Neutrino and Air Shower Simulations in IceCube

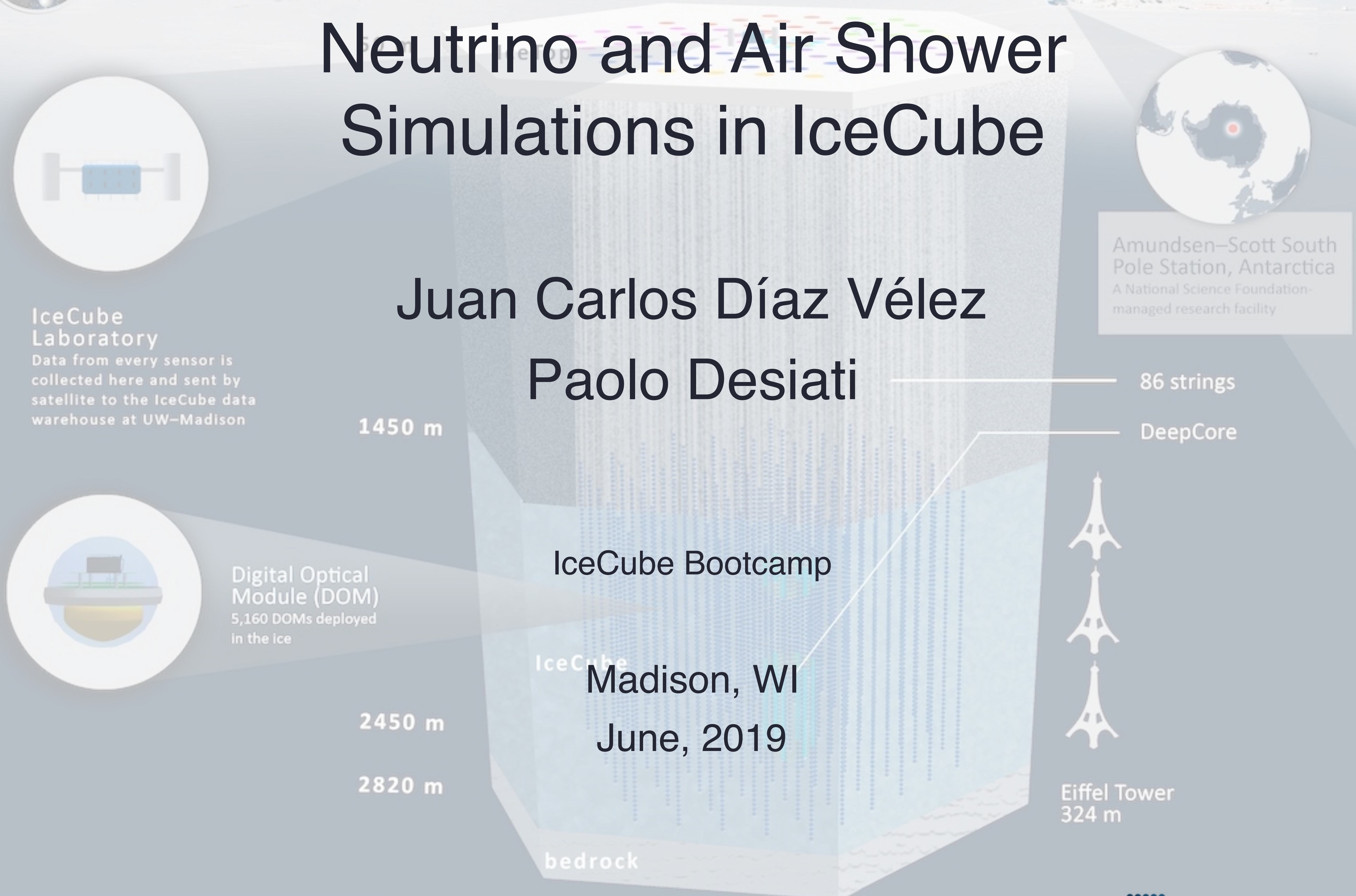
Juan Carlos Díaz Vélez

Paolo Desiati

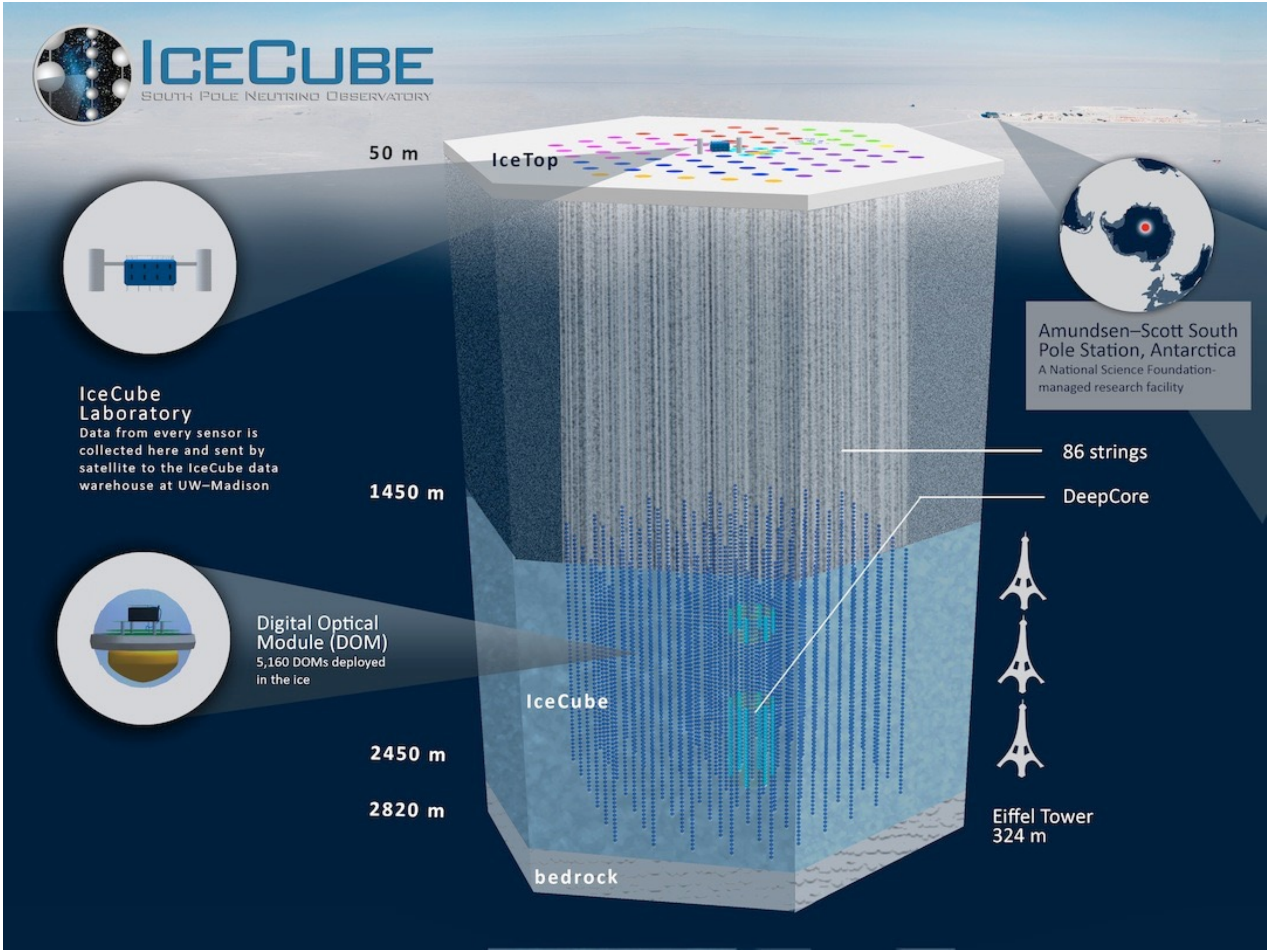
IceCube Bootcamp

Madison, WI

June, 2019

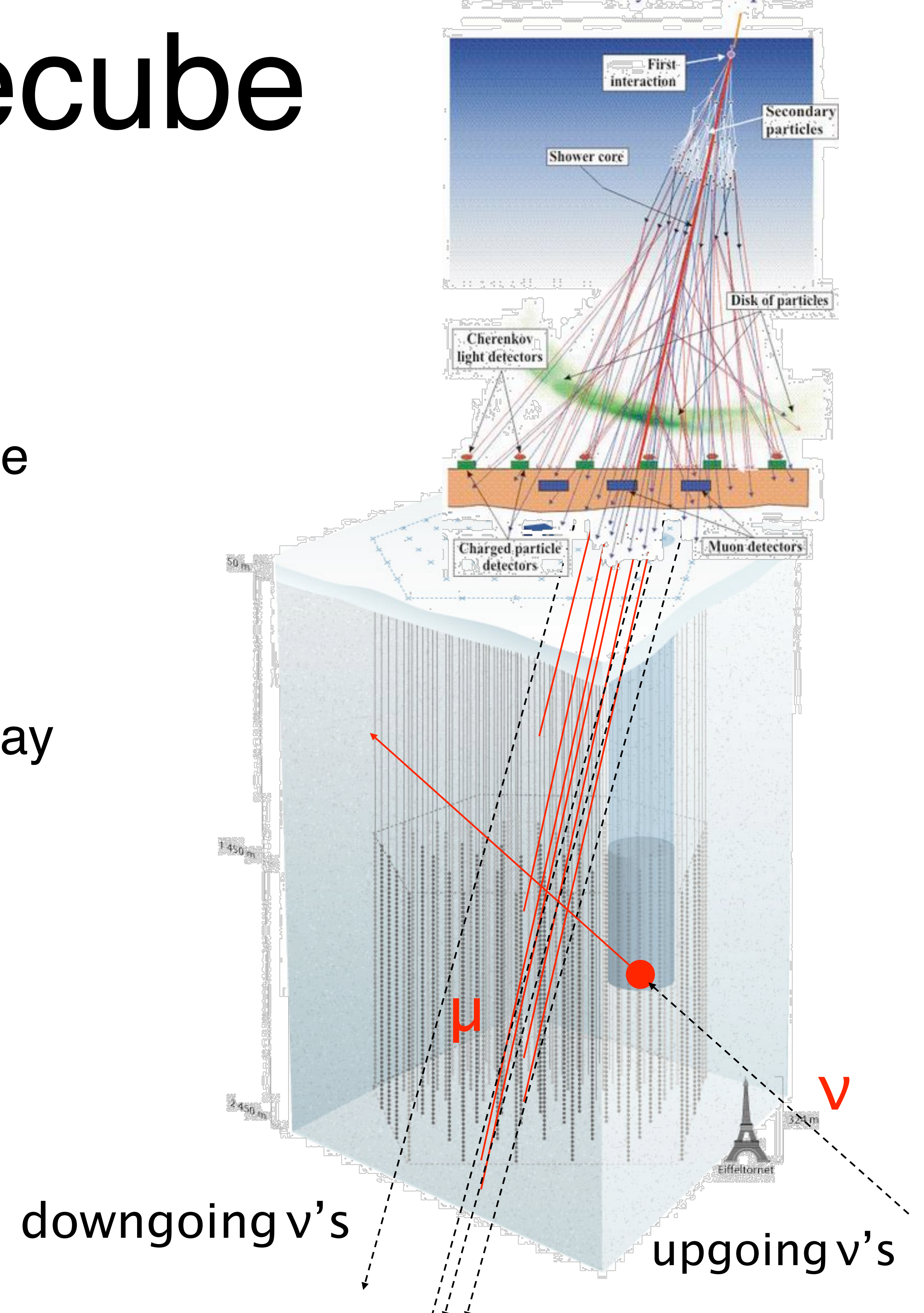


The IceCube Observatory

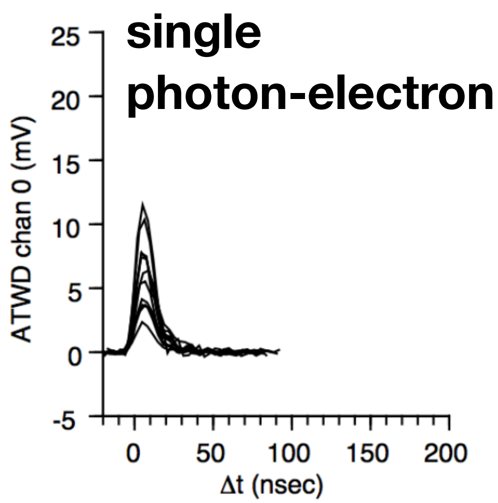


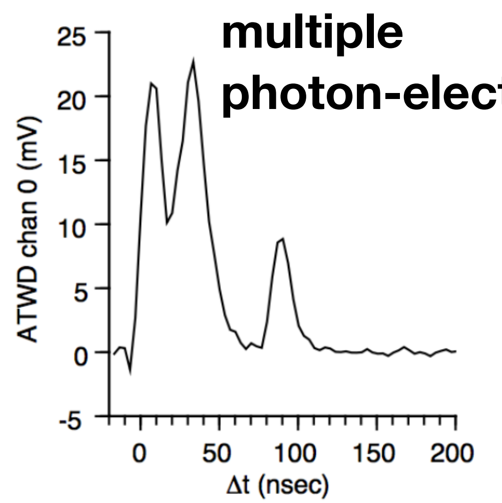
Events in icecube

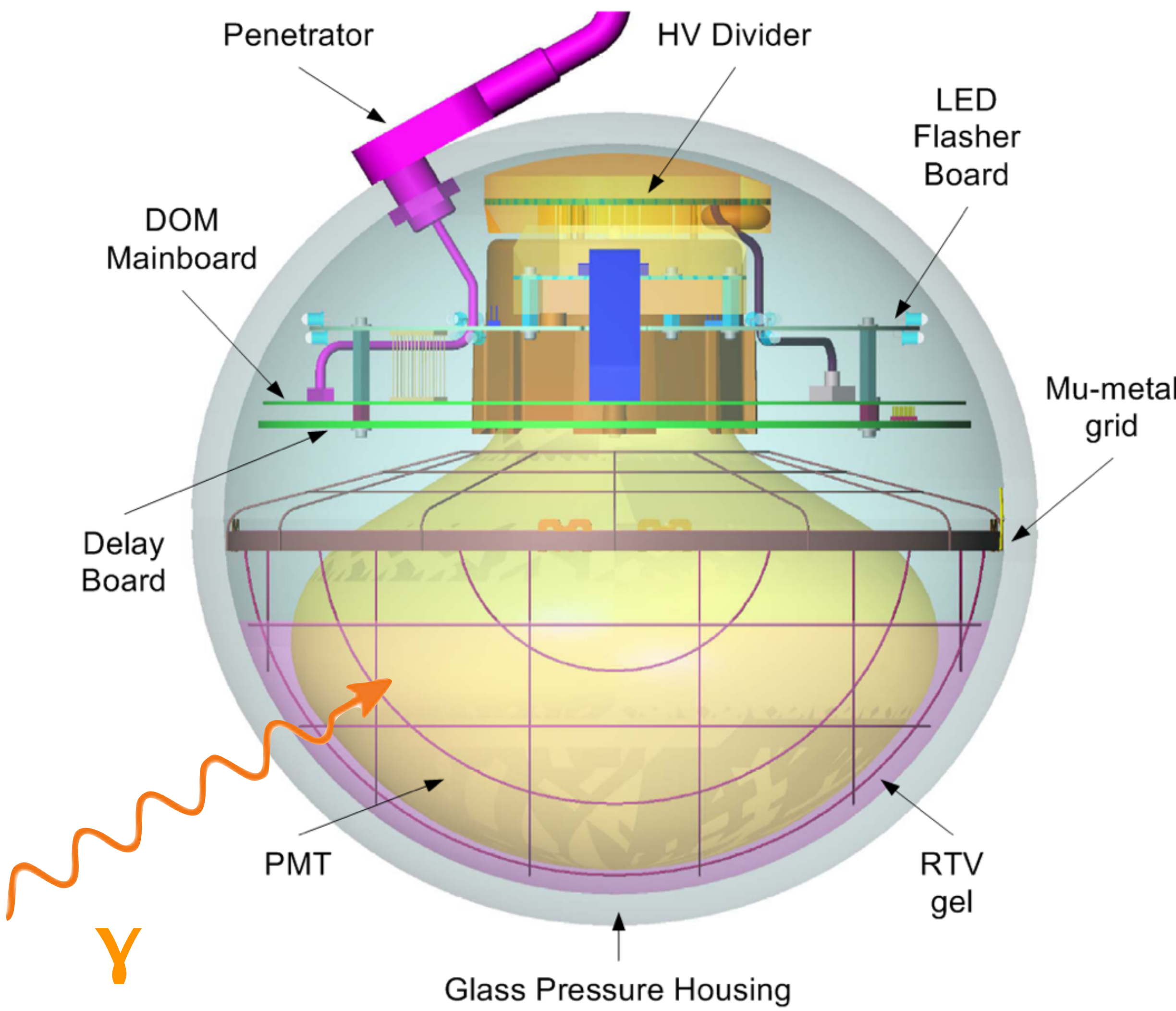
- Air shower detection @ surface
- Penetrating muon detection in deep ice
- Events dominated by cosmic ray muons : 10^6 μ for every ν that interacts in IceCube
- Atmospheric ν's



Digital Optical Module
the signal digitization

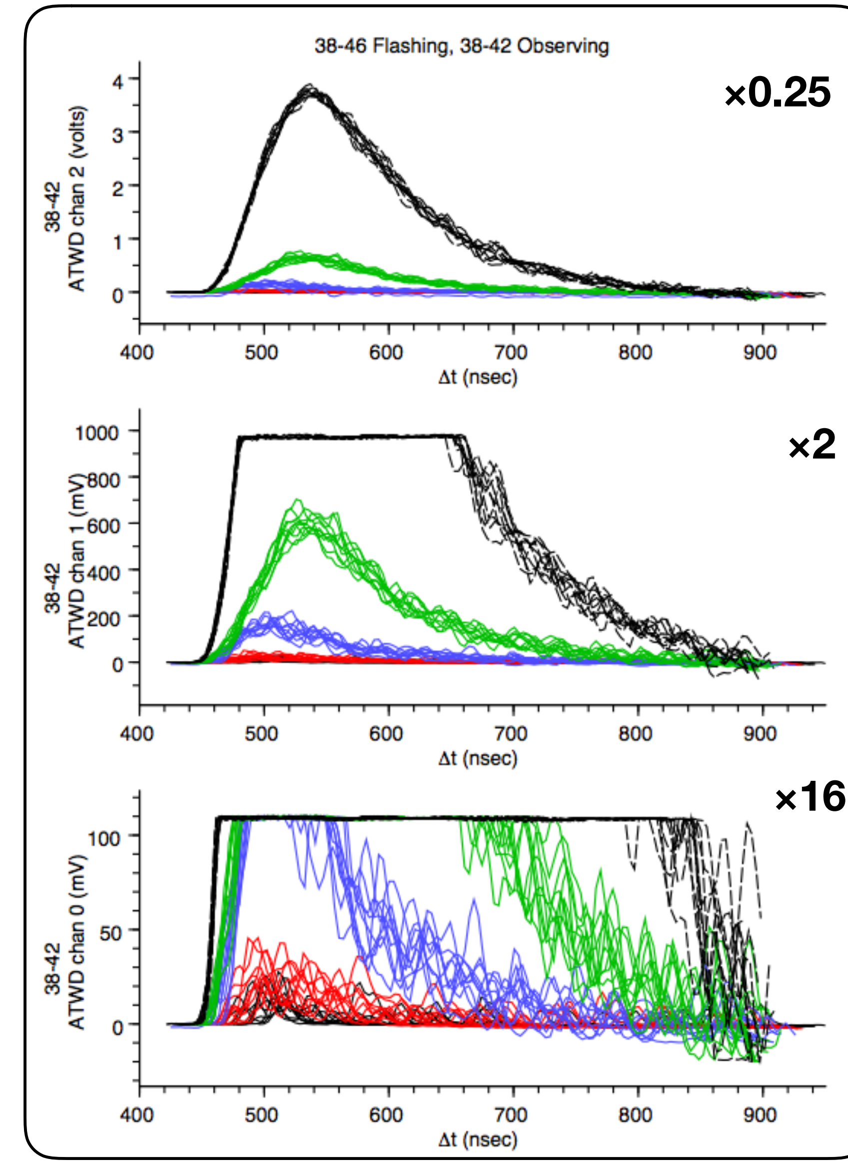






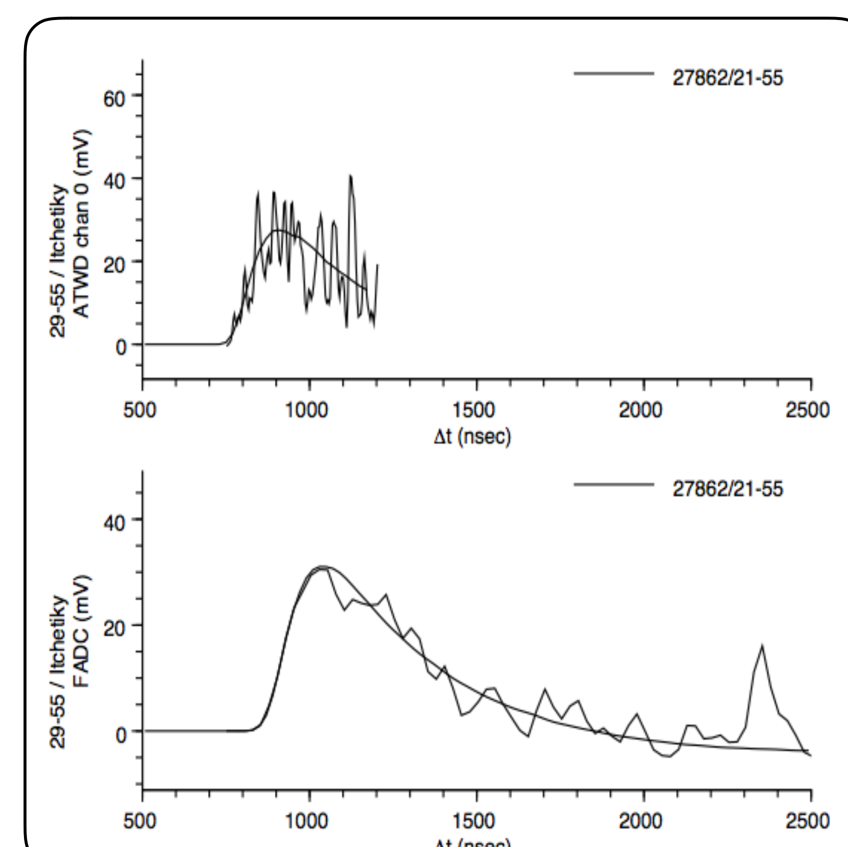
different gains

for wide dynamic range
in amplitude



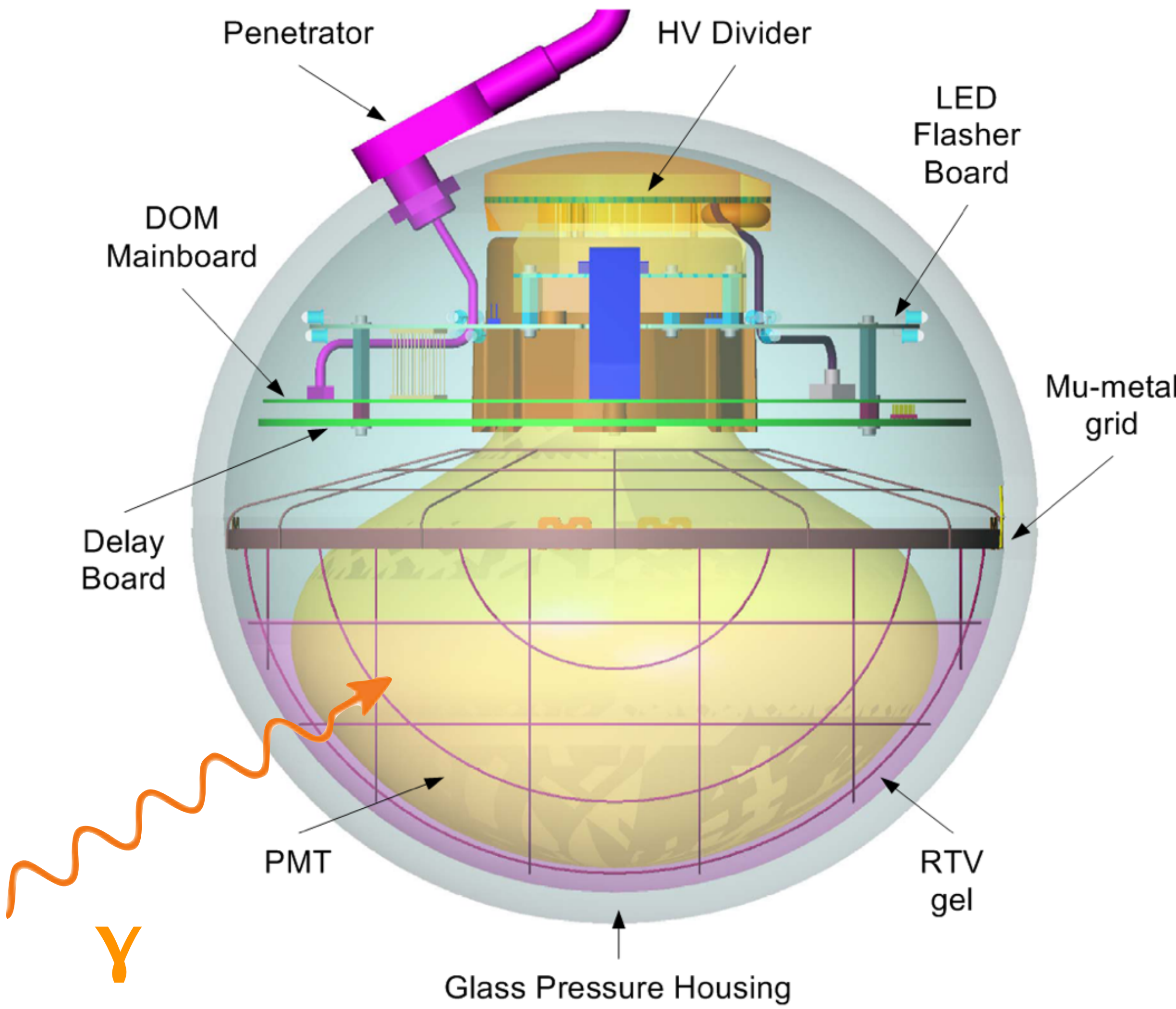
different time range

for long waveforms

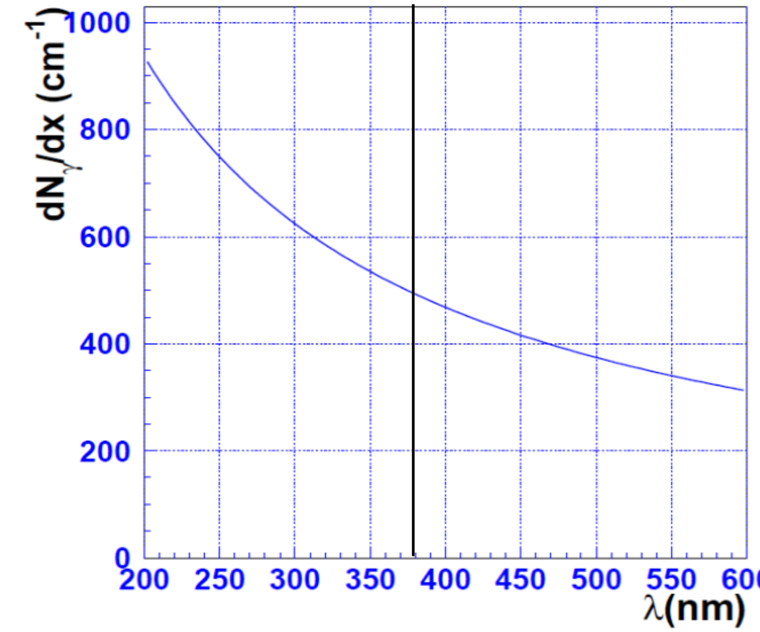


Digital Optical Module

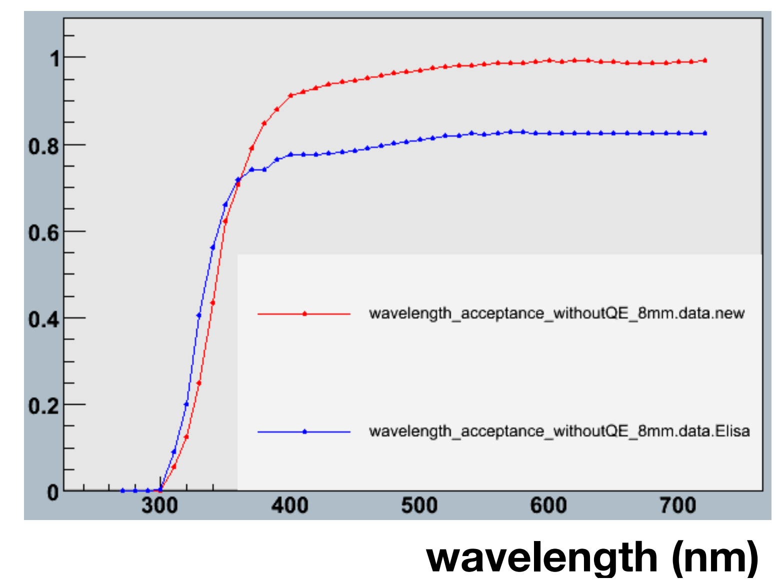
the photon sensitivity



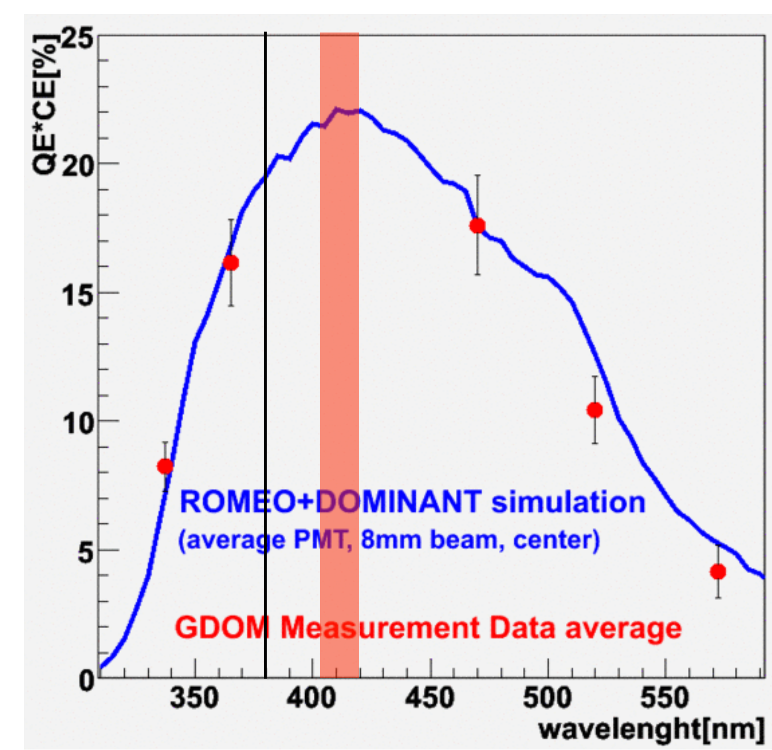
Cherenkov photon wavelength spectrum

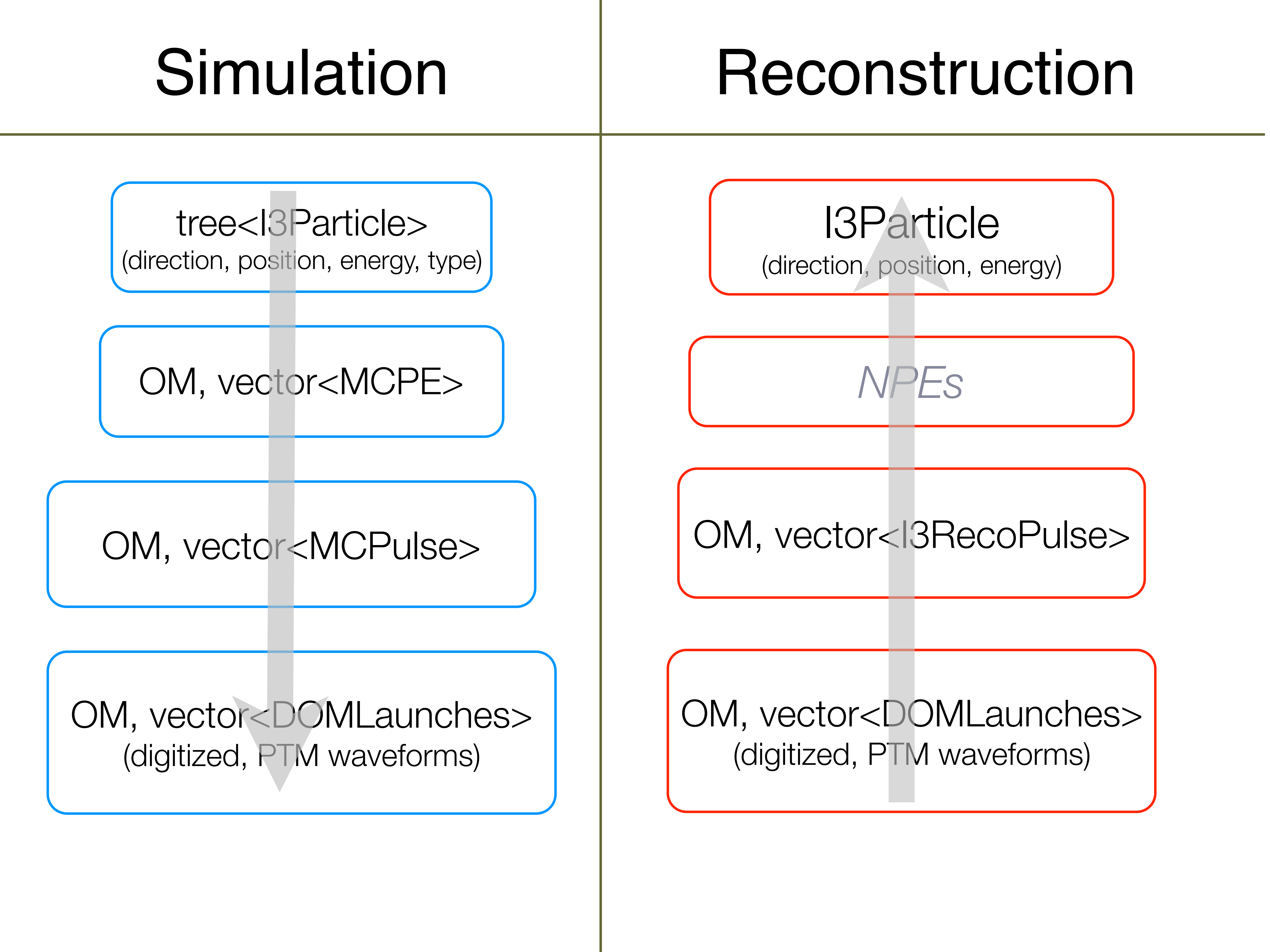


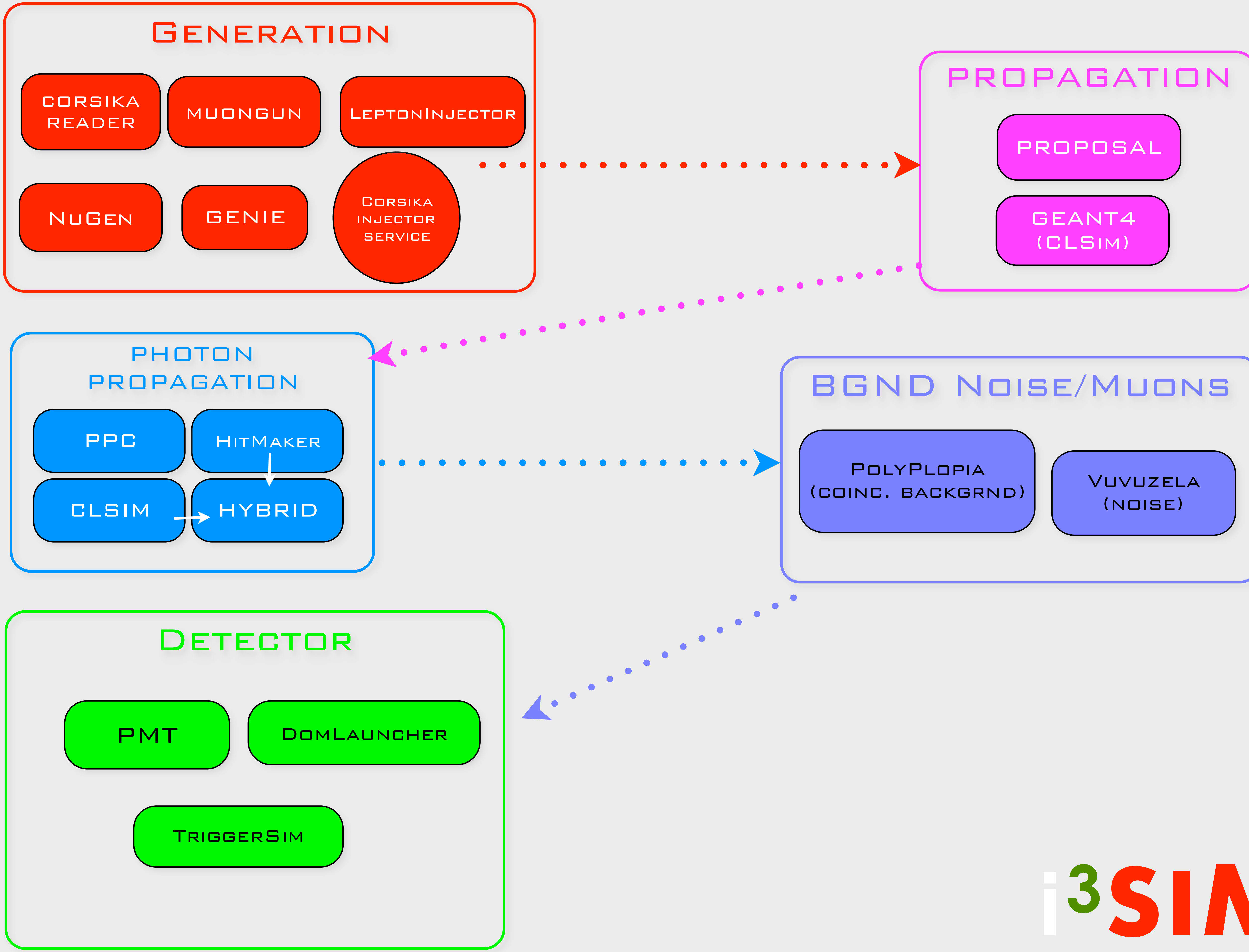
glass/gel transmittance low wavelength cut-off



PMT Quantum Efficiency × Collection Efficiency

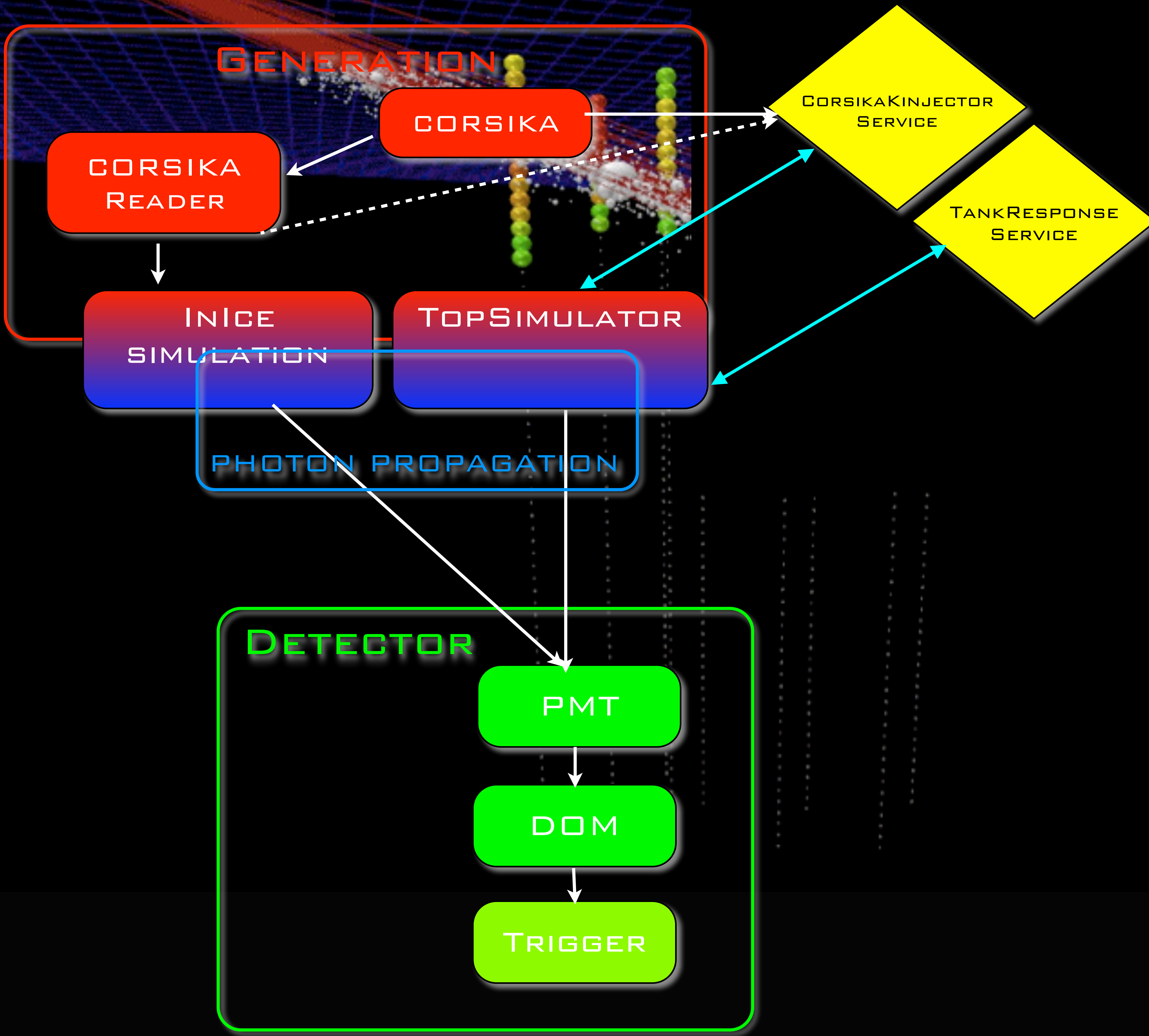



Simulation
Reconstruction
tree<I3Particle>
(direction, position, energy, type)
OM, vector<MCPE>
OM, vector<MCPulse>
OM, vector<DOMLaunches>
(digitized, PTM waveforms)
I3Particle
(direction, position, energy)
NPEs
OM, vector<I3RecoPulse>
OM, vector<DOMLaunches>
(digitized, PTM waveforms)


Generation
CORSIKA READER
MUONGUN
LeptonInjector
NuGen
GENIE
Corsika injector service
PROPAGATION
PROPOSAL
GEANT4 (CLSim)
PHOTON PROPAGATION
PPC
HitMaker
CLSIM
HYBRID
BGND Noise/Muons
PolyPlopia (coinc. backgrnd)
Vuvuzela (noise)
Detector
PMT
DomLauncher
TriggerSim

i^3SIM

simulaton chain (IT)



Generators

- Cosmic-ray Air Showers:
  - **CORSIKA** (FORTRAN stand-alone)
  - **corsika-reader**: IceTray reader for standard format
  - **CorsikaInjectorService** (IceTop)
- Muons:
  - **MuonGun**: parametrization of flux of atm. muons under the ice.
- Neutrinos:
  - **neutrino-generator**: injects neutrinos, propagates them through Earth, forces interaction in detector volume.
  - **genie-icetray**: detailed simulation of neutrino interactions with GENIE. (Used for low-energy simulations)
  - **LeptonInjector / NuFSGen** (not yet available): weighted leptons+weights to account for flux models, interaction models, in-earth propagation, etc.

Generators

- Cosmic-ray Air Showers:
  - **CORSIKA** (FORTRAN stand-alone)
  - **corsika-reader**: IceTray reader for standard format
  - **CorsikaInjectorService** (IceTop)
- Muons:
  - **MuonGun**: parametrization of flux of atm. muons under the ice.
- Neutrinos:
  - **neutrino-generator**: injects neutrinos, propagates them through Earth, forces interaction in detector volume.
  - **genie-icetray**: detailed simulation of neutrino interactions with GENIE. (Used for low-energy simulations)
  - **LeptonInjector / NuFSGen**: weighted leptons+weights to account for flux models, interaction models, in-earth propagation, etc.

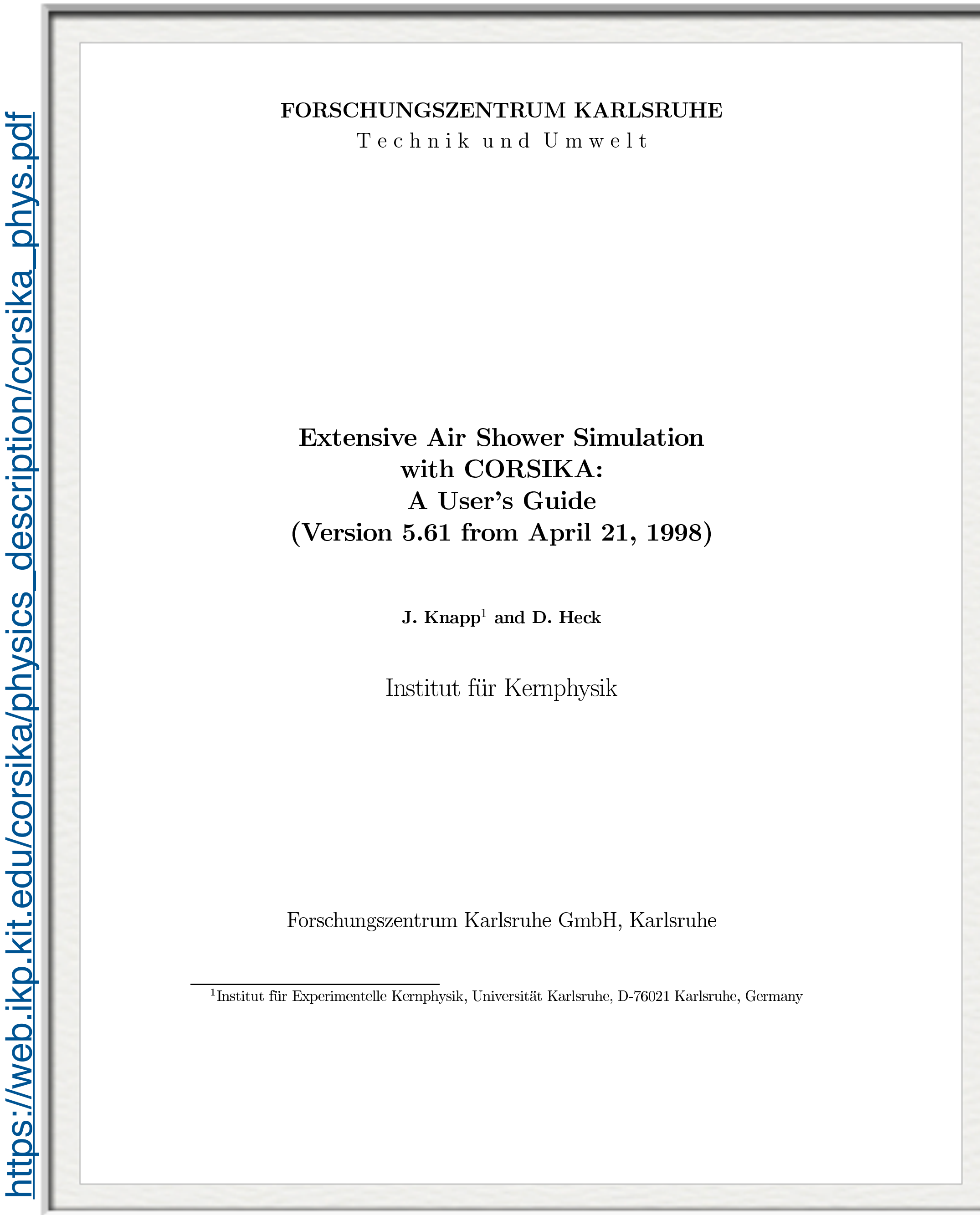
Generators (cont.)

- Other:
  - **wimpsim-reader**: IceTray interface for WimpSim (FORTRAN stand-alone)

Generators : CORSIKA

(**CO**smic **R**ay **SI**mulations for **KA**scade)

- Particles are tracked through the atmosphere until they undergo reactions with the air nuclei or - in the case of instable secondaries - decay.
- The hadronic interactions at high energies may be described by several reaction models alternatively:
  - *VENUS*, *QGSJET*, and *DPMJET* (Gribov-Regge theory),
  - SIBYLL (minijet model).
  - *neXus*, *EPOS* (combination of *QGSJET* and *VENUS*).
  - *HDPM* (Dual Parton Model).
- Hadronic interactions at lower energies:
  - FLUKA, *GHEISHA*, or *UrQMD* models.
- For electromagnetic interactions
  - Tailored version of *EGS4*.
  - Analytical *NKG* (Nishimura-Kamata-Greisen) formulas.



FORSCHUNGSZENTRUM KARLSRUHE
Technik und Umwelt

Extensive Air Shower Simulation with CORSIKA: A User's Guide (Version 5.61 from April 21, 1998)

J. Knapp[1] and D. Heck

Institut für Kernphysik

Forschungszentrum Karlsruhe GmbH, Karlsruhe

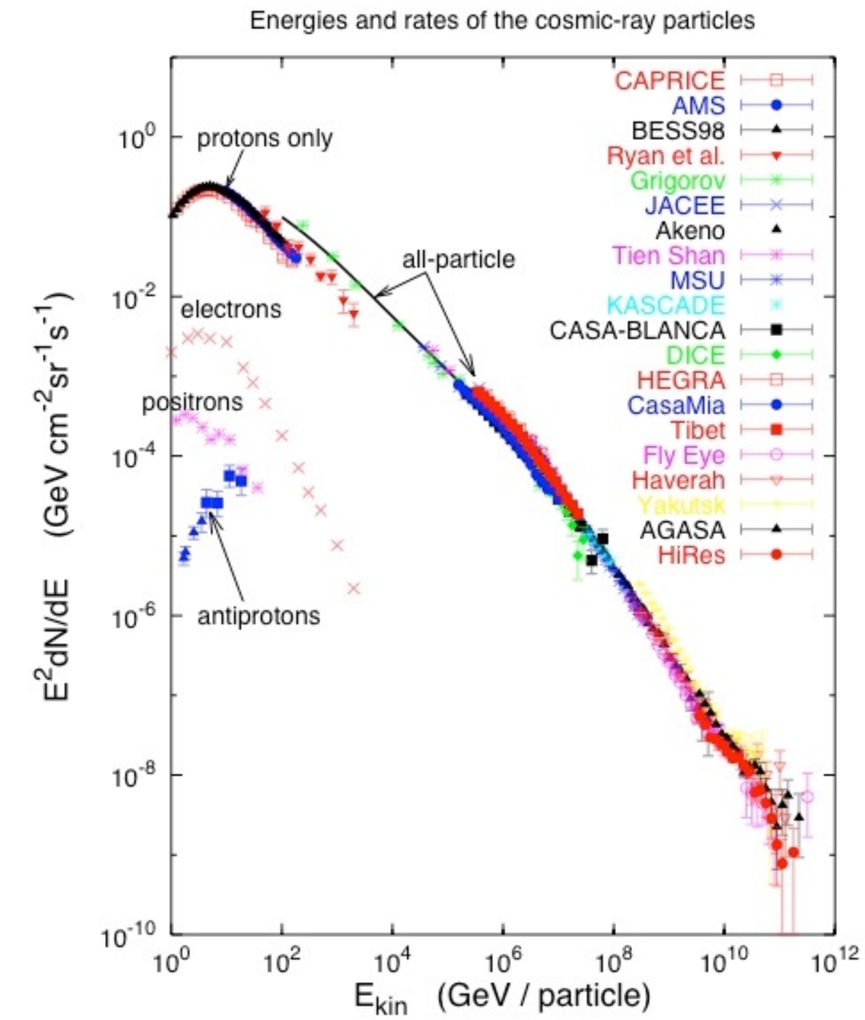
[1]Institut für Experimentelle Kernphysik, Universität Karlsruhe, D-76021 Karlsruhe, Germany

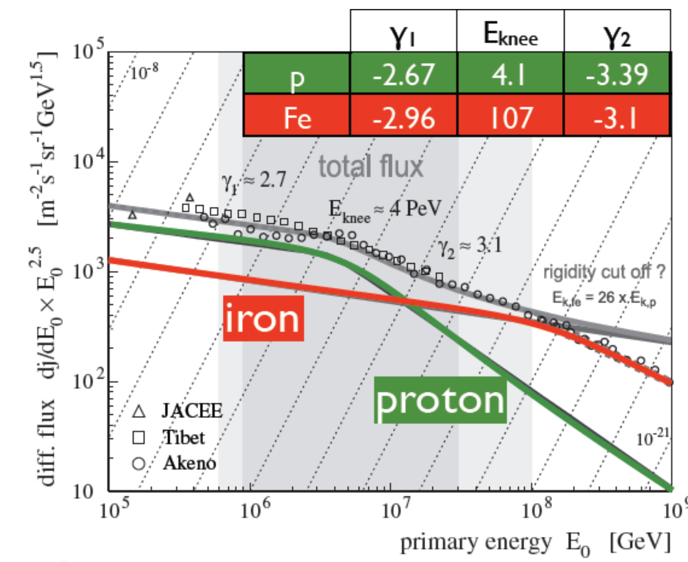
https://web.ikp.kit.edu/corsika/physics_description/corsika_phys.pdf

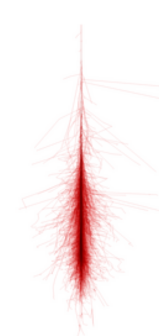
Generators : CORSIKA

(**CO**smic **R**ay **SI**mulations for **KA**scade)

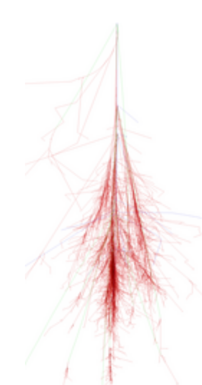
- weighted events : artificially flat spectrum
  - better livetime efficiency @ 10 TeV but poor efficiency @ TeV
  - energy-targeted generation of (H,He,CNO,Mg,Fe) with $E^{-1(2)}$





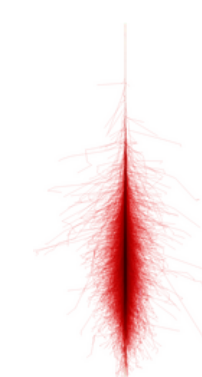


gamma shower



Proton shower

Iron shower



Muon shower

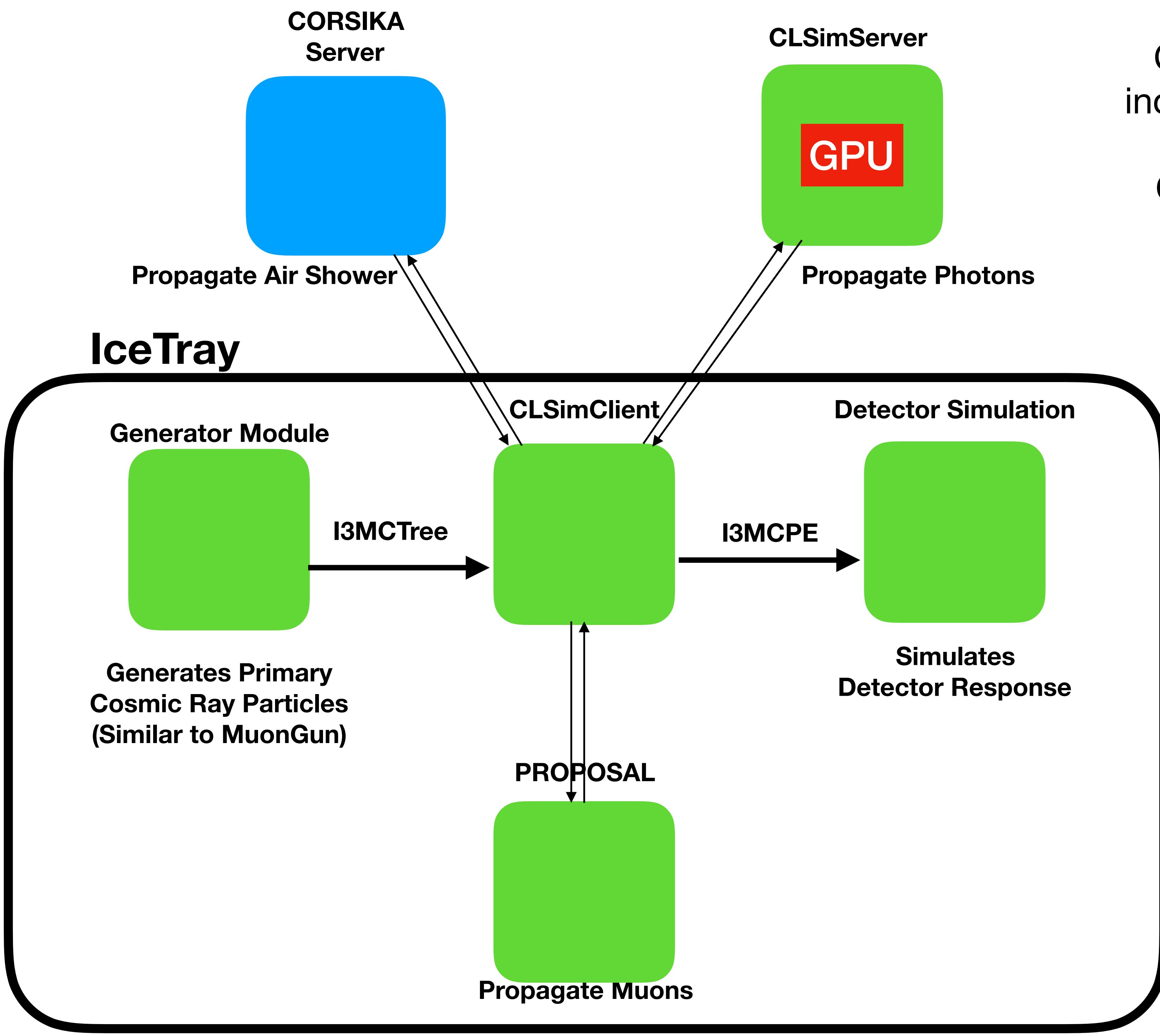
DYNSTACK in CORSIKA

Kevin Meagher & Jakob van Santen

- Replaces CORSIKA's post-reaction particle stack with a C++11 plugin
- General API for doing things like the neutrino kill threshold, plus helpful extras (take configuration from the steering card, manipulate event headers/trailers, etc)
- In mainline CORSIKA since 7.56 (modulo typos)
- Write plugins in C++11 without touching corsika.F, depend only on the standard library
- Build a better CORSIKA for in-ice background simulation.
- Reduce memory and disk requirements for high energy simulations.

Analysis-specific, targeted background simulation

DYNSTACK in CORSIKA

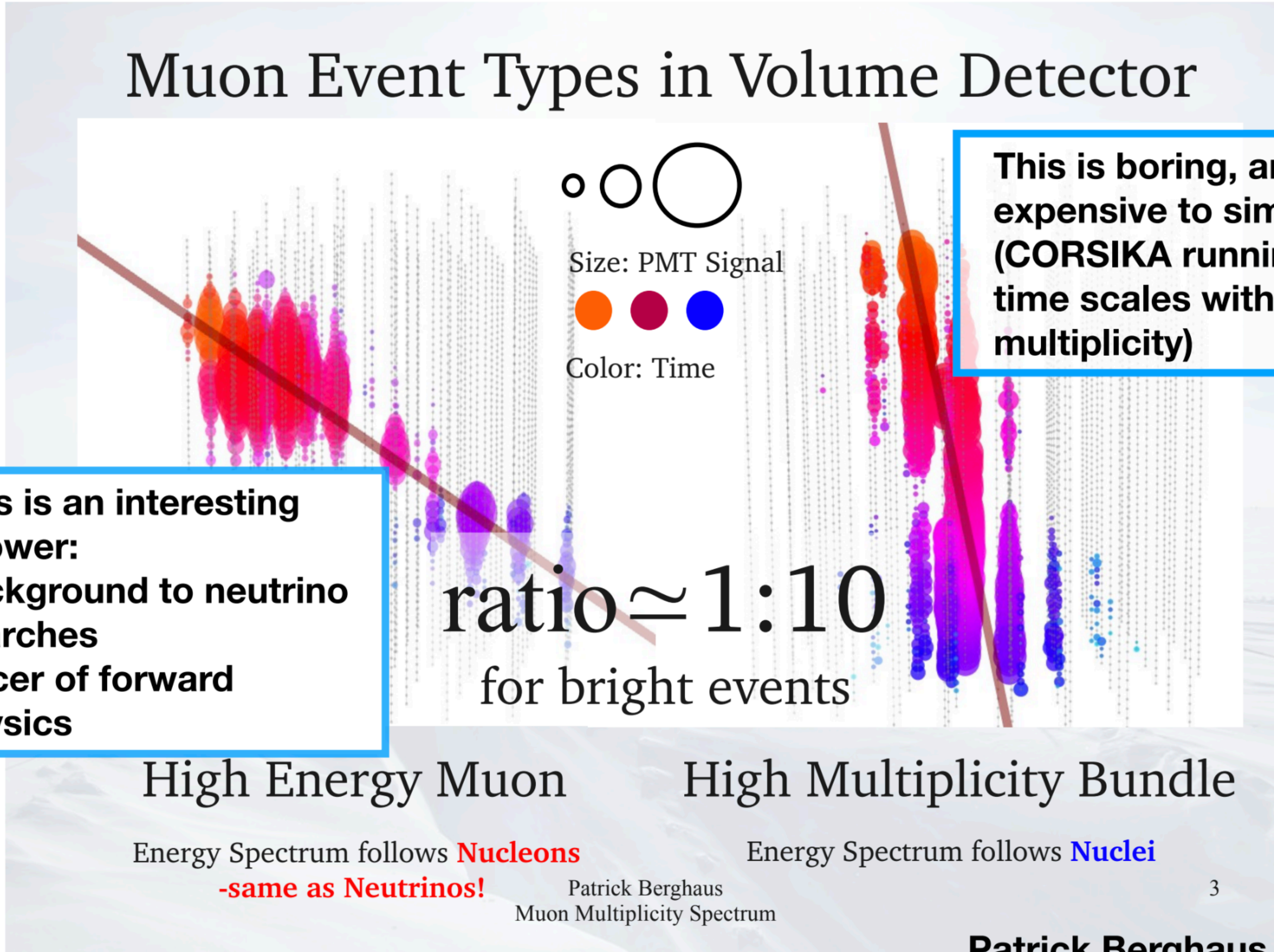


CLSimClient passes individual particles from the MCTree to the CORSIKA Server, to PROPOSAL to the CLSimServer

I3MCPE are created directly from the output of each individual CLSim propagation

K. Meagher

application of triggered CORSIKA



This is boring, and expensive to simulate (CORSIKA running time scales with multiplicity)

This is an interesting shower:

- **background to neutrino searches**
- **tracer of forward physics**

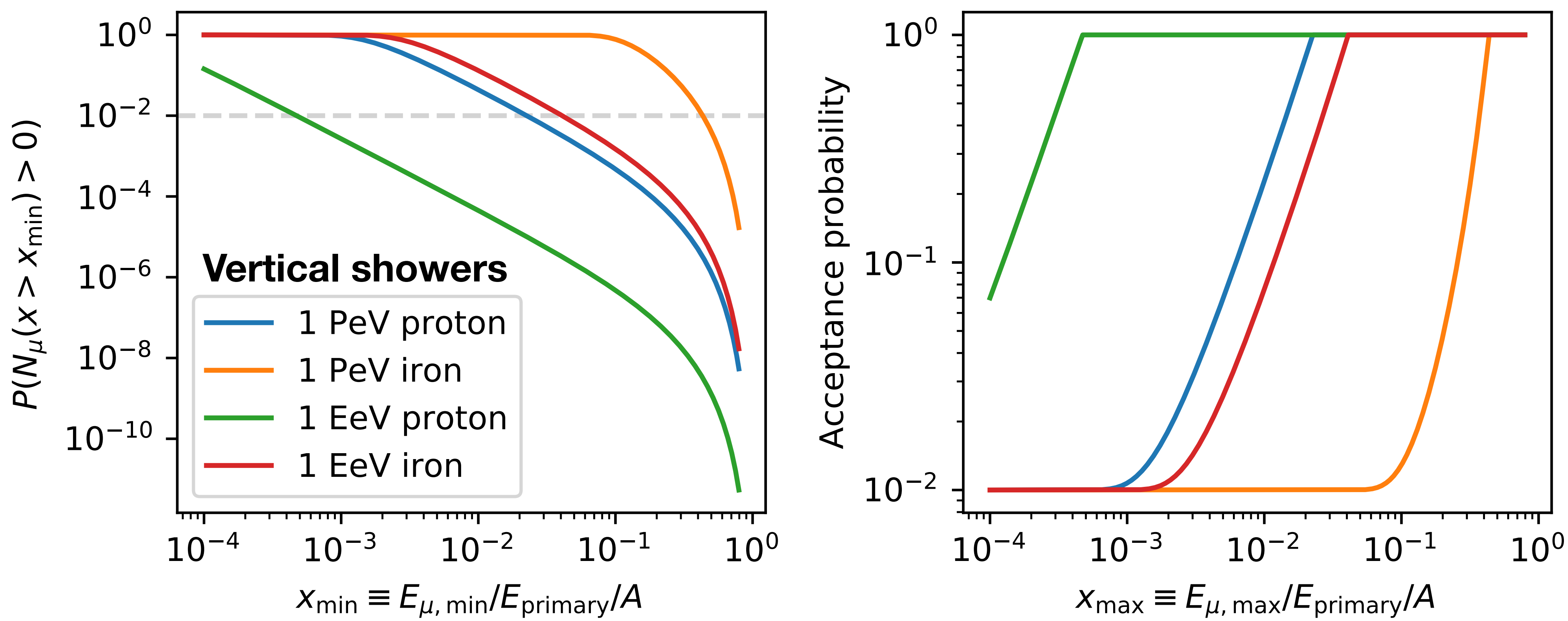
Patrick Berghaus, 2012

DYNSTACK in CORSIKA (example)

J. van Santen

Biasing scheme

- **User specifies a target fraction of showers to accept ("bias factor," e.g. 0.01)**
- **Plugin uses the Elbert formula to pick a muon energy threshold for each shower**



- Shower is killed with a probability **(always < 1!)** based on the highest-energy muon in the shower
- Kill probability increases monotonically with energy, so shower can be killed before the first muon is produced.

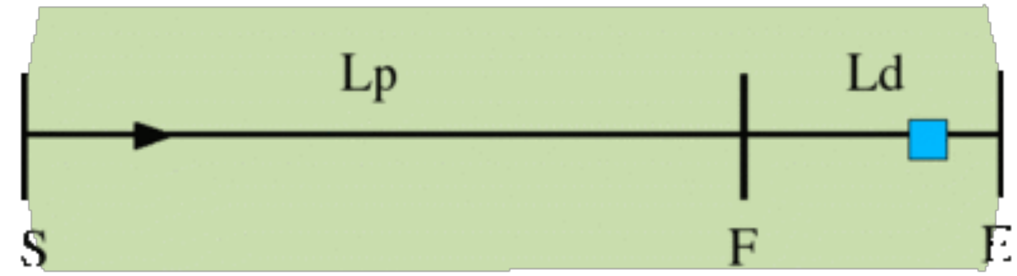
Generators

- Cosmic-ray Air Showers:
  - **CORSIKA** (FORTRAN stand-alone)
  - **corsika-reader**: IceTray reader for standard format
  - **CorsikaInjectorService** (IceTop)
- Muons:
  - **MuonGun**: parametrization of flux of atm. muons under the ice.
- Neutrinos:
  - **neutrino-generator**: injects neutrinos, propagates them through Earth, forces interaction in detector volume.
  - **genie-icetray**: detailed simulation of neutrino interactions with GENIE. (Used for low-energy simulations)
  - **LeptonInjector / NuFSGen** (not yet available): weighted leptons+weights to account for flux models, interaction models, in-earth propagation, etc.

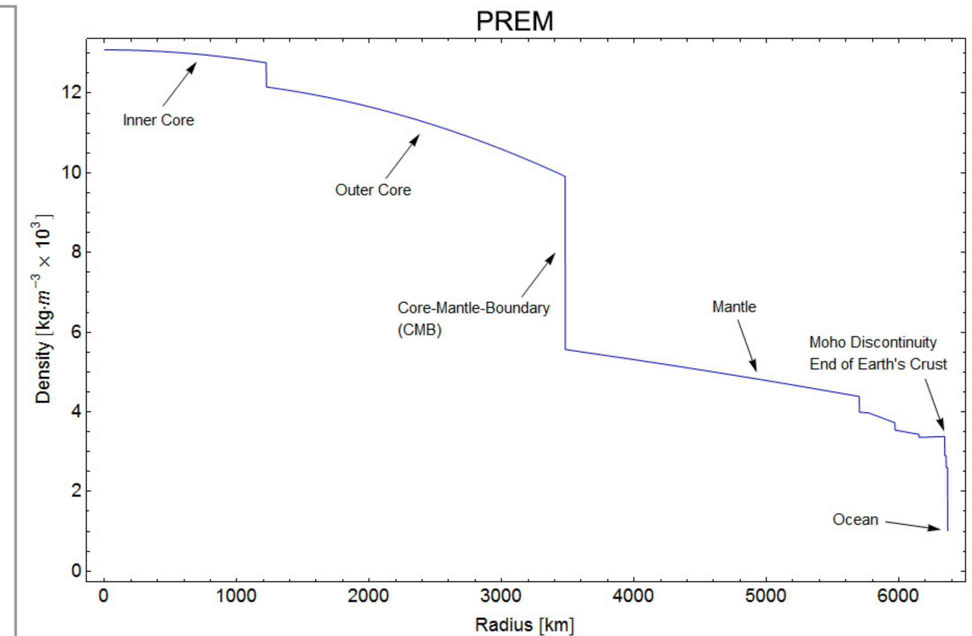
Generators

- Cosmic-ray Air Showers:
  - **CORSIKA** (FORTRAN stand-alone)
  - **corsika-reader**: IceTray reader for standard format
  - **CorsikaInjectorService** (IceTop)
- Muons:
  - **MuonGun**: parametrization of flux of atm. muons under the ice.
- Neutrinos:
  - **neutrino-generator**: injects neutrinos, propagates them through Earth, forces interaction in detector volume.
  - **genie-icetray**: detailed simulation of neutrino interactions with GENIE. (Used for low-energy simulations)
  - **LeptonInjector / NuFSGen** (not yet available): weighted leptons+weights to account for flux models, interaction models, in-earth propagation, etc.

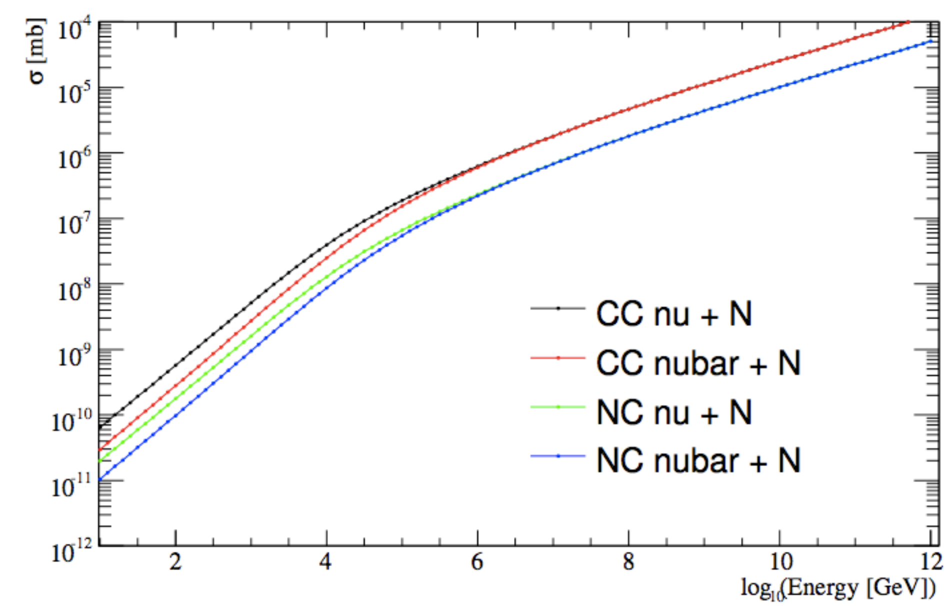
neutrino-generator



1. Calculate total path length inside the Earth using injected neutrino geometry.
    a. Separate the total path length into propagation area (SF) and
    b. detection volume (FE).
2. Define a step length dx[m] using propagation area and step number.
3. For each step:
    a. Calculate a column depth and Earth's density at the step point.
    b. Calculate a total cross section at the step point.
    c. Calculate a probability that the injected neutrino interacts within the step. Try Monte-Carlo, and decide whether an interaction happened within the step.
    d. If interaction occurred: choose interaction randomly.
        i. If CC-interaction is selected with injection particle NuMu or NuE, break (event is killed).
        ii. else, generate secondaries and continue to next step.
    e. If nothing happens, continue next step.
4. Finish propagation when injected neutrino + secondaries reach surface of detection volume (point F), then process a weighted interaction.

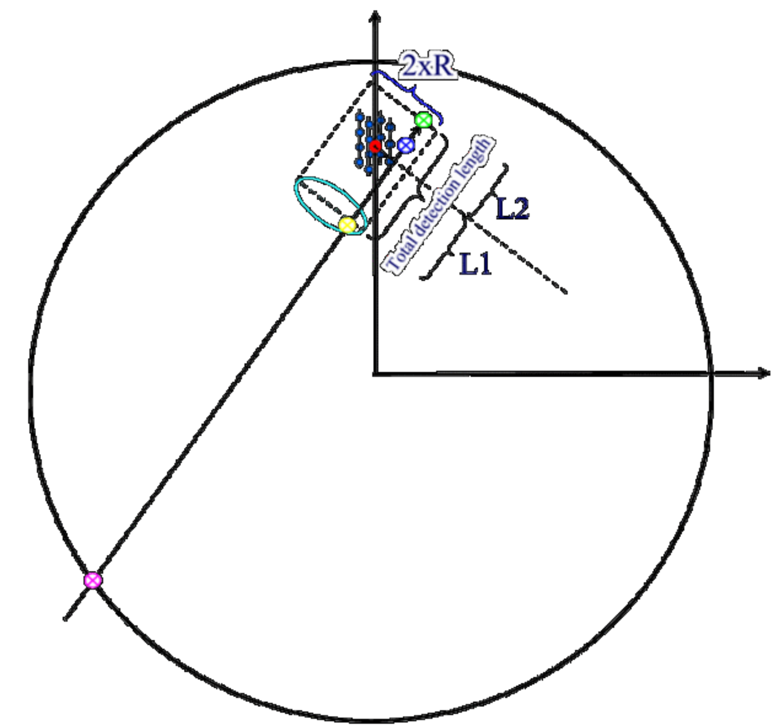


- produce a $E^{-\gamma}$ ν_μ, ν_e, ν_τ with
    - PREM Earth's density model



    - parton distribution functions
    - prop & interaction of neutrinos into a weight

neutrino-generator

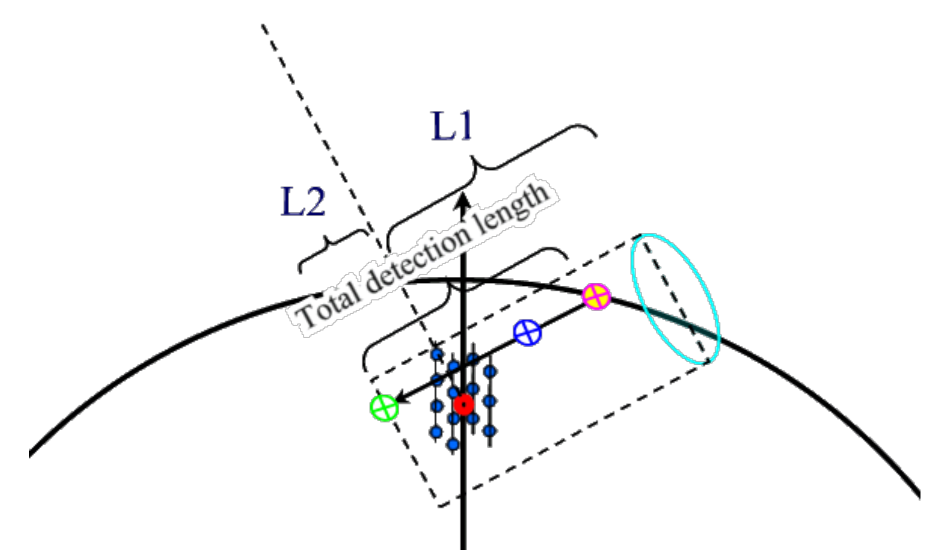


- Calculates the propagation probability (i.e. that the neutrino will reach the detector)

- It forces an interaction within a volume around the detector and computes the probability of this interaction

$$OneWeight = \left(\frac{P_{int}}{E^{-\gamma}}\right) \cdot \int_{E_{min}}^{E_{max}} E^{-\gamma} dE \cdot Area \cdot \Omega \cdot T[GeV \cdot cm^2 \cdot sec \cdot sr]$$

where $P_{int} = TotalInteractionProbabilityWeight$, $E^{-\gamma}$ is the neutrino generation energy spectrum shape, E_{min} and E_{max} is the minimum and maximum generation energy of neutrinos, $Area$ is the generation surface, Ω the generation solid angle and $T = 1 sec$ is the timescale.



- The weight corresponding to a given theoretically motivated neutrino flux is

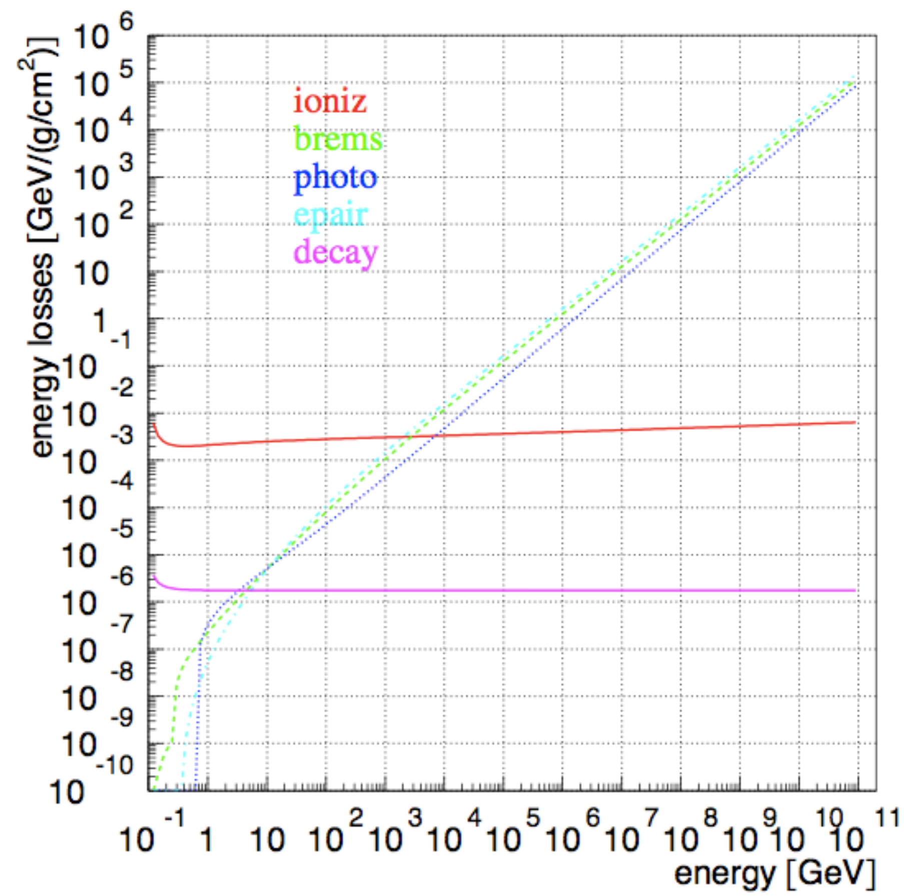
$$w_i = \frac{OneWeight_i}{NEvents} \times \frac{d\Phi_\nu(E_\nu)}{dE_\nu}$$

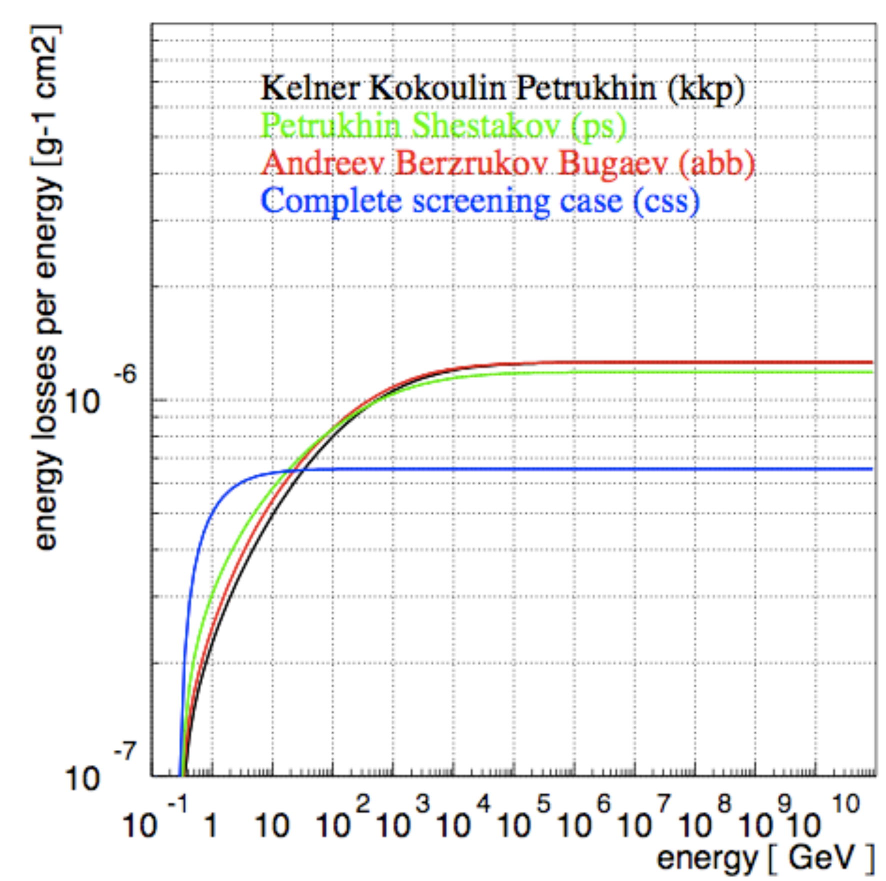
- For more details on how to use OneWeight see:

https://docushare.icecube.wisc.edu/dsweb/Get/Document-44937/OneWeight.pdf

Lepton propagation

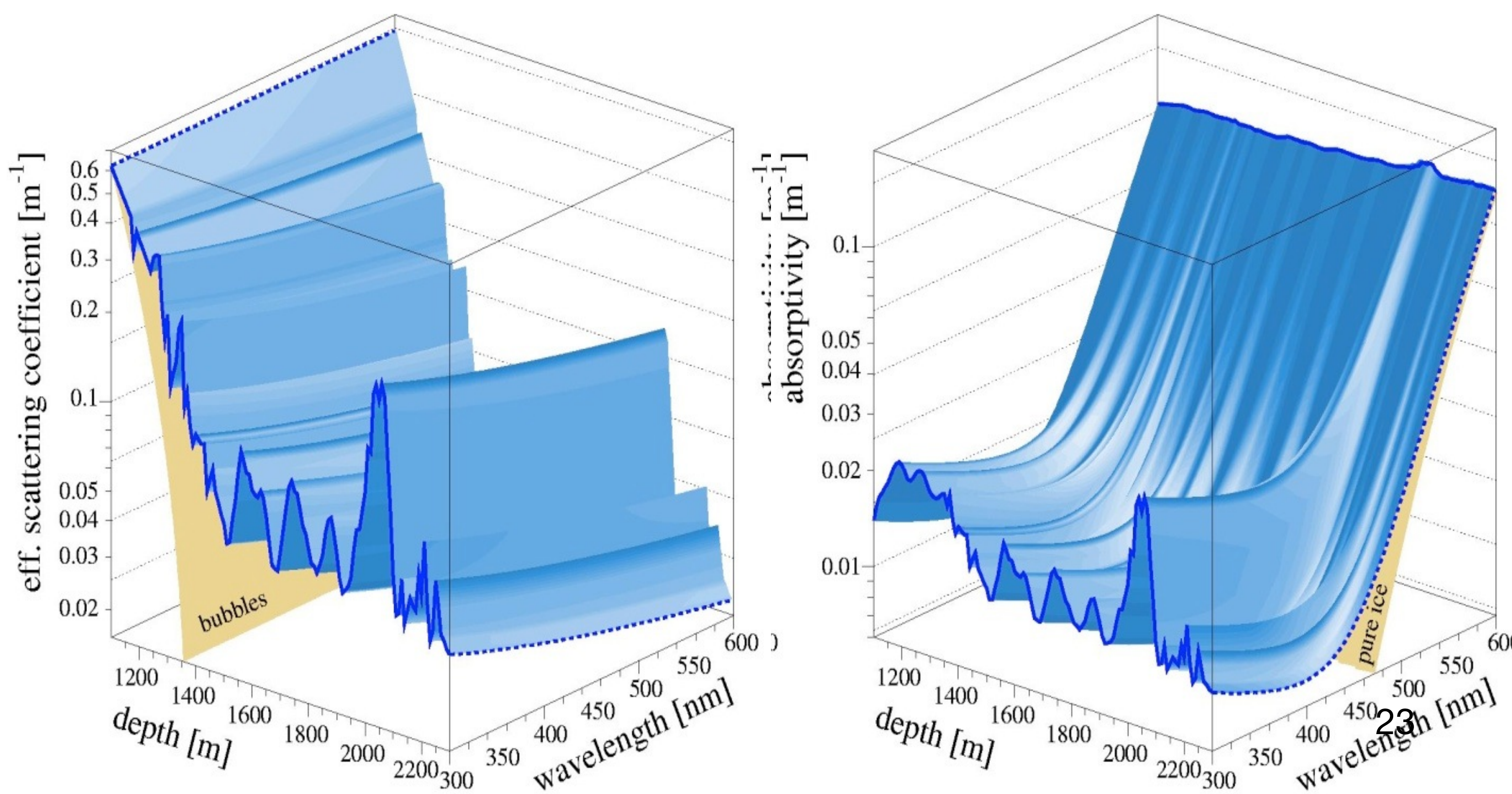
- PROPOSAL: parametrized interactions with the medium.
  - Stochastic energy losses include:
    - ionization
    - e^+e^- pair production
    - bremsstrahlung
    - photo-nuclear interaction
    - decay
- GEANT4: Detailed particle propagation in media.
  - 3rd-party G4 library used by CLSim to propagate leptons for low-energy simulations (CPU-intensive).



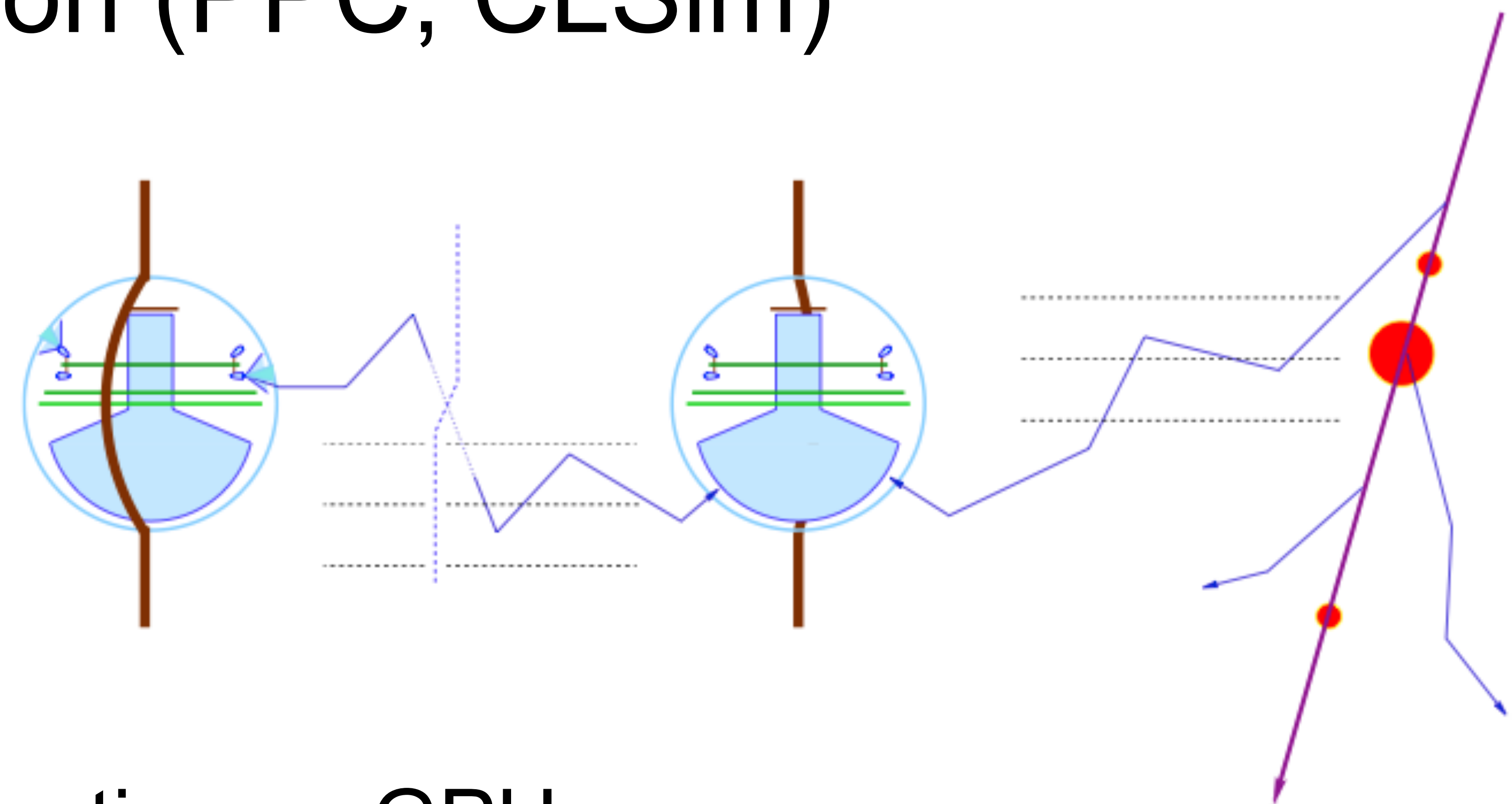


Photon Propagation

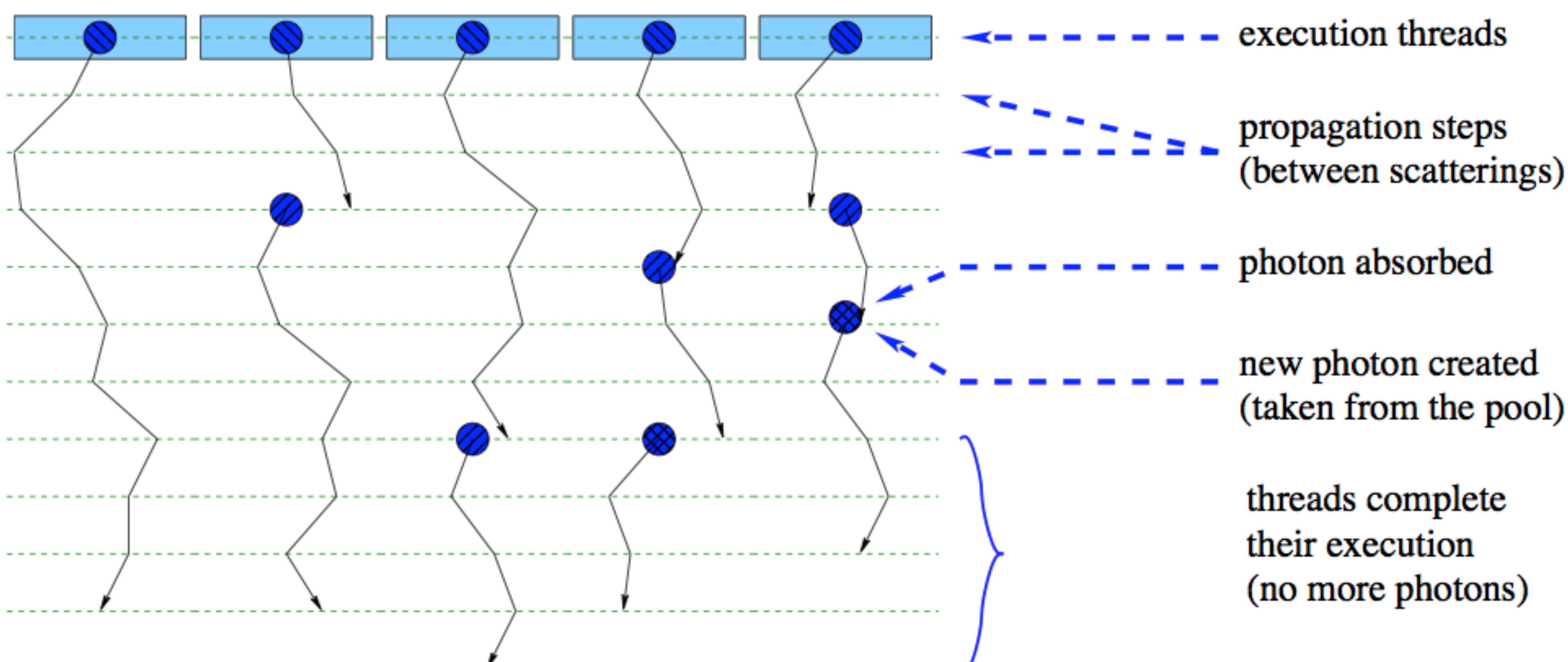
- μ energy lost + cascades ➙ photons ➙ p.e.
  - Photon propagation : ice properties + PMT response + DOM glass/gel
    - Pre-generated lookup splined table :
      - I3PhotonicsHitMaker
      - Amplitude and time distribution
    - Direct photon tracking
      - PPC
      - CLSim
    - Hybrid photon tracking
      - HitMaker + CLSim



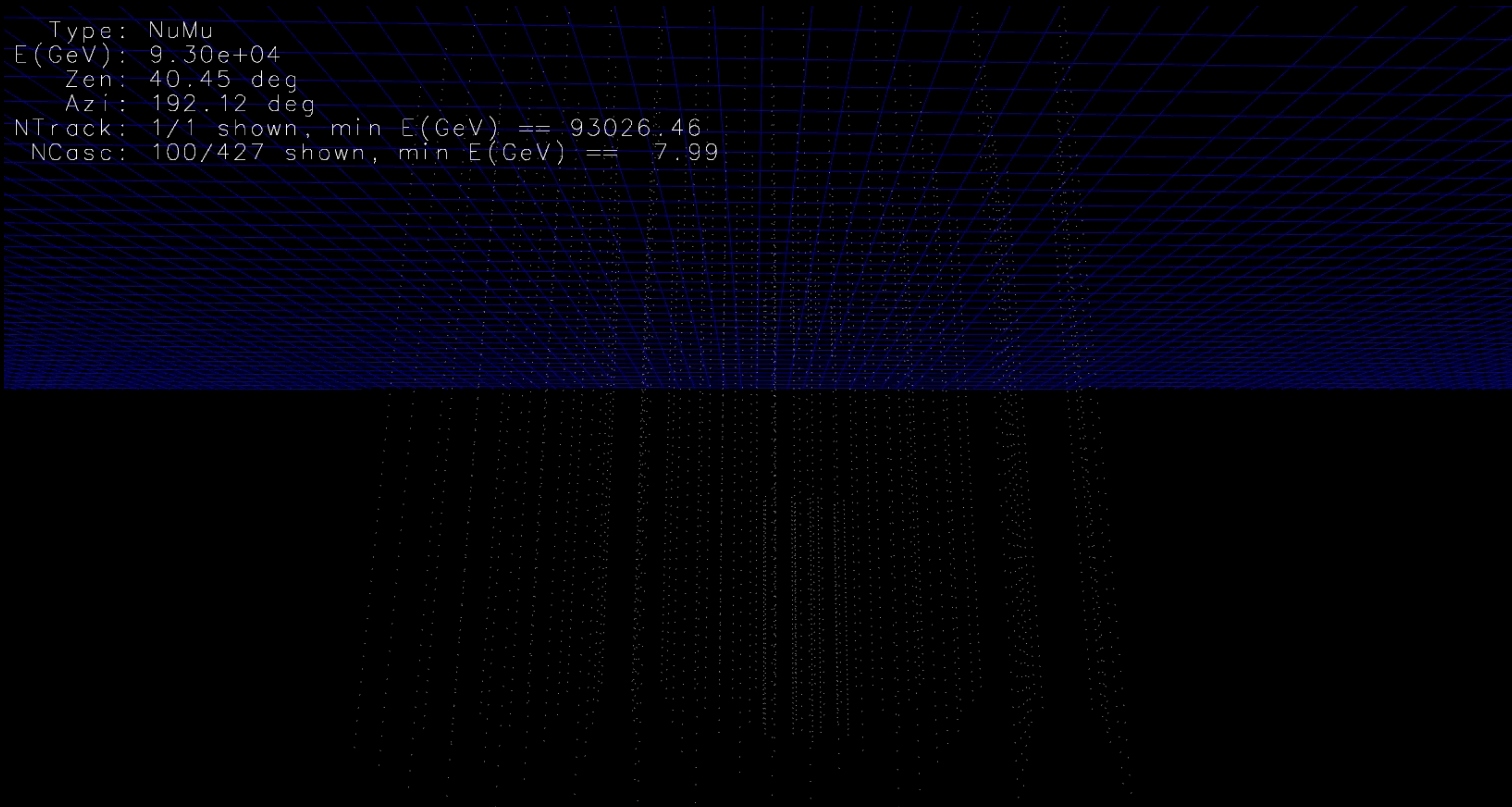
Photon Propagation (PPC, CLSim)



Direct photon propagation on GPU



Photon Propagation (PPC, CLSim)



http://icecube.wisc.edu/~ckopper/muon_with_photons.mov

Polyplopia

(from gr., πολύς - polús, "many,", and ὄψ-ops , "vision")

Coincident atmospheric shower events in IceCube

- poly**plopia::PoissonMerger**
  - Injects background event read from a separate file on top of primary events in the chain by sampling from a Poisson distribution over a time window Δt.
  - Also makes use of a *CoincidentEventService* that could be drop-in replaced with other event services such as a MuonGun-based service.
  - Writes a separate I3MCTree with background particles.
  - Writes a combined I3MCPE map for signal and background.

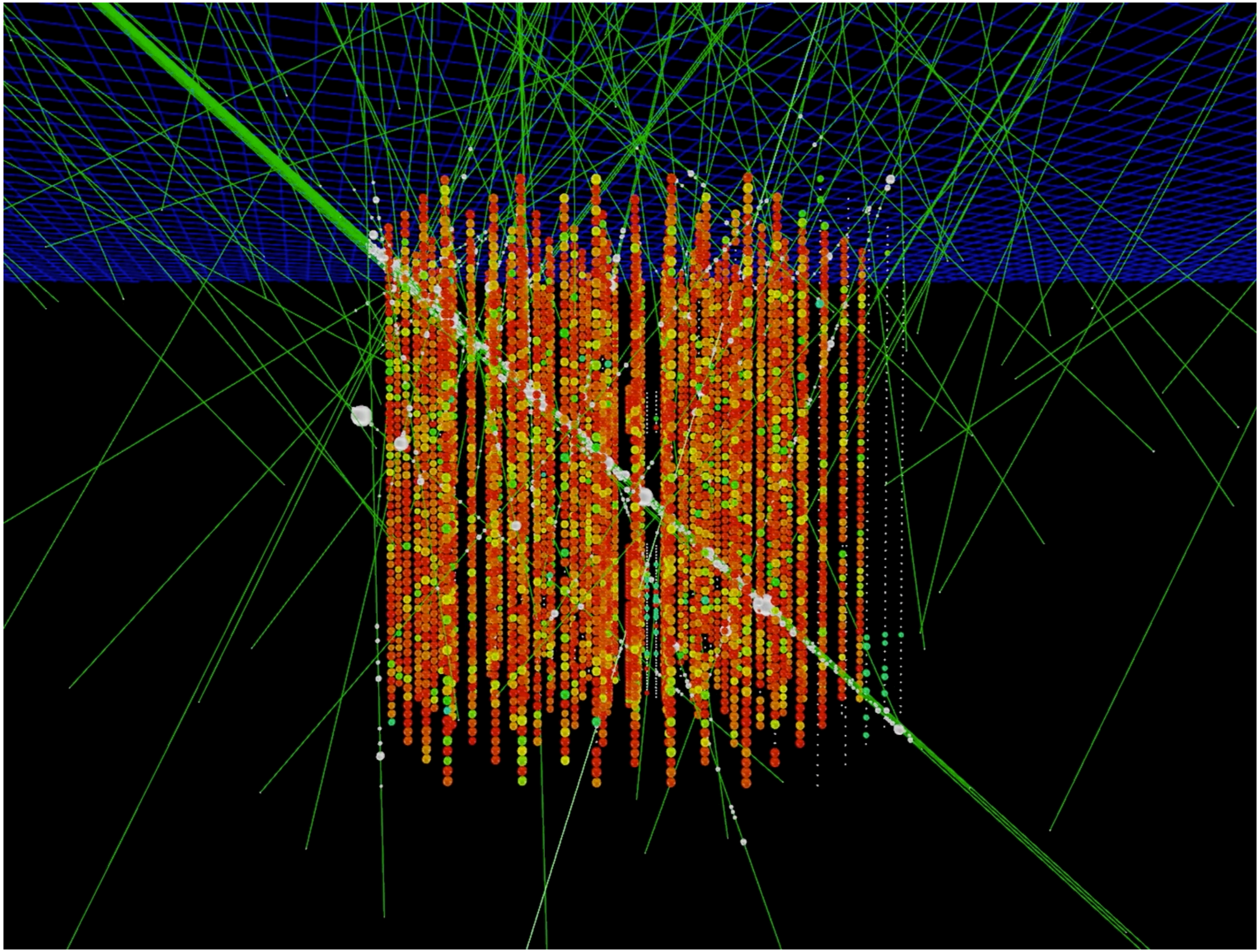
- poly**plopia::MPHitFilter**
  - Removes events that don't produce light in the detector and removes branches of I3MCTrees whose particles don't produce enough PEs in the detector,
  - Reduces the storage requirements.

- It is then up to the trigger-sim to split up Q-frames into P-frames events based on triggers.

Polyplopia

(from gr., πολύς - polús, "many,", and ὄψ-ops , "vision")

Coincident atmospheric shower events in IceCube



http://icecube.wisc.edu/~juancarlos/video/10msec_w_noise.mp4

Noise Generation

—> (MCPEs)

Noise Model

Thermal Noise (~few Hz)
[Poisson process]

~ ms Timescales

+

Radioactive Decay in Glass
[Poisson process]

~ ms Timescales

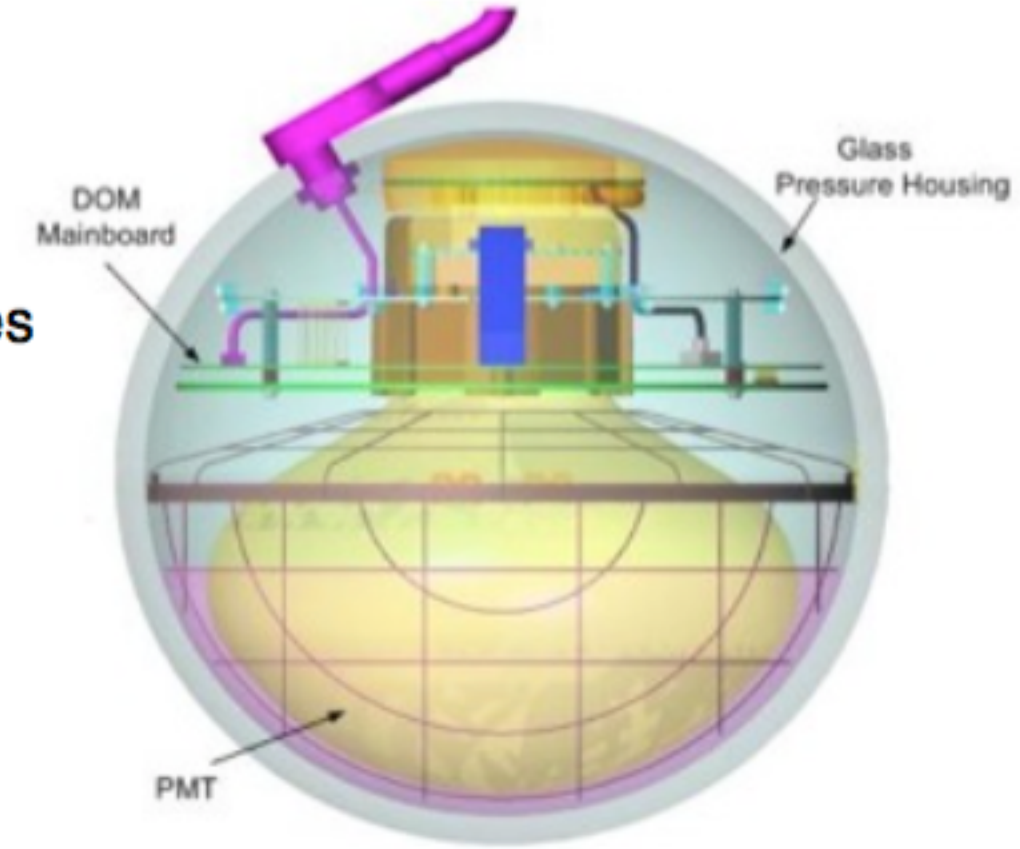
↓

Energy deposited in glass

↓

Glass scintillates/fluoresces
over long timescale
[Log-normal]

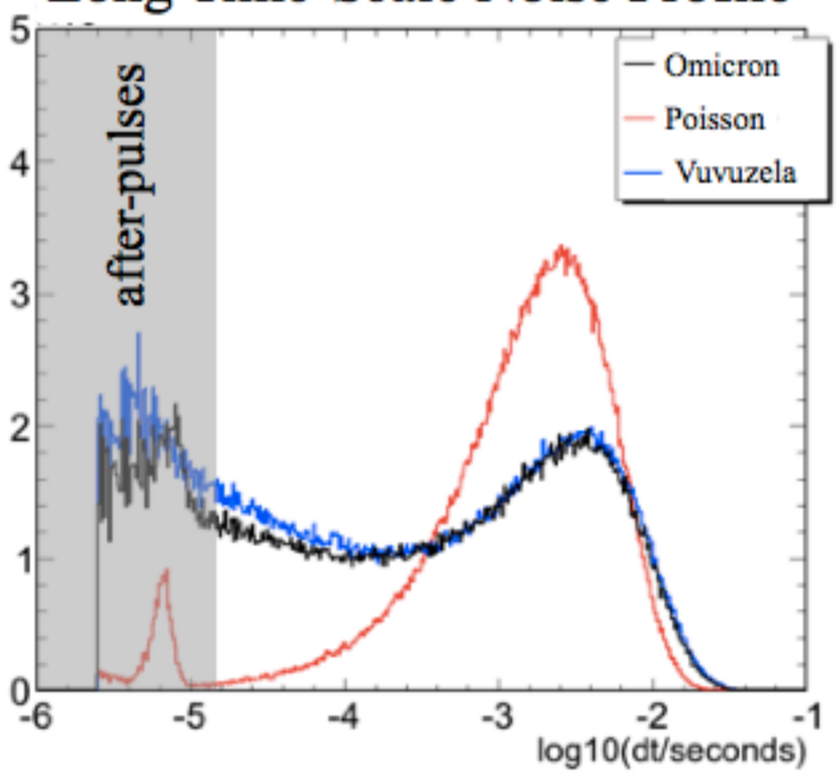
≲ 500 µs Timescales



Noise Generation

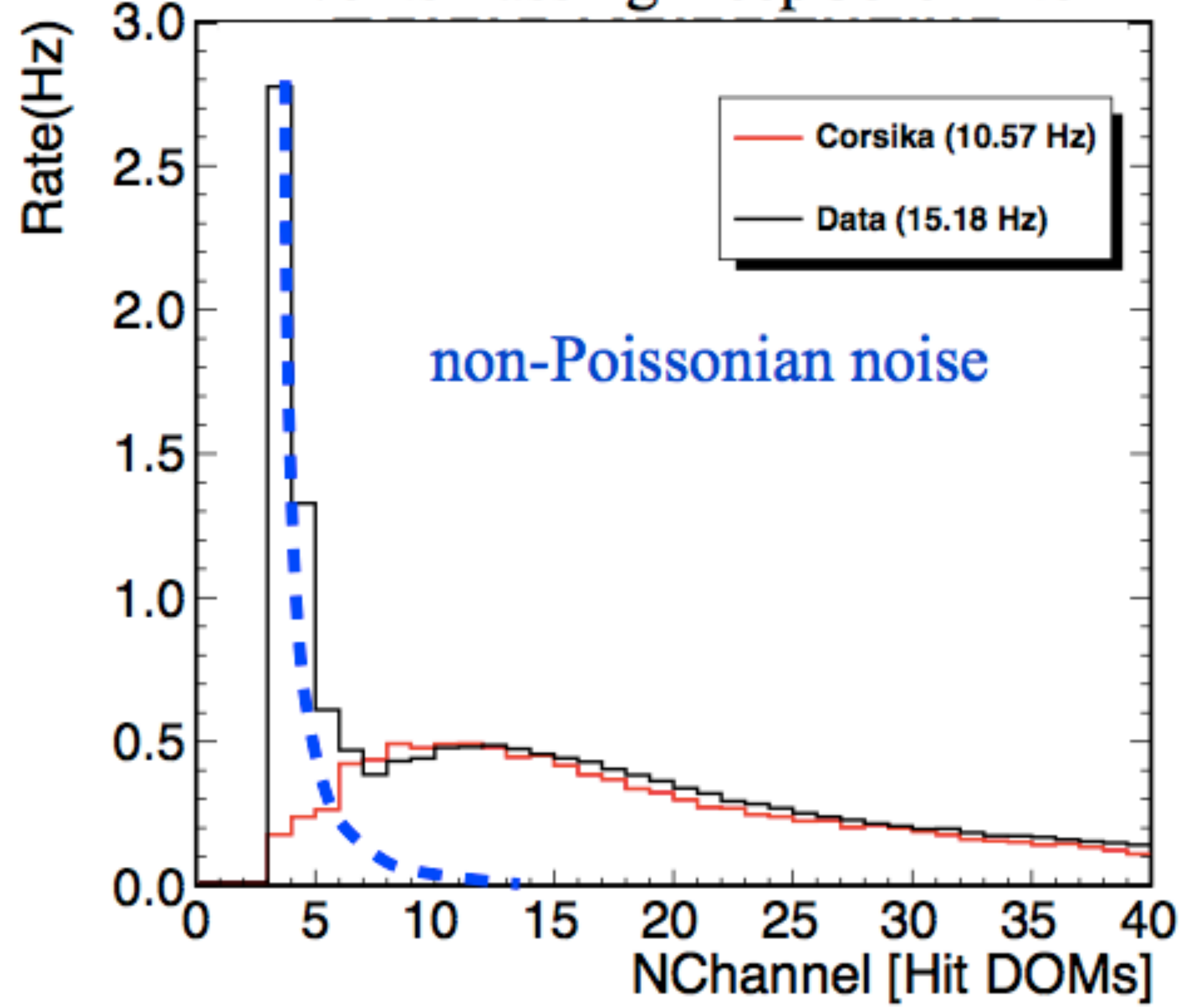
Previous simulation used simplified Poissonian model. Vuvuzela uses exponential for **thermal** and **radioactive decays** and log-normal for **scintillation**.

Long Time-Scale Noise Profile



*Courtesy of M.Larson (U.Alabama)

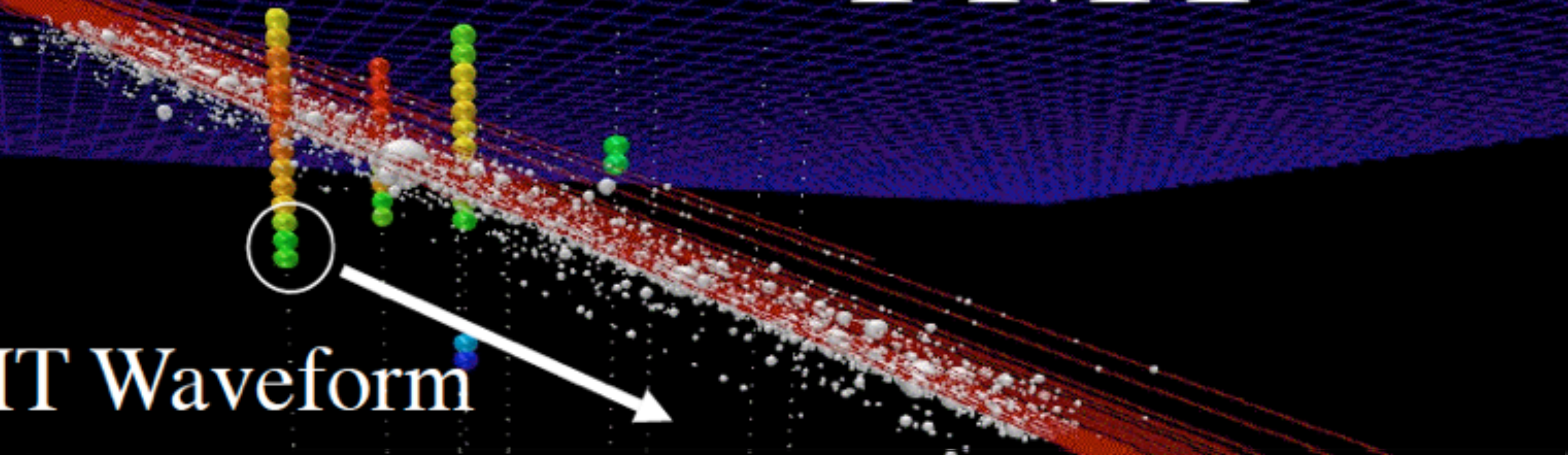
Events Passing DeepCore Filter



*Courtesy of J.Koskinen (PSU)

DOMLauncher::
PMTResponseSimulator

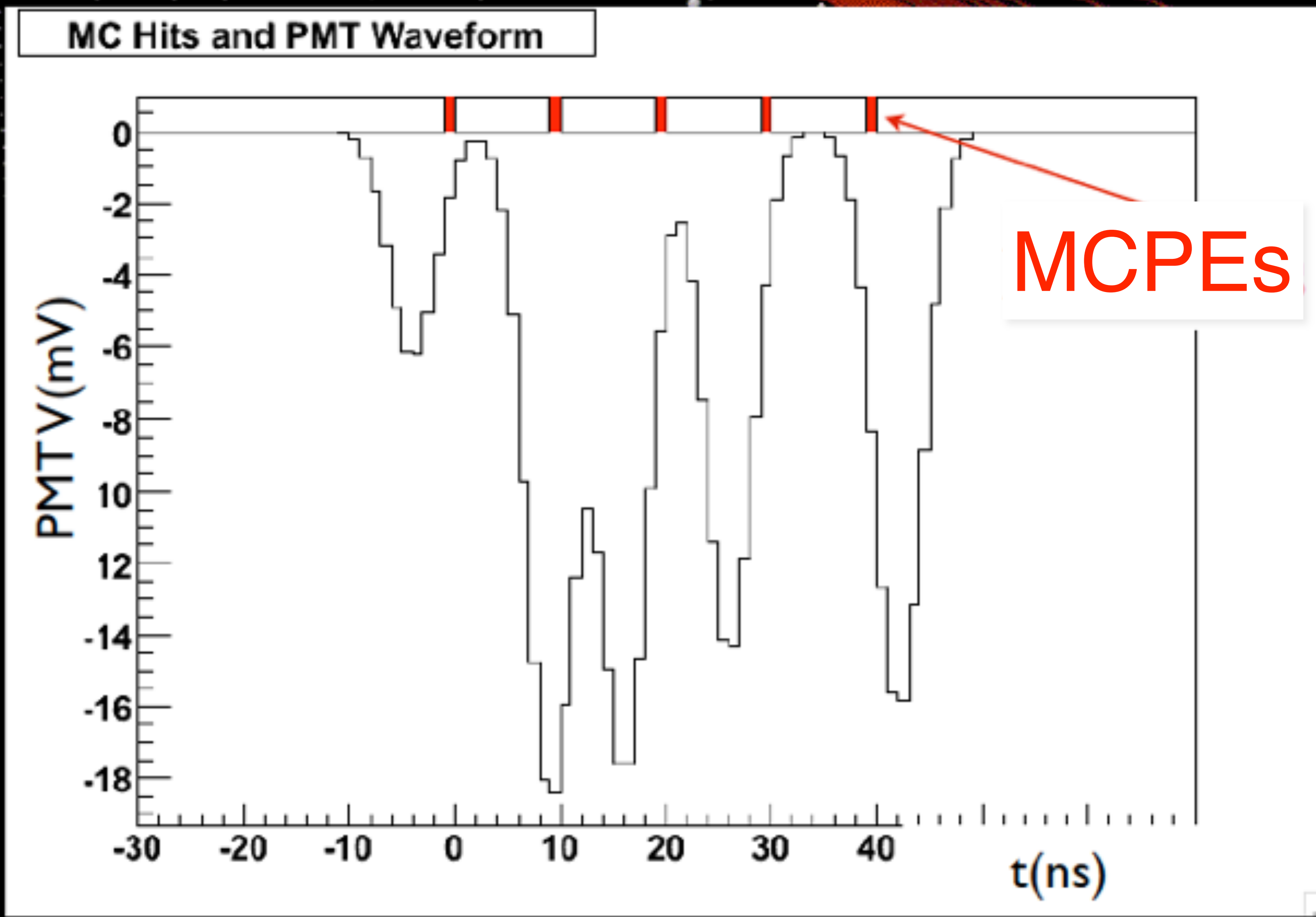
PMT



Generates PMT Waveform

From distribution of (combined) MCPEs.

Outputs I3MCPulseSeries for each DOM.

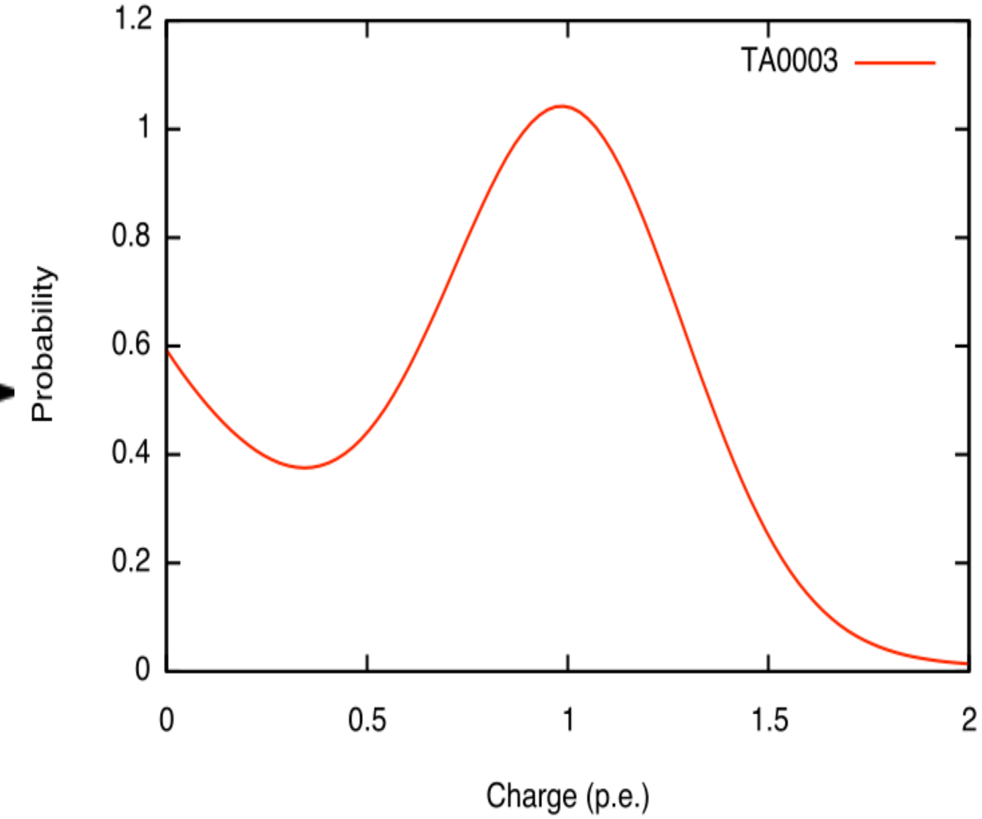


PMTResponseSimulator

Input: I3MCPEs
Output: I3MCPulses

Processing MCPEs :

- Give each MCPE a weight → corresponding to the pulse charge that photon would yield.
- Generate prepulses, late pulses and after pulses.
- Apply time jitter.
- Simulate the effect of saturation.



Weights fromSPE Charge Distribution

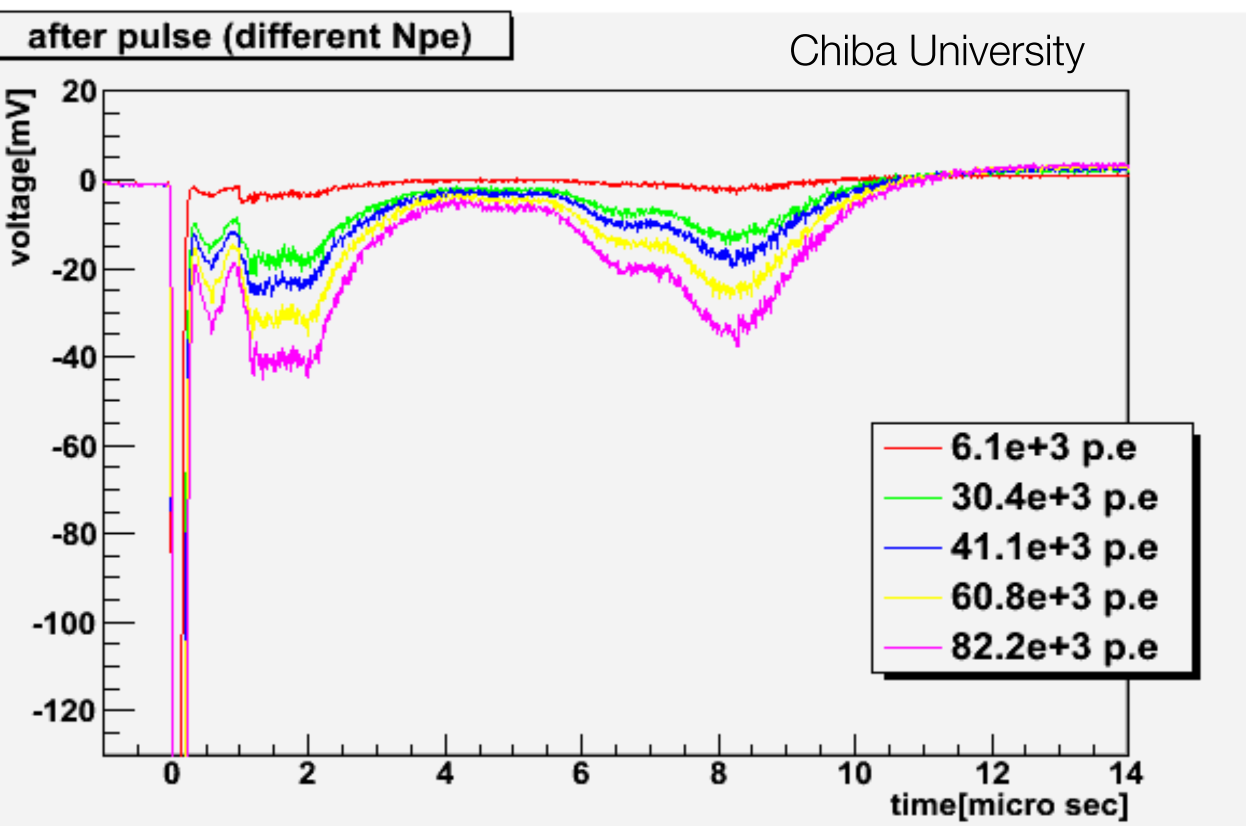
http://software.icecube.wisc.edu/documentation/projects/DOMLauncher/index.html

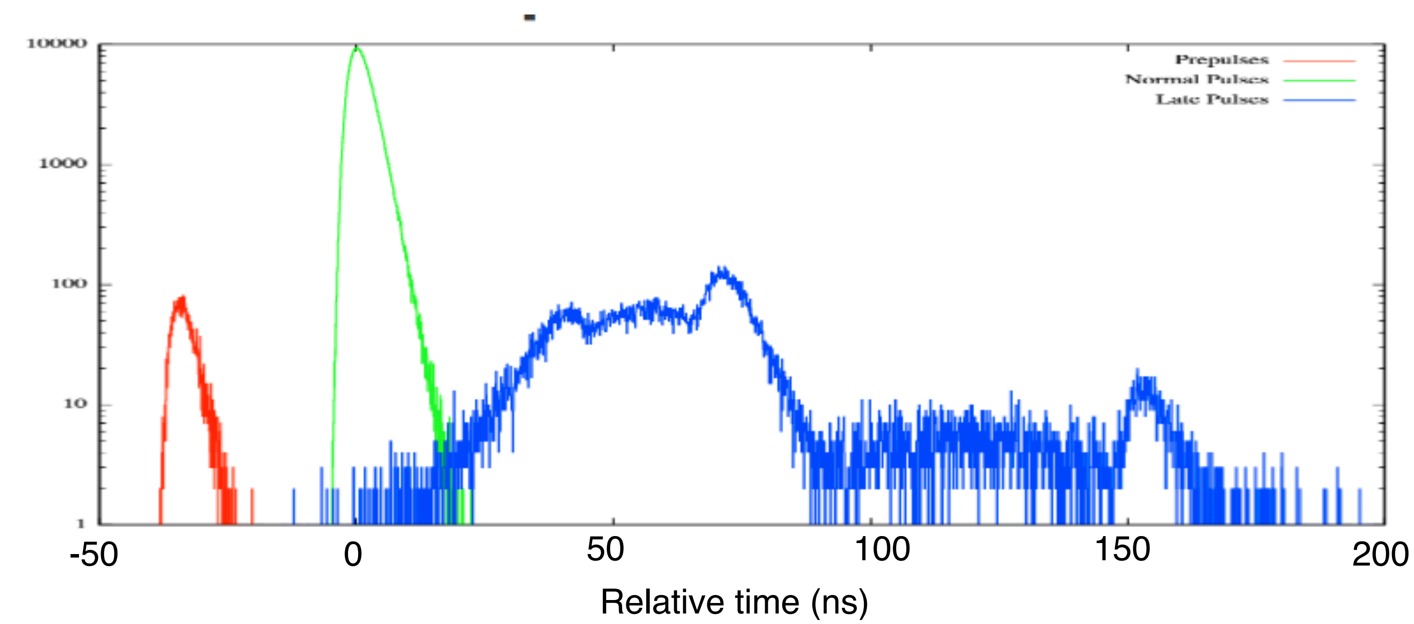
PAL pulses

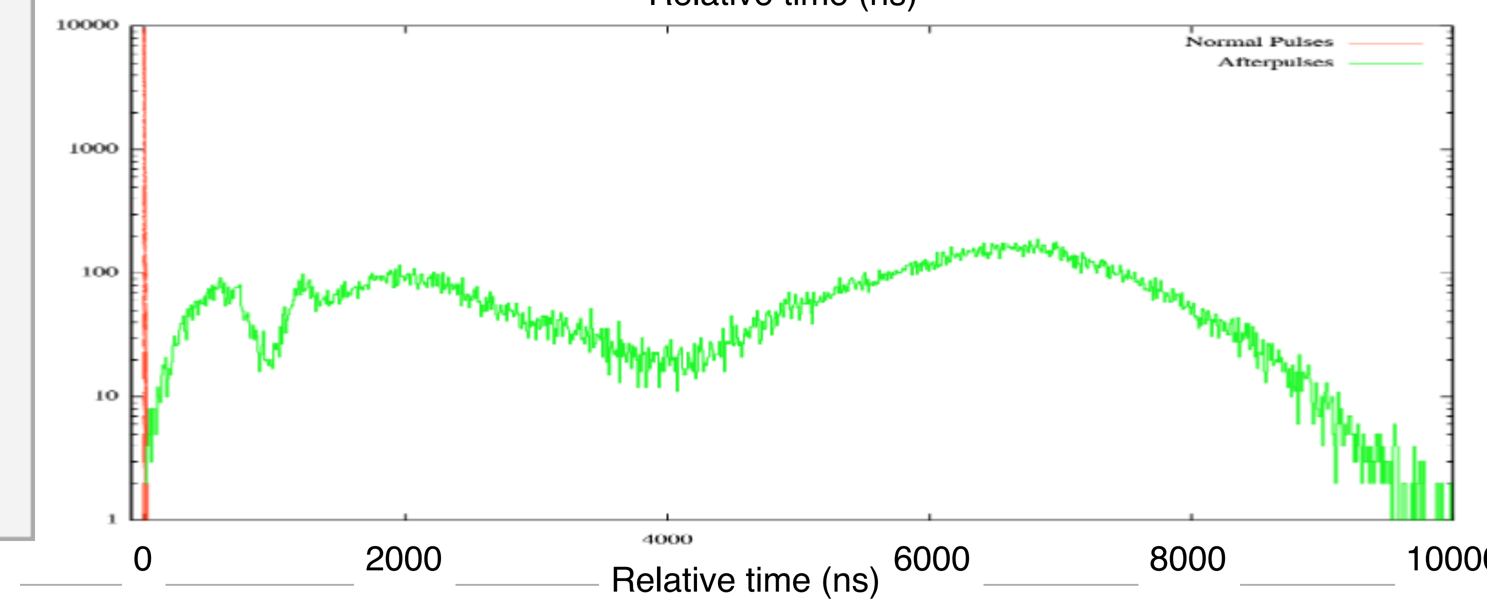
Pre-pulses: photoelectrons ejected from the first dynode,

Late pulses: electrons backscatter from dynode to cathode.

After-pulses: ionization of residual gases by electrons accelerated in the space between dynode.

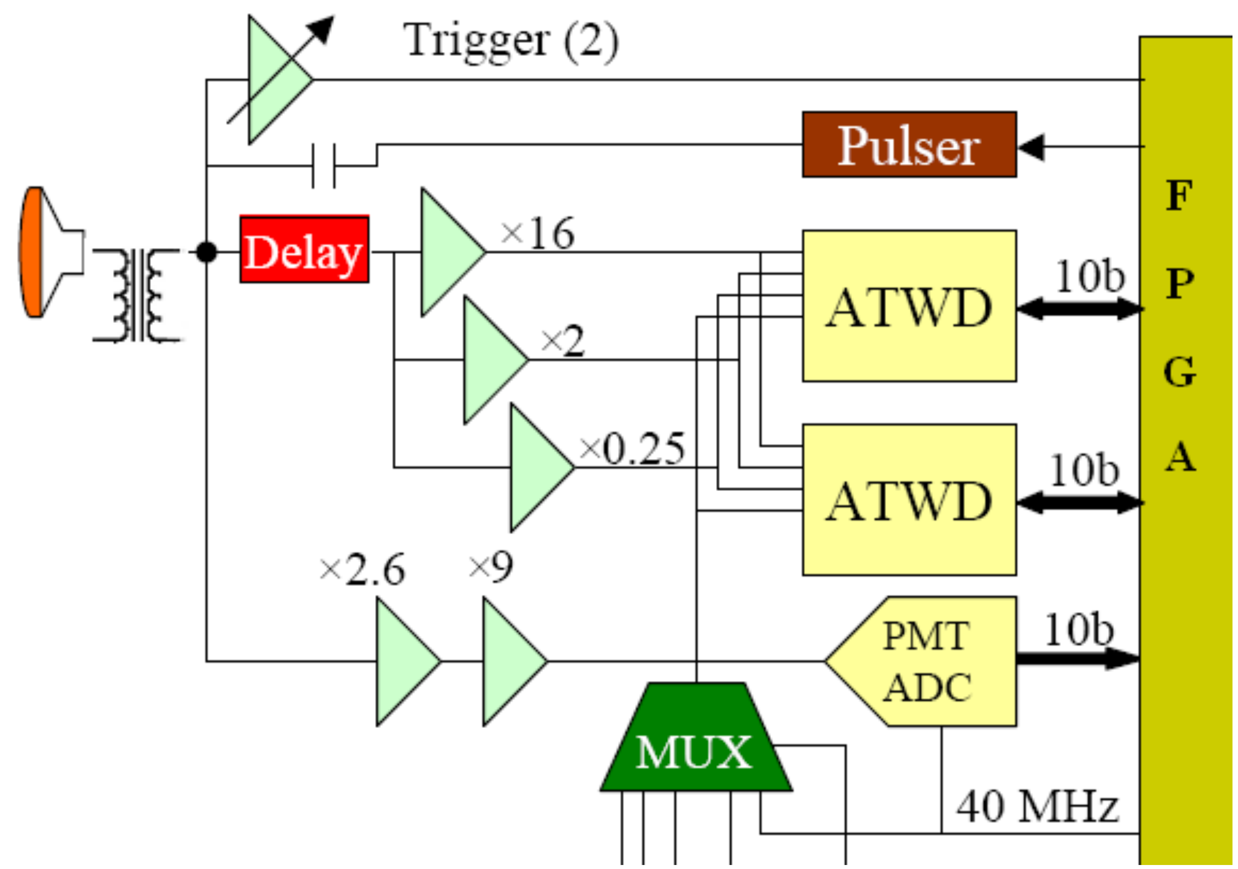


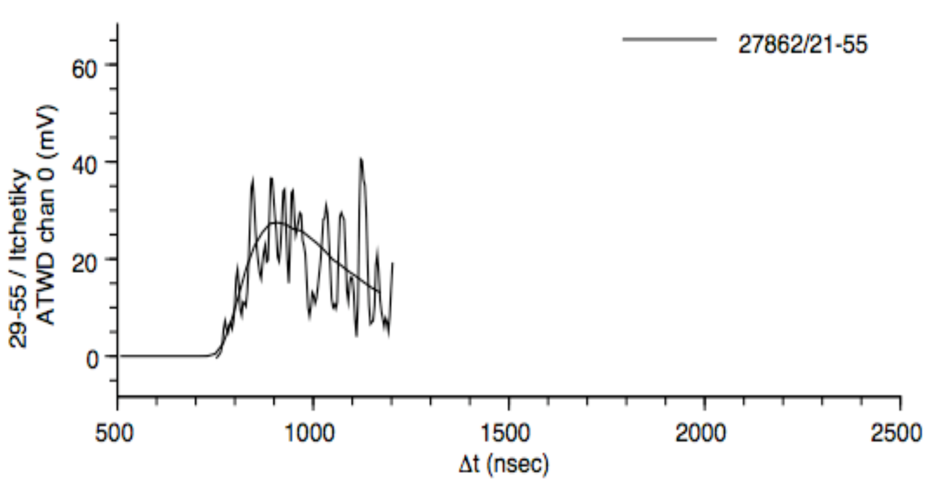


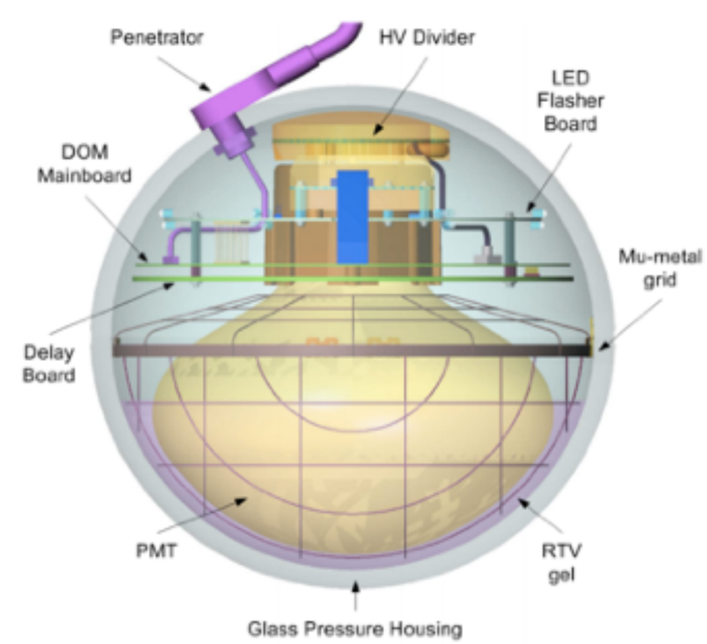


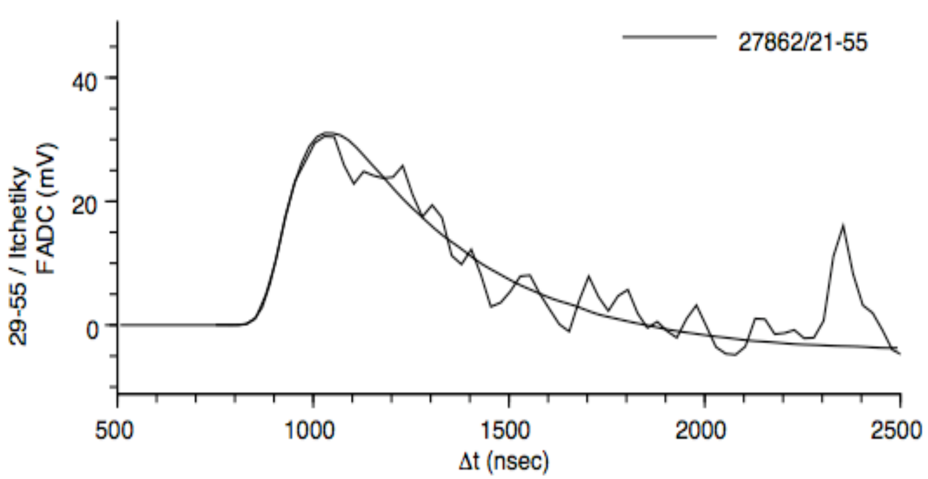
DOMLauncher: DOM electronics simulation

- Discriminator
- LC-logic
- Digitization
- Simulated effects
  - Electronic noise in the digitizers
  - Beacon launches (CPU triggered launches)
  - The FPGA Clock phase
  - RAPcal time uncertainty









Trigger Simulation

- **S**imple **M**ultiplicity **T**rigger (SMT)
  - *N* HLC hits or more in a time window
  - Example: InIce SMT8 with N_hits ≥ 8 in 5 μs
  - readout window around this captures early and late hits (-4 μs, +6 μs)

- **String** trigger (a.k.a. Cluster trigger in DAQ-land)
  - *N* HLC hits out of *M* DOMs on a string in a time window
  - Example: 5 hits from a run of 7 adjacent DOMs in a time window of 1500 ns

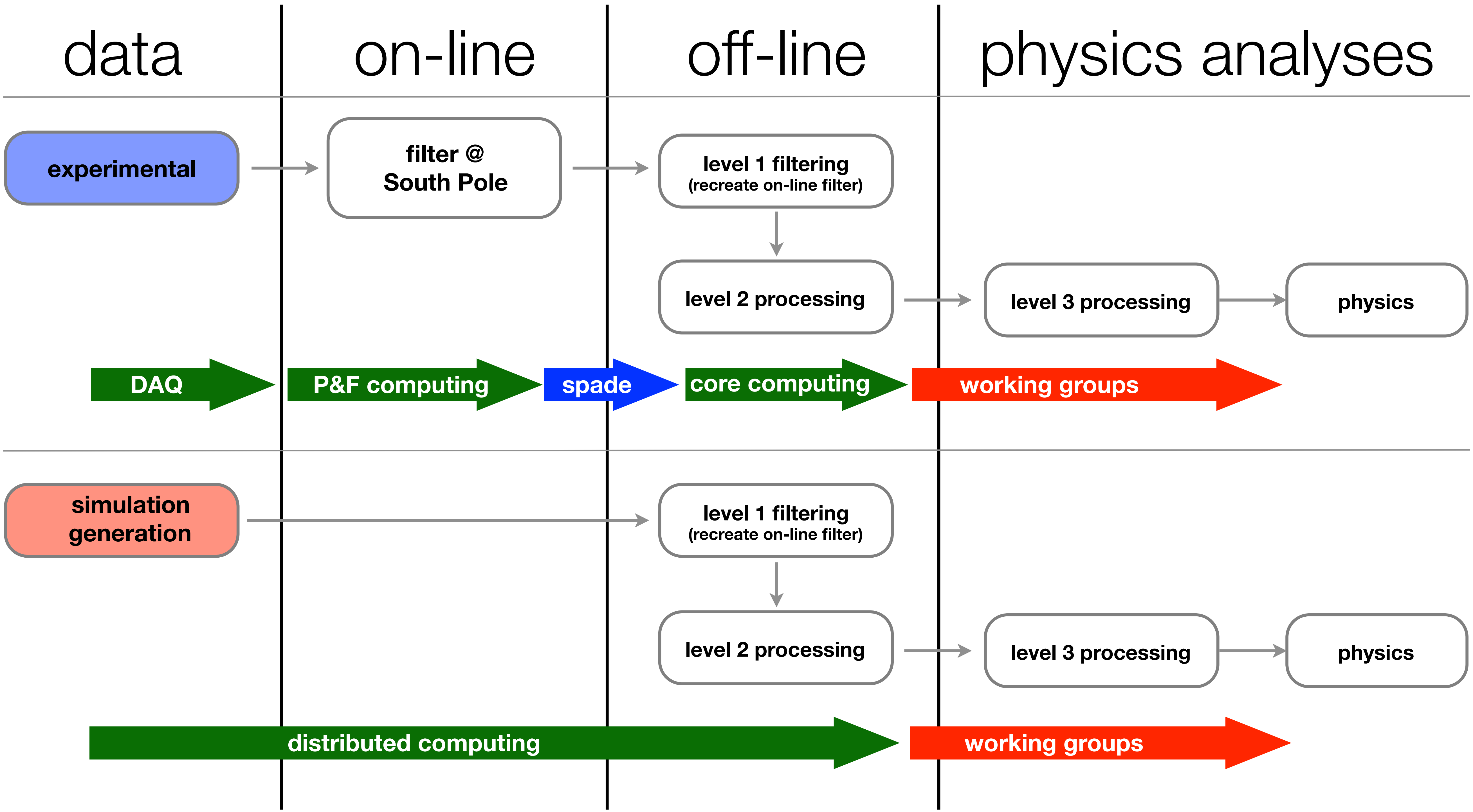
- **Volume** trigger (a.k.a Cylinder trigger in DAQ-land)
  - simple majority of HLC hits (SMT4) with volume element including one layer of strings around a center string
  - cylinder height is 5 DOM-layers (2 up and down from the selected DOM).

- **Slo**w **P**article trigger (SLOP)
  - slow-moving hits along a track
  - lengths of the order of 500μs and extending up to milliseconds

- ~~**F**ixed **R**ate trigger~~, **Min**imum **Bias** trigger, ~~**Calibration** trigger~~

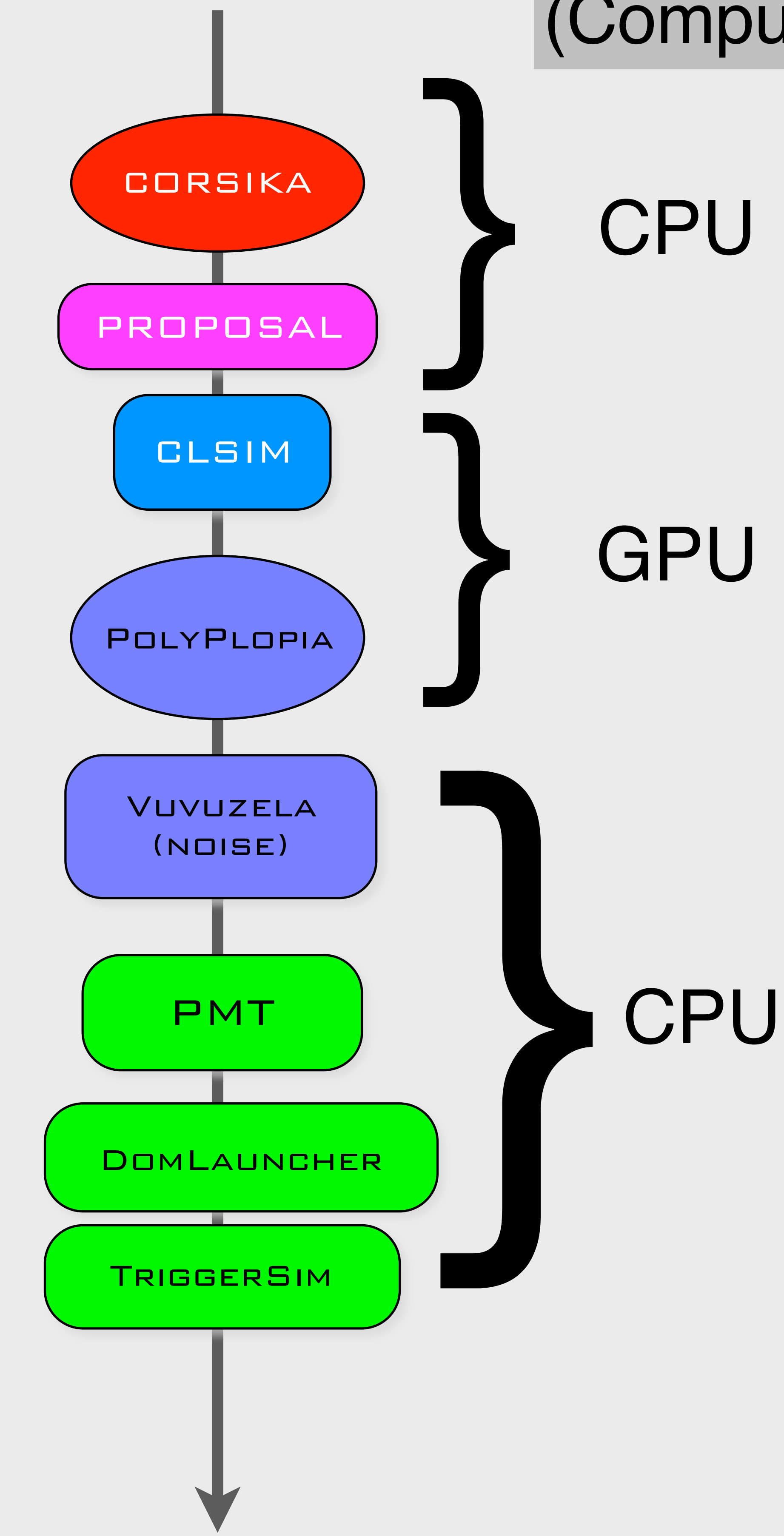
J. Kelley - DAQ

flow of experimental and simulation data



i3SIM

The Shish Kabob
(Computing Resource Optimization)



- Optimizing the shish kabob:
  - Different parts of the simulation chain have different resource requirements.
    - CORSIKA is CPU-intensive and requires little RAM
    - Photon propagation run almost exclusively on GPUs
    - Detector simulation is CPU bound and requires more memory.
- Things to keep in mind:
  - Running the whole chain on a GPU node will waste GPU resources and limit your throughput.
  - Intermediate storage:
    - breaking up chain requires transfering/storing intermediate files.
  - Reduce complexity in workflow

Simprod-Scripts

http://software.icecube.wisc.edu/documentation/projects/simprod_scripts/index.html

This project is a collection of scripts, tray segments and IceProd modules used in simulation production. The aim is to provide a central place with standard segments for running simulation in both production and privately.

- **Tray Segments**: IceTray meta-modules that contain several I3Modules with default parameters.
- **IceProd modules**: basic wrappers around tray segments that provide an interface for IceProd.
- **Scripts**: collection of python scripts used in simulation production
- **Examples**: The directory simprod-scripts/resources/examples contains a collection of example scripts for running IPModules
- **Tests:** are run on the build-bots to check that the different parts of the simulation are not broken with each commit to the software repository.

Simprod-Scripts

http://software.icecube.wisc.edu/documentation/projects/simprod_scripts/index.html

IceProd Modules

$I3_SRC/simprod-scripts/python/modules

- Corsika
- MuonGunGenerator
- NuGen
- GENIE
- CLSim
- PPC
- Detectors
- IceTop

Simprod-Scripts

http://software.icecube.wisc.edu/documentation/projects/simprod_scripts/index.html

Tray Segments

$I3_SRC/simprod-scripts/python/segments

Calibration

DetectorSim

GenerateAirShowers

GenerateCosmicRayMuons

GenerateFlashers

GenerateIceTopShowers

GenerateIceTopShowers

GenerateNeutrinos

GenerateNoiseTriggers

HybridPhotonicsCLSim

Polyplopia

PropagateMuons

simprod-scripts

http://software.icecube.wisc.edu/documentation/projects/simprod_scripts/examples/fullSimulation.html

Examples: $I3_SRC/simprod-scripts/resources/examples/fullSimulation.py

```
./fullSimulation.py -n 10 --seed=43682 --datasetnumber=1 --runnumber=1 --no-hybrid --icemodel=SpiceLea \
                    --detector=IC86 --unshadowed-fraction=0.99 --flavor=NuTau --outfile=taus.i3 --skip-calibration \
                    --from-energy=1000 --to-energy=10000000 --include-gcd-in-outfile
```

-n 10 Simulate 10 events (the output file might contain fewer events because not everything will trigger).

--seed=43682 A random number generator seed

--datasetnumber=1 --runnumber=1 A combination of "dataset" and "run" numbers. You can keep the same seed for all of them, each one will give a distinct independent set of events. (So you can think of the combination of seed, dataset and run as the actual random number seed. Or, put another way, there is no need to have different seeds for different runs.)

--**no-hybrid** Do not use hybrid simulation mode, i.e. propagate everything using direct simulation. There will be no photon tables and things like ice anisotropy/SpiceLea will work.

--icemodel=SpiceLea The other two models are "Spice1" and "SpiceMie". Including things like WHAM would be trivial if you need it.

--detector=IC86 This will select a GCD file from $I3_PORTS automatically. Currently works for IC86 and IC79.

--**unshadowed-fraction**=0.99 This is the "DOMEfficiency", currently named like this for compatibility with other tools that use the same name.

--flavor=NuTau You can set this also to "NuE" and "NuMu".

--**outfile**=... The name of your final .i3 file you want to generate.

--from-energy=1000 --to-energy=10000000 The energy range in GeV. Currently not implemented for MuonGun which uses fixed energy ranges. Should be fixed soon.

--**include-gcd-in-outfile** Use this option if you want to generate output files with GCD frames in them. It makes them much easier to use, but of course wastes some space..

simprod-scripts

Scripts: $I3_SRC/simprod-scripts/resources/scripts

(run the individual pieces as broken down by production tasks)

```
$  python nugen.py -h

                    Usage: nugen.py [options]

                    Options:
                      -h, --help            show this help message and exit
                      --no-execute          boolean condition to execute
                      --gcdfile=GCDFILE     GeoCalibDetStatus filename
                      --outputfile=OUTPUTFILE
                                            Output filename
                      --summaryfile=SUMMARYFILE
                                            XMLSummary filename
                      --mjd=MJD             MJD for the GCD file
                      --seed=RNGSEED        RNG seed
                      --procnum=RNGSTREAM
                    …
```

simprod-scripts

Exercise: Running scripts:

```
icecube@M16:~$ ssh submitter
[submitter]$
[submitter]$ condor_submit interactive_gpu.condor -interactive
Submitting job(s).
1 job(s) submitted to cluster 120263704.
Waiting for job to start…
Welcome to slot1@gtx-00.icecube.wisc.edu!

[gtx-00]$ cd $_CONDOR_SCRATCH_DIR
[gtx-00]$  /cvmfs/icecube.opensciencegrid.org/py2-v2/icetray-env simulation/V05-01-01
*******************************************************************
*                                                                 *
*                  W E L C O M E  to  I C E T R A Y               *
*                   Version combo.trunk     r147415               *
*                                                                 *
*                You are welcome to visit our Web site            *
*                      http://icecube.umd.edu                     *
*******************************************************************

[gtx-00]$ python $I3_BUILD/simprod-scripts/resources/scripts/nugen.py \
                 --gcdfile=/cvmfs/icecube.opensciencegrid.org/data/GCD/GeoCalibDetectorStatus_2016.57531_V0.i3.gz \
                 --outputfile=nutau.i3 --nevents=100 \
                 --seed=123 --procnum=0  --nproc=1 \
                 --FromEnergy=1e5 --ToEnergy=1e6 --NuFlavor=NuTau

[gtx-00]$ dataio-pyshovel nutau.i3
```

simprod-scripts

Exercise: Running scripts:

```
icecube@M16:~$ ssh submitter
[submitter]$
[submitter]$ condor_submit interactive_gpu.condor -interactive
Submitting job(s).
1 job(s) submitted to cluster 120263704.
Waiting for job to start…
Welcome to slot1@gtx-00.icecube.wisc.edu!

[gtx-00]$ cd $_CONDOR_SCRATCH_DIR
[gtx-00]$  /cvmfs/icecube.opensciencegrid.org/py2-v2/icetray-env simulation/V05-01-01
********************************************************************
*                                                                  *
*                   W E L C O M E  to  I C E T R A Y               *
*                    Version combo.trunk     r147415               *
*                                                                  *
*                 You are welcome to visit our Web site            *
*                         http://icecube.umd.edu                   *
********************************************************************

[gtx-00]$ python $I3_BUILD/simprod-scripts/resources/scripts/nugen.py \
                  --gcdfile=/cvmfs/icecube.opensciencegrid.org/data/GCD/GeoCalibDetectorStatus_2016.57531_V0.i3.gz \
                  --outputfile=nutau.i3 --nevents=100 \
                  --seed=123 --procnum=0  --nproc=1 \
                  --FromEnergy=1e5 --ToEnergy=1e6 --NuFlavor=NuTau

[gtx-00]$ dataio-pyshovel nutau.i3
```

simprod-scripts

Exercise: Running scripts**:**

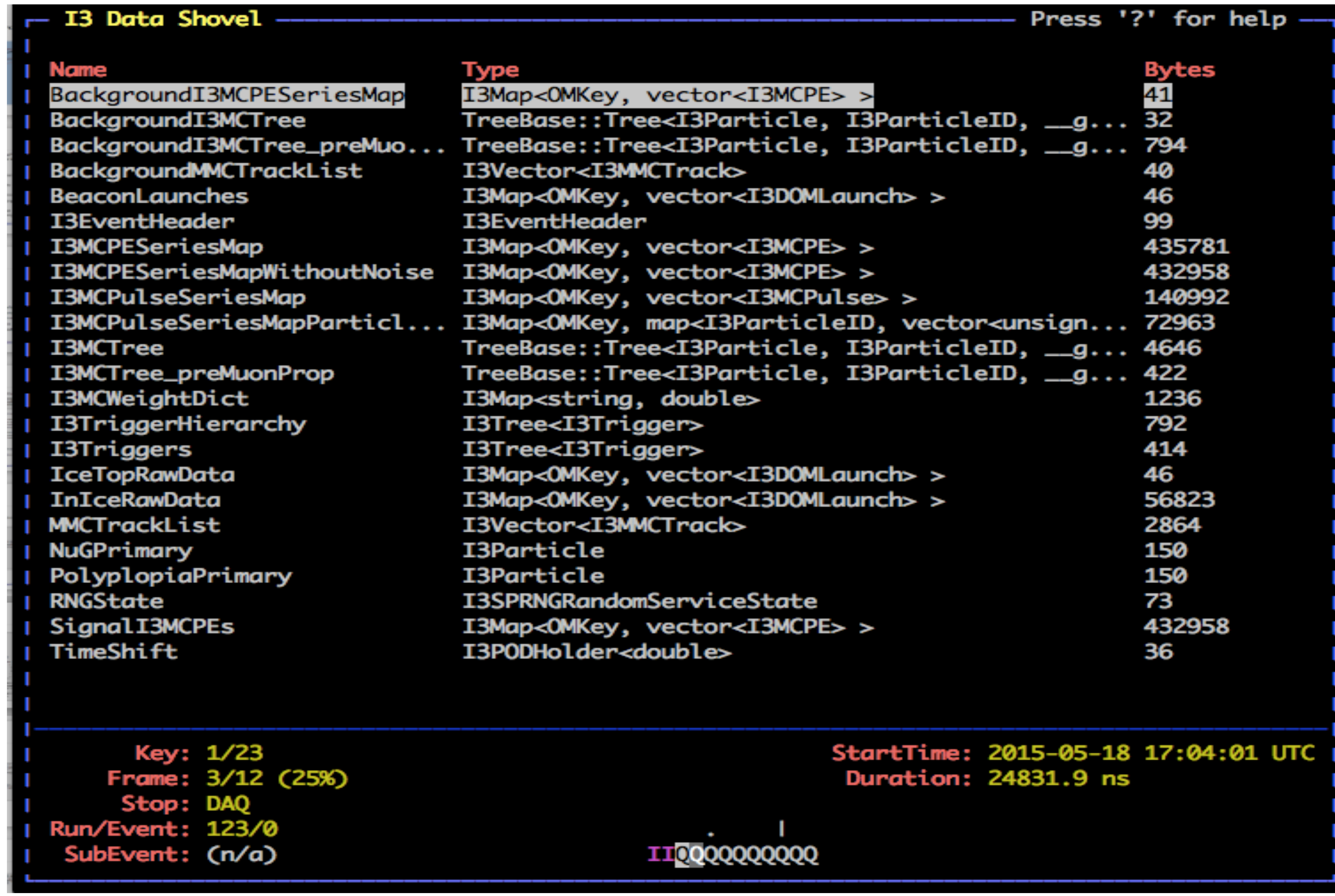
```
[gtx-00]$ python $I3_BUILD/simprod-scripts/resources/scripts/clsim.py \
                —gcdfile=/cvmfs/icecube.opensciencegrid.org/data/GCD/
GeoCalibDetectorStatus_2016.57531_V0.i3.gz \
                --inputfilelist=nutau.i3 --outputfile=mcpes.i3   \
                --seed=123 --procnum=0 --nproc=1 --no-RunMPHitFilter \
                --UseGPU

[gtx-00]$ dataio-pyshovel mcpes.i3

[gtx-00]$ python $I3_BUILD/simprod-scripts/resources/scripts/detector_ic86.py \
                --gcdfile=/cvmfs/icecube.opensciencegrid.org/data/GCD/
GeoCalibDetectorStatus_2016.57531_V0.i3.gz \
                --inputfile=mcpes.i3 --outputfile=det.i3 \
                --seed=123 --procnum=0 --nproc=1 --RunID=123

[gtx-00]$ dataio-pyshovel det.i3
```

simprod-scripts

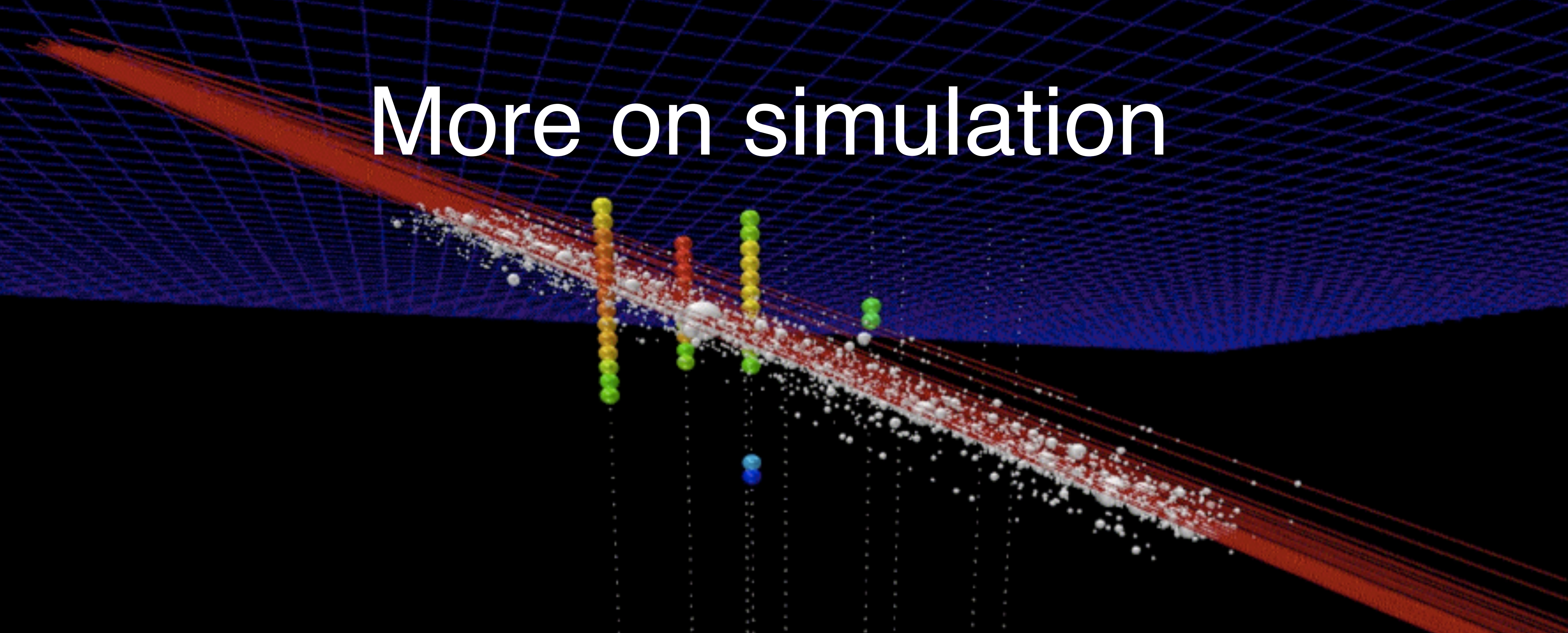
Exercise: Running scripts**:**

```
[gtx-00]$ python $I3_BUILD/simprod-scripts/resources/scripts/corsika.py \
                 --nshowers=10000 --outputfile=corsika_bg.i3 --seed=1234 \
                 --procnum=0 --nproc=1 --CORSIKAseed=123 --ranpri=2  \
                 --corsikaVersion=v6960-5comp \
                 --corsikaName=dcorsika --skipoptions=compress

[gtx-00]$ dataio-pyshovel corsika_bg.i3

[gtx-00]$ python $I3_BUILD/simprod-scripts/resources/scripts/polyplopia.py
\
               --gcdfile=/cvmfs/icecube.opensciencegrid.org/data/GCD/
GeoCalibDetectorStatus_2016.57531_V0.i3.gz \
               --inputfile=mcpes.i3 --outputfile=merged_pes.i3 \
               --seed=1234 --procnum=0 --nproc=1 --
backgroundfile=corsika_bg.i3 --mctype=NuTau

[gtx-00]$ python $I3_BUILD/simprod-scripts/resources/scripts/
detector_ic86.py \
                 --gcdfile=/cvmfs/icecube.opensciencegrid.org/data/GCD/
GeoCalibDetectorStatus_2016.57531_V0.i3.gz \
                 --inputfile=merged_pes.i3 --outputfile=det_wcoinc.i3 \
                 --seed=123 --procnum=0 --nproc=1 --RunID=123

[gtx-00]$ dataio-pyshovel det_wcoinc.i3
```

Frame Objects



More on simulation



1. http://wiki.icecube.wisc.edu/index.php/Simulation_Documentation_Wiki
2. Weighting tutorial: https://grid.icecube.wisc.edu/simulation/weighting/example.htm
3. http://software.icecube.wisc.edu/documentation/
4. http://wiki.icecube.wisc.edu/index.php/Simulation_Production
5. http://grid.icecube.wisc.edu/simulation/table
6. SLACK: #simulation