

Implementation, Deployment, and Use of NOvA Target Hall Geometry in FLUGG

Luke A. Corwin, Mike Martens, and Vamis Xhagjika

April 4, 2011
NOvA-doc-5387

Abstract

This note is intended to supplement the note by Alex Himmel on `g4numi_flugg` and summarize the changes made to use it for the NOvA target geometry

1 Introduction

FLUGG is an acronym that stands for FLUKA + Geant4 Geometry. FLUGG is a data simulation program that uses Geant4 geometry files with FLUKA physics simulation. It was developed at CERN [1, 2]. The program itself is documented and available for download at [9]. For MINOS, extensive geometry files and setup scripts were created by Alex Himmel and documented by him [4, 5].

We would like to extend our gratitude to Alex. Using his experience with FLUGG on MINOS, he has answered many of our questions and provided solutions to many of the problems we have had. Without him, installing and running FLUGG would have taken much longer.

The purpose of this note is to document the modifications of FLUGG and its usage for NOvA collaborators. Information that is not included in this document is probably in [9] or [4, 5].

Most people reading this document have already completed all of the steps in §2. Most of this document (until §6) is a record of what was done to produce a working deployment of FLUGG for NOvA, which is stored in the CVS repository `nusoft/numisoft`. If you just want to checkout, modify, and run `numisoft`, start with §6. Those instructions are also on the NOvA Redmine wiki [6].

All of the computer command given in this note are for the NOvA nodes at Fermilab; see the NOvA wiki for the current list [7].

2 Preliminary Steps

This steps are only necessary for those who do not have computer accounts or who are using a Windows PC. If you already have an account, this section can be skipped.

2.1 Getting Computer Accounts

See http://computing.fnal.gov/xms/Services/Getting_Started/Introduction_to_Computing_at_Fermilab

On-site Employees

1. Obtain a Fermilab ID during New Employee Orientation. (required before you can apply for computing accounts).
2. Read the Fermilab Policy on Computing.

3. You may then fill the Request Form for Computing Username and Primary Accounts. This form is used to request any of the following:
 - a username for all your computer accounts at the lab
 - a Fermilab Email address se
 - a Kerberos Principal
 - a CRYPTOCARD (Dont need this.)
 - an IMAP account (for email)
 - an account on Fermilab's central UNIX system, FNALU

2.2 If You are Using a Windows PC

Logging into flxi03 from Windows PC

You first need to get a Kerberos Ticket. Then, with Exceed, you must setup the xhost file.

Exceed → Tools → Configuration → Security, Access Control → Edit (xhost.txt file). Add lines

```
flxi02.fnal.gov # flxi02
flxi03.fnal.gov # flxi03
flxi04.fnal.gov # flxi04
flxi05.fnal.gov # flxi05
flxi06.fnal.gov # flxi06
flxi07.fnal.gov # flxi07
flxi08.fnal.gov # flxi08
flxi09.fnal.gov # flxi09
```

2.2.1 Setup Hummingbird Network

Hummingbird Network → New HostExplorer Profile

Connection → Telnet → Add new host (add flxi03.fnal.gov)

Security → General → (Kerberos)

Security→Kerberos (Kerberos client: MIT)

Security→Kerberos (Kerberos: Forwarding)

Then run the profile to log on.

2.2.2 Using PuTTY

For host name use: `martens@flxi03.fnal.gov`

Connection Type: SSH

For access to the FLUGG code you need to be a member of the group nova. (System admin needs to add you.)

I change my group to nova and run in the bash shell

```
$> newgrp nova
```

```
$> bash -l
```

Using the -l option with bash executes `~/profile`

3 FLUGG Installation

The central FLUGG installation for NOvA is located in `/grid/fermiapp/nova/novasrt/flugg/`. We are currently using `flugg_2009_3`. In order to run FLUGG, one must have CLHEP, FLUKA, Geant4, and FLUGG installed [4, 5]. All four of these packages are installed in `/grid/fermiapp/nova/novasrt`, and they need to be specified if one is installing a private installation of FLUGG, details in §3.1. However, there should be no need to install a private version of FLUGG since all of the changes relevant to NOvA are contained in `numisoft` (§4).

For access to the FLUGG code you need to be a member of the group nova. (System admin needs to add you.)

3.1 Configuration and Compilation

We downloaded the latest tarfile from the FLUGG website [9].

In the `/grid/fermiapp/nova/novasrt/flugg/flugg_2009_3/FLUGG` directory, we unpacked the tar file, configured and installed FLUGG. After unpacking the file with `tar`, the commands below (in `bash` shell) were used.

```
$> Configure
$>.  configure.sh
$> Install
```

For this instantiation of flugg, the following versions of code are used.

```
FLUGGINSTALL=/grid/fermiapp/nova/novasrt/flugg/flugg_2009_3/FLUGG
FLUPRO=/grid/fermiapp/nova/novasrt/fluka/fluka2008.3c
G4SYSTEM=Linux-g++
G4INSTALL=/grid/fermiapp/nova/novasrt/geant4/geant4.9.3
G4LIB_BUILD_SHARED=yes
CLHEP_BASE_DIR=/grid/fermiapp/nova/novasrt/clhep/2.0.3.2_32bit
```

In the `FLUGGINSTALL/source/Wrappers/src` code directory, we edited `WrapReg2Name.cc` to include two lines of debugging output:

```
#ifdef G4GEOMETRY_DEBUG
    G4cout << "===== RG2NWR =====" << G4endl;
    G4cout << "mreg=" << mreg << G4endl;
#endif
```

When compiling **only** the FLUGG configuration script should be run. **Do not** run any other setup scripts, otherwise havoc may result.

The actual compilation (`make` command) is run in `FLUGGINSTALL/source`.

3.2 Changes Made to Run in Scientific Linux 5 and 64-bit architecture

Note that we need to use a 32 bit version of CLHEP (as seen in §3.1) because Fluka is not yet compatible with a 64-bit system.

In updating the code to compile and run on Scientific Linux 5 (SL5), we decided to stick with `g77` rather than `gfortran` for fortran compilation. We had to copy the static library `libg2c.a` into `g4numi_flugg`. We also appended `-L.` to the `LDFLAGS` environment variable in `binmake.gmk` so that library is accessed at compilation. After compiling on an SL5 machine, users need to use the `-onlySL5` flag when submitting to the grid.

4 numisoft and g4numi_flugg setup

In the `g4numi` and `g4numi_flugg` directories we commented out the references to `hbook` in the `GNUmakefile`, so FLUGG only produces `ROOT` output for NOvA.

In `numisoft/g4numi/directory/GNUmakefile`, we commented out the lines

```
#ROOTLIBS := $(filter-out -lNew,$(ROOTLIBS))
#ROOTLIBS := $(filter-out -lThread,$(ROOTLIBS))
#ROOTLIBS := $(filter-out -lpthread,$(ROOTLIBS))
```

because the `Thread` routines are used.

There was some trouble getting the material definitions correct when running FLUGG. This seemed to be solved by installing a later version of FLUGG and by not copying the `flugg_patch` code from Alex Himmel's `g4numi_flgug` directory.

As originally setup, compiling `g4numi_flgug` required write access to the `flugg` code area. However, by setting the variable environment variable `G4WORKDIR` to `nusoft/numisoft/g4numi_flgug` (this is done in the `g4numi_flgug/scripts/setup.sh`), the code is now compiled and located in the users local area. This is a cleaner solution and prevents people from overwriting each other's code.

5 Translating NuMI Into Code

The goal of FLUGG is to simulate the interactions and decays of particles as they propagate and interact with material and each other. The goal of the code contained in `numisoft` is to reproduce the location and composition of the NuMI target hall in terms of material definitions and geometric representations that FLUGG uses.

The geometry is encoded in the GEANT4 files found in the `g4numi/src` and `g4numi/include` directories. These files are copied into the appropriate locations in `g4numi_flgug` when the setup script is run.

The materials used in the geometry are defined in the `*.inp` files contained in `g4numi_flgug`.

5.1 Materials Definitions

In the latest version of `g4numi_flgug`, the material `CALCIUM` is defined twice in the `g4numi_flgugaMat.inp` file that is generated by FLUGG. This causes fluka to crash. As a work around we have hand edited the input file and renamed one of the materials from `CALCIUM` to `CALCIUM2`. These `*.inp` files are included in the CVS repository, so there shouldn't be any need to edit them further.

5.2 Geometry

We have configured the NOvA implementation of `numisoft` such that we can run simulations with either the low energy (i.e. MINOS) or medium energy (i.e. NOvA) target and configuration. Many files stored in `g4numi/src`, `g4numi/include`, and `/g4numi_flgug/for` end with the suffixes `.le_target` and `.me_target`, which encode the low energy and medium energy configuration, respectively. Switching between these two is accomplished by `g4numi_flgug/scripts/setup.sh`, details are in §6.2.1.

5.2.1 Selected Changes for the Medium Energy Target

Most of the effort in encoding the NOvA configuration was put into translating the drawings of the NOvA target design into GEANT4 code. In addition, several other changes and improvements were made to make the simulation as close to the reality of metal and concrete as possible. The coordinates and distances in this section are given in a coordinate system in which (0,0,0) is the center of the upstream end of Horn 1. Downstream refers to the direction in which the beam travels (+z direction), and upstream means the -z direction.

For the medium energy target, the target fin geometry and Be window thickness need to be finalized after some additional engineering work. No big changes are expected, but the individual target fins may be somewhat longer while keeping the overall target length the same.

We have implemented the thinner outer conductor for horn 1. The outer conductor for horn 1 was reduced from 1 inch to 5/8 inch. No change is expected for horn 2.

Horn 2 and the notch in the shielding block above it have been moved to the NOvA position.

There is no horn stripline material in the model and there is no fringe magnetic field in the model. We don't know if this is significant. Horn 1 will have its stripline modified, and the fringe magnetic field may be of concern.

For the NOvA geometry, we have implemented the 150 cm long baffle.

The target hall shielding has a different configuration above the horn 2 location for MINOS compared to the rest of the target chase. Probably this reflects the different shielding above the horn 2 module. We did not change this for the NOvA geometry.

In the `/g4numi_flgug/for/fluscw.*` files, the width of several formatted printing statements were increased to account for increased variable values in the NOvA configuration

6 Checking out and Running `g4numi_flgug`

In general, FLUGG is rather temperamental about having the proper environment variables set. So, if you are having problems performing any of the tasks below, try logging in to a fresh terminal and running only the setup scripts specified in the instructions for that task.

6.1 Checking Out

In the directory where you wish to install `numisoft`, perform the following steps.

```
$> export CVSROOT=cvsuser@cdcvs:/cvs/cd
$> export CVS_RSH=/usr/krb5/bin/rsh
$> cvs checkout nusoft/numisoft/g4numi
$> cvs checkout nusoft/numisoft/g4numi_flgug
```

Modify `g4numi_flguga.sh` script directory to match your location.

6.2 Setting Up and Compiling

Compilation of `g4numi_flgug` needs to be run if you modify the geometry or switch between the MINOS and NOvA geometries. It should also be done after you check out `numisoft` before you will be able to run simulations. To do this, perform the following steps.

- login to a NOvA node with no setup scripts
- run `. g4numi_flgug/scripts/setup.sh [experiment]`, where `[experiment]` is either `minos` or `nova`.
- compile `g4numi_flgug`

6.2.1 What the Setup Script Does

The script `g4numi_flgug/scripts/setup.sh` does several things. It sets up several environment variables and runs a setup script in the FLUGG directory so that FLUGG can compile. As originally setup, compiling `g4numi_flgug` required write access to the FLUGG code area. However, by setting the variable environment variable `G4WORKDIR` (this is done in the `g4numi_flgug/scripts/setup.sh`), the code is now compiled and located in the users local area. This solution prevents people from overwriting each other's code. Several other environment variables are set so that FLUGG uses the right version of programs on which it depends, such as `Fluka` and `Geant4`.

The script `setup.sh` configures the `g4numi` source files for either MINOS geometry with the low energy target, or the NOvA geometry with the medium energy target and modified horn 1. The geometry files are stored and should be modified in `g4numi/src` and `g4numi/include`. As alluded to in §5.2, several of these files end with the suffixes `.le_target` and `.me_target`. These are the versions of the code for the MINOS and NOvA configurations, respectively. The setup script copies the proper versions of these files into the `g4numi_flgug/src` and

`g4numi_flgug/include` directories; the copies do not have the target suffixes. The `[experiment]` parameter specifies which set of files are copied for compilation.

You need to recompile after running `g4numi_flgug/scripts/setup.sh`

6.3 Modifying the Geometry

The geometry files are stored in `nusoft/numisoft/g4numi/src` and `nusoft/numisoft/g4numi/inc`. Those that are different for MINOS and NOvA are designated by a suffix after `.cc` (`.me_target` for NOvA and `.le_target` for MINOS). These are geometry files that are formatted for GEANT4. Basic instructions for the format and how to modify the geometry are found in the GEANT4 Users Guide [3].

Unless one is varying some physical component of the target, horns, decay pipe, etc., there should be no need for most users to modify the geometry.

6.4 Modifying Materials Definitions

The materials are defined in the `*.inp` files stored in `nusoft/numisoft/g4numi_flgug`. Unless one is adding a new material to the simulation, there should be no reason to modify these files. Any such modifications should be done in the users' personal copy of FLUGG; no modifications of these files should be made in the official FLUGG installation except with the permission of the person in charge of FLUGG.

Several different materials files are currently used; each one is used by a different configuration. They are called by `nusoft/numisoft/g4numi_flgug/scripts/g4numi_fluka.sh` in this section of code:

```
if [ $TGT_GEOM == "minos" ] ; then
  if [ X$SPECIAL == "X_airhrn" ]; then
    echo "Air Horn run"
    cat ${source}/g4numi_flukaMat_airhrn.inp >> ${inp}
  elif [ $PERIOD -gt 2 ]; then
    echo "Helium run"
    cat ${source}/g4numi_flukaMat_helium.inp >> ${inp}
  else
    echo "Vacuum run"
    cat ${source}/g4numi_flukaMat_shield.inp >> ${inp}
  fi
elif [ $TGT_GEOM == "nova" ] ; then
  cat ${source}/g4numi_flukaMat_nova.inp >> ${inp}
fi
```

There are two ways to go about changing materials definitions. The simplest way is to simply hand-edit the existing files. We have done this to work around the CALCIUM repetition mentioned in §5.1. We also needed to add an entry for low energy cross sections (LOW-MAT) for the second calcium definition.

We replaced

```
MATERIAL 20.0 40.090 9.990e-01 43.0 CALCIUM
```

with

```
MATERIAL 20.0 40.090 9.990e-01 43.0 CALCIUM2
```

```
LOW-MAT 43.0 CALCIUM
```

FLUGG uses existing input files to create a temporary new one as part of its running. One can take advantage of this feature to bootstrap new input files.

- Edit `nusoft/numisoft/g4numi_flgug/g4numi.start` and put the `STOP` command on the last line.

- Run FLUGG; it will finish very quickly.
- In the resulting `Run##` directory, a file `*flukaMat.inp` shall appear. Copy this file back to `g4numi_flgug` and either overwrite one of the current files or give it a new name. If you create a new materials file this way, be sure to add it to the section of code in `/g4numi_fluka.sh` above.
- Edit `CALCIUM` to `CACIUM2` to avoid error and add `LOW-MAT` card as shown above.
- Remove the `STOP` command from `g4numi.start`
- Run FLUGG normally.

Please note that this procedure has not be fully validated, so experiment with caution.

6.5 Running Simulations

Once `g4num_flgug` has been compiled, it is ready to generate simulated events. The generation is done by running `g4numi_flgug/scripts/g4numi_fluka.sh`. This script provides several environment variables that can be used to configure the geometry, output, and other options. They are defined and described in [4, 5].

There are three ways to run FLUGG simulations: interactively, on the local condor queue, or on the grid.

The easiest way to run simulations is with an executable wrapper script that sets the environment variables and calls `g4numi_fluka.sh`. It is best to set `umask 002` in the highest level wrapper script to ensure files are group writeable. FLUGG requires that four of the environment variables be set, or it will crash with an error message: `RUN`, `EVTS`, `PERIOD`, and `TGT_GEOM`. Their functions and allowed values are described in Table 1.

6.5.1 Quick Start (Interactive or Local Condor Queue)

An example of such a script (for running interactively or on the local condor queue), which we shall call `wrapper_script.sh`:

```
export RUN=$PROCESS
export EVTS=$1
export SPECIAL=$2
export TGT_GEOM=nova
export PERIOD=1
[Insert path to your script directory]/g4numi_fluka.sh
```

To run interactively, first make the directory `fluxfiles` at the same level where `nusoft` is installed. This is where the output from FLUGG is placed by default. To run the script above interactively

```
$> export PROCESS=0
$> ./wrapper_script.sh 1000 _script_test
```

This will generate 1000 events in a root file in the directory `fluxfiles/flugg_mn000z200i_script_test/Run000`. For the meaning of `mn000z200i` and the overall naming convention for simulations, see §6.5.4.

For jobs much larger than this, it is much more convenient to use either the condor queue or the grid. The script above can be submitted to the condor queue as follows:

```
$> source /grid/fermiapp/nova/condor-scripts/setup_nova_condor.sh
$> nova_jobsub ./wrapper_script.sh 100000 _script_test
```

The output will be placed in the same directory as if it were run interactively.

Table 1: Required environment variables for FLUGG running

Variable	Example	Description
RUN	10	The run number for this simulation. It defines the output directory name and sets the random seed. If running in the condor queue, set this variable = \$PROCESS
EVTS	100000	The number of protons-on-target (POT) to simulate.
PERIOD	3	For the MINOS configuration, these set various conditions (e.g. Helium in the decay pipe) to correspond to the different MINOS Run Periods (currently 1-4). This variable is required to be set to something for the NOvA simulation, but it does not have any effect as yet.
TGT_GEOM	nova	This can be <code>minos</code> or <code>nova</code> ; this determines the name of the FLUGG output directory, the default horn current values, and default target z position.

6.5.2 Quick Start (Grid)

For the grid, a few lines should be added to the beginning of the script:

```
#!/bin/sh

umask 002 #Ensure that files are user and group writable by default

mkdir ${_CONDOR_SCRATCH_DIR}/output
export OUTPUTDIR=${_CONDOR_SCRATCH_DIR}/output/

export SHARED_DIR=/nova/data/flux/flugg #Use this path for official MC only
# you can make your own directory in the /nova/data area
export GRID=1
```

The resulting script can be submitted to the grid as follows:

```
$> source /grid/fermiapp/nova/condor-scripts/setup_nova_condor.sh
$> nova_jobsub -g ./wrapper_script.sh 100000 _script_test
```

Remember, after compiling on an SL5 machine, users need to use the `-onlySL5` flag when submitting to the grid.

The output should appear in `$$SHARED_DIR`.

Three environment variables are required if running on the grid:

`GRID` is a flag to let the FLUGG scripts know you are running on the grid. Set to 1 if running on the grid; leave undefined otherwise.

`OUTPUTDIR` is the directory to which the output is written on the grid node running the job. It is temporarily stored here to be copied back to a shared area after the job is finished. The standard way to set this is to have the following lines in a wrapper script, as shown above:

```
mkdir ${_CONDOR_SCRATCH_DIR}/output
export OUTPUTDIR=${_CONDOR_SCRATCH_DIR}/output/
```

Use `setenv` rather than `export` for sh scripts.

SHAREDDIR is shared area to which the files will be copied and permanently stored after the grid job is finished. For official MC production, this should be `/nova/data/flux/flugg`

6.5.3 More Details

Several other optional and useful environment variables are also available. Note the convention for SPECIAL

CURRENT sets the horn current (in kA), 185 by default for MINOS; 200 by default for NOvA.

TARGETZ sets the target z position in $-cm$, 10 by default for MINOS, 0 by default for NOvA.

STEPL sets the maximum step size in cm. Include `.` (decimal point), for example `STEPL="1.0"` Default is 1.0 if it is not set.

SPECIAL The contents of special is added to the run type. For example, if `SPECIAL="_sh"`, then the files would output to `flugg_1e010z185i_sh/` For official NOvA MC, SPECIAL should be set to `_[date]_[description]`. `[date]` is in the format YYYYMMDD, and `[description]` is optional.

LOWTH: Set to true to remove the 1 GeV tracking threshold. The string `"_lowth"` will be appended to SPECIAL.

TARGFILE: Set to true to output the target hadron ntuples. This is meant to provide a way to study the hadron production and other behavior of the target. When this variable is set to true, a second root file `target*.root` is produced in the `Run##` directory. This file contains an ntuple named `h3`, which contains 23 variables, several of which are specific to particle properties as they exit the "target," where the target is defined as a cylinder drawn around the target with a radius of 15 cm and length of approximately 150 cm. The variables and their definitions are contained in the main FLUGG note [4, 5].

The final lines of output from a successfully executed simulation look like the following

```
Processing /nas-pool/e929/users/corwin/nusoft/numisoft/g4numi_flugg/root/fill_
flux.C("flugg_me_nova_design_20100309_sizetest_010.root")...
Read and stored 323 events to g4numi001_flux.dat
rm: cannot remove 'g4numi001_GNUMI': No such file or directory
/nas-pool/e929/users/corwin/nusoft/numisoft/g4numi_flugg/scripts/g4numi_fluka.
sh:line 443: return: can only 'return' from a function or sourced script
```

6.5.4 Directory Structure and Naming Conventions

Official NOvA MC is stored in `/nova/data/flux/flugg/` Whether in this directory or on your personal area, FLUGG will create subdirectories of the form `flugg_[l/m] [e/n] [nnn]z[nnn] i[SPECIAL]`. SPECIAL is described in §6.5.3.

The names of simulation configurations take the following form: `[l/m][e/n][nnn]z[nnn]i`; this is an extension of the MINOS convention

- `[l/m]`: l = low energy horn/target configuration, m = medium energy horn/target configuration
- `[e/n]`: e = MINOS target, n = NOvA target. In the event of target design changes, we have 24 letters left in the alphabet
- `[nnn]`: z = pull back (unchanged for MINOS target, relative to nominal pos. for NOvA target)
- `[nnn]`: i = horn current in kA (same as for MINOS)

A standard NOvA run will be `mn000z200i`.

Within `flugg_[l/m] [e/n] [nnn]z[nnn] i[SPECIAL]`, each run will be placed inside of a directory `Run##`, with `##` being the number assigned to the RUN environment variable. Inside of

Run##, a successful run will produce several files of the form `flugg_[l/m] [e/n] [nnn]z[nnn] i[SPECIAL]_##.*`, with the `*.root` file containing the FLUGG simulation. A second ROOT file will also appear if the `TARGFILE` variable is set, as described earlier.

The variables in the final FLUGG ntuple and their definitions are contained in the main FLUGG note [4, 5]. Note that all of the detector related variables are for the **MINOS** detector, not the NOvA detector.

6.6 Visualizing the Simulated Geometry

To visualize the geometry encoded in FLUGG, we use the HepRApp viewer [8]. To install, first go to the `numisoft` directory.

```
$> mkdir HepRApp
$> cd HepRApp
$> wgethttp://www.slac.stanford.edu/~perl/HepRApp/HepRApp.jar
$> chmod +x HepRApp.jar
$> cd ../g4numi
```

To run the visualization if you have it installed or are using the official NOvA installation, you need to perform the next steps in `numisoft/numisoft/g4numi`.

```
$> source /grid/fermiapp/nova/novaart/novasoft/releases/development/setup/setup_
novasoft_ifcluster.sh
$> . setup.sh [experiment]
$> gmake clean
$> gmake
$> ./g4numi < vis.mac
$> ../HepRApp/HepRApp.jar G4Data0.heprep
```

As before, `[experiment]` can be `minos` or `nova`. The `vis.mac` macro file is setup to generate a HepRep and to create particle trajectories. This creates a file `G4Data0.heprep`.

The final line of code above actually runs the viewer to see the visualization. The viewer is pretty self explanatory. Right clicking on the mouse allows you to choose the orientation. Scroll bars on the side and bottom allow you to pan and rotate.

The visibility tree on the right allows you to make parts of the geometry visible or invisible. As an example of how to use the visibility tree, the decay pipe volume is located in the HepRApp visibility tree at `[top of tree] → Detector Geometry → Detector Geometry and SubTypes[0] → TUNE → TUNE and SubTypes[0] → DPIP → DPIP and SubTypes[0] → DVOL → DVOL and SubTypes[0] → DVOL[0]`

7 Common Problems and Solutions

- `mainG4NuMI.cc:5:28: FGeometryInit.hh: No such file or directory`
This probably means you have the wrong environment variables set. Try logging into a fresh window, running no setup scripts (except as instructed in §6.2, and recompiling.
- If running interactively, make sure you have `RUN` set to something. If you are using the example wrapper files, you can either manually set `RUN` to a value in the file or set `PROCESS` on the command line before running the wrapper.
- If your job ends normally but produces a root file with only 1 entry, your materials definitions could be wrong. Perform the steps in §6.4

8 Unresolved Issues

In the latest version of `g4numi_flugg`, the material `CALCIUM` is defined twice in the `g4numi_flukaMat.inp` file that is generated by `flugg`. This causes `fluka` to crash. As a work around we have hand edited the input file and renamed one of the materials from `CALCIUM` to `CALCIUM2`.

References

- [1] M. Campanella, A. Ferrari, P. R. Sala, and S. Vanini. Reusing code from `fluka` and `geant4` geometry. ALT-SOFT 98-039, CERN, 1998.
- [2] M. Campanella, A. Ferrari, P. R. Sala, and S. Vanini. First calorimeter simulation with the `flugg` prototype. ALT-SOFT 99-004, CERN, 1999.
- [3] Geant4 Collaboration. <http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/html/ch02s02.html>.
- [4] A. I. Himmel. The numi beam simulation with `flugg`. MINOS Document 6316, California Institute of Technology, 2009.
- [5] A. I. Himmel. The numi beam simulation with `flugg`. NOvA Document 5388, California Institute of Technology, 2009.
- [6] NOvA Collaboration. https://cdcv.s.fnal.gov/redmine/wiki/nova-cvs/Running_NOvA_Beam_Simulations.
- [7] NOvA Collaboration. https://cdcv.s.fnal.gov/redmine/wiki/nova-cvs/Accessing_NOvA_software_at_FNAL.
- [8] J. Perl. <http://www.slac.stanford.edu/BFROOT/www/Computing/Graphics/Wired/>.
- [9] P. R. Sala and S. Vanini. <http://www.fluka.org/content/tools/flugg/>.

A Variable Definitions

Ndecay	Process
1	$K_L^0 \rightarrow \nu_e + \pi^- + e^+$
2	$K_L^0 \rightarrow \bar{\nu}_e + \pi^+ + e^-$
3	$K_L^0 \rightarrow \nu_\mu + \pi^- + \mu^+$
4	$K_L^0 \rightarrow \bar{\nu}_\mu + \pi^+ + \mu^-$
5	$K^+ \rightarrow \nu_\mu + \mu^+$
6	$K^+ \rightarrow \nu_e + \pi^0 + e^+$
7	$K^+ \rightarrow \nu_\mu + \pi^0 + \mu^+$
8	$K^- \rightarrow \bar{\nu}_\mu + \mu^-$
9	$K^- \rightarrow \bar{\nu}_e + \pi^0 + e^-$
10	$K^- \rightarrow \bar{\nu}_\mu + \pi^0 + \mu^-$
11	$\mu^+ \rightarrow \bar{\nu}_\mu + \nu_e + e^+$
12	$\mu^- \rightarrow \bar{\nu}_\mu + \nu_e + e^-$
13	$\pi^+ \rightarrow \nu_\mu + \mu^+$
14	$\pi^- \rightarrow \bar{\nu}_\mu + \mu^-$
999	Other

Table 2: The decay codes stored in Ndecay.

Particle	Fluka	Geant	PDG
	Hadron File	Flux File	SNTF
γ	7	1	22
e^+	4	2	-11
e^-	3	3	11
μ^+	10	5	-13
μ^-	11	6	13
π^0	23	7	111
π^+	13	8	211
π^-	14	9	-211
K_L^0	12	10	130
K^0	24	10/16	311
\bar{K}^0	25	10/16	-311
K^+	15	11	321
K^-	16	12	-321
n	8	13	2112
p	1	14	2212
\bar{p}	2	15	-2212
K_S^0	19	16	310
Λ	17	18	3122
Σ^+	21	19	3222
Σ^0	22	20	3212
Σ^-	20	21	3112
Ξ^0	34	22	3322
Ξ^-	36	23	3312
Ω^-	38	24	3334
\bar{n}	9	25	-2112
$\bar{\Lambda}$	18	26	-3122
$\bar{\Sigma}^-$	31	27	-3222
$\bar{\Sigma}^0$	32	28	-3212
$\bar{\Sigma}^+$	33	29	-3112
$\bar{\Xi}^0$	35	30	-3322
$\bar{\Xi}^+$	37	31	-3312
Ω^+	39	32	-3334
τ^+	41	33	-15
τ^-	42	34	15
$\bar{\nu}_e$	6	52	-12
ν_e	5	53	12
$\bar{\nu}_\mu$	28	55	-14
ν_μ	27	56	14
$\bar{\nu}_\tau$	44	-	-16
ν_τ	43	-	16

Table 3: The particle codes across the three schemes used in MINOS.

Code	Material
5	Beryllium
6	Carbon
9	Aluminum
10	Iron
11	Slab Steel
12	Blu Steel
15	Air
16	Vacuum
17	Concrete
18	Target
19	Rebar Concrete
20	Shotcrete
21	Variable Density Aluminum
22	Variable Density Steel
23	1018 Steel
24	A500 Steel
25	Water
26	M1018 Steel
28	Decay Pipe Vacuum
31	CT852

Table 4: The material codes as defined by Gnuni and used in the fluxfiles, old and current.

Variable	Description
run	Run number (not used)
evtno	Event number (proton on target)
Ndxdz Ndydz	Neutrino direction slopes for a random decay
Npz	Neutrino momentum (GeV/c) along the z -axis (beam axis)
Nenergy	Neutrino energy (GeV) for a random decay
NdxdzNea NdydzNea	Direction slopes for a neutrino forced towards the center of the MI-NOS or NOvA near detector
NenergyN	Energy for a neutrino forced towards the center of the MINOS or NOvA near detector
NWtNear	Weight for a neutrino forced towards the center of the MINOS or NOvA near detector
NdxdzFar NdydzFar	Direction slopes for a neutrino forced towards the center of the MI-NOS or NOvA far detector
NenergyF	Neutrino energy (GeV) for a decay forced to the center of the MINOS or NOvA far detector
NWtFar	Neutrino weight for a decay forced to the center of the MINOS or NOvA far detector
Norig	Not used in flux file
Ndecay	Decay process that produced the neutrino (Table 2)
Ntype	Neutrino flavor. $\nu_\mu = 56, \bar{\nu}_\mu = 55, \nu_e = 53, \bar{\nu}_e = 52$
Vx Vy Vz	Neutrino production vertex (cm)
pdPx pdPy pdPz	Momentum (GeV/c) of the neutrino parent at the neutrino production vertex (parent decay point)
ppdxdz ppdydz	Direction of the neutrino parent at its production point (which may be in the target)
pppz	z momentum (GeV/c) of the neutrino parent at its production point
ppenergy	Energy (GeV) of the neutrino parent at its production point
ppmedium	Code for the material the neutrino parent was produced in (Table 4)
ptype	Neutrino parent species (GEANT codes, see Table 3)
ppvx ppvz	Production vertex (cm) of the neutrino parent
muparpx muparpy muparpz	Momentum (GeV/c) of the neutrino grandparent at the grandparent decay point (muons) or grandparent production point (hadrons).
mupare	Energy (GeV) of the neutrino grandparent, as above
Necm	Neutrino energy (GeV) in the center-of-mass frame
Nimpwt	Importance weight of the neutrino

Table 5: The entries stored in the neutrino ntuple files. There is one entry for every neutrino produced. For variables specific to a detector, the MINOS detector positions are used when `setup.sh minos` or `setup.sh me` are run. The NOvA positions (including the underground near detector positions) are used when `setup.sh nova` is run.

Variable	Description
xpoint	
ypoint	Debugging hook – unused
zpoint	
tvx	
tvx	Position (cm) of the neutrino ancestor as it exits target (possibly, but not necessarily, the direct neutrino parent)
tvx	
tpx	
tpy	Momentum (GeV/c) of the ancestor as it exits target
tpz	
tptype	Species of the ancestor exiting the target (GEANT codes)
tgen	Neutrino parent generation in cascade. 1 = primary proton, 2 = particles produced by proton interaction, 3 = particles from interactions of the 2's, etc.
tgtype	Species of the parent of the particle exiting the target (GEANT codes)
tgppx	
tgppy	Momentum (GeV/c) of the parent of the particle exiting the target at the parent production point.
tgppz	
tprivx	
tprivy	Primary particle interaction vertex (not used)
tprivz	
beamx	
beamy	Primary proton origin (cm)
beamz	
beampx	
beampy	Primary proton momentum (GeV/c)
beampz	
Vr	$\sqrt{V_x^2 + V_y^2}$
pdP	$\sqrt{pdPx^2 + pdPy^2 + pdPz^2}$
pdPt	$\sqrt{pdPx^2 + pdPy^2}$
ppp	Total momentum (GeV/c) of the neutrino parent at its production point.
pppt	Transverse momentum (GeV/c) of the neutrino parent at its production point ($= \sqrt{ppp^2 - pppz^2}$).
ppvr	$\sqrt{ppvx^2 + ppvy^2}$
muparp	$\sqrt{muparx^2 + mupary^2 + muparz^2}$
muparpt	$\sqrt{muparx^2 + mupary^2}$
tvr	$\sqrt{tvx^2 + tvy^2}$
tp	$\sqrt{tpx^2 + tpy^2 + tpz^2}$
tpt	$\sqrt{tpx^2 + tpy^2}$

Table 6: The entries stored in the neutrino ntuple files. There is one entry for every neutrino produced.

Variable	Description
x	
y	Position (cm) of the particle as it exits target
z	
px	
py	Momentum (GeV/c) of the parent as it exits target
pz	
type	Species of the particle leaving the target (Fluka codes)
weight	Weight
gener	Generation
montype	Species of the parent of the particle exiting the target (Fluka codes)
momp	
px	Momentum (GeV/c) of the parent of the particle exiting the target
mpy	at the parent decay point
mpz	
protvx	
protvy	Primary particle interaction vertex (not used)
protvz	
protx	
proty	Primary proton origin (cm)
protz	
protpx	
protpy	Primary proton momentum (GeV/c)
protpz	
event	Event number

Table 7: The entries stored in the hadron ntuple files. There is one entry for each hadron that exits the target volume.