nuSIM-2021-02 revision 1 July 8, 2021

Installing and setting up nuSIM

P. Kyberd and K. Long

Introduction

This document summarises the steps needed to set-up and run nuSIM. A summary of the tasks that nuSIM performs may be found in [1]. nuSIM has been developed in python; python 3 is assumed.

Getting the code

nuSIM is maintained using the GitHub version-control system. The latest release can be downloaded from the nuSTORM wiki (https://www.nustorm.org/trac/wiki/Software-and-computing).

Dependencies and required packages

nuSIM requires the following packages:

- Python modules: `scipy`, `matplotlib`, and `pandas`;
- CERN programme library: `pyroot` (which may be installed using the standard `root` installers, see the documentation at https://root.cern/install/).

It may be convenient to run nuSIM in a "virtual environment". To set this up, after updating your python installation to python 3, and installing root, execute the following commands:

1. `python3 -m venv --system-site-packages venv`
   - This creates the director `venv` that contains files related to the virtual environment.
2. `source venv/bin/activate`
3. `python -m pip install pandas scipy matplotlib`

To exit from the virtual environment, execute the command `deactivate`.
The command `source venv/bin/activate` places you back into the virtual environment.

Unpacking the code, directories, and running the tests

After downloading the package from GitHub, or cloning the repositiry, you will find a "`README.md`" file which provides some orientation and instructions to run the code. In particular, a `bash` script "`startup.bash`" is provided which:

- Sets the "`nuSIMPATH`" environment variable so that the files that hold constants etc. required by the code can be located; and
- Adds "`01-Code`" (see below) to the PYTHONPATH. The scripts in "02-Tests" (see below) may then be run with the command "python 02-Tests/< filename >.py".

Below the top directory, the directory structure in which the code is presented is:

`01-Code`: contains the python implementation as a series of modules. Each module contains a single class or a related set of methods.

`02-Tests`: contains self-contained test scripts that run the various methods and simulation packages defined in the code directory.

`11-Parameters`: contains the parameter set used in `02-Tests/RunSimulation.py` to generate muon decays in the production straight.

The instruction in the `README.md` file should be followed to set up and run the code.

1 Running the code

The file in 02-Tests/RunSimulation.py - will run the code and produce a root data set.

The file **RunSimulation.py** contains:

- the definition of the root output file for the generated dataset
  rootfilename = os.path.join(nuSIMPATH, 'Scratch/nuSIM-RunSimulation.root')
- the definition of csv input file to control the running of the Simulation
  filename = os.path.join(nuSIMPATH, '11-Parameters/nuSTORM-PrdStrght-Params-v1.0.csv')
- the call to the Simulation class with; the number of events to generate; the central energy to generate; and the filenames
  Smltn = Simu.Simulation(5000, 6., filename, rootfilename)

Most of the entries for the file **nuSTORM-PrdStrght-Params-v1.0.csv** are self explanatory but it is worth noting:

- Run Type, rType, 1, i, numSim-2021-01
  1 generates a muon decay and 2 generates a pion beam

- Momentum acceptance,pAcc,10,%,nuSIM-2021-01
  Generates a parabolic distribution with a half width given by the number. For standard generation 10 should be used for pions and 15 muons

1.1 Plots

There is a file **01-Code/Plots.py**. It produces a separate root file with histograms filled by running the programme. The plots are written to **plots.root** in the directory from which the job is run. In the default case it will produce a plot of the energy of the ν_μ created by either the μ or π beam. You can either modify this file or produce your own file. The calls are made from **Simulation.py** and Plots.py is included with the line
import Plots as plots

The class contains three methods: __init__; fill; histdo.
__init__ is called with no parameters and makes the calls to create a root histogram, the single histogram acts as an example
fill(self, < array or class >), where the array or class contains the values to be plotted for each event. The example is an array, but it will work with a class which has suitable *get() methods.*
histdo is called with no parameters and writes out all the histograms. The example shows writing out the single histogram

Making a contribution

nuSIM is archived in the git repository `longkr/nuSTORM`. To clone the code using `git clone` you will need your own account on GitHub and permission to clone the code. Instructions to request such permission is posted on the nuSTORM wiki.

References

[1] P. Kyberd and K. Long, "nuSIM: parameters for first simulation of neutrino spectra," Tech. Rep. nuSIM-2021-01, March, 2021. https://www.nustorm.org/trac/raw-attachment/wiki/Software-and-computing/Documentation/2021/nuSIM-doc-01.pdf.