How to Make a Monte Carlo Production

Kuunal Kelash Mahtani
June 19th, 2023

Stony Brook University

Geant4 & EDepSim

- Simulations of particles interacting in matter
  - Geant - toolkit using MonteCarlo techniques
  - https://geant4.web.cern.ch/
- Requires input information
  - Detector geometry, materials
  - Most functions (wrapper)

- EDepSim - the **E**nergy **Dep**osition **Sim**ulation
  - Provides ROOT file to record Geant output
  - Can be linked as a library
  - Implements energy deposition as ionizing and non-ionizing energy loss
  - https://github.com/ClarkMcGrew/edep-sim

Producing a Geant4 Output

- Using EDepSim as a wrapper
- To use EDepSim, we provide it with the following:

Geometry (can check for overlaps)	Provided by .gdml file
Number of events	Command line argument
Random seed (Reproducibility)	Command line argument
Physics list (Intranuclear Cascade Model)	Command line argument
Macro file	Provided by a .mac file

```
edep-sim \
    -C \
    -g ${GEOMETRY} \
    -o ${OUTDIR}/${PROTONAME}.edepsim_${CONFIGG}_${PHYSICSLIST}_${LOWEND}MeV_${HIGHEND}MeV.root \
    -e ${NEVT} \
    -s ${RNDSEED} \
    -p ${PHYSICSLIST} \
    -u \
    LANL_${LOWEND}MeV_${HIGHEND}MeV.mac
```

- Shell script produced for easy implementation
  - launchjobarray.sh
- Produces 1 MeV intervals by default

Produce a MC Output

- https://github.com/neutron-lanl tb-analysis/neutronsimulation
  - Clone this repository
  - All relevant scripts in this github

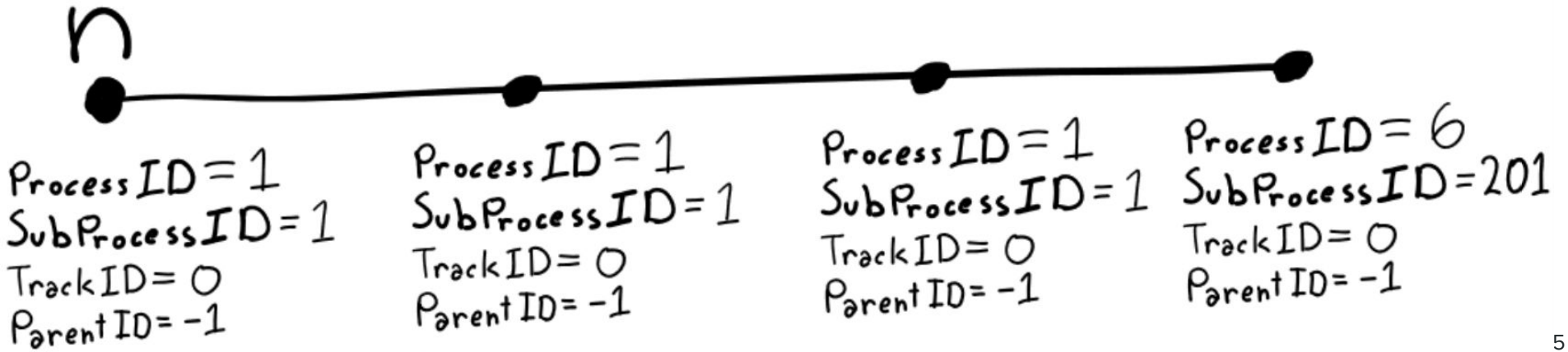
- Edit shell script:
  - Outfile to /your/path/to/directory/
  - Rest of naming convention to be followed

```
$ bash run_edepsim_LANL.sh 300 301
```

- MCProduction Directory
  - Geometry File LANL_24x8x48_wFiber_wDead_realSize_wTyvek_noGap_v7.gdml
  - Macro File neutron-flat3.mac
  - Shell script for single run run_edepsim_LANL.sh

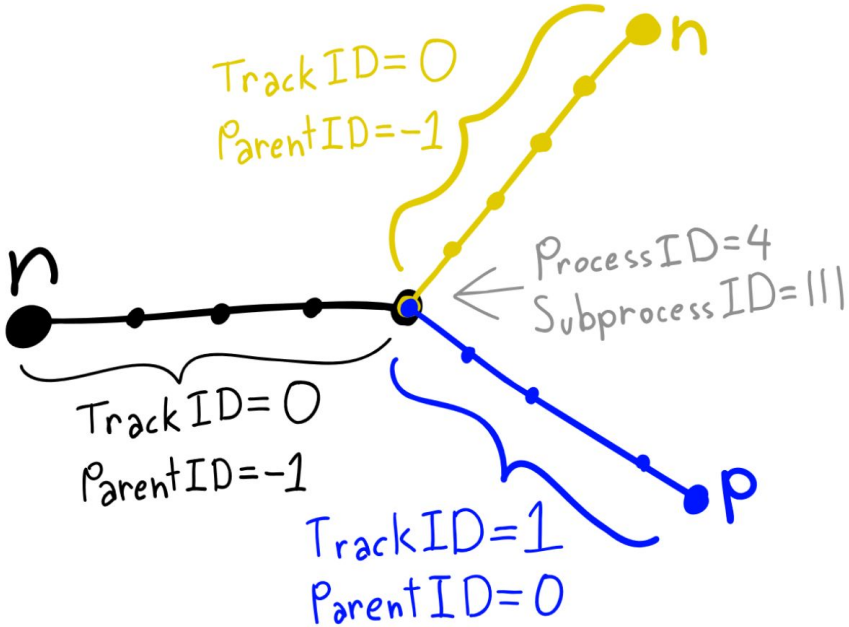
Event Organization in Geant4

- 1 Trajectory/tracked particle
  - Particle number in stack unique to trajectory
- Kinematics information stored per point
  - Process, SubProcess, total energy, momentum, position, etc.



Example: Neutron Hadronic Elastic Scattering off Free Hydrogen

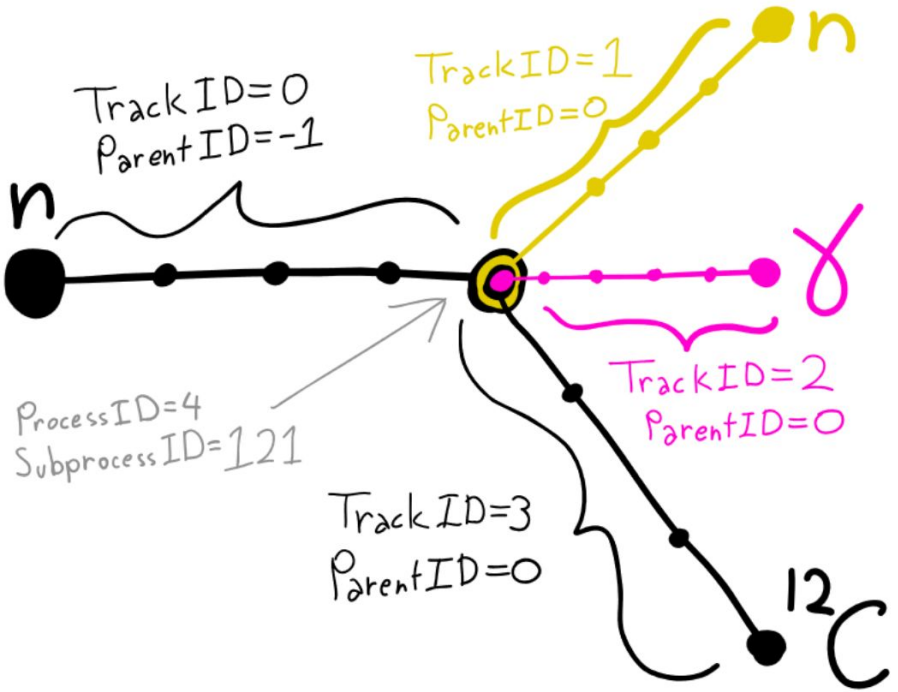
- 1 Trajectory per particle: **Incoming Neutron**, **Outgoing Neutron**, **Outgoing Proton**
- Each trajectory has a TrackID and a ParentID; these index the particle as well as it's parent
- Each trajectory has multiple points, and each point has a Subprocess ID and a Process ID
- The Process ID: **interaction process**
- The Subprocess ID: **interaction type**



Process	SubProcess
Hadronic	Elastic Interaction
Transport	Inelastic Interaction
etc.	etc.

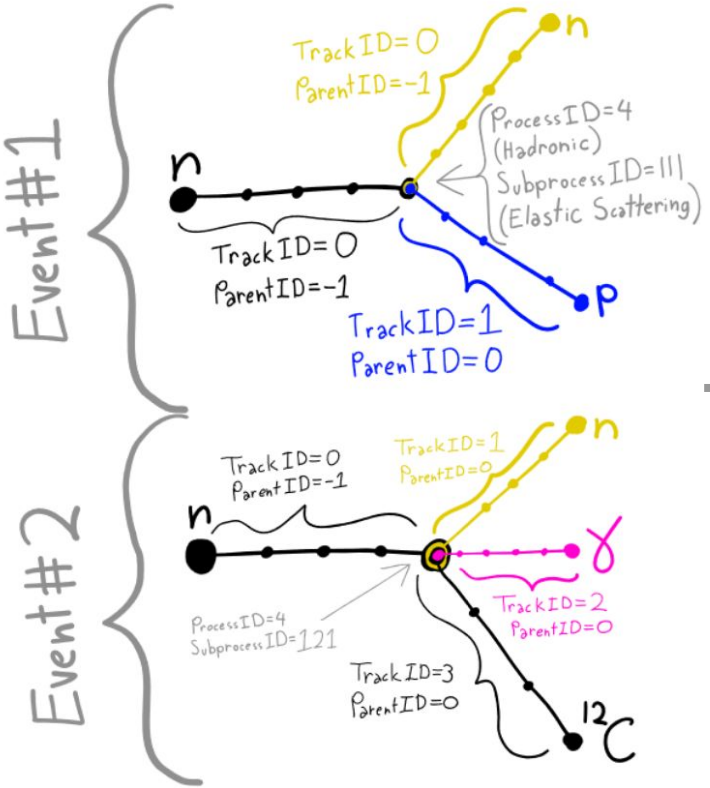
Neutron Hadronic Inelastic Scattering off Carbon

- Incoming & outgoing neutron have different tracks
  - Geant4's handling of intranuclear interactions
- Interaction Vertex
  - Interaction type
  - Relative to start position of outgoing particles ➔ Vertex Cut
  - Kinematics



Dumping to a Tree - General

- Reformat simulation to resemble data



Row	Instance	neut_inc_	neut_inc_
0	1	1	91
1	0	1	91
1	1	1	91
2	0	1	91
2	1	1	91
3	0	1	91
3	1	1	91
4	0	1	91
4	1	1	91
5	0	1	91
5	1	4	111
5	2	1	91
6	0	2	22
6	1	1	91
6	2	1	91
7	0	1	91
7	1	1	91
7	2	1	91
8	0	1	91
8	1	4	121
8	2	1	91
9	0	1	91
9	1	4	121
9	2	1	91
10	0	2	22

Structured

Unstructured - Tree of values

Dumping to a Tree - Specifics

- Loop over all hits in all events
- Identify particle characteristics for each event
- Store in predefined variables
  - Most are arrays of ints/floats, size 200

```
for iloop in nTraj: #nTraj is an array of track ID's for each trajectory (@1) for each event, so each iloop is a TrackID value
    if hit.Contrib[0] == iloop and iN < 200:
        t_id_maxContrib[iN] = iloop
        t_length_maxContrib[iN] = (hStop-hStart).Mag()
        trjXStartPosHitSeg[iN] = hStart.x() #the x, y, z positions at which the hit segment begins
        trjYStartPosHitSeg[iN] = hStart.y()
        trjZStartPosHitSeg[iN] = hStart.z()
        trjXEndPosHitSeg[iN] = hStop.x() #the x, y, z positions at which the hit segment ends
        trjYEndPosHitSeg[iN] = hStop.y()
        trjZEndPosHitSeg[iN] = hStop.z() #these will give the start and the end of the hit segments
        tgeo.FindNode( hStart.x(),hStart.y(),hStart.z() )
        t_node[iN] = tgeo.GetCurrentNodeId()
        trjStartTMaxContrib[iN] = hit.GetStart().T() #start time
        trjEndTMaxContrib[iN] = hit.GetStop().T() #stop time
        trjEDepMaxContrib[iN] = hit.GetEnergyDeposit()
        trjPdgMaxContrib[iN] = event.Trajectories[iloop].GetPDGCode()
        trjLenMaxContrib[iN] = hit.GetTrackLength() #Hit length, NOT THE TRAJECTORY LENGTH, this is track length per hit
        trjIdMaxContrib[iN] = event.Trajectories[iloop].GetTrackId()
        trjSubPMaxContrib[iN] = event.Trajectories[iloop].Points[0].GetSubprocess()
        trjProcMaxContrib[iN] = event.Trajectories[iloop].Points[0].GetProcess()
        t0_MaxContrib[iN] = int(1800 * int((ient - current_time * int(347/freq)) * freq + aRann));
        t_spill_MaxContrib[iN] = current_time;
        t_gamma_time_MaxContrib[iN] = t0_MaxContrib[iN] + flightDis/0.3  ;
```

- Construct vector of hit segments objects to loop over

```
for hit in hits:
    hStart = ROOT.TVector3( hit.GetStart().X()/10., hit.GetStart().Y()/10., hit.GetStart().Z()/10. )
    hStop = ROOT.TVector3( hit.GetStop().X()/10., hit.GetStop().Y()/10., hit.GetStop().Z()/10. )
```

- Obtain information from hit segments

```
hits = []
counter = 0
for key3 in event.SegmentDetectors:
    if "volCube" in key3.first:
        if (counter == 0):
            hits = key3.second
            counter = 1
        else:
            hits += key3.second
TrckIDMaxContrib = 100*[-999]
```

Dumping to a Tree - Specifics

- Loop over all hits in all events
- Identify particle characteristics for each event
- Store in predefined variables
  - Most are arrays of ints/floats, size 200
- **`TG4HitSegment`** Objects
  - Hit segment corresponds to a single voxel
  - Script specific to SFGD & Prototype
- Multiple voxels per event
- Vector of hit segments

```
hits = []
counter = 0
for key3 in event.SegmentDetectors:
    if "volCube" in key3.first:
        if (counter == 0):
            hits = key3.second
            counter = 1
        else:
            hits += key3.second
TrckIDMaxContrib = 100*[-999]
```

- Obtain information from hit segments

```
for hit in hits:
    hStart = ROOT.TVector3( hit.GetStart().X()/10., hit.GetStart().Y()/10., hit.GetStart().Z()/10. )
    hStop = ROOT.TVector3( hit.GetStop().X()/10., hit.GetStop().Y()/10., hit.GetStop().Z()/10. )
```

Dump MC Production to a Tree

- One Line implementation
- ```
  python3 dumpTreeNeutron_LANL4.py
  --topdir='/path/to/directory'
  --first_run=X --last_run=Y
  --outfile=/path/to/output/file.root
  ```
- X = start energy, Y = end energy (MeV)
- Let's do 300, 301 for consistency with previous example

- DumpTree Directory
  - python3 file
  - Note line **791**
    - Naming convention manually input here
  - Important information located and dumped in loop with line **415**
  - Branches to store information defined from line **558**
    - **DATA TYPE MUST BE CONSISTENT**
  - Variables branches point to initialized from line **71**
  ```

Restructuring Events

- Selection cuts on each entry in tree
  - Organize information to represent neutron interactions
- Event objects
  - Class of objects containing particle interaction information
- Output:
  - Resembles organization of information we expect from data
  - Usable file format for analyses

```
Float_t trjXStartPosHitSeg[200];
Float_t trjYStartPosHitSeg[200];
Float_t trjZStartPosHitSeg[200];
Float_t trjXEndPosHitSeg[200];
Float_t trjYEndPosHitSeg[200];
Float_t trjZEndPosHitSeg[200];
Float_t trjEDepMaxContrib[200];
Float_t trjLenMaxContrib[200];
Int_t trjPdgMaxContrib[200];
Int_t trjIdMaxContrib[200];
Int_t trjSubPMaxContrib[200];
Int_t trjProcMaxContrib[200];
```

```
t.SetBranchAddress("trjXStartPosHitSeg", trjXStartPosHitSeg);
t.SetBranchAddress("trjYStartPosHitSeg", trjYStartPosHitSeg);
t.SetBranchAddress("trjZStartPosHitSeg", trjZStartPosHitSeg);
t.SetBranchAddress("trjXEndPosHitSeg", trjXEndPosHitSeg);
t.SetBranchAddress("trjYEndPosHitSeg", trjYEndPosHitSeg);
t.SetBranchAddress("trjZEndPosHitSeg", trjZEndPosHitSeg);
t.SetBranchAddress("trjEDepMaxContrib", trjEDepMaxContrib);
t.SetBranchAddress("trjStartTMaxContrib", trjStartTMaxContrib);
t.SetBranchAddress("trjEndTMaxContrib", trjEndTMaxContrib);
t.SetBranchAddress("trjPdgMaxContrib", trjPdgMaxContrib);
t.SetBranchAddress("trjLenMaxContrib", trjLenMaxContrib);
t.SetBranchAddress("trjIdMaxContrib", trjIdMaxContrib);
t.SetBranchAddress("trjSubPMaxContrib", trjSubPMaxContrib);
t.SetBranchAddress("trjProcMaxContrib", trjProcMaxContrib);
```

Defining variables and reading from branches

```
for (int ivt = 0; ivt < 200; ivt++)
{
  event->SetHitSegStart(ivt, 0, (float)trjXStartPosHitSeg[ivt] + 12.12);
  event->SetHitSegStart(ivt, 1, (float)trjYStartPosHitSeg[ivt] + 4.1);
  event->SetHitSegStart(ivt, 2, (float)trjZStartPosHitSeg[ivt] + 25.24);
  event->SetHitSegStart(ivt, 3, (float)trjStartTMaxContrib[ivt]);
  event->SetHitSegEnd(ivt, 0, (float)trjXEndPosHitSeg[ivt] + 12.12);
  event->SetHitSegEnd(ivt, 1, (float)trjYEndPosHitSeg[ivt] + 4.1);
  event->SetHitSegEnd(ivt, 2, (float)trjZEndPosHitSeg[ivt] + 25.24);
  event->SetHitSegEnd(ivt, 3, (float)trjEndTMaxContrib[ivt]);
  event->SetTrajectoryMaxLen(ivt, (float)trjLenMaxContrib[ivt]);
  event->SetTrajectoryMaxPdg(ivt, (int)trjPdgMaxContrib[ivt]);
  event->SetTrajectoryMaxEDep(ivt, (float)trjEDepMaxContrib[ivt]);
  event->SetTrajectoryMaxId(ivt, (int)trjIdMaxContrib[ivt]);
  event->SetTrajectoryMaxSubP(ivt, (int)trjSubPMaxContrib[ivt]);
  event->SetTrajectoryMaxProc(ivt, (int)trjProcMaxContrib[ivt]);
```

Using Setter functions after selection
Defined in Events.hh (touched upon later)

Restructure from Tree

- Build:
  - `"data_preprocessing/build/"`
  - Remove all files, `"cmake ../"`
  - `"make"`
- `/bin/EventStructure_MC_update 1000 "/path/to/input/file.root" "/path/to/output/file.root"`
- Select output file of dumptree (`dumpTreeNeutron_LANL4.py`) as input file of this script (`"/path/to/input/file.root"`)

- Event Structure Directory
  - Executable located in `"data_preprocessing/bin/"`
  - Information setting starts from ln **778**
  - Setter functions defined in `"Event.hh"`, located in `"analysis/src/classes/"`
    - Variables, setter and getter functions defined
    - **DATA TYPE MUST BE CONSISTENT**
  - C++ code: must be compiled

Sbatch

- Shell scripts produced to batch many jobs
  - Located in shell_scripts/ directory
  - Each shell script launches Dumptree or Event Structure scripts
  - Hardcoded to batch 800 jobs, 8 jobs at a time - can be changed
- Standard format is to use 1 MeV intervals
- Standard format is to use naming convention specified

```
#!/bin/bash
#
#SBATCH --cpus-per-task=1
#SBATCH --mem=8G
#SBATCH --array 0-799%8

#The path to the version of root and geant 4 need to be sourced on your local machine, these are predefined for
our NNHome cluster
source /home/riccioc/root/6.24.06/root_install/bin/thisroot.sh
source /home/riccioc/geant4/geant4.10.06.p03-install/bin/geant4.sh

echo "Job number ${SLURM_ARRAY_TASK_ID}"

i=${SLURM_ARRAY_TASK_ID}

#The path to the directory where the MC production was created 'topdir' should be changed accordingly
#The path to the directory where the output file is stored and the name of the output file 'outfile' should be
changed accordingly

python3 ../DumpTree/dumpTreeNeutron_LANL4.py --topdir='/storage/shared/LANL/geant4output/2019/
bertini-upstream_material' --first_run=$((i)) --last_run=$((i)) --outfile=/storage/shared/mahtani/
Conversion_Results/bertini-upstream_material/DumpTree_Output/DT_$((i))_MeV.root;
```

Example: Dumptree shell script

Note: Always be careful when submitting batch jobs to not take up too many computing resources

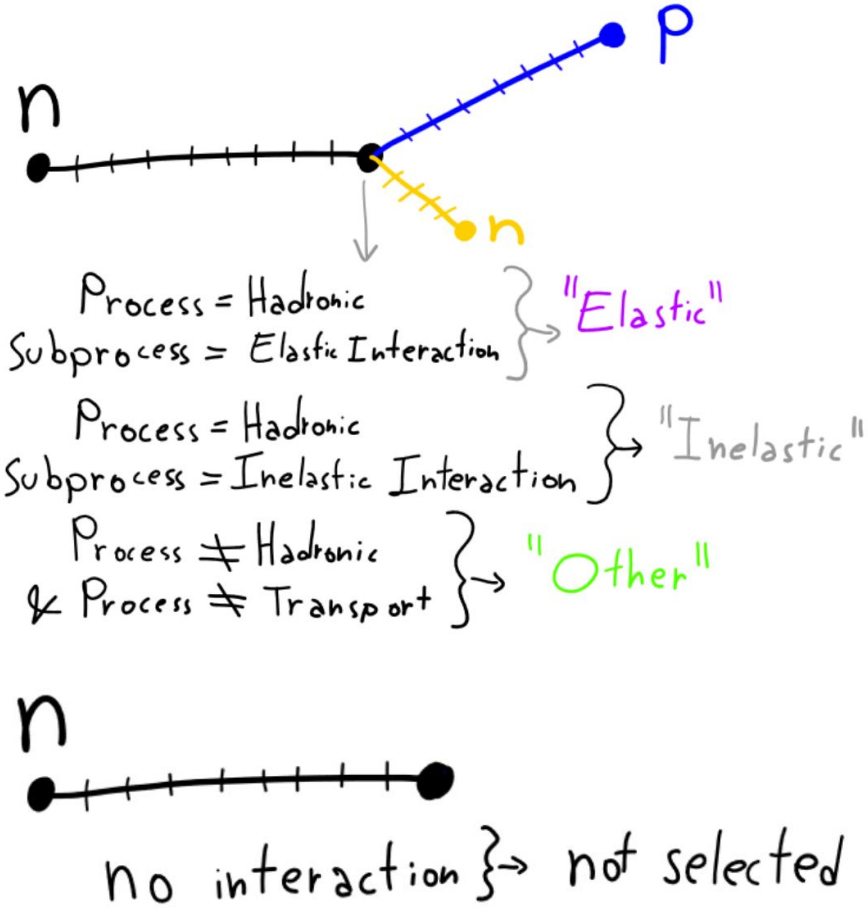
Notable saved variables

- Can determine most particle kinematics with this information
- Hit object
  - Computational object resembling ‘hit’ from a single detector cube

Variable Name	Meaning
trjXStartPosHitSeg	Start position in X of EACH hit segment
trjYStartPosHitSeg	Start position in Y of EACH hit segment
trjZStartPosHitSeg	Start position in Z of EACH hit segment
trjStartTMaxContrib	Start time of EACH hit segment
trjXEndPosHitSeg	End position in X of EACH hit segment
trjYEndPosHitSeg	End position in Y of EACH hit segment
trjZEndPosHitSeg	End position in Z of EACH hit segment
trjEndTMaxContrib	End time of EACH hit segment
trjLenMaxContrib	Length of EACH hit segment
trjPdgMaxContrib	Particle type contributing to each hit segment
trjEDepMaxContrib	Energy deposition in each hit segment
trjIdMaxContrib	Track ID
trjSubPMaxContrib	Subprocess for the FRIST POINT in EACH trajectory
trjProcMaxContrib	Process for the FIRST POINT in EACH trajectory

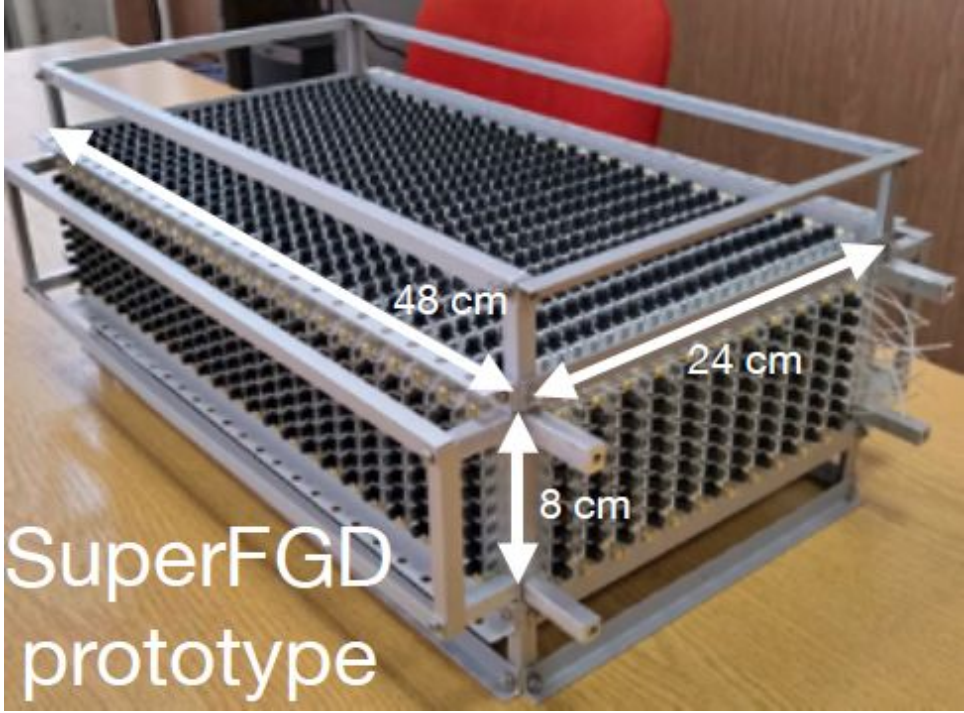
Backup

- For our selection, we only look at outgoing particles from the first Hadronic interaction from the incoming neutron
- If the incoming neutron has:
  - An Elastic Hadronic interaction, we classify this as “**Elastic**”
  - An Inelastic Hadronic interaction, we classify this as “**Inelastic**”
  - An Non-Hadronic interaction, we classify this as “**Other**”
  - **No interaction** (if we only have Transport), we do not consider these events
- Since we don’t store the events with no neutron interaction, our stacked histogram for incoming neutron energy is not flat
  - If we were to store these in another category, we would have a flat stacked distribution



Prototypes

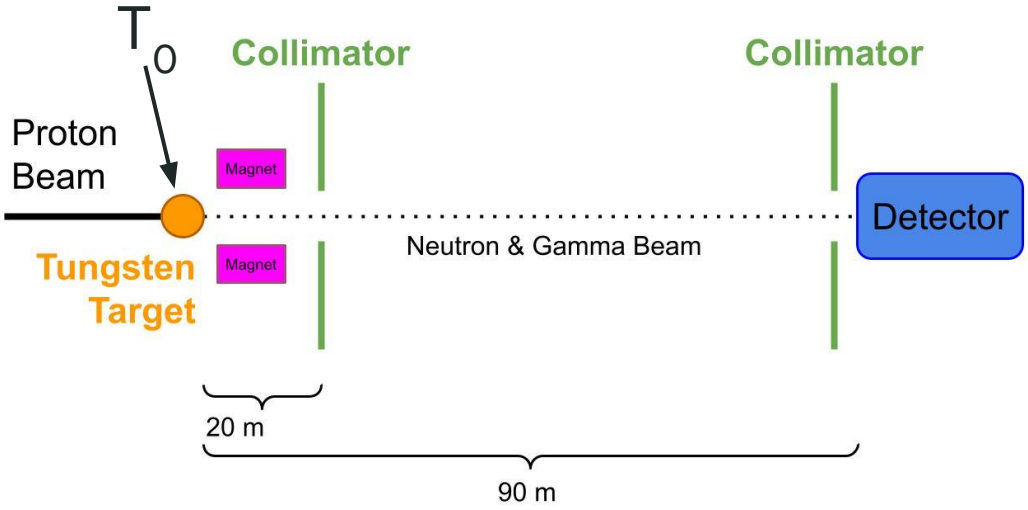
- 2 Prototypes constructed to prove SuperFGD technology & study detector performance
  - SuperFGD - 24x8x48 cubes
  - US-Japan - 8x8x32 cubes
- Assembled at CERN
- Tested at Los Alamos National Lab (LANL) Weapons Neutron Research (WNR) facility
  - Data taken in 2019, 2020

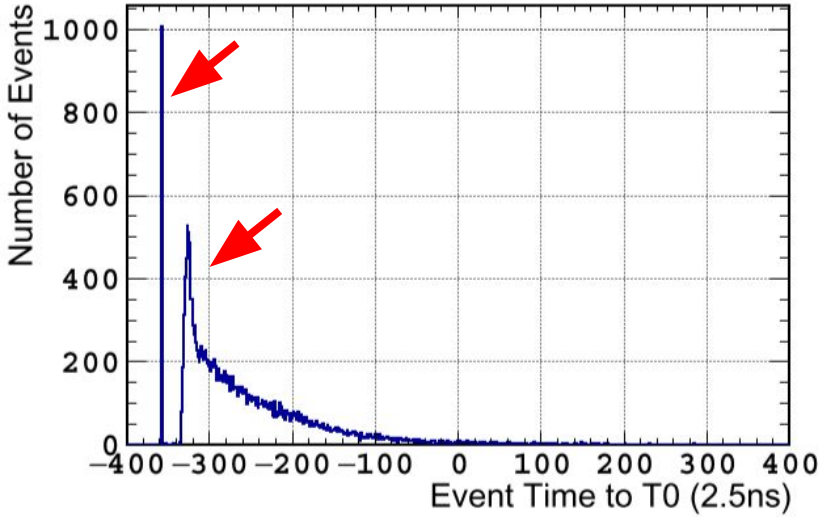


Credit: Ciro Riccio

LANL Beam Test

- Exposure to Neutron beam, 0-800 MeV
- Neutron arrival time relative to gamma ➔ neutron (ToF)
- Measured total neutron cross section on CH
- Results published in Phys. Lett. B
- Proved Capability of SuperFGD to measure neutron kinematics using ToF
- First physics result!

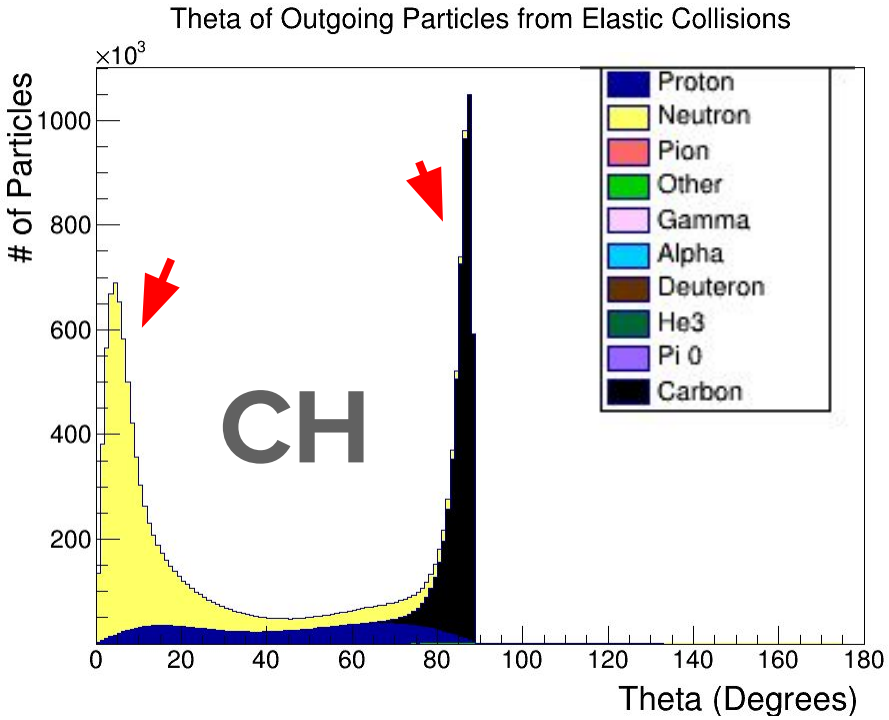




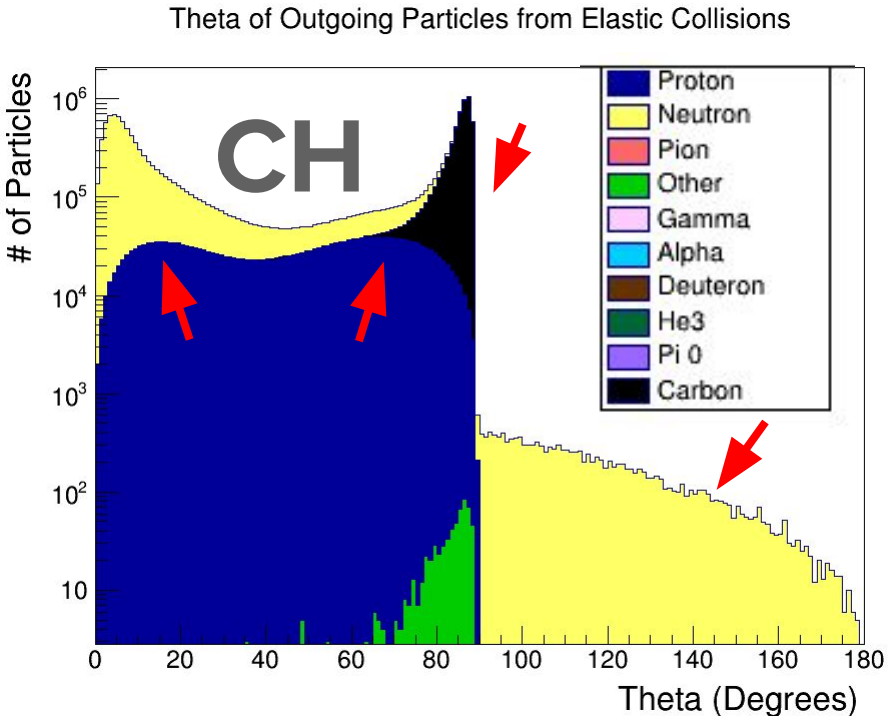
A. Agarwal et al. Total neutron cross-section measurement on CH with a novel 3D-projection scintillator detector. Phys. Lett. B, 840:137843, 2023.

θ of Outgoing Particles from Elastic collisions on CH

(Stacked)



Linear Scale



Logarithmic Scale

Single Track Selection

- Recall: LANL Test Beam Paper Aims:
  - Measured total neutron cross section looking at depletion of # of events along detector
    - Selected neutron interactions with 1 outgoing charged particle - clear vertex identification
- Reduction & restructuring of MC simulations to resemble data
- Computed distances between reconstructed tracks and truth
  - Verify matching between reco & true
- Percentage of particle types contributing to single track events

Purity Analysis

- Largest distance between true and reconstructed tracks is 8.3mm, < half diagonal cube
- Purity analysis
  - Particle with maximum energy deposition per single track event
  - Particle type using MC information

Particle Type	Purity
p	84.7%
$\pi^{\pm}$	5.9%
α	2.7%
e^{-}	2.5%
${}^{8}C - {}^{13}C$	1.3%
${}^{2}H$	1.0%
e^{+}	0.3%
${}^{3}He$	0.3%
μ^{+}	0.01%
μ^{-}	$< 0.01\%$
γ	$< 0.01\%$
n	$< 0.01\%$
Others	1.19%

First Interaction Process (`ProcessID`)	First Interaction Type (`SubProcessID`)	Purity
Hadronic	Inelastic Scattering	56.0%
	Elastic Scattering	41.2%
	Hadron at Rest	$< 0.01\%$
Electromagnetic	Compton Scattering	2.3%
	Gamma Conversion	0.3%
	Ionization	$< 0.01\%$
Decay	Decay	0.2%

Ongoing & Future Work

- SuperFGD
  - Neutron kinematics measurements for the first time!
  - Data taking in November
- Analyses on simulations of neutron interactions on CH show features consistent with ENDF data
  - Features of Geant4 version require further investigation
- Protons contribute most to single track events
- 56% of single track events from inelastic scattering, 41.2% from elastic

- Comparisons of analysis on MC simulations to 2019, 2020 LANL beam test data
  - Particle multiplicity, outgoing particle angles, kinetic energies, etc.
- Investigations into different software versions to understand limitations of simulation analyses better

Outline

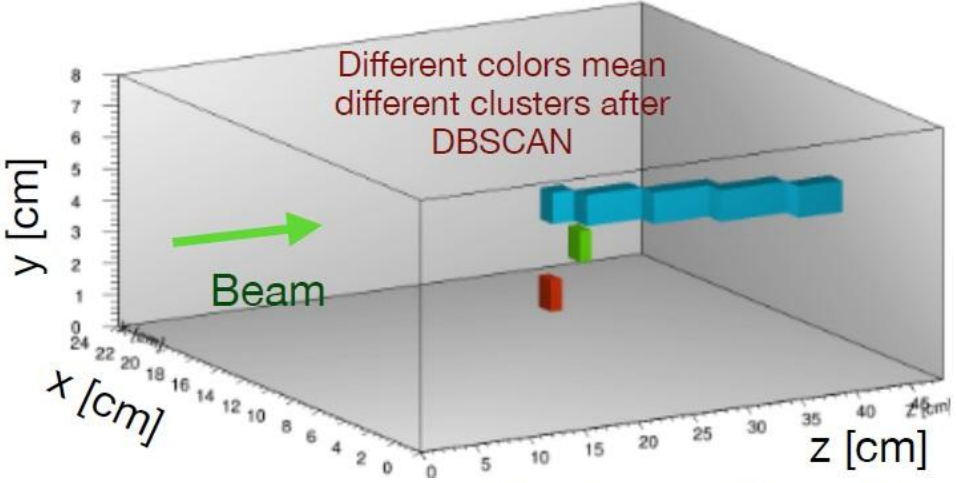
- G4 & EDep Background
- G4 Theory (particle assignments)
- What we want to do in a nutshell (g4, dt, edep - cover this in one slide)
- How do? G4
- How do? DT
  - Variables that are saved
- How do? EDep
  - Note that previous variables are saved, note that we can save the beam flux here too

Geant4 & EDepSim

- We aim to create simulations of particles interacting in matter
  - Neutron beam test at LANL
- In order to do this, we employ Geant - a toolkit used to simulate particle interactions in matter
- Geant uses MonteCarlo techniques and requires information of the geometry and materials in the detector to perform these simulations
- We use the 4th version of Geant - Geant4
- For our purposes, we are also interested in using EDepSim - the **E**nergy **Dep**osition **Sim**ulation - a wrapper for Geant4
- Geant4 requires users to provide most programs, and edep-sim serves as a wrapper to do so, allowing users to concentrate on physics
- edep-sim provides a ROOT based file to record the output of GEANT4
- Ede-sim can also be linked as a library to another application and makes the same information as a class
- EdepSim implements energy deposition as ionizing and non-ionizing energy loss, using the EM saturation model provided for Geant4 for all models except for liquid argon (which uses NEST)

Single Track Selection

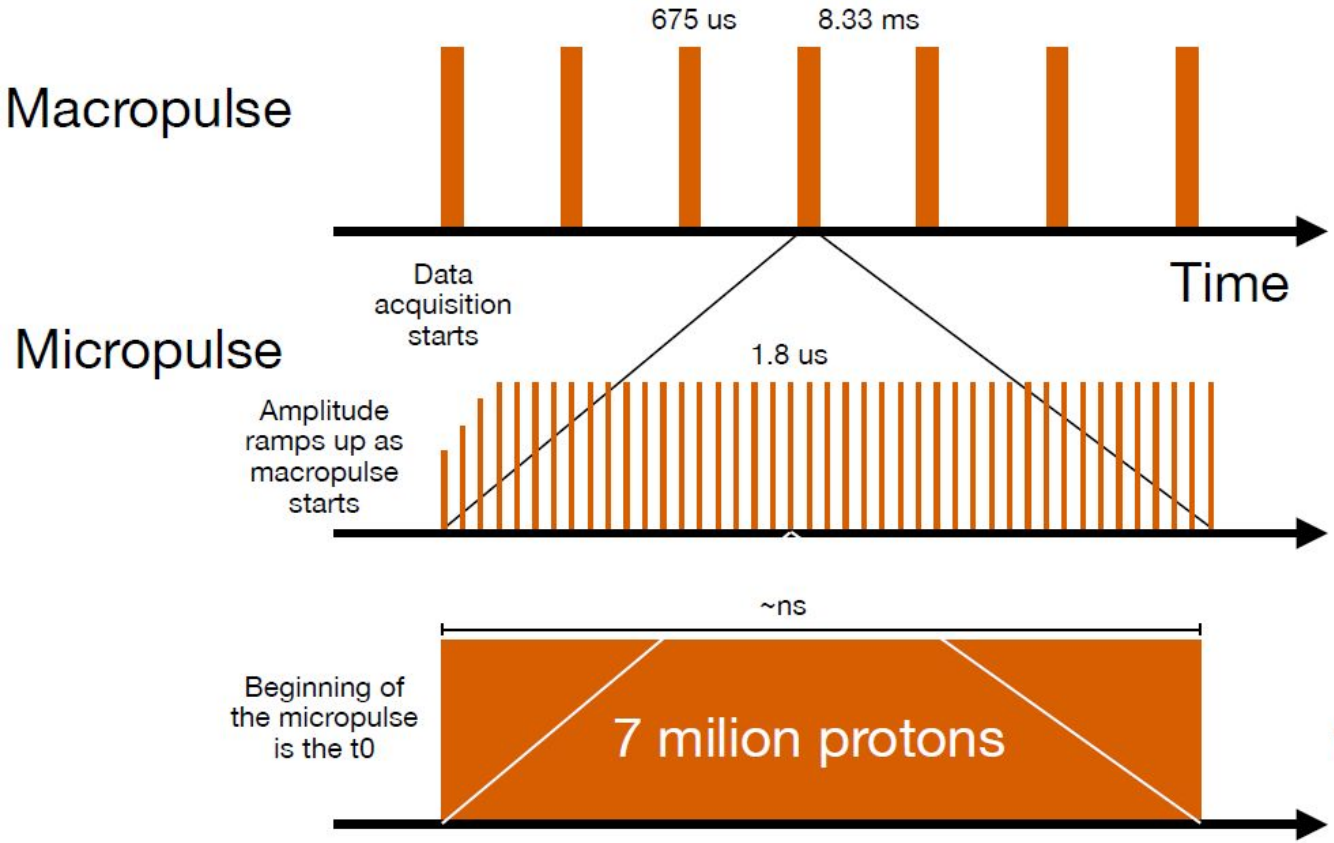
- Minimum energy deposit threshold
- >3 voxels (cubes), each with PE>20
- Within fiducial volume
- Clustering algorithm (DBSCAN)
- Cluster width within 1.7cm (cube diagonal)



Credit: Eric Zhong

Credit: Ciro Riccio

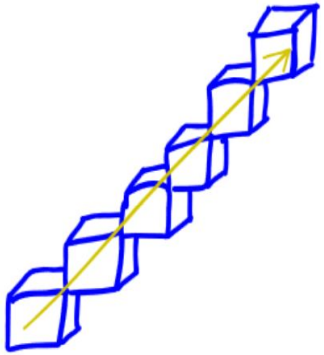
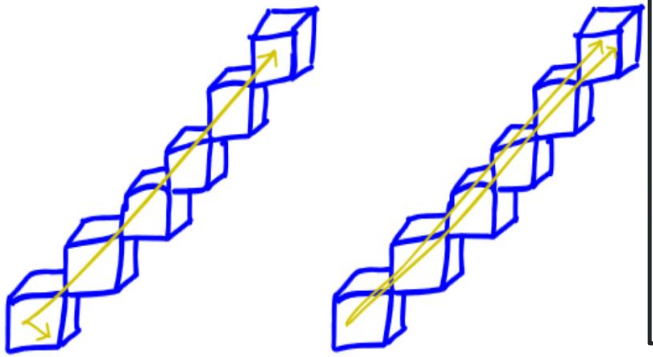
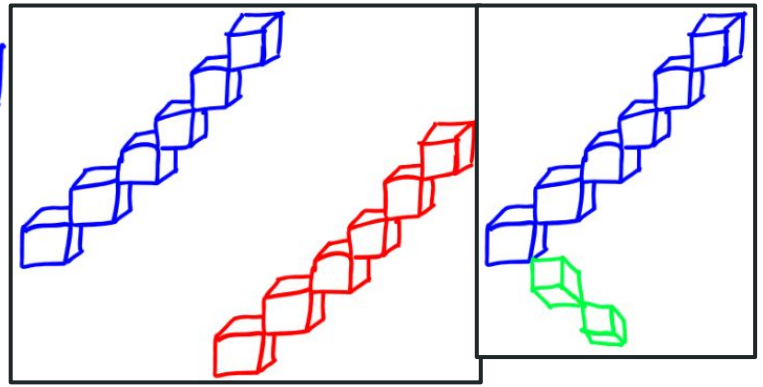
Beam Structure



Protons impinge on the target and produce neutrons and gammas.

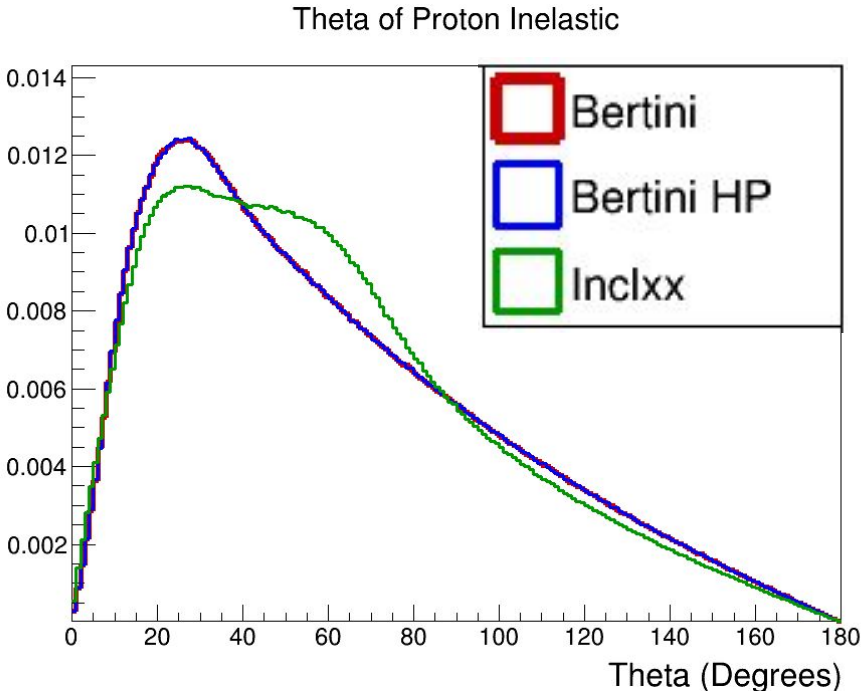
Credit: Ciro Riccio

Single Track Selection: Physics Perspective

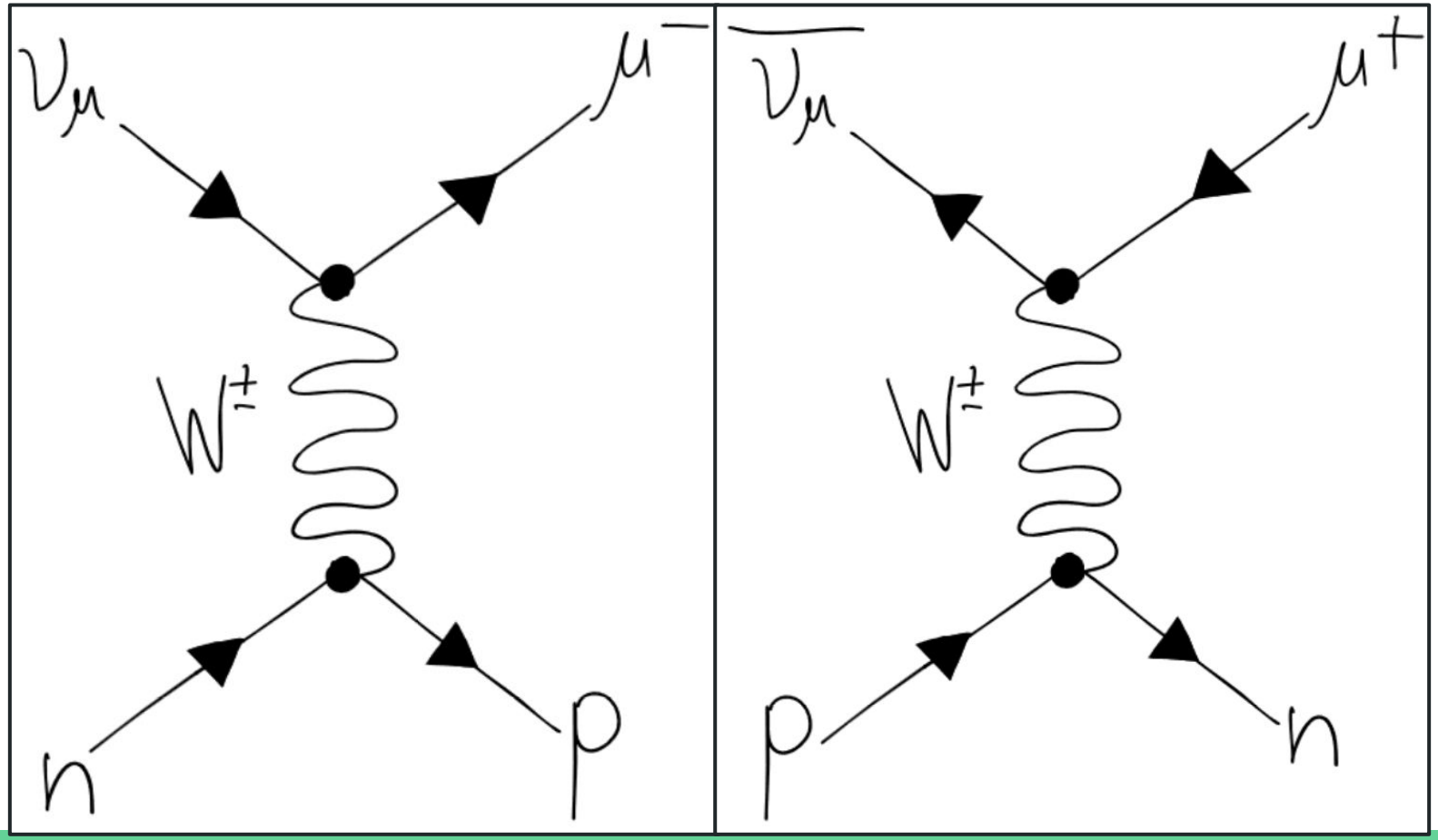
	Appears as Single Track	Appears as Multiple Track
Is Single Track		
Is Multiple Track		

Physics List Comparisons

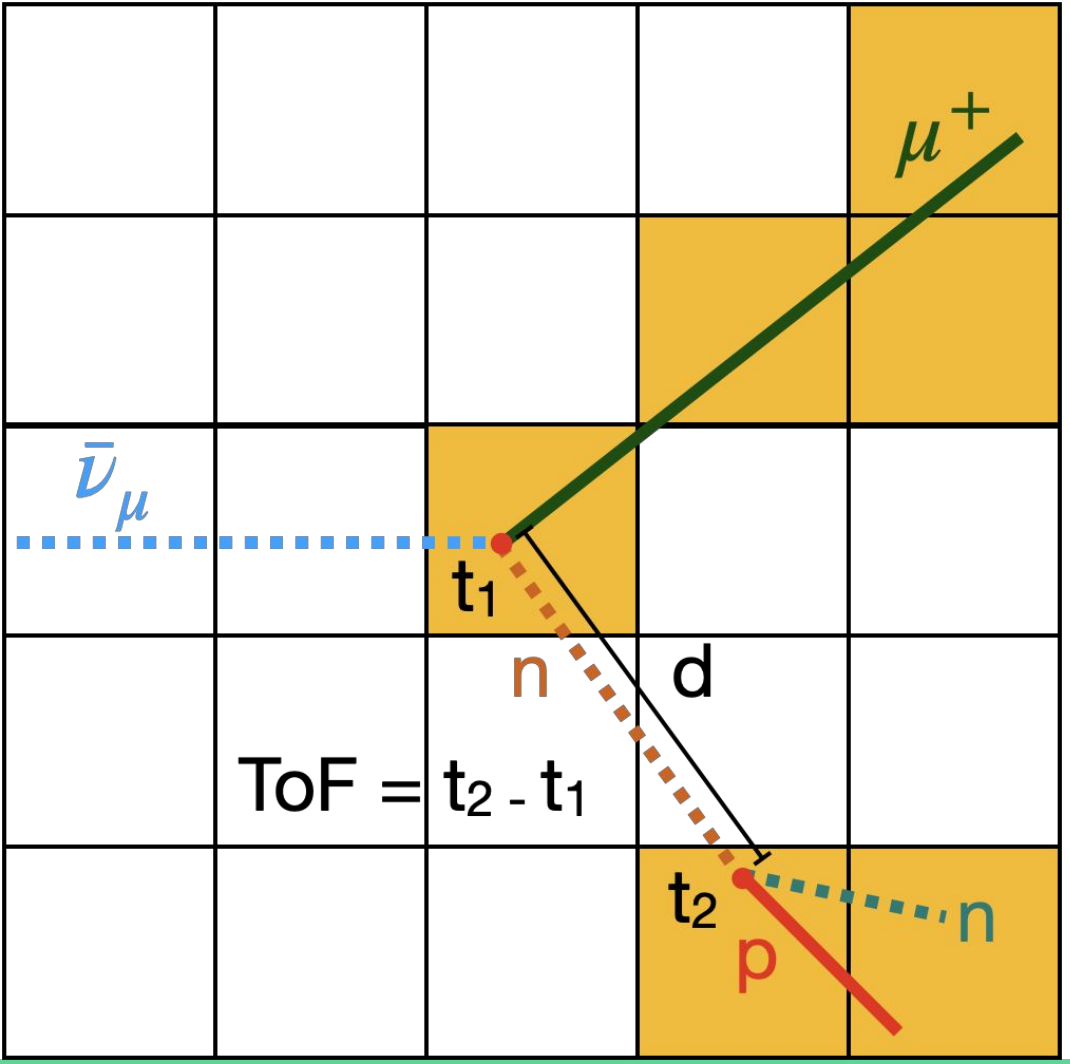
- Different models for particle-nuclei interactions
  - Expect: mostly effect neutron inelastic scattering on carbon
- Bertini
  - Uses Fermi Gas model
  - Small nucleon size relative to medium size
- Bertini High Precision (HP)
  - Extension of Bertini to 0-20 MeV
- Inclxx
  - Leige Intranuclear Cascade model
  - Reactions induced by light nuclei



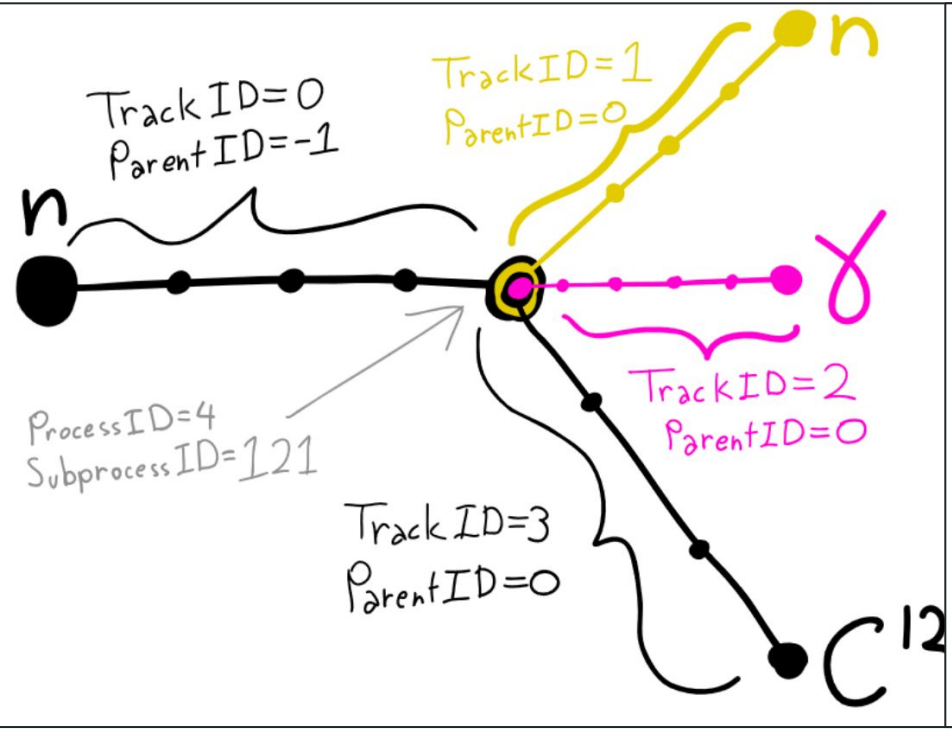
Examples of CCQE interactions

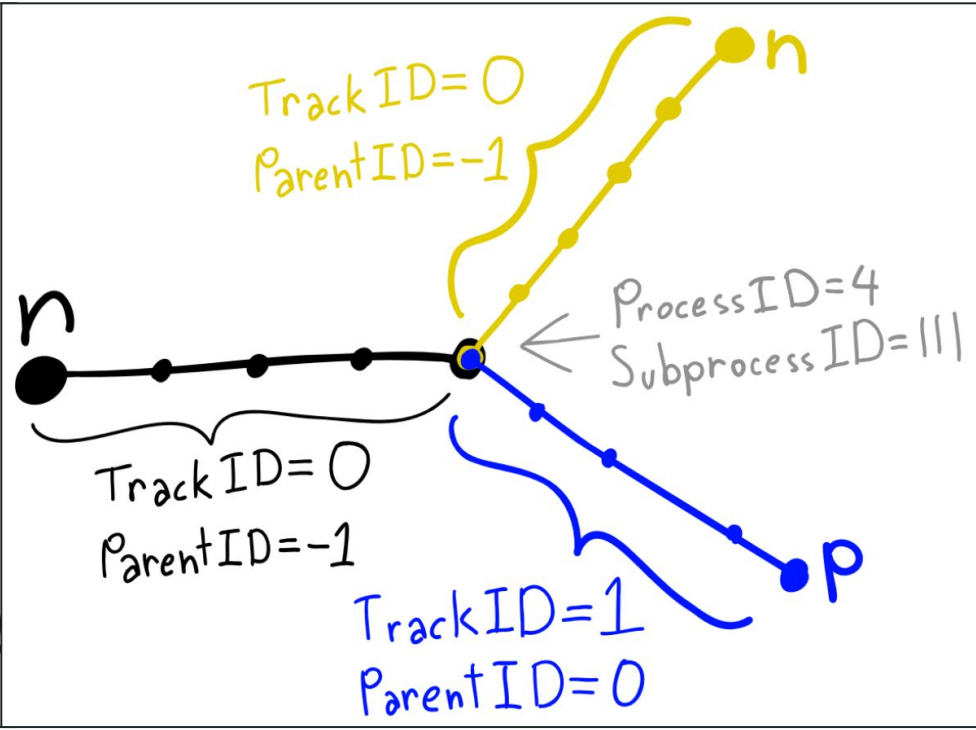


Neutrino Interactions in a Cubic Detector

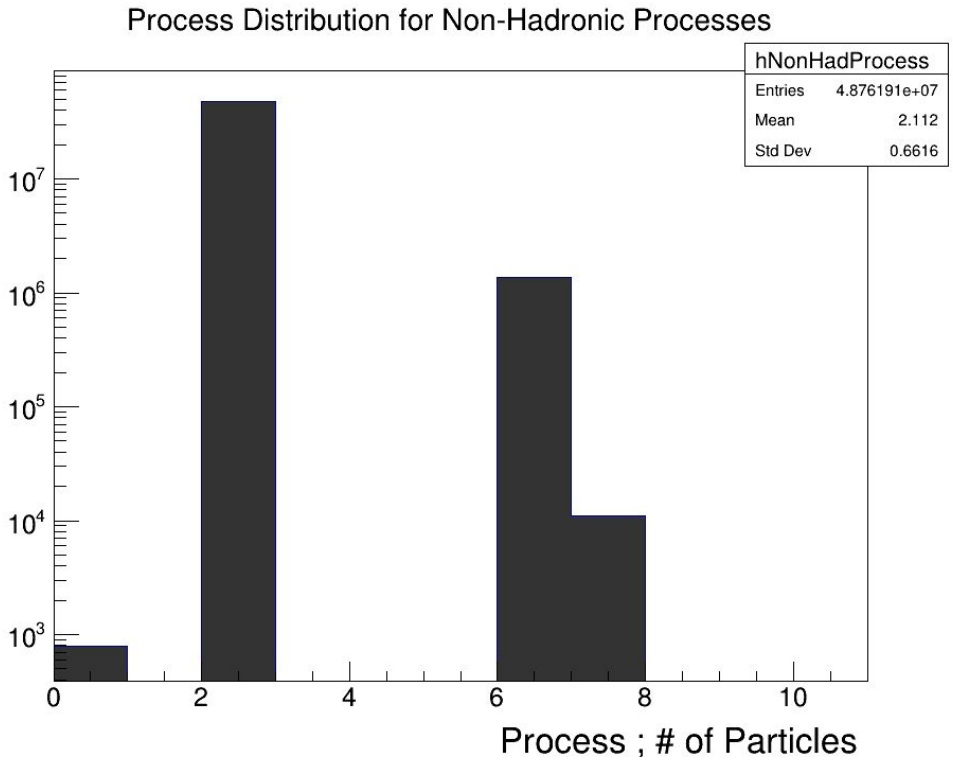


Geant4 Interaction Process Examples





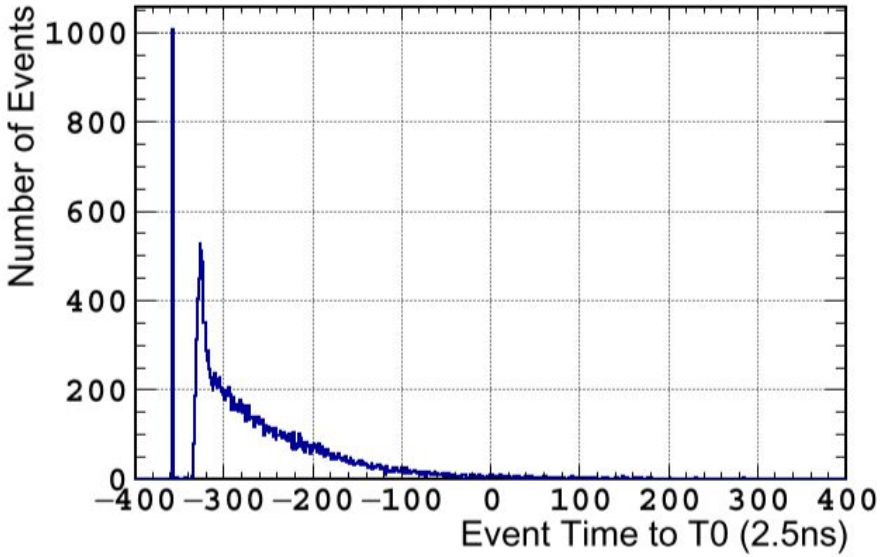
Non-Hadronic Process Interactions for Incoming Neutrons



Process ID	Process Type
0	NotDefined
1	Transportation
2	Electromagnetic
3	Optical
4	Hadronic
5	Photolepton_hadron
6	Decay
7	General
8	Parameterisation

LANL Beam Test Results

- Improvements to measurements of neutron cross section for 500-688 MeV
  - $\sigma_{99\text{-}688MeV}$ = 0.36±0.05 barn
  - χ^2/d.o.f = 22.03/38
- Proved Capability of SuperFGD to measure neutron kinematics using ToF
- First physics result!



A. Agarwal et al. Total neutron cross-section measurement on CH with a novel 3D-projection scintillator detector. Phys. Lett. B, 840:137843, 2023.

Near Detector

- UA1 Magnet
  - Measure momenta with good resolution
  - Measure sign of charged particles
- Pi-Zero Detector
  - Measures ν_μ + n → ν_μ + n + π^0 + X with the same neutrino beam flux that reach SK
- Time Projection Chamber (TPC)
  - Determines number, orientation, **momenta of charged particles**
  - Determines event rate as fxn of neutrino energy, ionization left for each particle
    - PID from ionization
- Fine Grain Detector (FGD)
  - Tracks charged particles
- Electromagnetic Calorimeter
  - Photon detection, energy and direction measurement
- Side muon range Detector
  - Records muons escaping with high angle relative to the beam (θ)
  - Identify beam-related event interactions in cavity walls and magnet