DEEP UNDERGROUND NEUTRINO EXPERIMENT

LArSoft user tutorial

by a non-expert developer

SNIPER meeting – 4/7/2024

Emanuele Villa

CERN/Université de Genève

emanuele.villa@cern.ch

References

I am still on the noob side, this is just a very basic tutorial based on experience and other trainings like these, but more are being prepared now that we are using linux 9

- https://cdcvs.fnal.gov/redmine/projects/dunetpc/wiki/_Tutorial_:
- https://dune.github.io/computing-training-basics/index.html

Verbose version of this presentation is here https://docs.google.com/document/d/14ORCEtpXWSIT_1hXJxtW2PMGMVWozzRK1GxgPKaptCk/edit

lxplus

You can work on fnal cluster, that will be a bit slower than lxplus. Also, it seems like some things are not properly set up yet after switching to Alma9

In lxplus, dune cannot be set up in rhel9, you need a centos7 container like:

```
/cvmfs/oasis.opensciencegrid.org/mis/apptainer/current/bin/apptainer shell --shell=/bin/bash -B
/cvmfs,/opt,/run/user,/etc/hostname,/etc/hosts,/etc/krb5.conf --ipc --pid
/cvmfs/singularity.opensciencegrid.org/fermilab/fnal-dev-sl7:latest
```

Then all should be good and ready to run.

Vocabulary

- **art**: framework (originally developed by CMS).

- **larsoft**: large software for lar TPCs. The *lar* command comes from here

  Larsoft is widely used to actually refer to **dunesw,** including by me.
- **dunesw**: project containing the dune-specific repositories

- **fcl** (aka fhicl) files: configuration files for larsoft

- **producer**: it's a module that takes a certain input and acts on it to generate an output, following the standard chain of operations in a simulation. The *TriggerPrimitiveFinder* is an example

- **analyzer**: a module that is used to read the products of a producer and perform operations on it. The *TPDumper* is an example.

Concept

Larsoft is a software to simulation liquid argon detectors, used for several experiments including Nova and DUNE.

The idea is that the simulation happens in 4 stages:

- **generation**: use an event generator to simulate neutrinos interacting with matter and the daughter particles.
- **g4**: Geant4 response, considering the fact that we are in liquid Argon
- **detsim**: this simulates the detector response given what is produced in g4, that is basically electron clouds that reach the readout. Here many things can be changed in order to simulate waveforms that resemble the way they are produced in the DAQ.
- **reco1**: normally this is where the waveforms are read and the hitfinding is performed. The result are hits, sometimes already grouped.
- **reco2**: higher level analysis, track reconstruction etc.

Set up and use a dunesw version

The first thing to do when using larsoft is to load the ups products.

```
# load ups products
source /cvmfs/dune.opensciencegrid.org/products/dune/setup_dune.sh
```

Then, set up a dunes version, this means that you load all the packages and you can use them in your bash session.

```
# setup a dunes version
export DUNESW_VERSION=v09_79_00d02;
export DUNESW_QUALIFIER=e26:prof;
echo 'Setting up dunesw version' $DUNESW_VERSION', qualifier' $DUNESW_QUALIFIER '...'
setup dunesw $DUNESW_VERSION -q $DUNESW_QUALIFIER
echo 'Done!'
```

I suggest putting these two things in a script, that has to be sourced, not executed.

Install a local dunesw version with chosen repos

You want to do it when using branches or code versions that are not the ones in a certain release.

The procedure is the following:

1. mkdir my_larsoft
2. cd my_larsoft/
3. mrb newDev
4. source localProducts_XXXX/setup
5. cd $MRB_SOURCE
6. mrb g dunesw # and all other needed packages that are required considering the dependencies, or mrbsetenv will return an error
7. mrb g your_package
8. Checkout to the tags of the branch you want to use!!!!
9. build
   a. cd $MRB_BUILDDIR
   b. mrbsetenv
   c. mrb b –jN   # build (can use $nproc to use automatically the number of cores of the machine)
   d. mrb i –jN # install
   e. mrbslp   # set up

fcl

Before running a simulation, you have to choose or create a configuration file.

You can choose a standard available in the dunesw that you set up, or you can create your own (ofter just overwriting some parameters of existing ones). If you have locally a fcl with the same name of the one in the dunesw, lar will look for the local one first and, if it exists, use it.

Quick guide about synthax https://cdcvs.fnal.gov/redmine/attachments/download/29136/quick_start_v3.pdf

Example to build a fcl: https://indico.ph.ed.ac.uk/event/130/contributions/1737/attachments/1083/1506/Simulation_Tutorial_LArSoft_Workshop_2022.pdf

To override a parameter in respect to what was set in the included config, use this syntax:

```
physics.producers.marley.marley_parameters.direction.x: 0.1
```

To see fully processed configuration:

```
fhicl-dump myConfig.fcl
```

Simple important thing: many fcls have maxEvents = 10. To usemaxEvents from command line, create a fcl like

```
#include “other.fcl”
source.maxEvents:  -1
```

Run a lar simulation

The simulation runs in several steps. You can execute one or all steps, they are all independent.

There are several options, the most used ones are

```
lar -c ${DATA_PATH}${GEN_FCL_CHANGED}.fcl -n $number_events -o
${DATA_PATH}${GEN_FCL}.root"
```

- -c: configuration, ie your fcl file
- -n: nEvents
- -o: output file name, that has to be given as input for the following stage.
- -s: input file (counterintuitive!)

The different stages are independent. For example, you can run a detsim stage once, keep the output file and then the reco1 stage several times starting from the same input file.

Generate TP samples

This is where the code for TP generation is https://github.com/DUNE/duneana/tree/evilla/tpstream , it is based on larsoft v79.

In the repo https://github.com/dune-sn-online-pointing/dunesw-config you can find instructions to run the simulation.

Once samples are generated, use the tools in the other repos of https://github.com/dune-sn-online-pointing/

Read outputs

The root output file is written in artroot, that means that makes use of larsoft dictionaries.

In other words, it’s not a simple tree that one can inspect with root.

There are different options then:

- create an analyzer to read the data

- look at the histogram files that are produced in parallel to the main root output
  - you have to enable the writing of the histograms in the configurations

- use gallery (adding instructions and template here at some point)