Running g4NuMI simulation on NuMiX nodes (A Newcomer's Guide)

Sudeshna Ganguly

02/25/2021

Running g4NuMI simulation on NuMI-X Nodes

- Aim:
  - Check out g4numi code
  - Build g4numi on NuMI-X node
  - Run g4numi : generate large flux samples on the grid
  - Plot: plot histograms

Running g4NuMI simulation on NuMI-X Nodes

- **Setting up area on NuMI-X**

ssh -AKXY sganguly@numixgpvm01.fnal.gov
pwd
/nashome/s/sganguly

- g4NuMI: a Geant4 based simulation of the beamline

- To checkout the g4numi codes (under /nashome/s/sganguly)

git clone ssh://p-numi-beam-sim@cdcvs.fnal.gov/cvs/projects/numi-beam-sim-g4numi

- But to get the latest code:

- Copy new target simulation & setup script to your /nashome area from either Nilay or Yiding (I copied from Nilay):

/nashome/n/nbostan/yiding_new_target

/nashome/n/nbostan/setup.sh

```
source /cvmfs/larsoft.opensciencegrid.org/products/setup
setup cmake v3_17_3
setup root v5_34_32 -q debug:e9:nu
```

Running g4NuMI simulation on NuMI-X Nodes

- cd **/nashome/s/sganguly/yiding_new_target/newtarget/g4numi**
- source setup_beamsim.sh
- make clean
- make
- make
- setup jobsub_client
- echo $EXPERIMENT
  - minerva
- export EXPERIMENT=numix

Before submitting jobs on the grid:
Note: scratch /pnfs area from which to send tar file (tarred up local area) to grid:
CACHE_PNFS_AREA= "/pnfs/{EXPERIMENT}/scratch/users/{USER}/grid_cache/".format(EXPERIMENT = os.getenv("EXPERIMENT")

I had to first create these areas under /scratch:
- mkdir -p /pnfs/numix/scratch/users/sganguly/
- mkdir -p /pnfs/numix/persistent/users/sganguly/grid_cache
- Submit jobs on the grid:
  - **python ProcessG4NuMI.py --pot 500000 --outdir /pnfs/numix/persistent/users/sganguly/highstat --beamconfig me000z200i --n_jobs 100**
  - me000z200i means Medium Energy Configuration and 200 kA Horn Current
- Test status of jobs:
  - jobsub_q --user sganguly -G fermilab

Running g4NuMI simulation on NuMI-X Nodes

- Parameters chosen to run the simulation (**what I understood by looking at the macro file**):
- POT : 500000 per job
- Njobs: 100
- Total: 5X10^7
- ME configuration
- Horn current : 200 kA

- Optimized target configuration

```
NUMBER_OF_FINS  = 48   #default ME
DISTANCE_BETWEEN_FINS     = 0.5  #mm
BUDAL_MONITOR_ME_POSITION = 0    #mm
WIDTH_ME_FIN  = 9.0  #mm
WINGED_FIN_ID1 = 1
WINGED_FIN_ID2 = 2
WINGED_FIN_ID3 = 3
WINGED_FIN_ID4 = 4
```

Running g4NuMI simulation on NuMI-X Nodes

- Output files written here: /pnfs/numix/persistent/users/sganguly/
- No of root files = No. of log files = No. of jobs
- Log files have details of the macro content

======== MACRO CONTENT ========
#This is an template macro to create ME G4numi files.
#v1.1.2.7 of this file contains all of the ME defaults
#as of 2017/11.
#The "BeamConfig" option has been depreciated in the ME.
#It can/should be used to change horn current but not
(not target Z position. Use targetPosition for that).

/control/suppressAbortion 2
/NuMI/run/DebugLevel 0
/NuMI/run/useNuBeam true
/NuMI/run/useNImpWeight true
/NuMI/det/RunPeriod 0
.....
......
....

Running g4NuMI simulation on NuMI-X Nodes

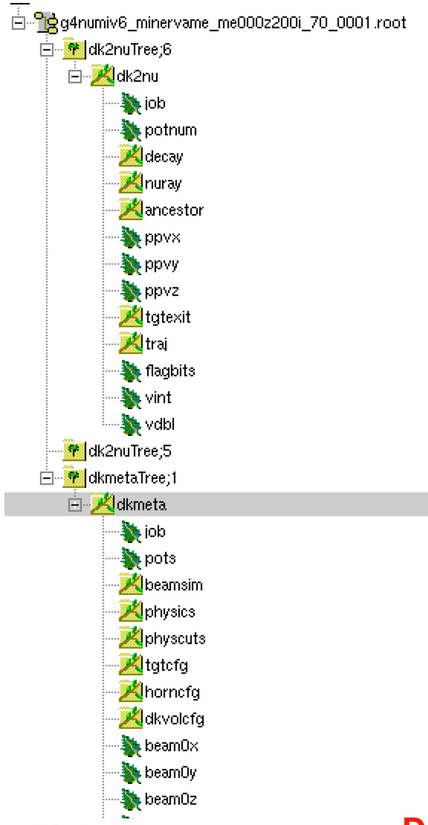
- Output root files have:
  - dk2nuTree
  - dkmetaTree

- Dk2nu ntuple is essentially a list of neutrinos that are created by beam simulation

```
A class that defines the "dk2nu" object used as the primary
*        branch for a TTree for the output of neutrino flux simulations
*        such as g4numi, g4numi_flugg, etc
```

- Each neutrino is associated with a dk2nu object which contains detailed information about neutrino kinematics & kinematics of its hadron ancestors

- Dk2nu output files generally contain two ntuples
  1. metadata ntuple
  2. neutrino ntuple
  - https://cdcvs.fnal.gov/redmine/projects/lbne-beamsim/wiki/Dk2nu_ntuples (this page has all details on Dk2nu branches, though it is for g4lbnf, but same applies for g4numi)



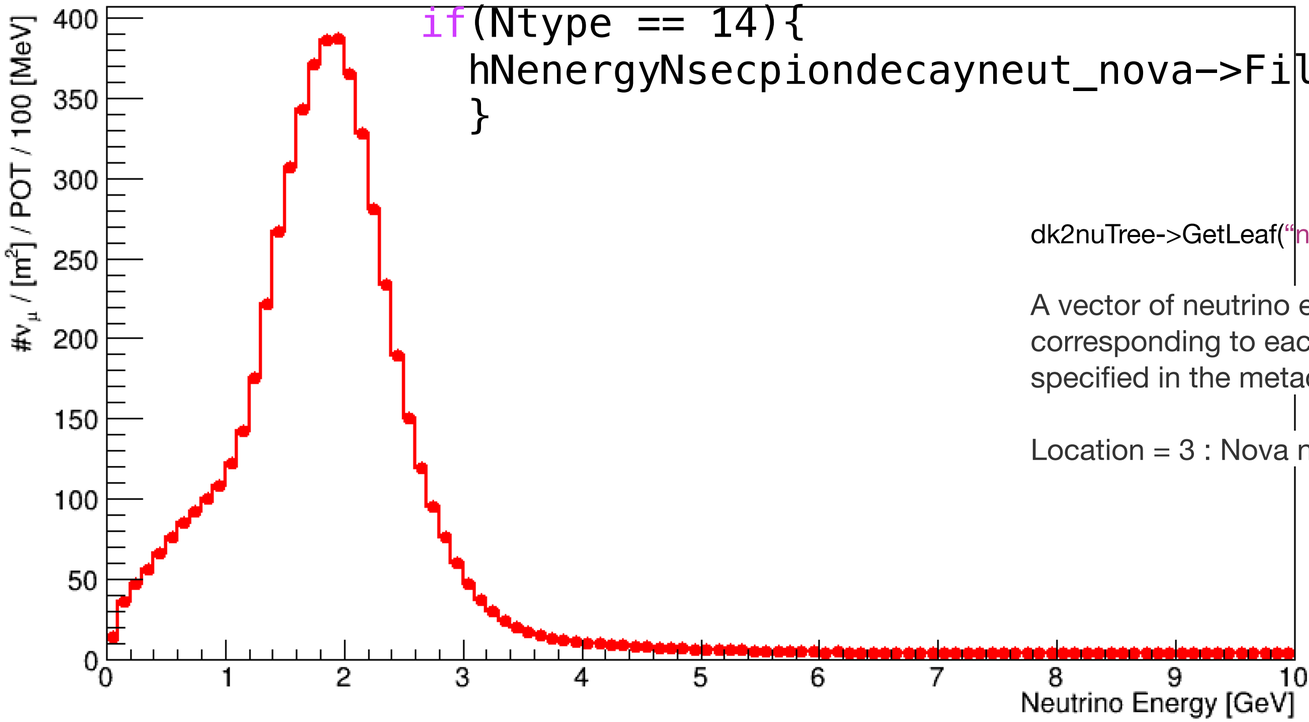
Description of branches

- dk2nutree
  - dk2nu: one entry corresponds to a neutrino decay
    - job: Simulation job number
    - potnum: Number of primary proton that generated the neutrino (number of protons simulated increme job to total number of protons simulated at end)
    - decay
      - norig: Neutrino origin; 1 = particle produced in target or baffle, 3 = muon decay, 3 = all others
      - ndecay: Decay code of decay that produced neutrino
      - ntype: GEANT particle code of neutrino
      - vx: x component of neutrino vertex position in cm
      - vy: y component of neutrino vertex position in cm
      - vz: z component of neutrino vertex position in cm
      - pdpx: x component of final momentum of neutrino parent in GeV
      - pdpy: y component of final momentum of neutrino parent in GeV
      - pdpz: z component of final momentum of neutrino parent in GeV
      - ppdxdz: px/pz of parent at parent production point
      - ppdydz: px/pz of parent at parent production point
      - pppz: pz of parent at parent production point in GeV
      - ppenergy: energy of parent at parent production point in GeV
      - ppmedium: empty branch
      - ptype: GEANT particle code of neutrino parent

Running g4NuMI simulation on NuMI-X Nodes

- An example of how to plot quantities from a g4numi dk2nu ntuple can be found here (Nilay's test macro in root): /nashome/s/sganguly/clean_backup_g4numi/g4numi/ana_scripts/test.C

- Note: Before running root in this area, I had to do these:
  - start a new terminal
  - pwd
  - /nashome/s/sganguly
  - . setup.sh
  - cd /nashome/s/sganguly/clean_backup_g4numi/g4numi/ana_scripts/
- Run script with: root -l test.C




```
if(Ntype == 14){
  hNenergyNsecpiondecayneut_nova->Fill(energy_nova,(NWtNear_nova * Nimpwt)/3.1416);
  }
```

dk2nuTree->GetLeaf("nuray.E"))->GetValue(3)

A vector of neutrino energies in GeV, corresponding to each of the locations specified in the metadata ntuple

Location = 3 : Nova near detector

dk2nuTree->GetLeaf("nuray.wgt"))->GetValue(3)

dk2nuTree->GetLeaf("decay.nimpwt"))->GetValue();

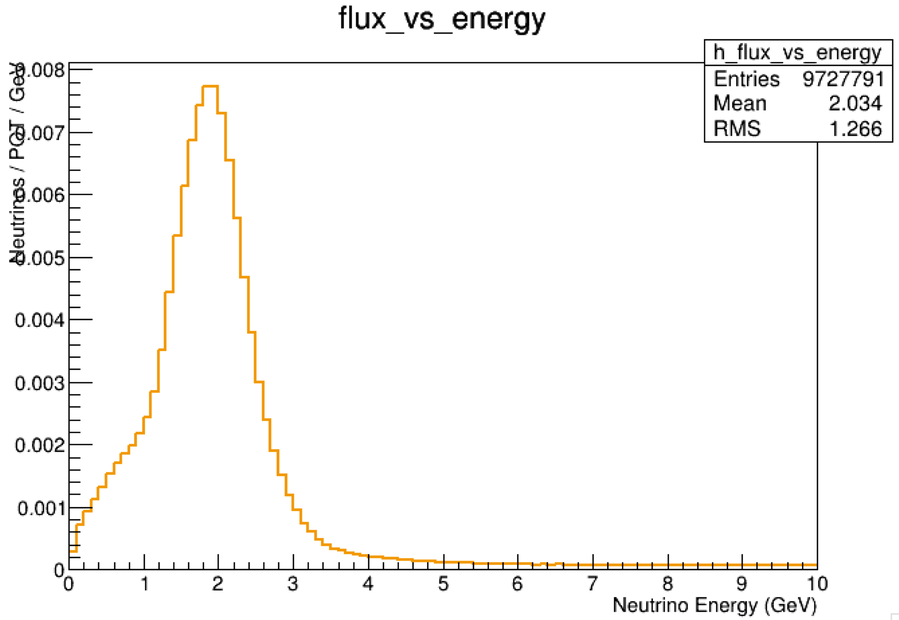
Running g4NuMI simulation on NuMI-X Nodes

- An example of how to plot quantities from a g4numi dk2nu ntuple can be found here:

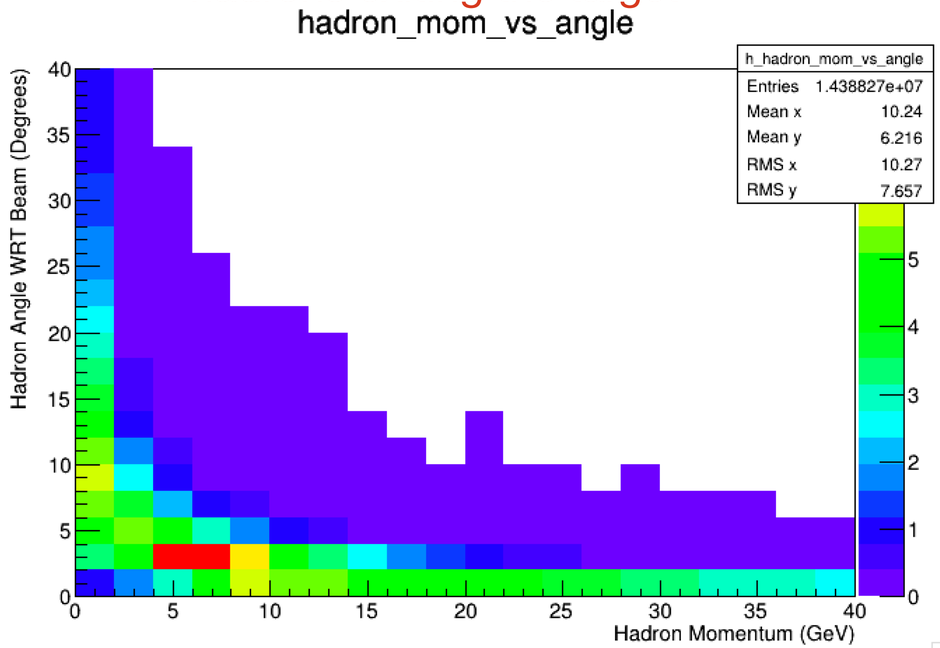
For python lovers, I've added a script here: /nashome/s/sganguly/clean_backup_g4numi/g4numi/ana_scripts/plottingExample.py

- Note: Before running python in this area, I had to do these:
  - start a new terminal
  - pwd
  - /nashome/s/sganguly
  - . setup.sh
  - cd /nashome/s/sganguly/clean_backup_g4numi/g4numi/ana_scripts/
- Run script with: python plottingExample.py

Plot un-oscillated flux of muon neutrinos at Nova near detector versus energy



Plot energy and angle of all hadrons exiting the target



Running g4NuMI+MM simulation on NuMI-X Nodes

Run g4NuMI simulation+ Muon Monitor Simulation

- Parameters chosen to run the simulation POT : 500000 per job
- Njobs: 100
- Total: 5X10^7
- ME configuration
- Horn current : 200 kA

- Optimized target configuration

```
NUMBER_OF_FINS  = 48   #default ME
DISTANCE_BETWEEN_FINS     = 0.5  #mm
BUDAL_MONITOR_ME_POSITION = 0    #mm
WIDTH_ME_FIN  = 9.0  #mm
WINGED_FIN_ID1 = 1
WINGED_FIN_ID2 = 2
WINGED_FIN_ID3 = 3
WINGED_FIN_ID4 = 4
```

- This default version has gdml files with hadron absorbers

Running g4NuMI+MM simulation on NuMI-X Nodes

- cd **/nashome/s/sganguly/clean_backup_g4numi/g4numi (Copied from Yiding)**
- source setup_beamsim.sh
- make clean
- make
- make
- setup jobsub_client
- echo $EXPERIMENT
  - minerva
- export EXPERIMENT=numix

- Submit jobs on the grid:
  - **python ProcessG4NuMInu2mu.py --pot 100000 --outdir /pnfs/numix/persistent/users/sganguly/muonmonitor --beamconfig me000z200i --n_jobs 100**
  - me000z200i means Medium Energy Configuration and 200 kA Horn Current
- Test status of jobs:
  - jobsub_q --user sganguly -G fermilab

- Note:

After submitting jobs, I was getting this error:
ERROR:User authorization has failed: --group 'numix' not found.
To solve: In the ProcessG4NuMI.py script, changed the group to be fermilab
submit_command = (“jobsub_submit {GRID} {MEMORY} -N {NJOBS} -d G4NUMI {OUTDIR} ”
“-G fermilab ”

Running g4NuMI+MM simulation on NuMI-X Nodes

Run g4NuMI simulation+ Muon Monitor Simulation

- Run script ProcessG4NuMInu2mu.py to run g4numi and muon monitor simulation at one go

- This script:
  - Runs g4numi(nu mode) at first step
  - After getting dk2nu files from g4numi simulation, runs script in dk2nu_local: ./dk2nu_local/bin/nu2mubatch
    This script gets pion information from dk2nu and generates muons
  - Runs g4numi(muon Input mode) using those muons from nu2mubatch. Macro from here: /macros/template_muon.mac

In ProcessG4NuMInu2mu.py script, above 3 steps are already being executed

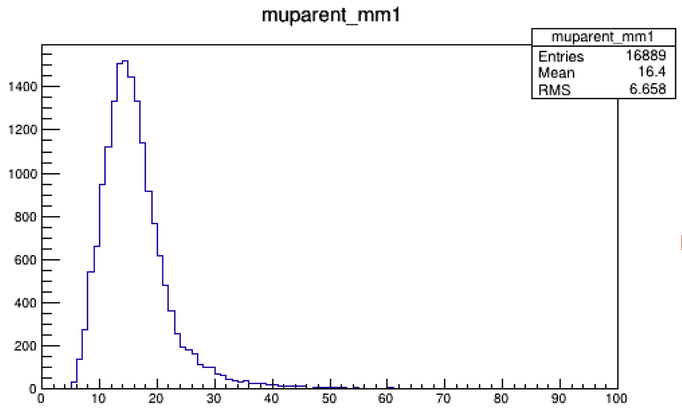
- **There is a page with muon monitor simulation information:**

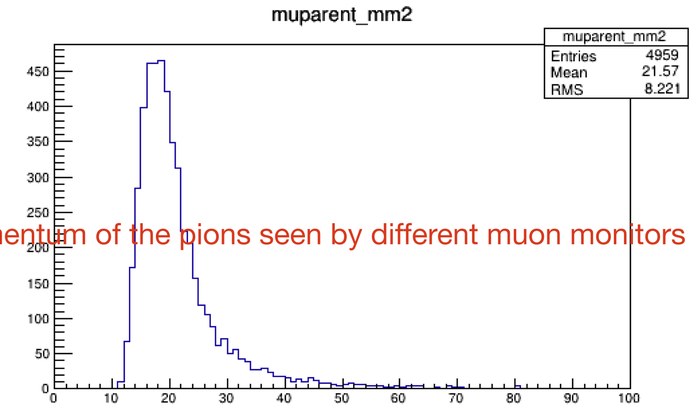
https://cdcvs.fnal.gov/redmine/projects/numi-beam-sim/wiki/Muon_Monitor_Simulations

Running g4NuMI+MM simulation on NuMI-X Nodes

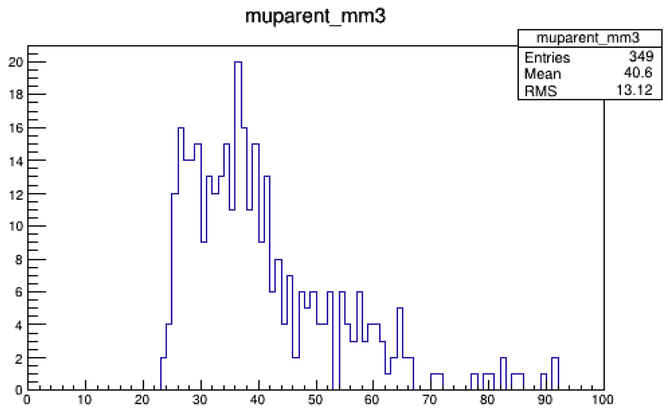
Output files written here: /pnfs/numix/persistent/users/sganguly/muonmonitor

- Generated 2 types of root files:
  - g4numiv6_minervame_me000z200i_*.root dk2nu ntuples
  - hadmmNtuple_me000z200i_*.root. muon monitor ntuples
- **Inside /nashome/s/sganguly/clean_backup_g4numi/g4numi/ana_scripts: python script by Yiding: run_anatuples.py**
- **Used that to take a quick look inside muon monitor ntuples**
- Note: Before running python script, I had to do these:
  - start a new terminal
  - pwd
  - /nashome/s/sganguly
  - source /cvmfs/minerva.opensciencegrid.org/minerva/software_releases/v22r1p1/setup.sh
  - cd /nashome/s/sganguly/clean_backup_g4numi/g4numi/ana_scripts
  - make clean
  - make
- Run script with: python run_anatuples.py





momentum of the pions seen by different muon monitors



Next step:

- Create large simulation samples for different target status to find out correlations b/w target incidents and Muon Monitor observations
  - Related to this topic is what Katsuya suggested: perform a systematic study to find correlation among individual pixel of MM with a specific physics condition ~ might give us a better handle on how many POTs to be generated

- In terms of documentation:
  - Add description of mm ntuples
  - Create/update wiki with latest instructions from these slides, for people who would want to run these simulations on NuMI-X

Big thanks to Nilay and Yiding for sharing your knowledge!