Using LeptonInjector or NuGen detector mode simulation examples

2019/9/14

Diffuse Global-fit Meeting in ERI Tokyo

Kotoyo Hoshina

What do you want to do with detector simulations?

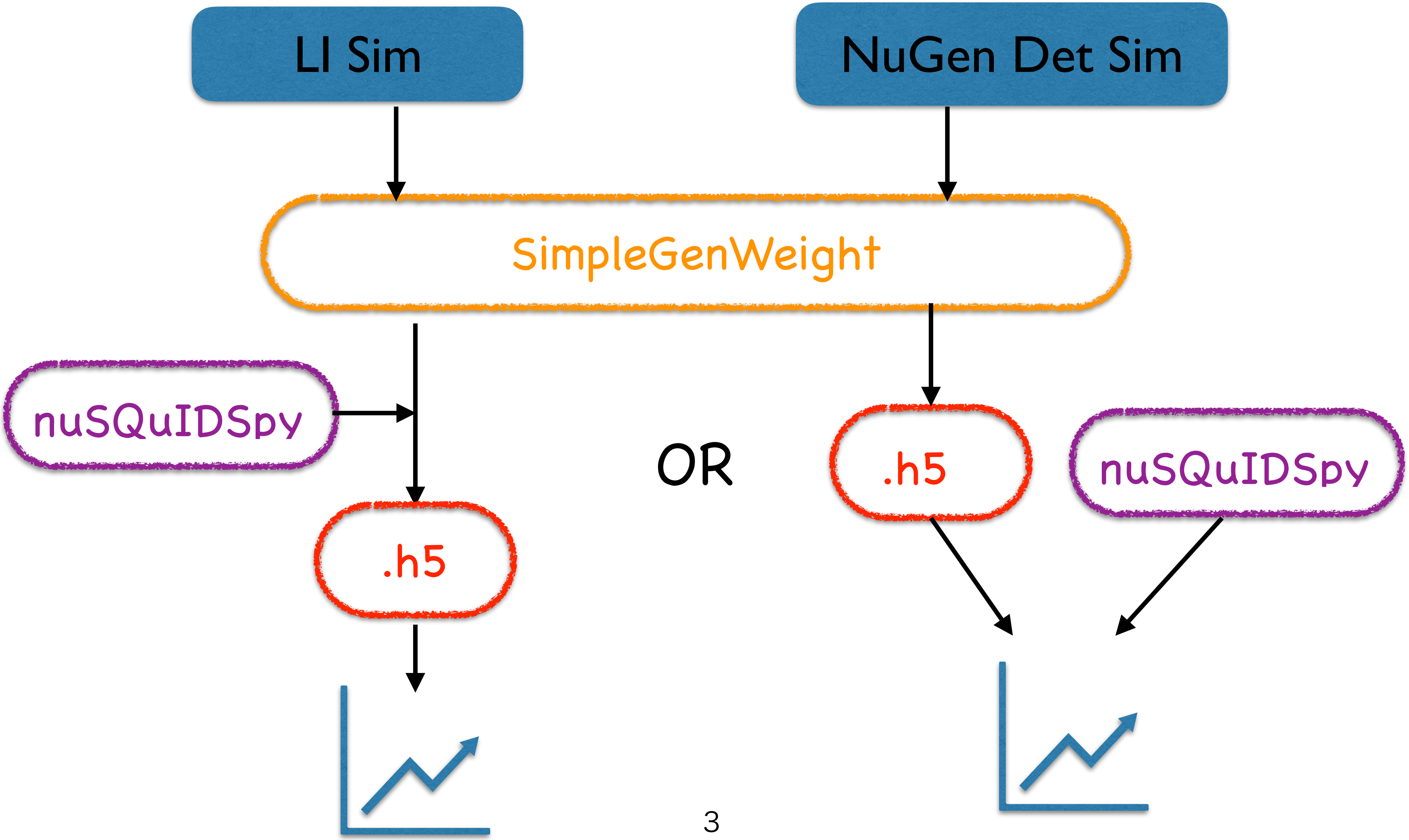
I just want to use them with a standard cross section and PREM Earth, and the gamma index of primary neutrino spectrum is almost fixed!

I may test several cross sections or just scaling total cross sections (assuming the effect of NC interaction during propagation in Earth to the arrival spectrum is negligible), or test several oscillation parameters. I want to use someone's program. I'm good at C++.

I need to change Earth model or gamma index of input neutrino flux drastically, but I don't change cross sections in a fitting loop!

You are a light user. Do it once and then forget everything!

All Python



LI Sim

NuGen Det Sim

SimpleGenWeight

nuSQuIDSpy

.h5

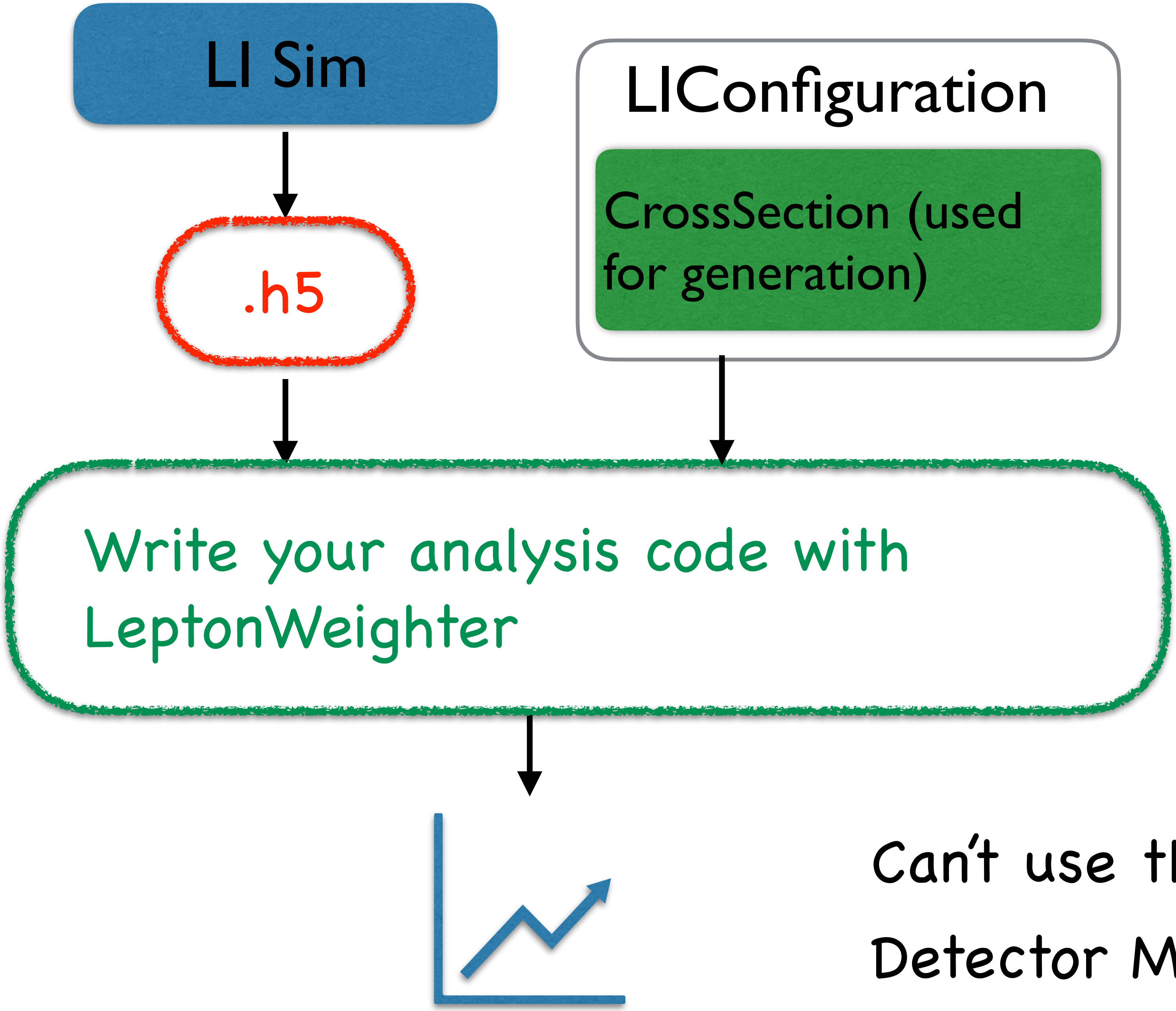
OR

.h5

nuSQuIDSpy

Start coding from LeptonWeighter or just use GolemFit

C++



LI Sim

.h5

LIConfiguration

CrossSection (used for generation)

Write your analysis code with LeptonWeighter

Can't use them for NuGen Detector Mode Sim

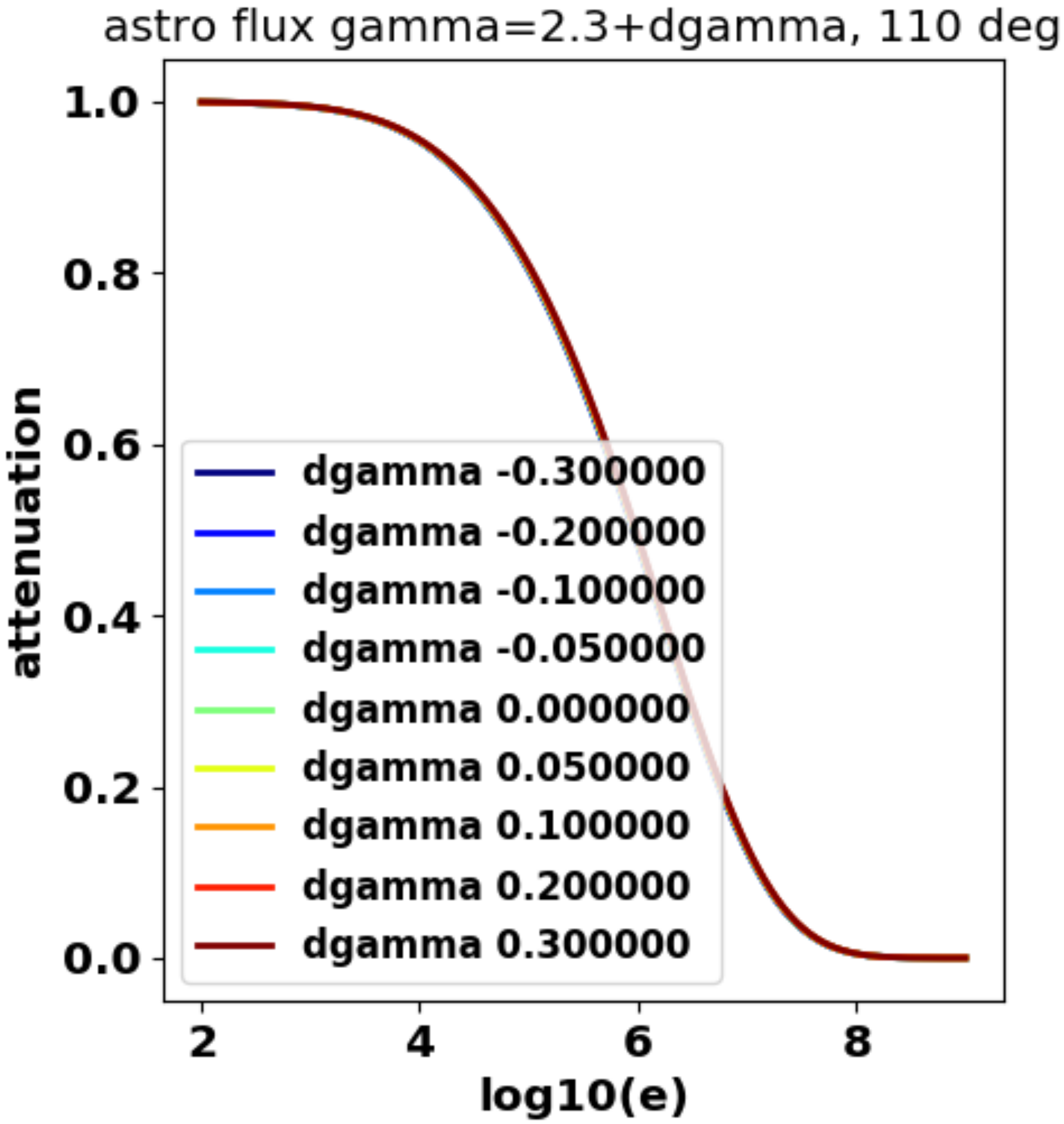
Write your program from scratch. nuFATE may be a good choice.*

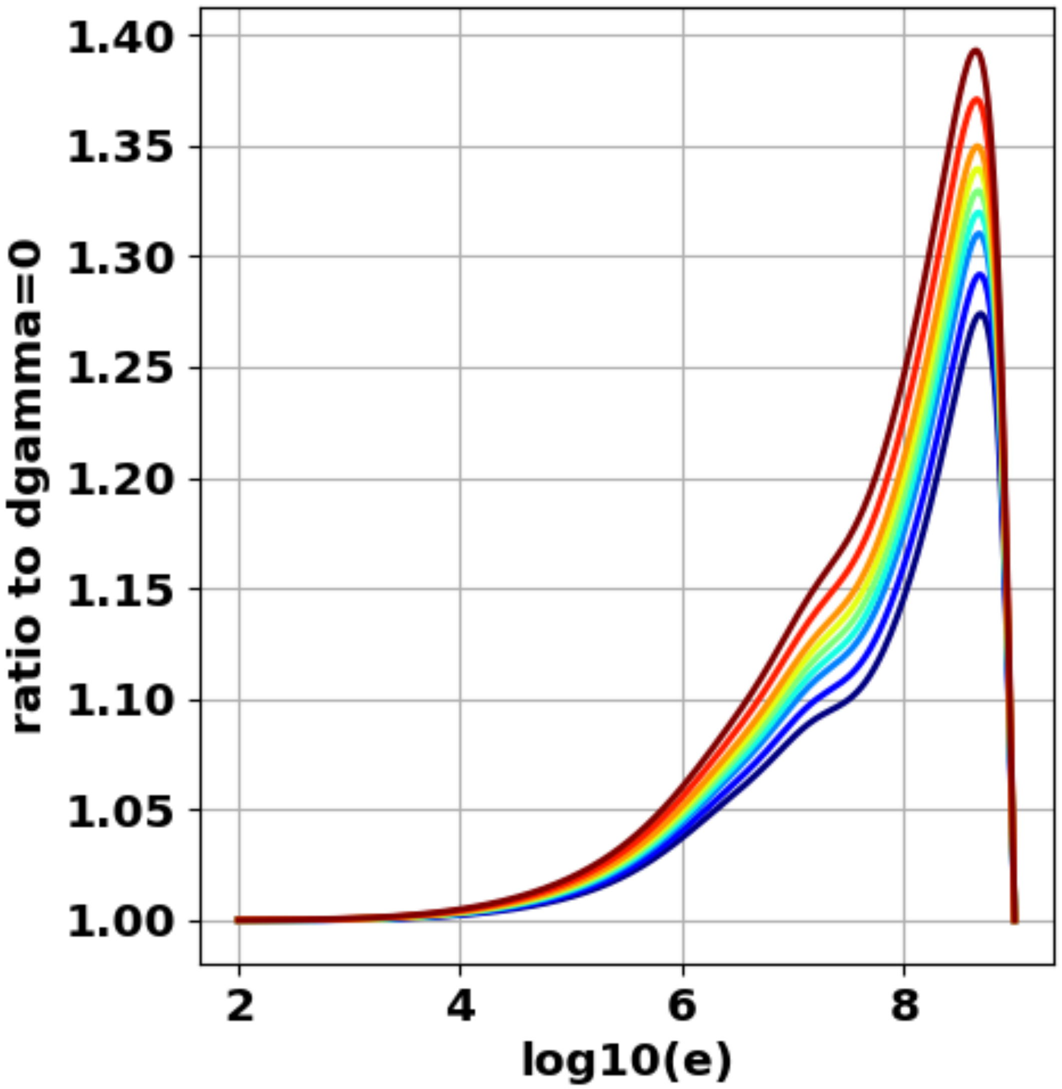
*to experts: Does GolemFit do it?

Python or C++

- ✦ If you need to change gamma index of primary flux drastically, you will need to re-calculate propagation in Earth for each loop in fitting process. For nuSQuIDS call EvolveState for every change of initial flux. For nuFATE need to solve square system Ax = b using Eigenvector and Eigenvalues for every change of initial flux. nuFATE could be faster (Aaron is trying to speed up solver).
- ✦ Or, just pre-calculate all possible propagations in advance and create spline curve for all possible primary flux, and use the spline while fitting.
- ✦ If you want to do the process on-the-fly, I have example codes (not stable yet). Tell me what you want to do, I may give some advice.

I don’t know which is me…





- ✦ If you care about the difference, you are

For hackathon

I prepared example codes, learn it tomorrow.
https://github.com/kotoyo/NuWeighter/tree/hackathon

Ask experts
(Carlos and authors of LeptonWeighter, GolemFit)

There are good instructions in software pages too:

https://github.com/IceCubeOpenSource/LeptonWeighter

https://github.com/IceCubeOpenSource/GolemFit

Ask me what you want to do.

Prerequisites for running example

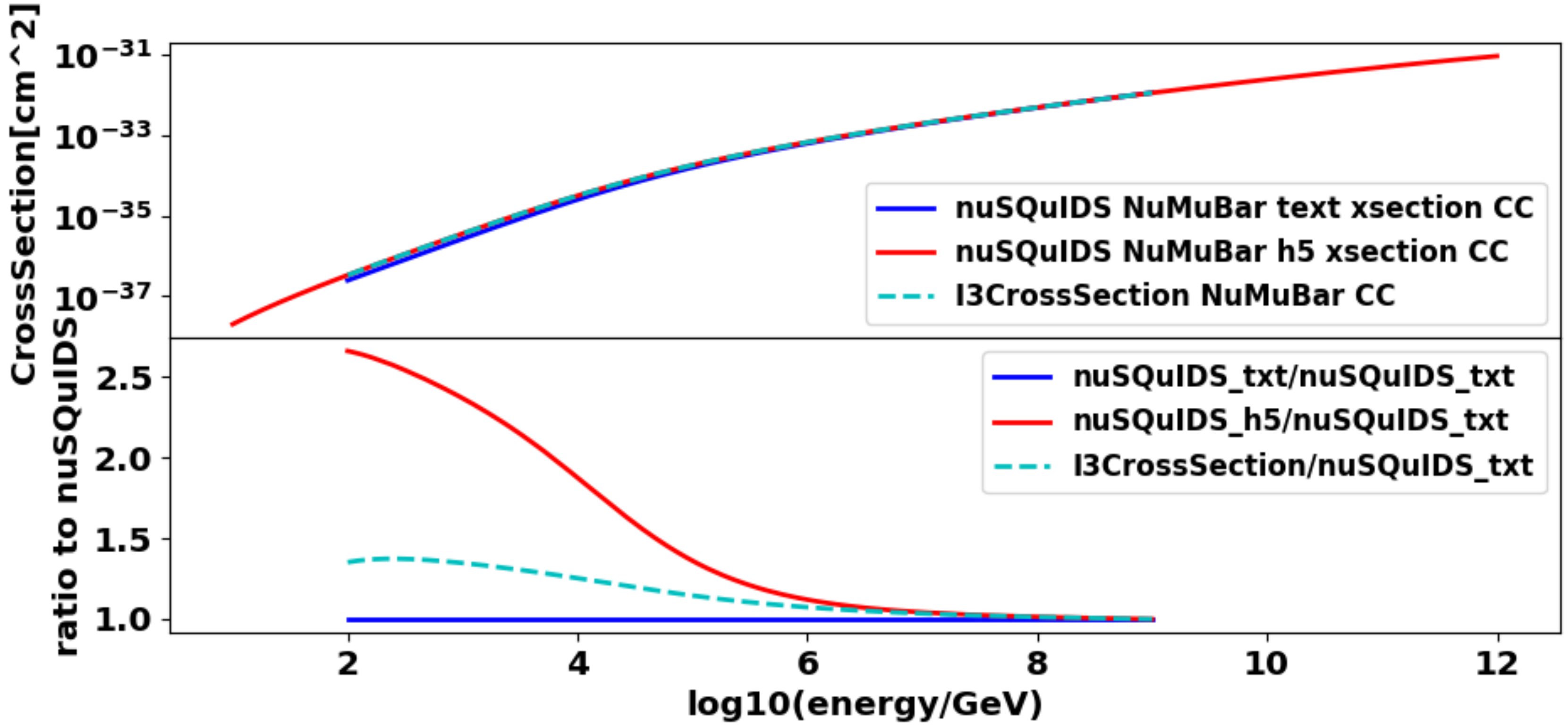
- cvmfs environment. If you want to run it local, just follow the page below.
  https://wiki.icecube.wisc.edu/index.php/CVMFS
- A compiled COMBO meta-project with LeptonInjector.
  (I don’t know why COMBO still doesn’t include LeptonInjector…)
- A running flux module (NewNuFlux, MCEq, etc) if you want to calculate atmospheric spectrum.

To do

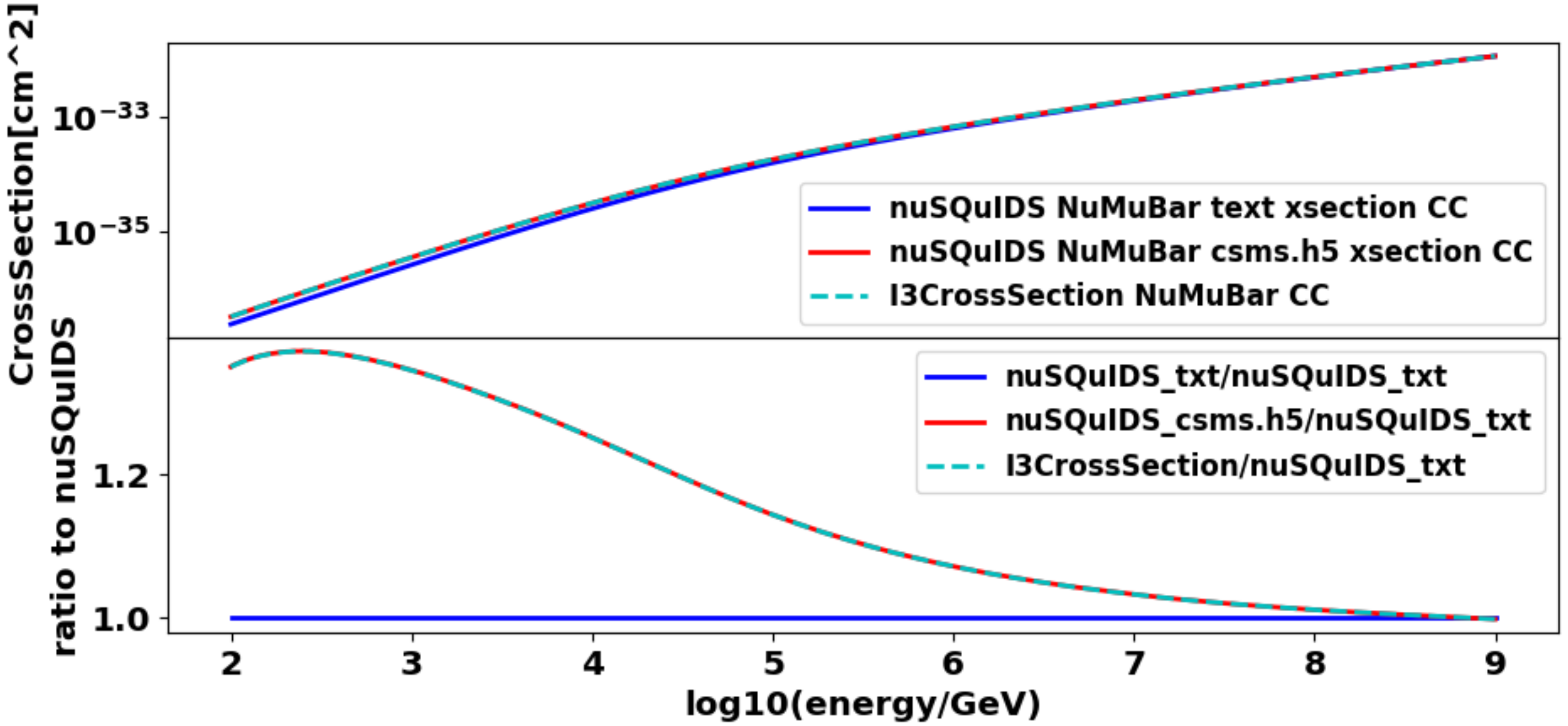
- Install NuWeighter. Follow README to prepare it. https://github.com/kotoyo/NuWeighter/tree/hackathon
- Run H00x examples. Read comments on top of each file.

this slide will be updated by tomorrow…

Update : This problem was solved!!!
(sorry it was my script bug…)



buggy



fixed

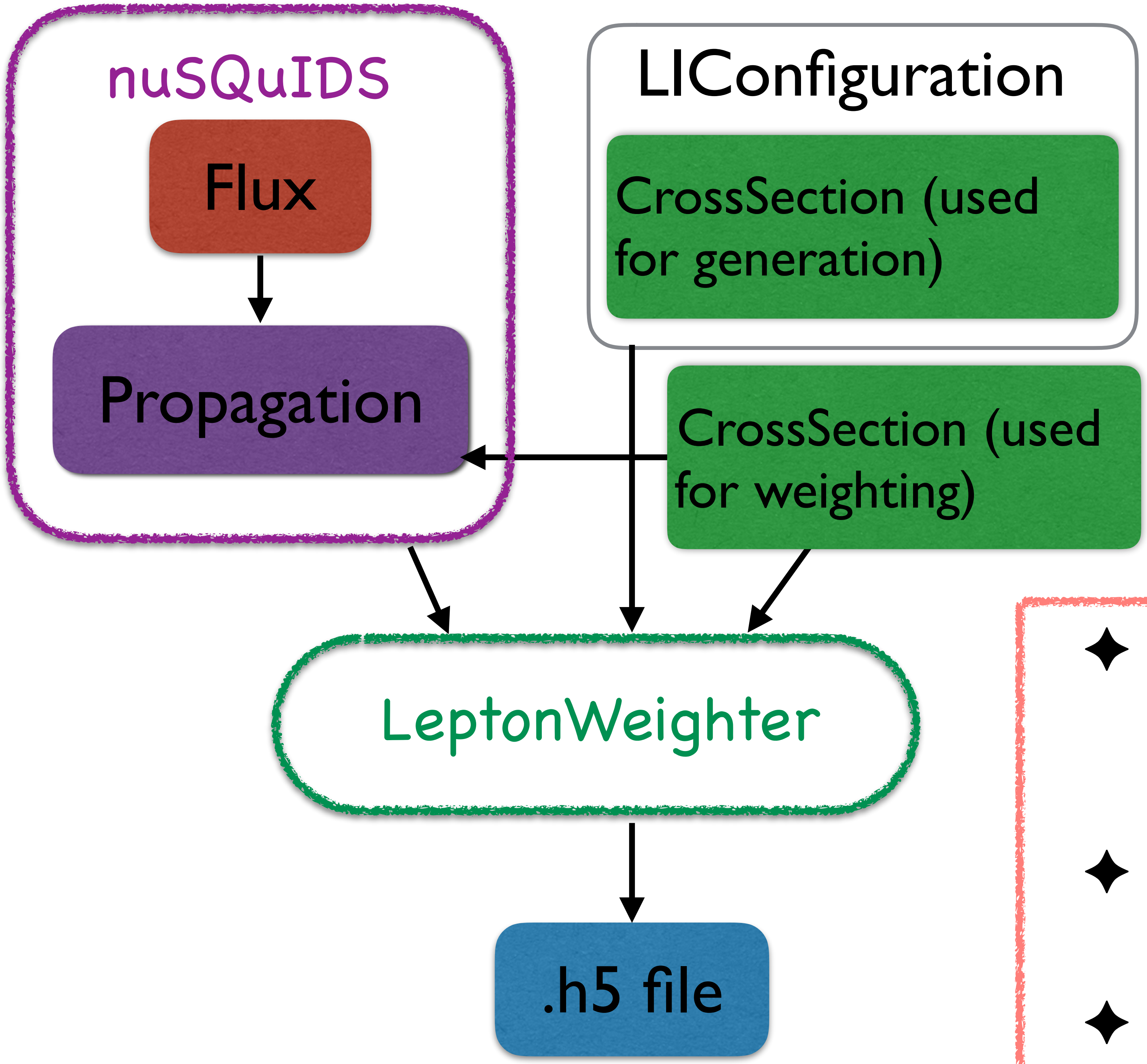
Final Remark

Do not weight Detector-mode simulations (LeptonInjector, NuGen DetectorMode) in a BLACK BOX!

It is useful, but can go wrong very easily!

backups

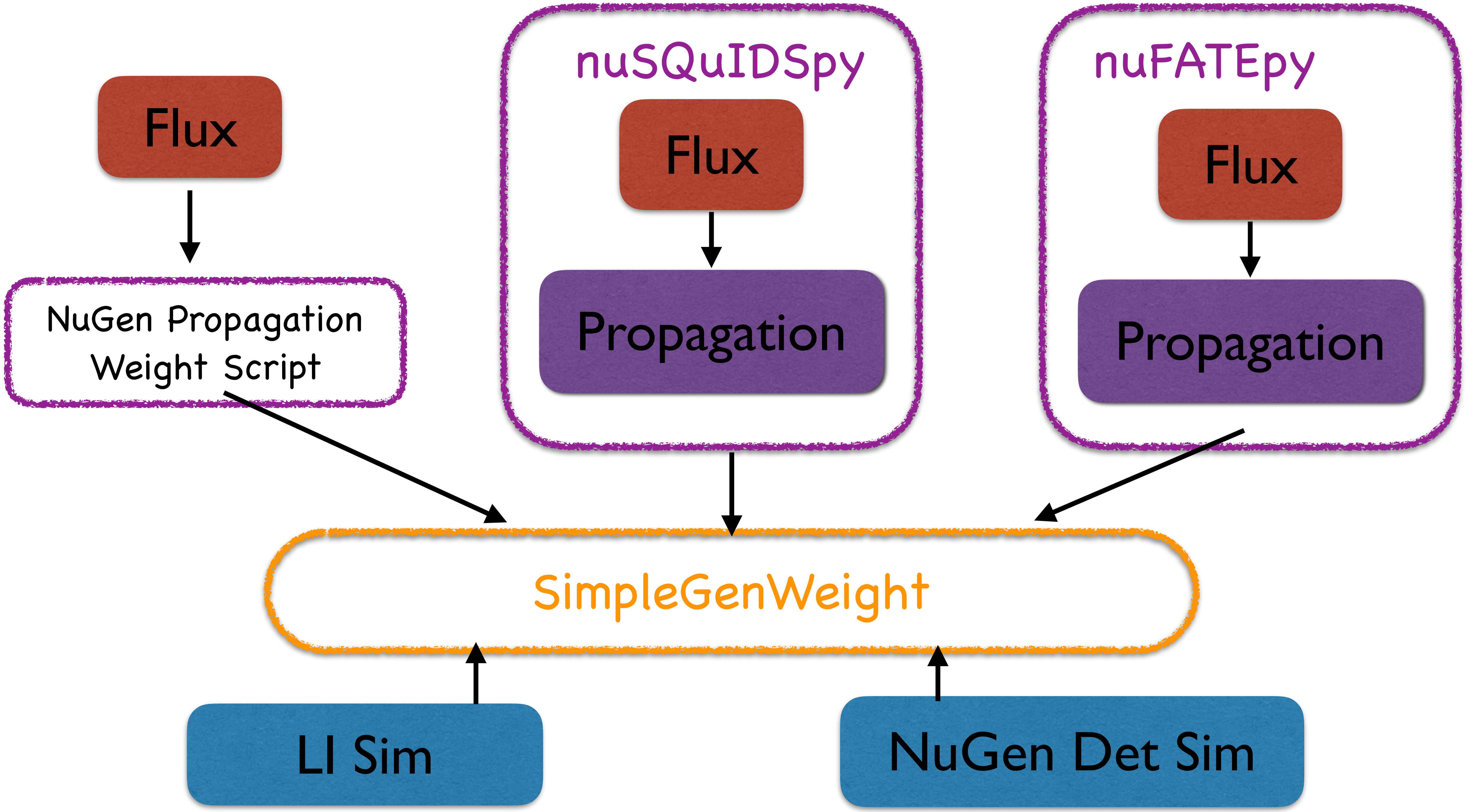
Option 2 : using nuSQuIDS + LeptonWeighter


nuSQuIDS
Flux
Propagation
LIConfiguration
CrossSection (used for generation)
CrossSection (used for weighting)
LeptonWeighter
.h5 file

- ✦ MEOWS analysis is using it

- ✦ Need to re-write your analysis program (may be from the scratch)
- ✦ You may need to write them in C++
- ✦ Not applicable for NuGen's detector mode simulation

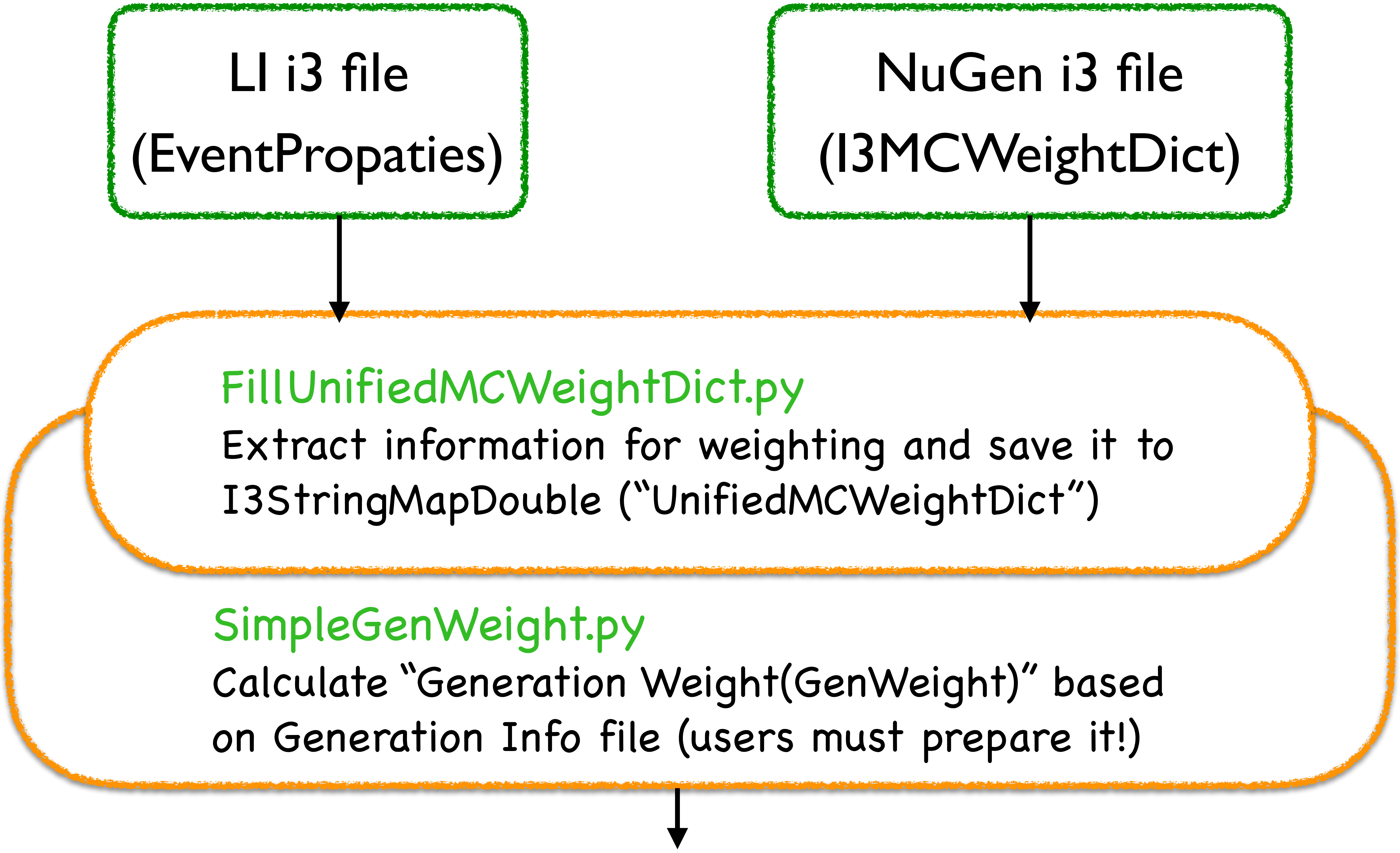
Option 3: using SimpleGenWeight and whatever



- All python based
- Supports both NuGen and LI

- Need i3 file
- Could be slower than C++

What the SimpleGenWeight does?



LI i3 file
(EventPropaties)

NuGen i3 file
(I3MCWeightDict)

FillUnifiedMCWeightDict.py
Extract information for weighting and save it to I3StringMapDouble ("UnifiedMCWeightDict")

SimpleGenWeight.py
Calculate "Generation Weight(GenWeight)" based on Generation Info file (users must prepare it!)

Weight simulation with GenWeight in your script, with propagation module (nuSQuIDS, nuFATE, etc)

Remind : OneWeight

CAUTION:
If you use multiple datasets, no overlap in generation space is allowed!

Astro Flux (per nu + nubar combined)

n(Hz) = norm * (primaryE/1e5) $^{-\text{gamma}}$ * $\underline{\text{OneWeight}}$ / NGenEvents

flux at Earth's Surface

Atmospheric Flux (per primary type)

n(Hz) = AtmFlux$_{\text{at\_Earth's\_surface}}$ * $\underline{\text{OneWeight}}$ / (NGenEvents/2)

$\underline{\text{OneWeight}}$ =

NAtms * TotalCrossSection * ➡ final interaction probability
PropagationWeight * ➡ propagation in Earth
InjectionArea [cm^2] * ➡ normalization factor
SolidAngle [str] * ➡ normalization factor
PrimaryEnergyIntegral / GenerationSpectrum ➡ energy norm. factor

Docs: Read Gary and Chad's documents
https://wiki.icecube.wisc.edu/index.php/Weights_in_Simulation_Data

Breakdown of GenWeight

All weights are calculated "per primary type"

n(Hz) = Flux_at_detector * DifferentialCrossSection / GenerationWeight

Generation Weight =

$\sum$for_all_datasets {

1 / NAtms *
DifferentialCrossSection / TotalCrossSection *
1 / InjectionArea *
1 / SolidAngle *
GenerationSpectrum / PrimaryEnergyIntegral *
NGenEvents_for_the_dataset

}

- ✦ GenerationWeight can have overlap in generation space.
- ✦ To understand why it works, read Gary and Chad's documents in https://wiki.icecube.wisc.edu/index.php/Weights_in_Simulation_Data

OneWeight and GenWeight gives consistent results

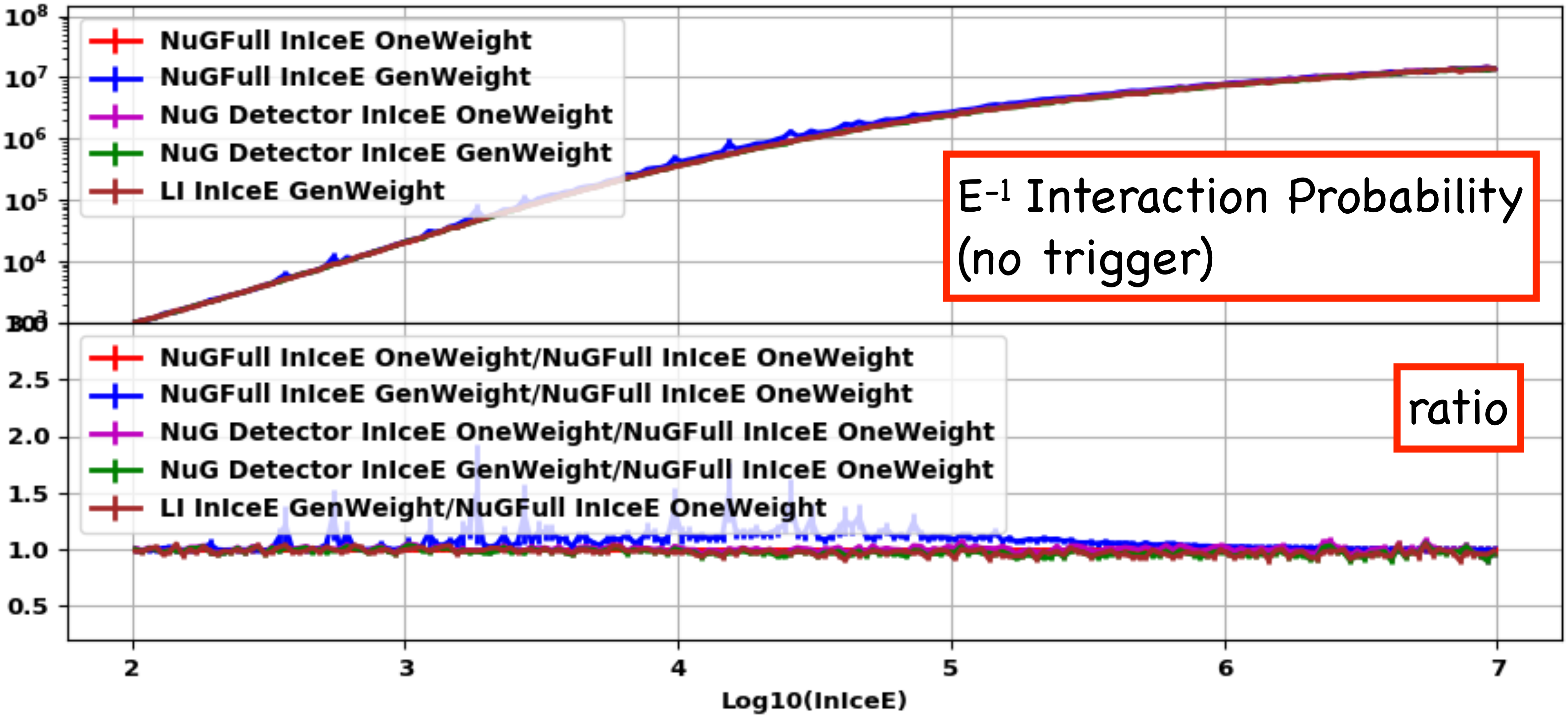
NuGen Full Sim * OneWeight

NuGen Full Sim * GenWeight (may be affected by low statistics?)

NuGen DetectorSim * OneWeight

NuGen DetectorSim * GenWeight

LeptonInjector Sim * GenWeight



E^{-1} Interaction Probability (no trigger)

ratio

Flux at Detector

n(Hz) = Flux_at_detector * Differential CrossSection / GenerationWeight

nuSQuIDS

- For AstroFlux, use nuSQUIDS
- For Atmospheric, use nuSQUIDSAtm (array of nuSQUIDS for each zenith)
- Input energy and neutrino type, then get flux at detector

nuFATE

- It basically gives "attenuation" of flux
- One nuFATE module is required per neutrino type
- Input total column depth of the Earth for the given zenith angle, then get attenuation

Calculate arrival flux FOR EACH POSSIBLE primary flux!
(e.g. need calculation for all possible gamma index for astrophysical flux)

Weighting with GenWeight and nuSQuIDSAtm

```
e = tables.open_file(open(“Ares_MuonL4GenW_00_22_01000.h5”))
truecosz = numpy.cos(
  e.root.UnifiedMCWeightDict.cols.PrimaryNeutrinoZenith[:])
primarytype =
e.root.UnifiedMCWeightDict.cols.PrimaryNeutrinoType[:]
trueE = e.root.UnifiedMCWeightDict.cols.PrimaryNeutrinoEnergy[:]
genw = e.root.SimpleGenWeight.cols.GenWeight[:]
diffxsec = e.root.csms.cols.InteractionDiffXsectionCGS[:]

import nuSQUIDSpy as nsq
import nuSQUIDSTools
nsq_atm = nsq.nuSQUIDSAtm(“atm_earth.hdf5”)
atmw = numpy.zeros(truecoszen.shape)
flavorID = 1 # muon flavor
nuID = 0     # neutrino or antineutrino
for i, cz in enumerate(truecoszen) :
   flavorID, nuID = get_nusqtype(primarytype[i])
   atmw[i] = nsq_atm.EvalFlavor(flavorID, cz, trueE[i]/1e-9,nuID)

atm_weights = atmw * diffxsec / genw

plt.hist(ene, bins=100, weights=atm_weights)
```

Pre-calculated propagation file

Weighting with GenWeight and nuFATE

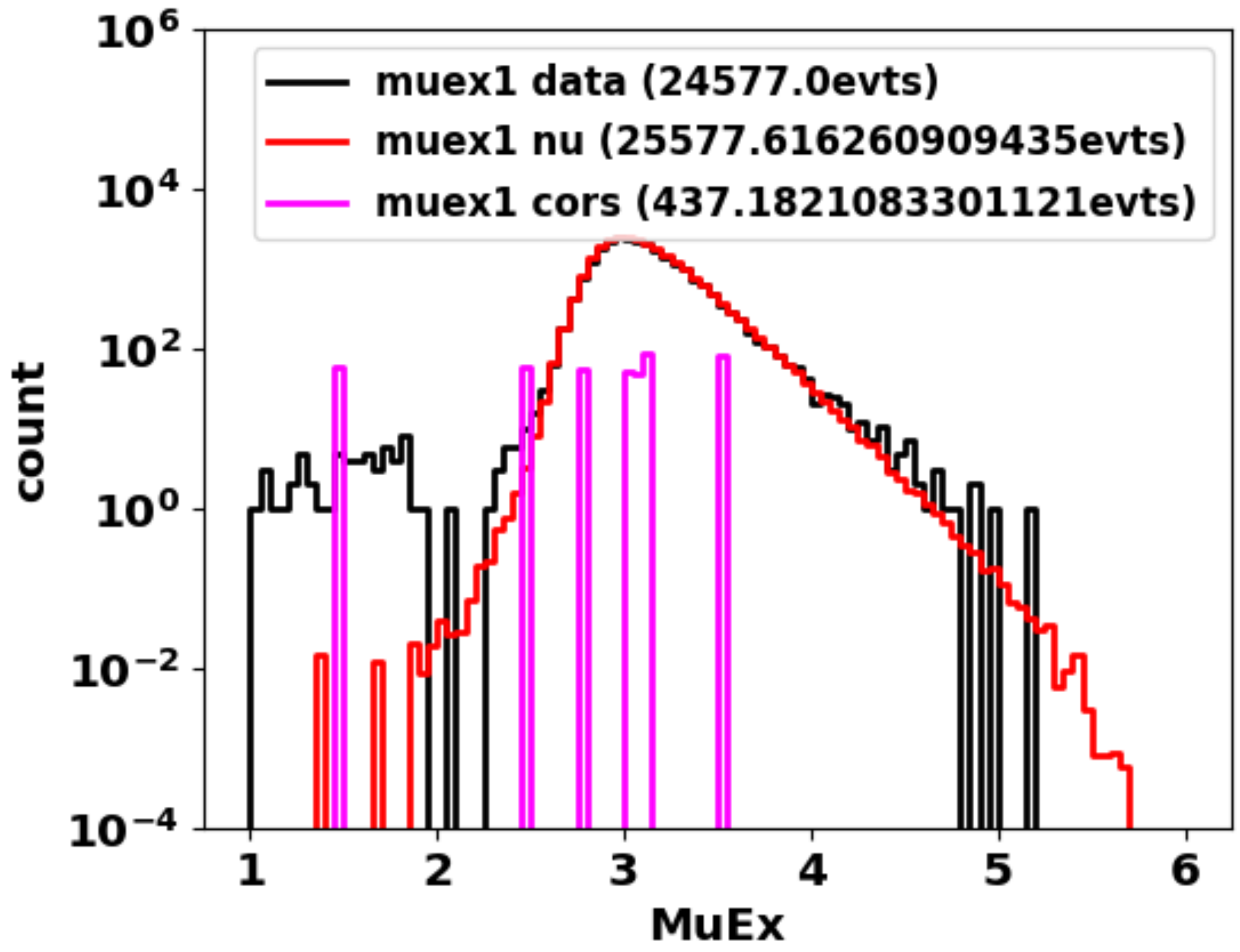
```
e = tables.open_file(open("Ares_MuonL4GenW_00_22_01000.h5"))
truez = e.root.UnifiedMCWeightDict.cols.PrimaryNeutrinoZenith[:]
…
import nuFATEpy as nufate
nuf = nufate.nuFATE(2, gamma, nucc, nunc, nudiffnc)
nubarf = nufate.nuFATE(-2, gamma, nubarcc, nubarnc, nubardiffnc)
atmw = numpy.zeros(truecoszen.shape)
NA = 6.0221409e+23

for i, z in enumerate(truez) :
    flux_at_z = …(get array of primary flux at given zenith)
    cdep = …(get column depth of the Earth at given zenith)
    if (primarytype[i]>0) :
        nuf.set_initial_flux(flux_at_z)
        atmw[i] = nuf.get_arrival_flux_at(cdep*NA, trueE[i])
    else :
        nubarf.set_initial_flux(flux_at_z)
        atmw[i] = nubarf.get_arrival_flux_at(cdep*NA, trueE[i])

atm_weights = atmw * diffxsec / genw

plt.hist(ene, bins=100, weights=atm_weight)
```

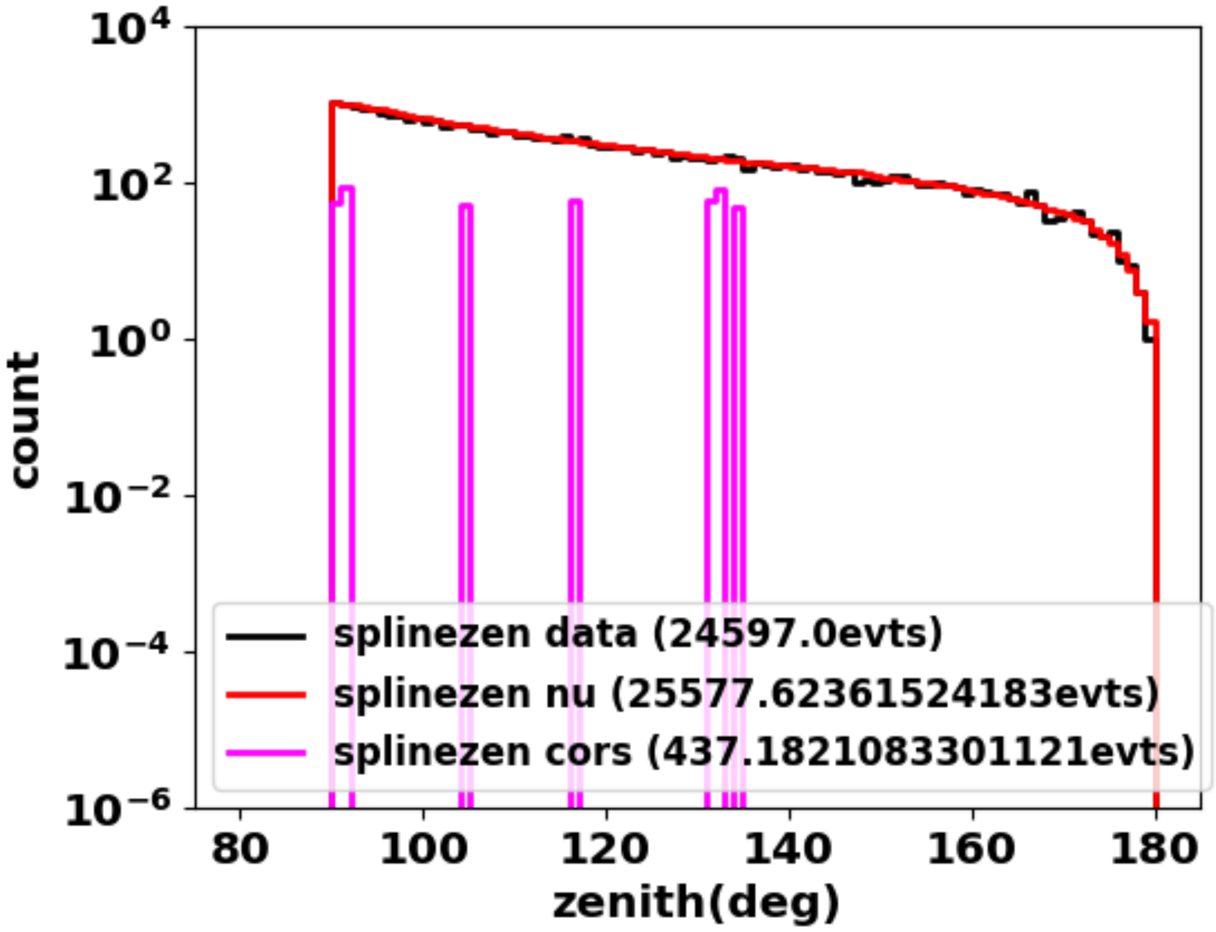
Cross section files

need Kotoyo's fork

Weighting with GenWeight and nuSQuIDSAtm

- Data : burnsample 2010-2018
- Nu Sim : Spencer’s LeptonInjector NuMu CC simulation, weighted with GenWeight and nuSQuIDSAtm
- It is working reasonably 😀





Spectrum Fitting

- When you fit "shape" of primary flux, the iterator changes primary flux for every iteration.
  - Both nuSQuIDS and nuFATE need to be recalculated when primary flux is changed. ➡ Too Slow !!
  - Worst case : Fitting atmospheric flux with various inclination of primary cosmic ray flux (changing delta-gamma)

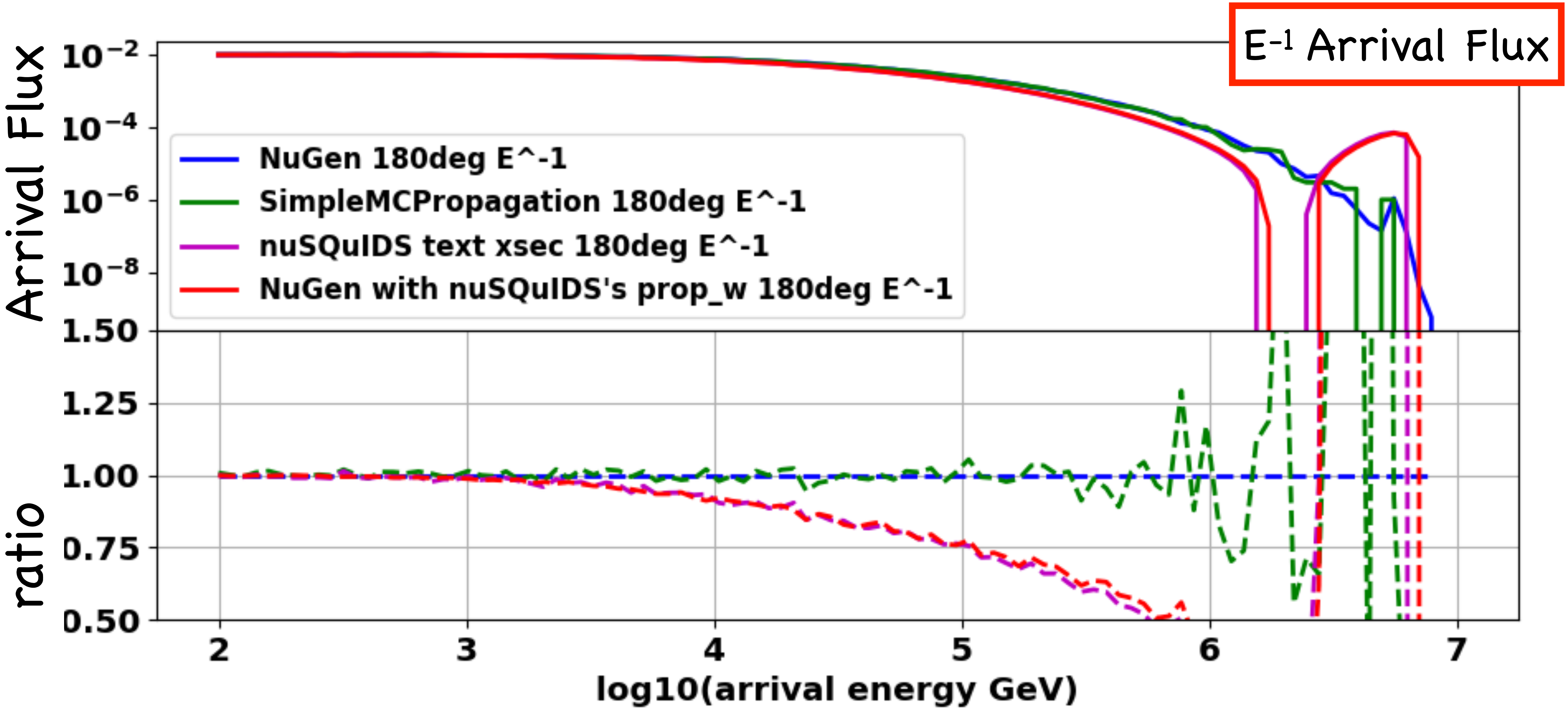
Possible Solutions ?

- Control order of fitting parameters (to reduce number of calls)
- Use kotoyo's fork branch of nuFATE (faster than original one, but still slower than legacy OneWeight with NuGen Full Mode)
- Pre-calculate all possible combination of primary fluxes and zenith, then make spline table

Another issue

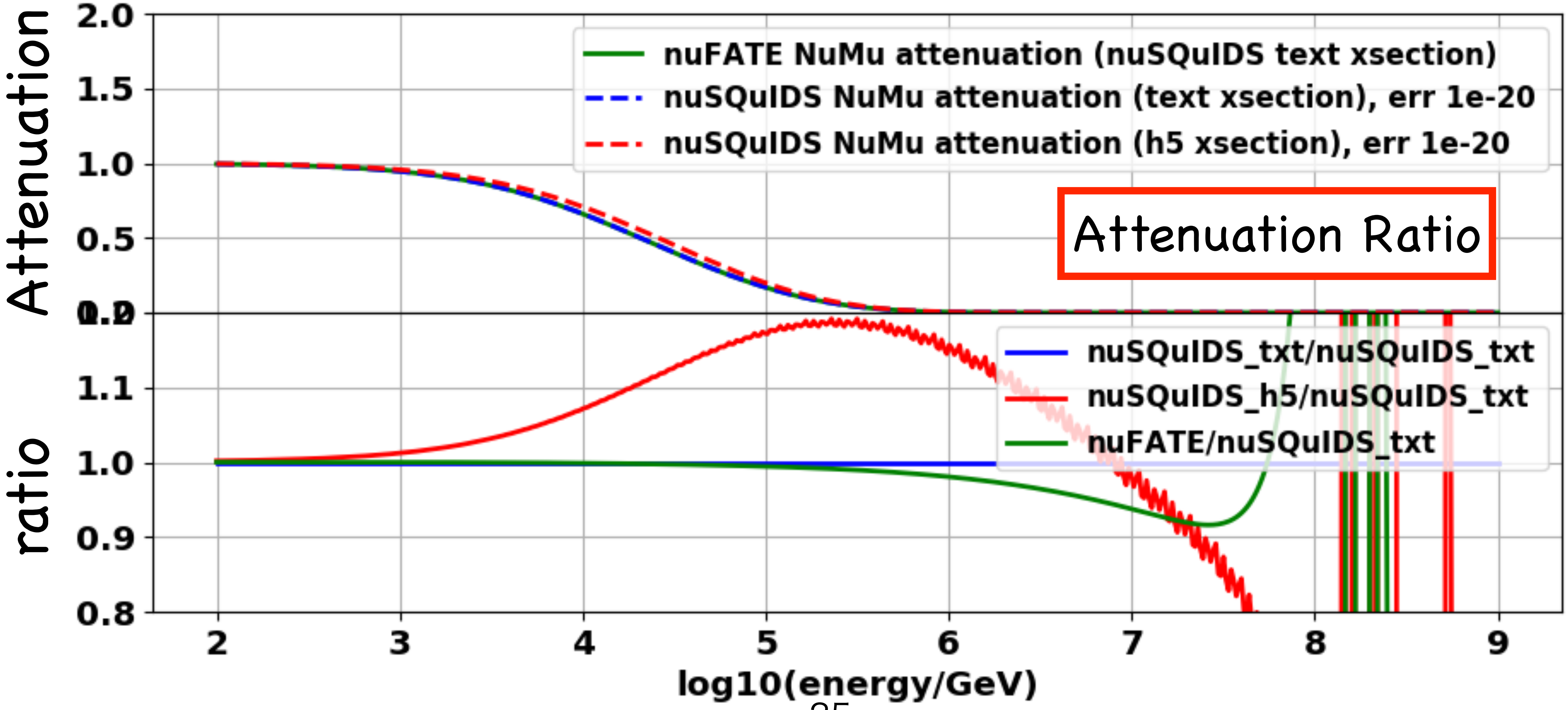
- ✦ nuSQuIDS with CSMS gives 20%~50% lower arrival flux than NuGen's CSMS simulation at 1PeV, 180 degree ?!
- ✦ <u>May not be visible for atmospheric spectrum </u>(see page13)
- ✦ Could it be serious for high-energy end? (need investigation)

E-1 Arrival Flux



More remarks for nuSQuIDS and nuFATE

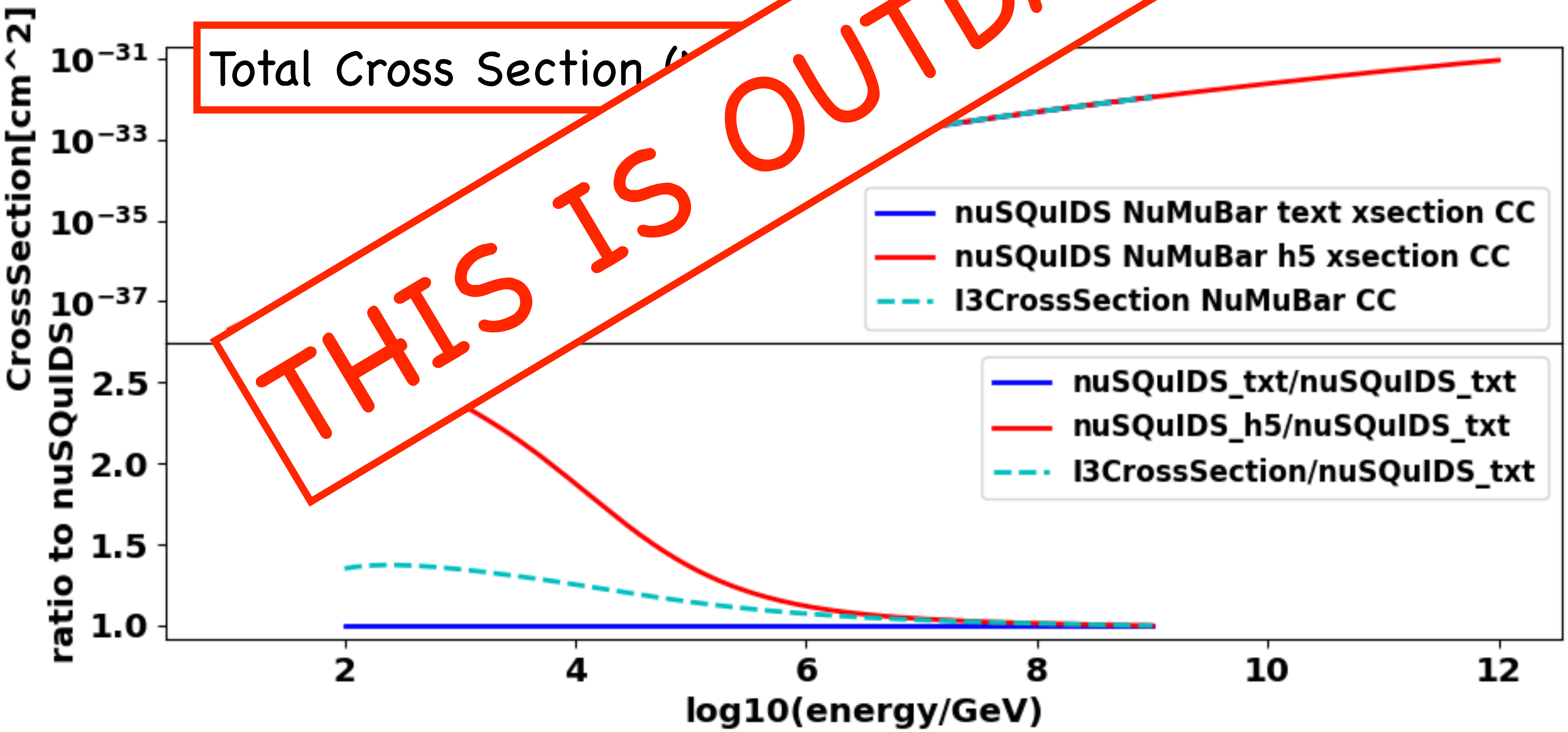
- nuFATE treats isoscalar Earth only, may affects Grashow interaction
- nuFATE seems to have binning effect? (compare blue and green)
- nuSQuIDS is sensitive to setting parameters (see page 24-26)
- However, absolute values of all differences are small and may not be an issue. We need check these behaviors for lower zenith angle above 1PeV.



Answer: Need new cross section module!

- I3CrossSection(.fits) and nuSQuIDS cross sections (.h5 and text files) are supposed to be generated from the same CSMS program
- However, even total cross sections didn't ma easy way to compare differential cross sections. We le to guarantee the identity of cross section files.

THIS IS OUTDATED



Topics skipped today

- ✦ How to run SimpleGenWeight
- ✦ How to write Text GenInfo File
- ✦ How to prepare pre-calculated nuSQuIDS file
- ✦ How to calculate Earth's column depth
- ✦ ….

To learn more, checkout my example project!

https://github.com/kotoyo/NuWeighter

this may be good to read

https://events.icecube.wisc.edu/getFile.py/access?contribId=108&sessionId=13&resId=0&materialId=slides&confId=100

Summary

- Introduced how to weighting DetectorMode simulations(LeptonInjector or NuGen Detector Mode) with SimpleGenWeight, a full python-based weighter.
- Weighting DetectorMode sim with a static primary flux is simple and works fine. Changes in existing analysis script is minimal (if you use SimpleGenWeight).
- Weighting DetectorMode sim with dynamic primary fluxes (in fitting loop) is not simple as we did with NuGen Full mode.
  - There are some second-best solutions, but may not be fast as legacy way and may require drastic change of analysis script (e.g. need to write C++ code from scratch, shift to GholemFit,…)
- Grashow resonance needs more works (for both NuGen and LI)
  - e.g. doesn't store number of electron-targets, nuFATE has some restrictions, etc.

Remind: Summary of the last spring

NuGen Full Mode

- Simple enough to start using it today for beginners, won't be super wrong as long as using FULL Mode or NuMu Detector Mode
- Generation may be inefficient depending on your analysis.
- To include Grashow and Tau Regeneration, need to use simulations for all three flavors

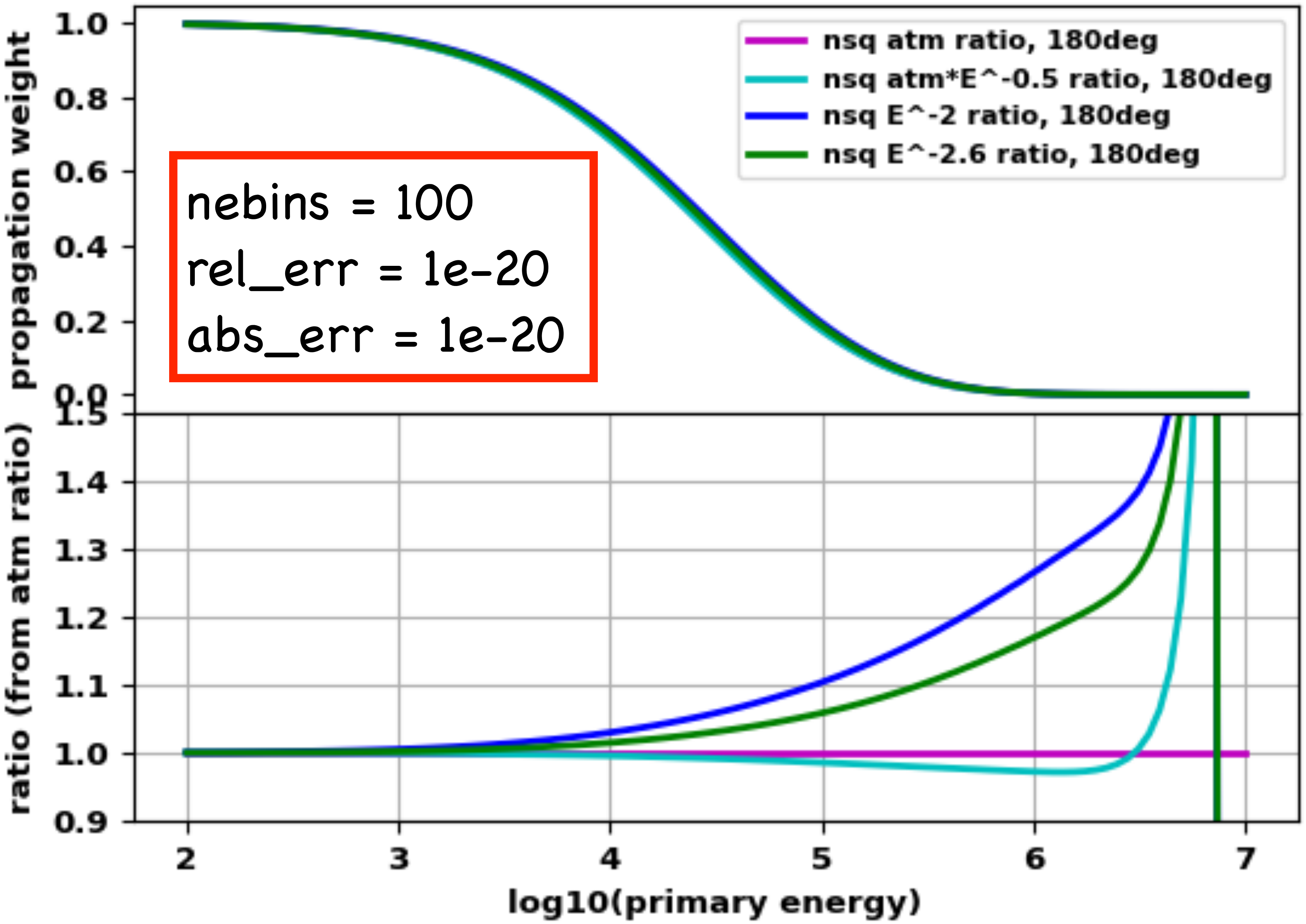
DetectorMode sim

- You have all controls
- Can generate simulations for target detection channel only, efficient for specific analysis
- You may get wrong answer if you don't know what you are doing
- You may need some exercises to learn how it works

nuSQuIDS : propagation weight

Propagation Weight = Flux_at_detector / Flux_at_Earth_surface
… will depend on shape of primary flux.

```
emin = 2
emax = 7
nebins = 100
ebins_eV = 1e9 *
np.logspace(emin,emax,nebins)

nsq_atm =
nsq.nuSQUIDSAtm(coszbins,
ebins_eV, 3,
nsq.NeutrinoType.both, True)

# set errors
# this is default value.
nsq_atm.Set_rel_error(1e-20)
nsq_atm.Set_abs_error(1e-20)
```

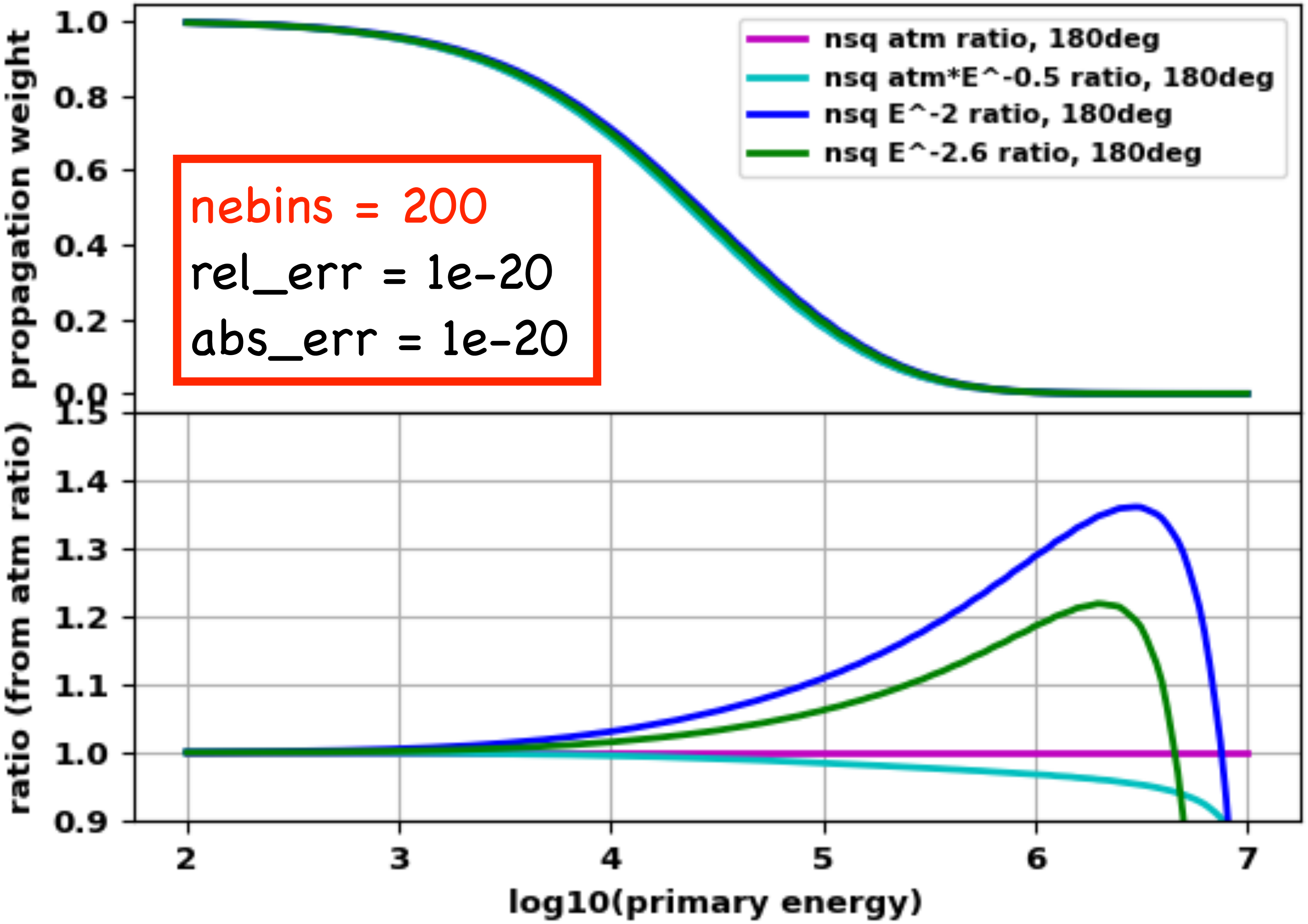


nebins = 100
rel_err = 1e-20
abs_err = 1e-20

nuSQuIDS : propagation weight

Modified Number of energy bins

```
emin = 2
emax = 7
nebins = 200
ebins_eV = 1e9 *
np.logspace(emin,emax,nebins)

nsq_atm =
nsq.nuSQUIDSAtm(coszbins,
ebins_eV, 3,
nsq.NeutrinoType.both, True)

# set errors
# this is default value.
nsq_atm.Set_rel_error(1e-20)
nsq_atm.Set_abs_error(1e-20)
```

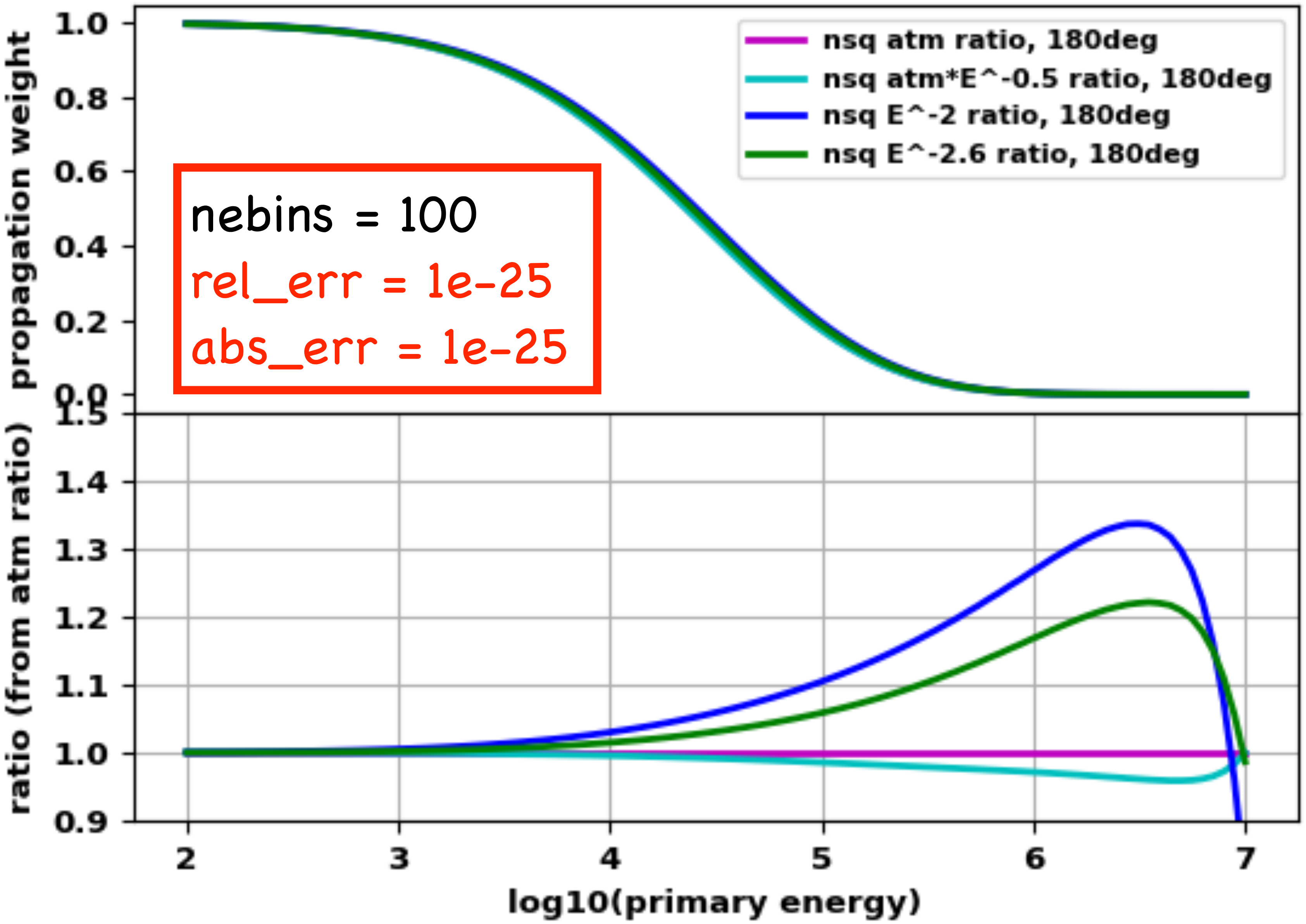


nebins = 200
rel_err = 1e-20
abs_err = 1e-20

nuSQuIDS : propagation weight

Modified relative and absolute errors

```
emin = 2
emax = 7
nebins = 100
ebins_eV = 1e9 *
np.logspace(emin,emax,nebins)

nsq_atm =
nsq.nuSQUIDSAtm(coszbins,
ebins_eV, 3,
nsq.NeutrinoType.both, True)

# set errors
nsq_atm.Set_rel_error(1e-25)
nsq_atm.Set_abs_error(1e-25)
```



nebins = 100
rel_err = 1e-25
abs_err = 1e-25