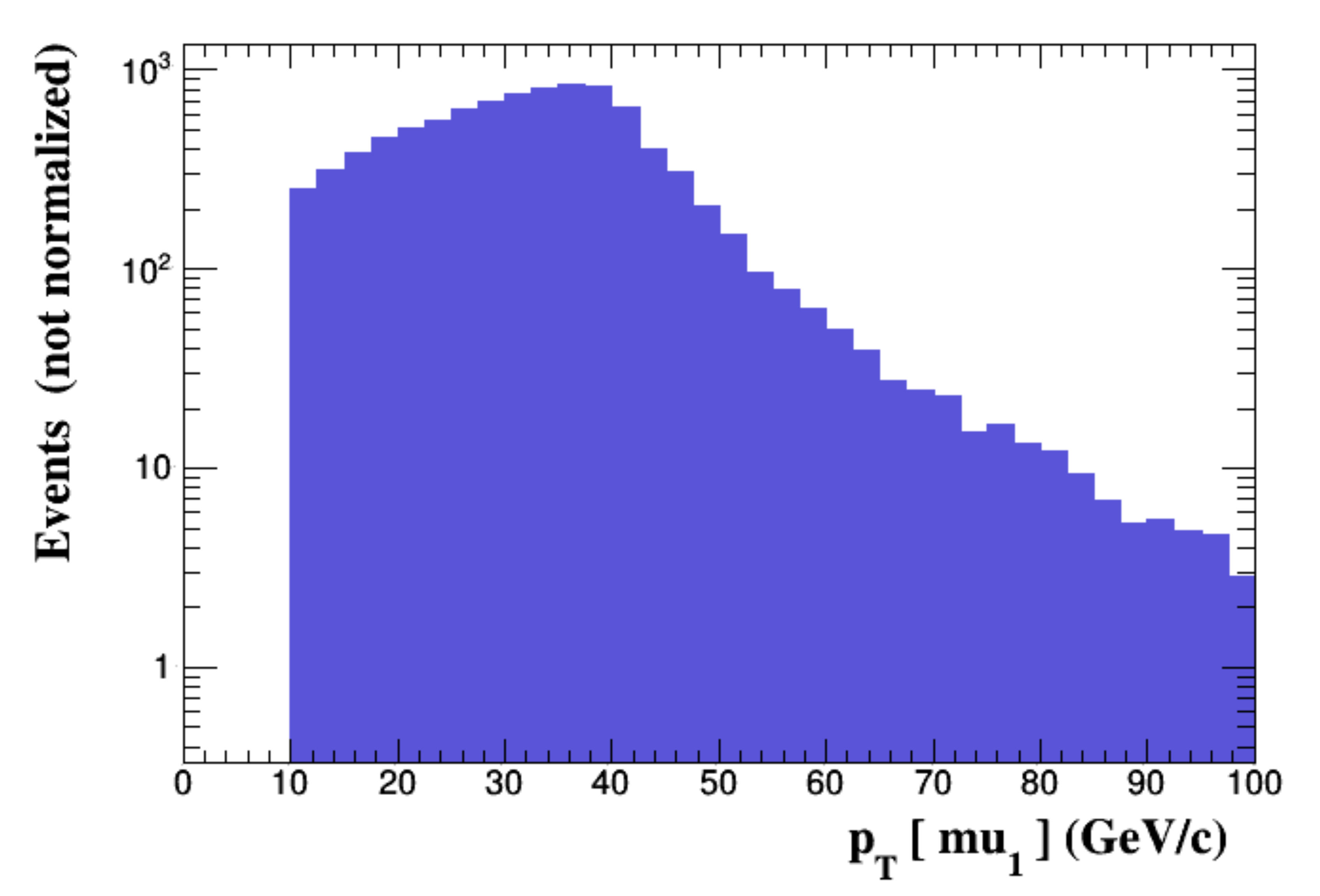
→ Reconstruction efficiencies for muon

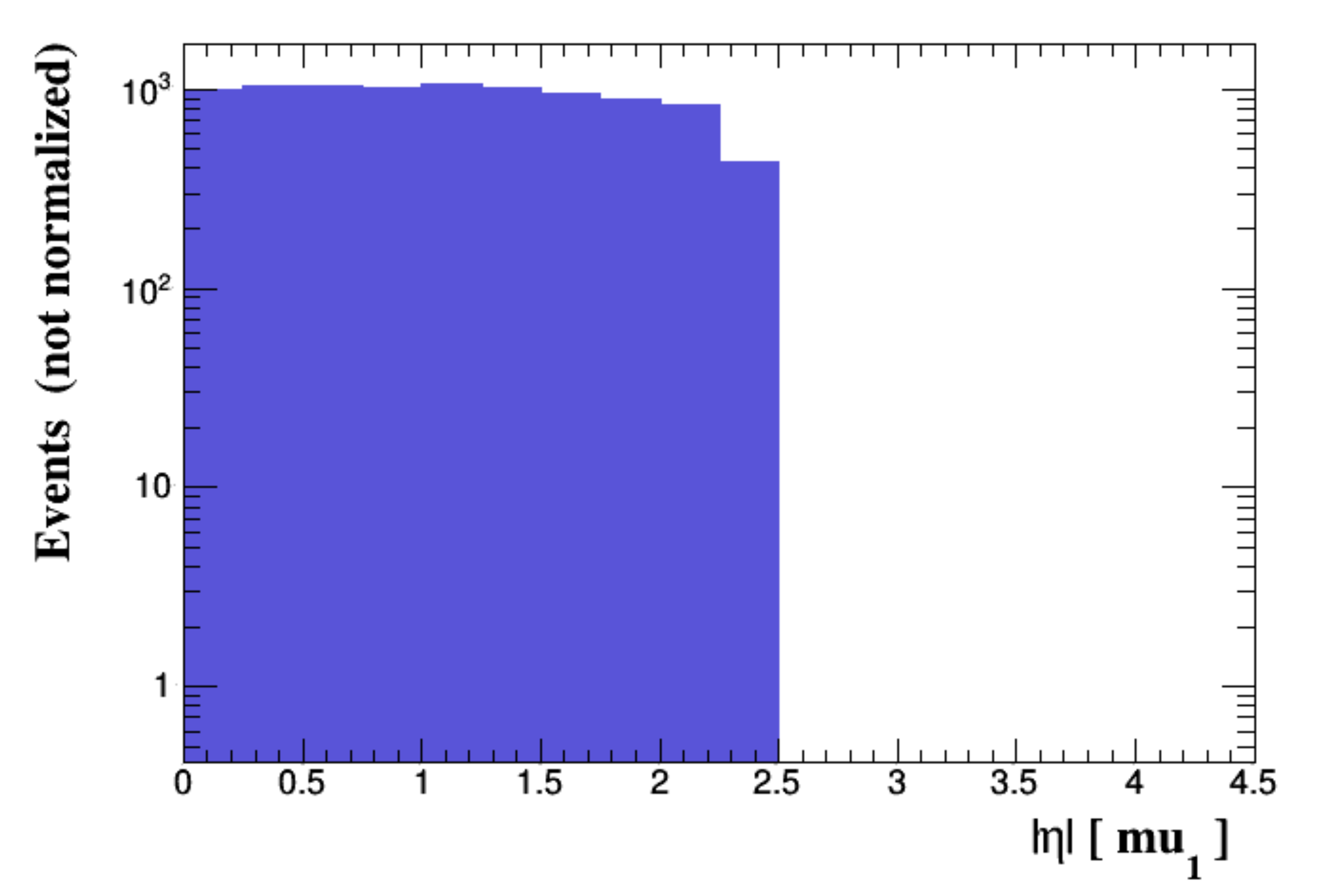
Let's plot the results!

```
import /PATH/my_sample.hepmc.gz

reject (l)  PT < 10
reject M (l+ l-) < 50.0
reject (hadronic)  PT < 20
reject (hadronic) DELTAR (l) < 0.4

plot PT (mu[1]) 40 0 100 [logY]
plot ABSETA (mu[1]) 18 0 4.5 [logY]

submit tutorial
```

Some important cuts

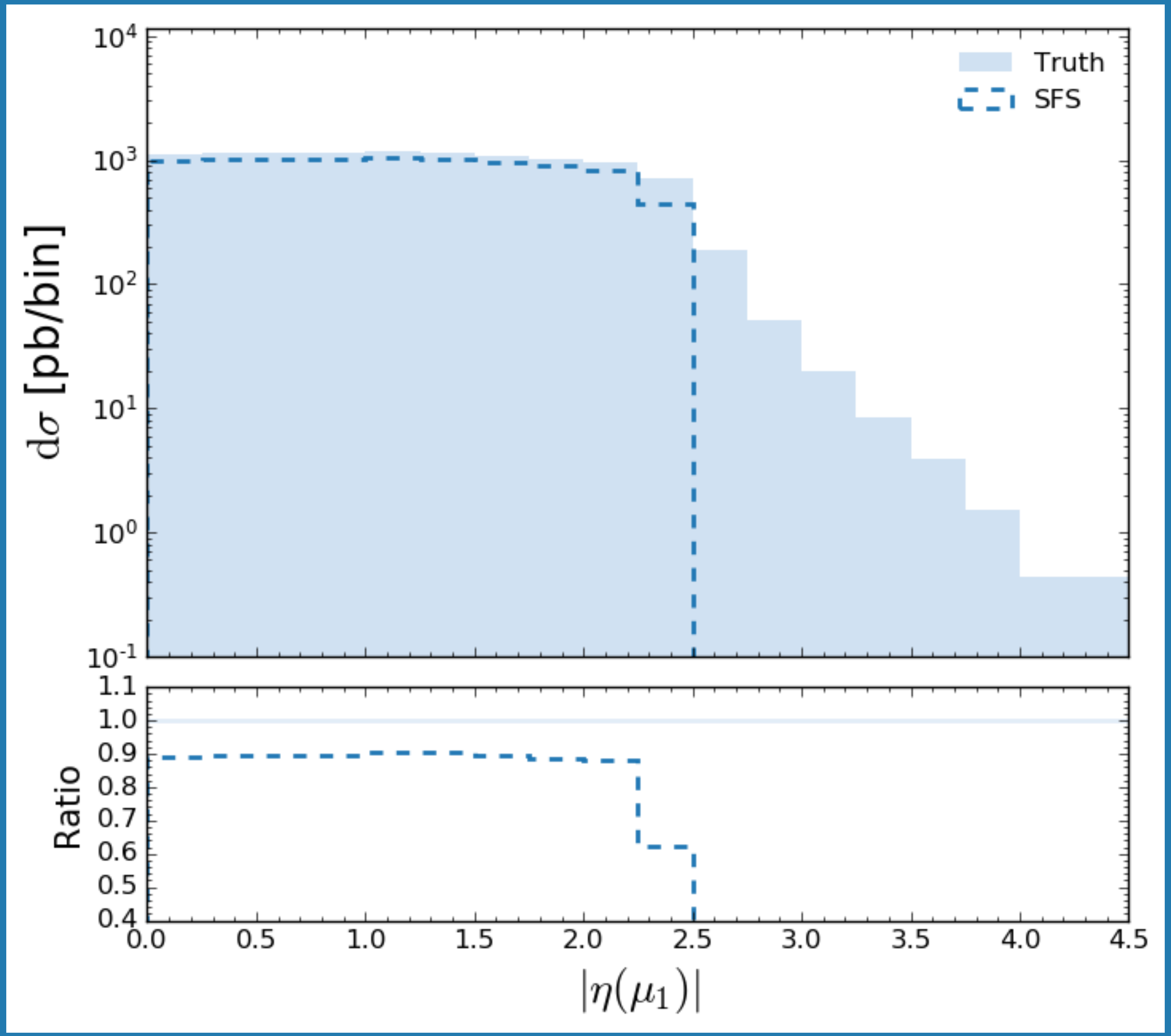
Here are the plots for p_T and $|\eta|$ for the leading muon

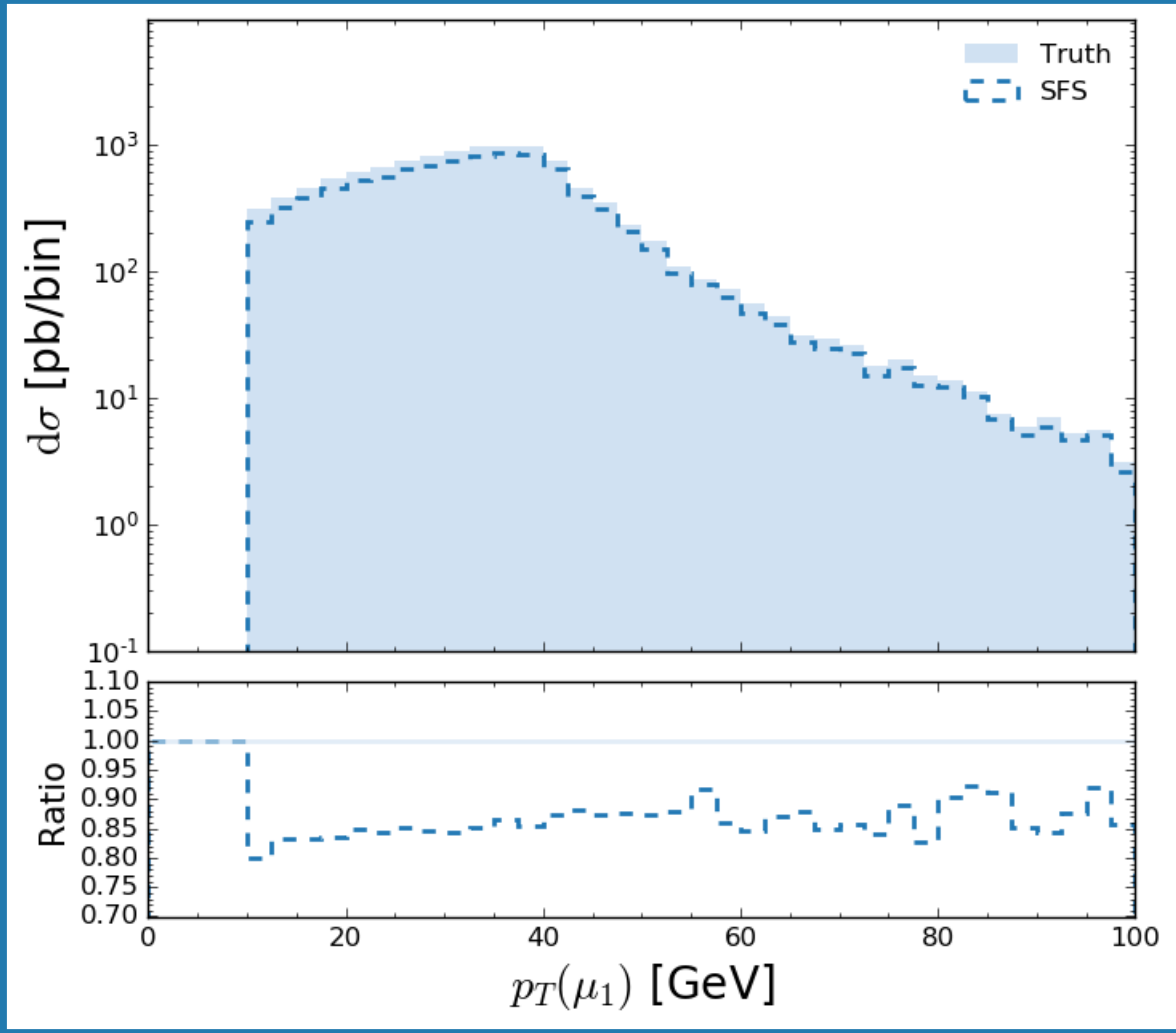




Voilà!

What's the difference?





I'm too lazy to write my own detector simulation! Are there any ready-to-use ones?

Yes! Run MadAnalysis 5 with the following command!

> ./bin/ma5 -R madanalysis/input/CMS_default.ma5

For CMS construction or

> ./bin/ma5 -R madanalysis/input/ATLAS_default.ma5

For ATLAS construction

I'm an expert, I want to write my own analysis embedded in a detector simulation. Is that possible?

Yes! Write your own SFS card or use a default one and then simply type

> ./bin/ma5 -Re my_folder my_analysis my_sfs.ma5

Where to find more information?

- ★ Tutorials
- ★ Normal mode reference card
- ★ SFS paper
- ★ Just give it a shot!