Software and documentation

Status and future plans

Akanksha Katil, Juan Pablo Yáñez

Arthur B. McDonald
Canadian Astroparticle Physics Research Institute

UNIVERSITY OF ALBERTA

Basic software needs

Simulation production

- Define a medium
- Define a detector geometry
- Generate neutrino interactions
- Propagate the resulting particles
- Produce photons
- Propagate the photons
- Mimic the response of the detector

Analysis

- Open data & MC files
- Explore contents
- Pull relevant information from them
- Reconstruct event properties
- Infer physical quantities

From previous experiences

IceCube's IceTray can solve many of these needs - already open source

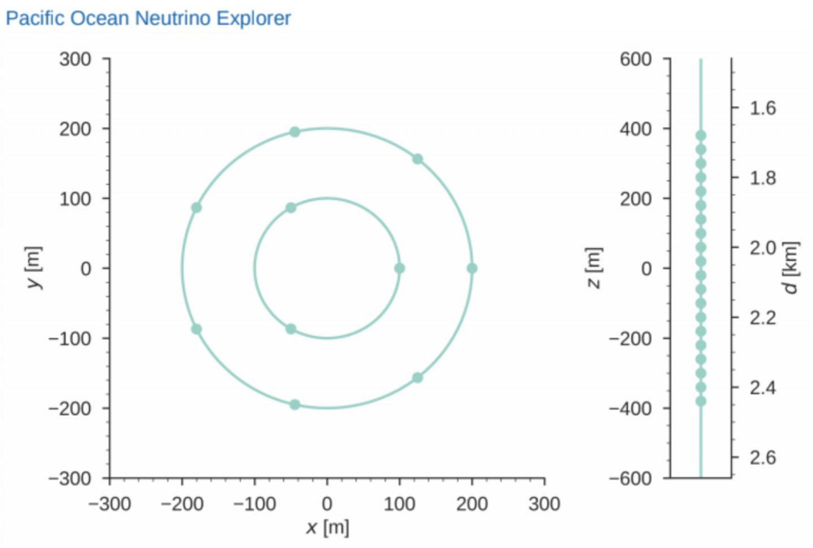
https://github.com/IceCubeOpenSource/offline_production

- Framework to simulate and store data from a neutrino telescope
- Core written in C++, users can access it and write code in python
- Pre-existing dataclasses and packages make it easy to use

Used for the reconstruction and tau search studies (with some differences)

The Simulation Chain

- Requires a geometry definition
- Pick an event generator
  - NuGen not part of open source package
  - Dev's of new "Lepton Injector" could be persuaded to make it open source



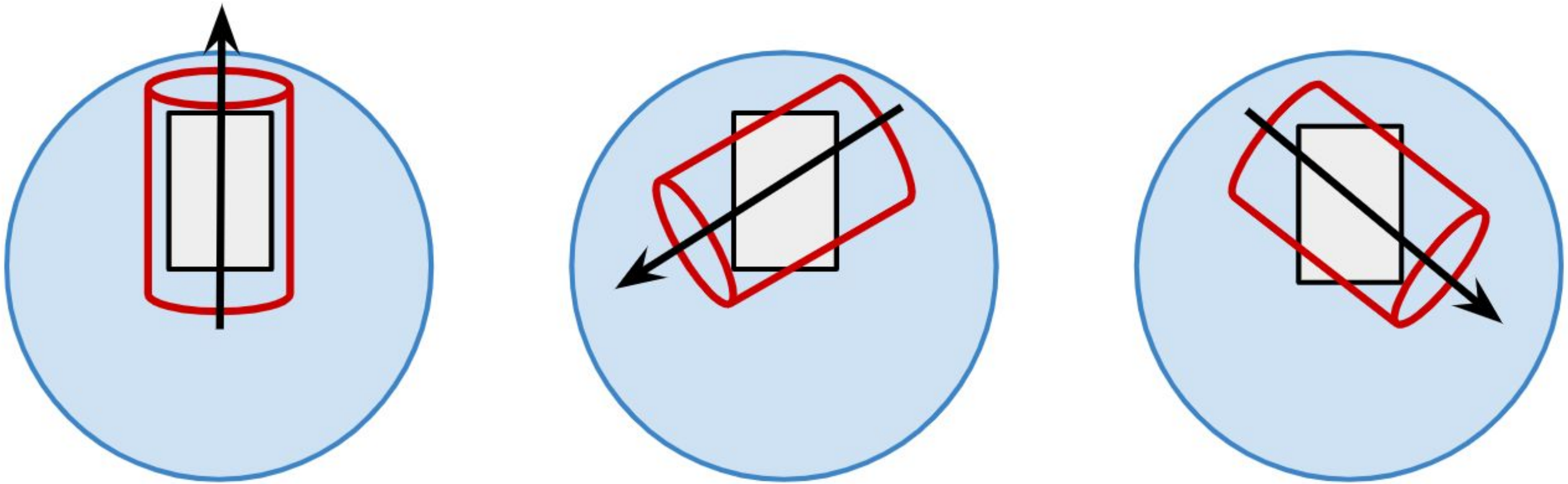
Neutrino Generator(NuGen) → **PROPOSAL** → **CLSim** → **PE Hit Generator** → **Noise Generator** → **RecoPulse Generator**

IceCube Modules (yellow)

Modules developed for P-ONE (green)

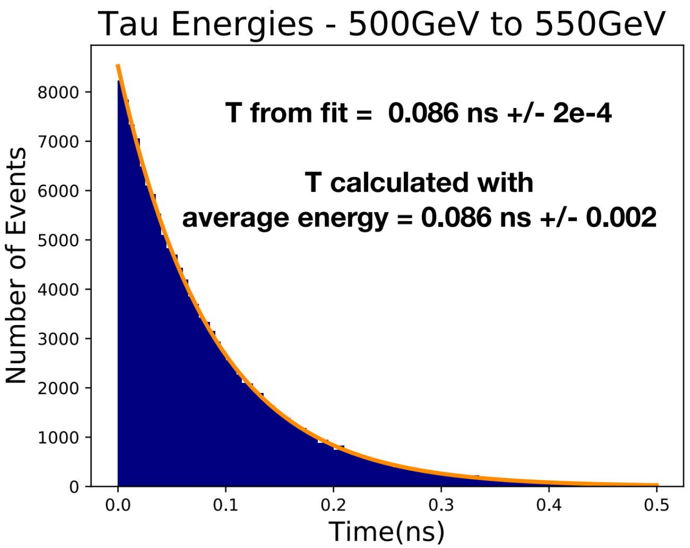
Neutrino Generator (NuGen)

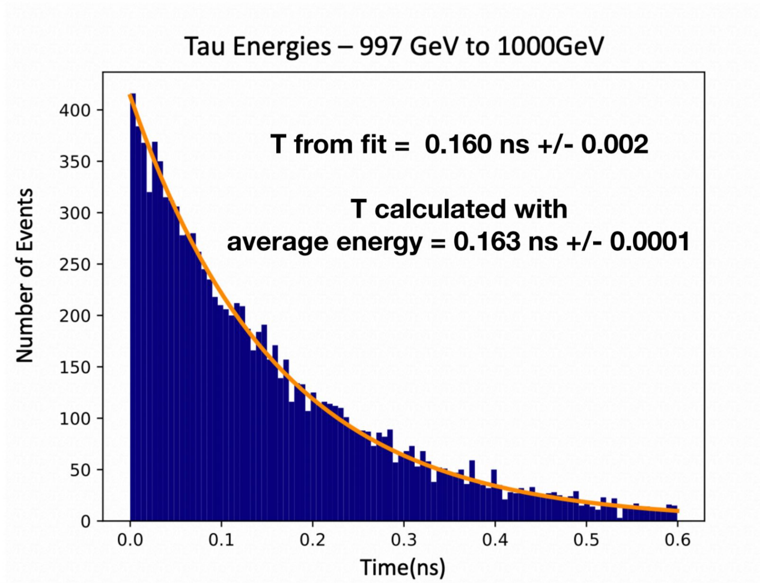
- Injected neutrinos are forced to interact within a given cylinder volume.
- Works well at neutrino energies above 100 GeV



PROPOSAL

- Propagates muon and tau leptons
- Medium of propagation not changed to water since the difference is negligible
- Side task: verified τ propagation





CLSim - Changing the Medium

- Generate and propagate photons.
- To propagate photons in CLSim, medium should be changed to water.
- The optical properties in clsim are defined by the following equations

$$b_e = b_{e,400}(\frac{\lambda}{400})^{-\alpha}$$

$$a = a_{dust,400}(\frac{\lambda}{400})^{-\kappa} + \boxed{Ae^{-B/\lambda}(1 + 0.01\Delta T)}$$

→ Ignored, need more data

b_e - effective scattering coefficient at a given wavelength λ

$b_{e,400}$ - scattering coefficient at wavelength 400 nm

a - absorption coefficient at a given wavelength λ

$a_{dust,400}$ - absorption coefficient at wavelength 400 nm

Exponential component of wavelength dependence is ignored

Effective Scattering Length

- The simulation takes effective scattering length.

$$\lambda_{\text{sct}}^{\text{eff}} \equiv \frac{\lambda_{\text{sct}}}{1 - \langle \cos\theta \rangle}$$

$$\langle \cos\theta \rangle = \eta \cdot \langle \cos\theta \rangle_{molecular} + (1-\eta) \cdot \langle \cos\theta \rangle_{particulate}$$

- $\langle \cos\theta \rangle_{\text{molecular}} = 0$
- $\langle \cos\theta \rangle_{\text{particulate}} = 0.924$ (from Antares paper)
- $\eta = 0.132$ (Matthew Man's Analysis)

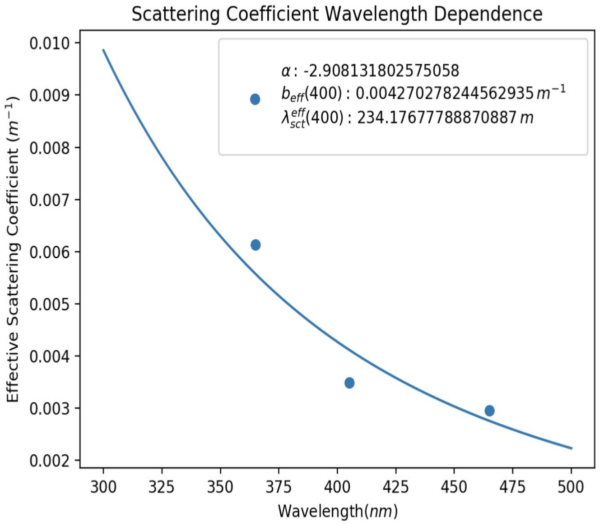
Wavelength(nm)	Scattering Length(m)	Effective Scattering Length(m)	Absorption Length(m)
365	32.30	163.16	9.21
405	56.78	286.81	17.56
465	66.87	337.78	31.87

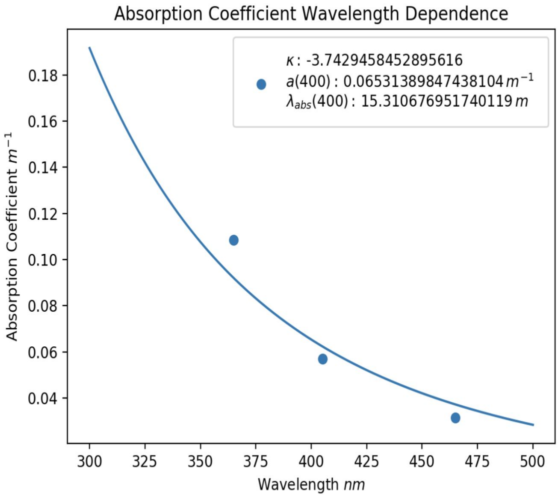
Note: Absorption and scattering length taken from Andreas Gaertner's analysis. Analysis still in progress

- Current STRAW analysis considers equal probability for forward and backward scattering.
- For a more correct approach η(fraction of molecular scattering) is included.

Determining Parameters

- The simulation uses scattering coefficient($b_{e,400}$), α , absorption coefficient($a_{e,400}$) and κ obtained from the fit of the data points.

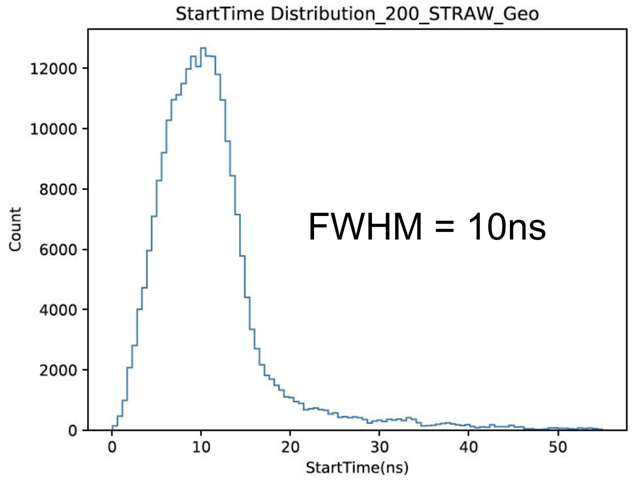


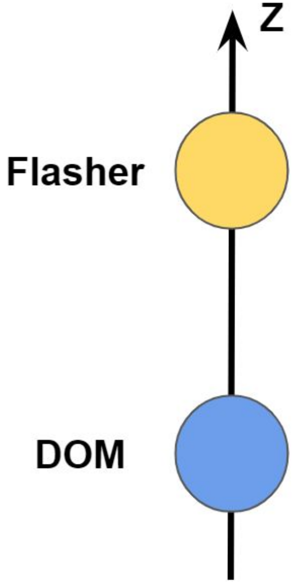


- Phase function in software defined by Simplified Liu(SL) and Henyey Greenstein(HG).
- Phase function of water closer to HG, thus SL ignored by setting f_{sl} = 0

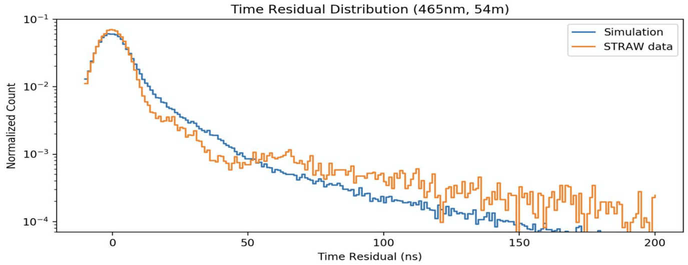
Verification: Simulating Flashers

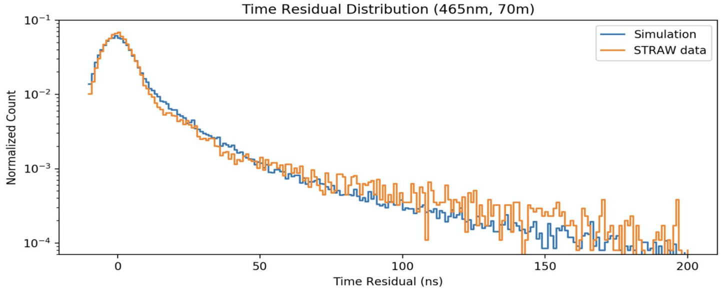
- Goal: compare output to STRAW data
- Simulating multiple flashers, spaced equally in cosine of zenith, for isotropic flashes.
- DOM position is fixed and flasher position is changed

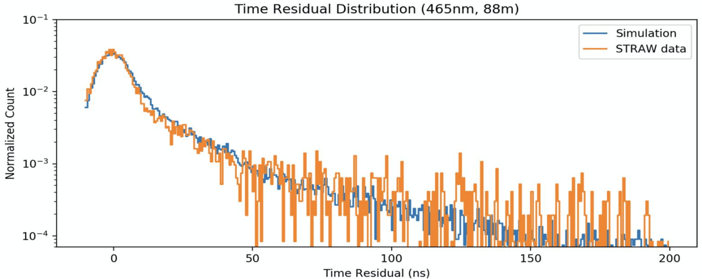


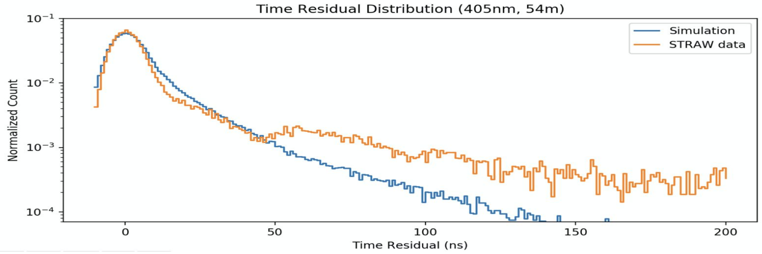


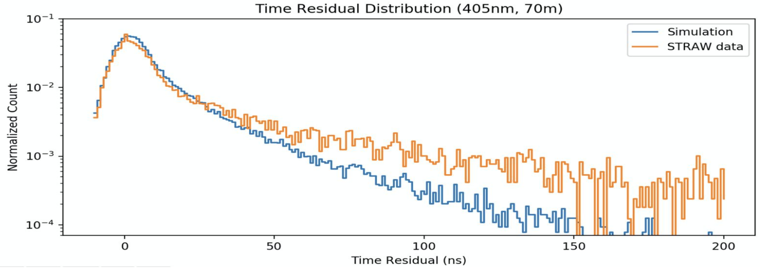
Time Residuals

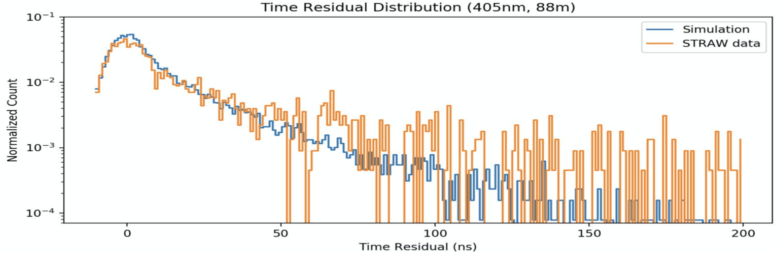


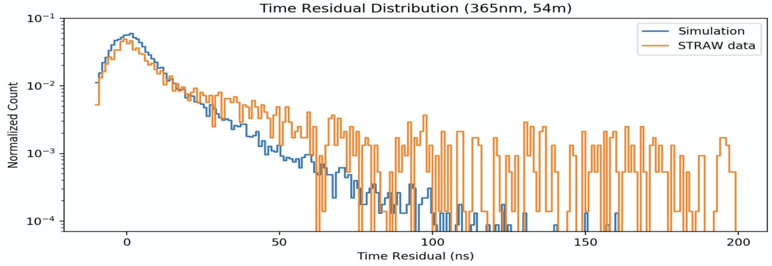








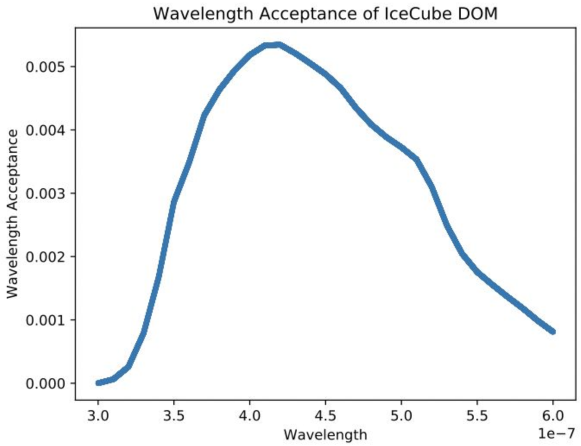


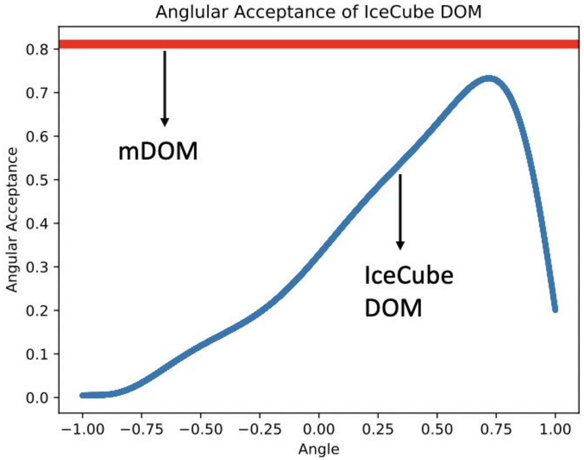


Great tool to test the inputs from STRAW analysis.

PE Hit Generator - custom mDOM

Probability of Hit = DOM eff × ang Acc × photon weight

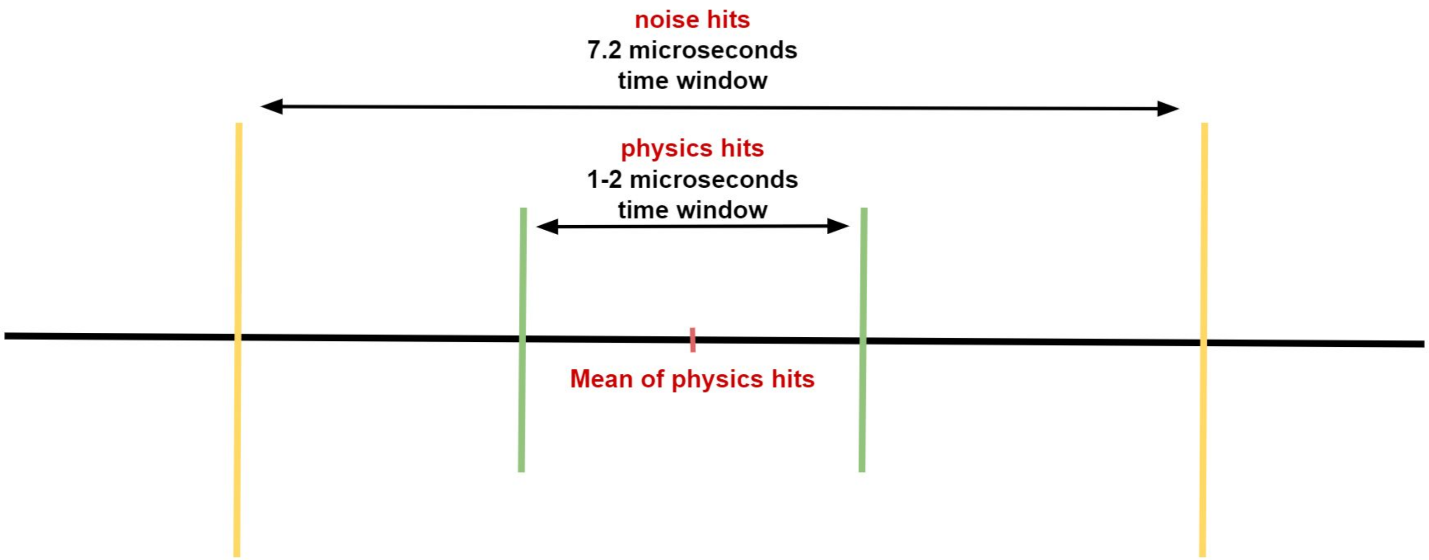


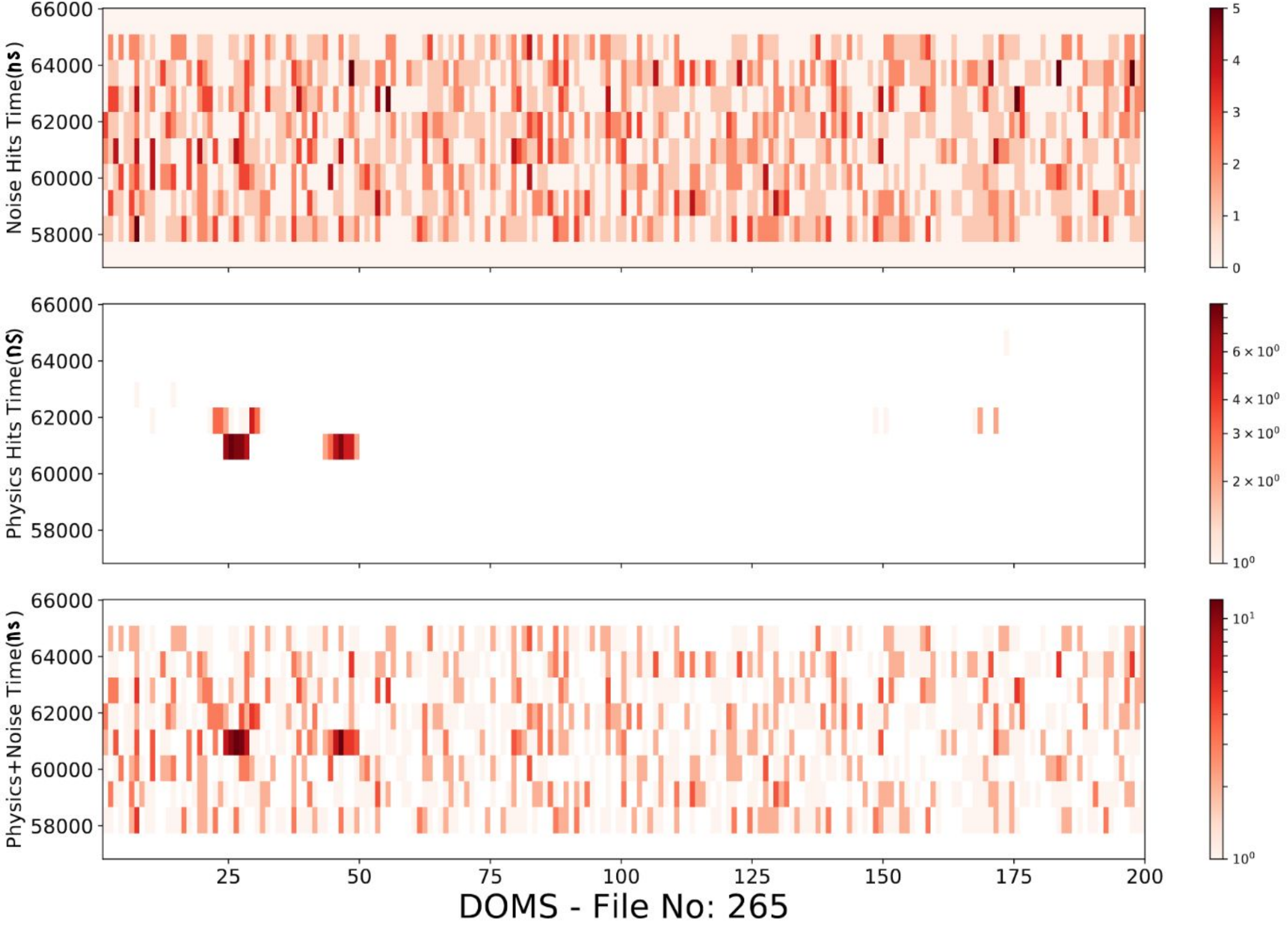


- Total acceptance for mDOM is 2.4x IceCube due to the photocathode area
- As of now, no granularity implemented. All hits are merged.

Noise Generator

- Injecting noise from STRAW data on event windows of 7.2us
- Since noise scales with photocathode area, taken 24 random chunks of 7.2us overlaid on each other, to generate noise in mDOM




Noise Hits Time(ns)
Physics Hits Time(ns)
Physics+Noise Time(ns)
66000
64000
62000
60000
58000
25
50
75
100
125
150
175
200
DOMS - File No: 265

RecoPulse Generator

- Precision and features of readout to be designed
- Using educated guesses from IceCube DAQ

- Simple simulation of pulses from the photoelectron hits
  - Hits within 3ns are merged as one
  - Charges are summed, need to be smeared still
- Random normal fluctuations added with 1.5ns width
- Logic and output could be reviewed

Output

- Files in "I3 Format"
  - Require 'offline processing' software to open
  - Can access all the information from the simulation
- Tables
  - IceTray-indepedent format
  - Only needs python/ROOT*

To get going

- Github repository: https://github.com/pone-software
  - Private. Happy to add local managers to add people.
  - Code already there:
    - Geant4 simulations (SFU)
    - Coincidence analysis (SFU)
    - Muon track reconstruction (Queens)
- Plan
  - Fork the offline_processing project from IceCube
  - Get an open source neutrino generator
  - Move simulation chain to repo (Katil, UofA)
  - Move tau analysis to repo (Katil, UofA)

Analysis code and documentation

Move into GitHub P-ONE repository (if P-ONE)

- You maintain ownership
- Makes it easy for others to use and look at

Code documentation (only code)

- Proposal: github pages
- Markdown syntax, living next to the code, hosted by github directly

Open software tasks

- Move to PMT hit in DOM
  - Needs some decision on module design
  - Information could turn out to be really useful
  - Software already has the capability of storing hits for multiple PMTs in a DOM
- Implement real correlated noise
  - Right now, noise pattern is entirely uncorrelated
  - Fine for tau study, but a better model is needed
  - Use Geant4 K40 + bioluminescence to produce conditional PDFs for PMTs in DOM?
- More realistic response, readout (time & charge)
  - Needs some design choices
- A trigger condition
  - Needs some design choices
- An event viewer?