NEUT

Yoshinari Hayato (Kamioka, ICRR, The Univ. of Tokyo)
for the NEUT developers

1. Overview of flux/geometry driver functionalities
2. Review of the output format
3. Simulation of ISI and FSI

1) Overview of Flux/Geometry driver and input formats

Available Flux/Geometry drivers

Simple flux driver

- 1D flux histogram (root TH1)
- Single nucleus (with Hydrogen)
- Uniformly distributed vertex in specified radius (2-dim.)
- Fixed direction specified by the configuration (card) file

Super-Kamiokande Atmospheric neutrino flux driver

- Not included in the NEUT distribution (SK library)
- Honda fluxes
  - (also supports old Gaisser, Lee, Naumov fluxes)
- H_2O target
- Uniformly distributed vertex in the SK detector
- Neutrino direction is determined by the input flux
  - 3D for Honda fluxes, 2D (zenith only) for the others.

1) Overview of Flux/Geometry driver and input formats

Available Flux/Geometry drivers

T2K-SK flux driver

- T2K neutrino vector (vector flux file) or 1D histogram
- Single nucleus (with Hydrogen)
- Uniformly distributed vertex in the SK detector
- Fixed direction (T2K beam direction, hard coded.)

T2K-ND280 (off-axis detectors) flux driver

- T2K neutrino vector (vector flux file)
  - 4-vector and position at the "incident" plane.
- ROOT Geometry (TGeoManager)
  - Target nucleus is taken from the geometry.
- Direction is given by the neutrino vector (flux)

1) Overview of Flux/Geometry driver and input formats

Available Flux/Geometry drivers

T2K neutrino vector

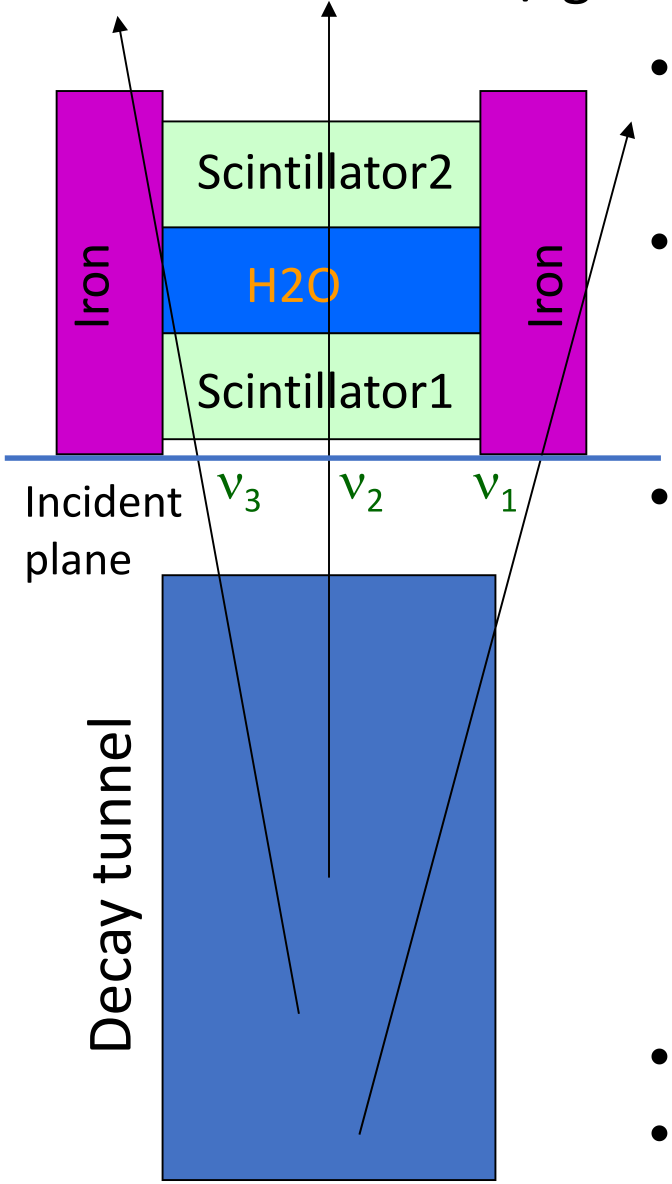
Output neutrino flux vector file from jnubeam (T2K beam simulation program).

This contains the following variables:

- Neutrino energy (E_ν),
- Neutrino direction (dx, dy, dz),
- Neutrino position (x, y) at the "incident plane" in the detector coordinate,
- Weighting factor to give **flux/Detector** with 1E21, and
- Many other variables (from the beam simulation).

1) Overview of Flux/Geometry driver and input formats

T2K-ND280 flux / geometry driver



- Read neutrino vector information

↓

- Calculate the path lengths of each volume and material, like L_{Iron}, $L_{scintillator1}$ etc...

↓

- Calculate the interaction probabilities in each volume (material), like

$$P_{Iron} = L_{Iron} \times \sigma_{Iron}(E_\nu)$$
$$P_{scintillator1} = L_{scintillator1} \times \sigma_{scintillator}(E_\nu)$$
$$\vdots$$
$$P_{total} = P_{Iron} + P_{scintillator1} + P_{scintillator2} + \dots$$

↓

- Store all the probabilities for each vector.
- Use stored information to generate events.

2) NEUT output data format

Output data formats

1) CERNLIB ZEBRA based bank (ZBS)
2) ROOT class
3) ROOT tree (T2K ND280 specific RooTracker format)
4) ROOT flat tree (similar to the ZBS format)

1) ~ 3) stores complete information including
the particle tracking information in the nucleus and
the physics (model) information and parameters
to generate the event, i.e. re-weightable.
4) does not keep the detailed tracking information.

2) NEUT output data format

NEUT output data structure

N_{part} # of particles stored in the particle stack

$N_{primary}$ # of particles from initial (primary) neutrino interaction

Particle stack
Particle #1
Particle #2
:
:
Particle $\#N_{primary}$
Particle $\#N_{primary}+1$
Particle $\#N_{primary}+2$
:
:
Particle $\#N_{part} = N_{primary} + N_{FSI}$

- Incident particles
  neutrino
  nucleon(s)
  (or nucleus for coherent π prod.)
- Particles produced by the initial (primary) neutrino interactions

- Secondary particles
  Particles produced by the hadron secondary interactions
- De-excitation γ and nucleons

2) NEUT output data format

NEUT output data format

Particle stack
Particle #1
Particle #2
:
:
Particle #$N_{primary}$
Particle #$N_{primary}$+1
Particle #$N_{primary}$+2
:
:
Particle #N_{part} =$N_{primary}$+N_{FSI}

Particle information
Particle ID (PDG code)
Mass
4-momentum
Status (Escaped from the nucleus =1 or not =0)
Status flag(detail) 0: Escaped from the nucleus 1: Decayed to the other particle 2: Escaped from the detector 3: Absorbed 4: Charge exchanged 5: Pauli blocked 6: N/A 7: Produced particles 8: Inelastically scattered
Initial vertex ID (in the detector)
Final vertex ID (in the detector)
Parent particle index #

2) Output data format

Example of output data

$$\nu_\mu + p \rightarrow \mu^- + p + \pi^+$$

$$\pi^+ + n \rightarrow \pi^0 + p$$

$$p + N \rightarrow p + p$$

This particle is not simulated.

Index	Particle	Status	Parent particle ID	Status (detail)
#1	Incident ν_μ	0	0	-1
#2	Incident Proton	0	0	-1
#3	Outgoing μ^-	1	1	0
#4	Outgoing Proton	0	2	7 (produced particles)
#5	Outgoing π^+	0	2	4 (charge exchange)
#6	Re-scattered π^0	1	5	0
#7	Re-scattered proton	1	4	0
#8	Re-scattered proton	1	4	0
#9	De-excitation gamma	1	2	0

3) Initial state interactions and final state interactions

Initial and final state interaction simulation parts are separated.
In the library, not only
- neutrino-nucleus interaction simulation program but
- pion-nucleus scattering simulation program and
- nucleon-nucleus scattering simulation program are included.

4) NEUT specific issues (difficulties for the users)

- NEUT uses gfortran.
- NEUT extensively uses CERNLIB.

  CERNLIB contains old versions of PYTHIA and JETSET.

  Makes it difficult for users to compile.

  Symbols may conflict with the other libraries.

Possible solution (just an idea)