Introduction to GENIE

Gabriel N. Perdue
Fermilab

Overview

- Getting help
- Installation
- Run the basic applications
- Simple output checks

The goal for this lecture: install GENIE, make a cross section spline file, generate some events. A green-field build of GENIE will take too long if we are only starting now, but start the build anyway. Hopefully you already have it, in which case focus on making splines and interactions.

Prefatory Remarks

- These presentations attempt to be consistent in how they refer to files, functions, and directories.
- When I remember to, I use back-ticks (``) to mark functions, classes, compiled programs, and code snippets, and single-ticks ('') to mark files and directories.
- Directory paths are generally understood to begin with '$GENIE/' - the installation root environment var.
- Directory paths into the source code sometimes start with 'src/' and sometimes don't. If it isn't clear, ask.
- Configuration xml will usually be assumed to be found in '$GENIE/config/'.
- Bash shell expansion will be used in the text (so, "File.{h,cxx}" is "File.h File.cxx" as would resolve under `ls`).

$ Commands will often (but not always) start with a dollar-prompt and use a different (Courier) font.

Getting Help

- http://genie.hepforge.org
- For most user and physics questions, check the Physics and Users Manual:
  - http://genie.hepforge.org/manuals/GENIE_PhysicsAndUserManual_20130615.pdf
  - When the manual is updated, just check the left hand side of the GENIE main page.
  - There is also a mailing list:
    - https://www.jiscmail.ac.uk/cgi-bin/webadmin?A0=NEUTRINO-MC-SUPPORT
- We are also working on a Developer's Manual aimed at implementing/tuning models:
  - http://home.fnal.gov/~perdue/GENIEDevelopersManual.html
  - When it is done it will live on the GENIE main page.
  - You can look at the "code" (it is written in AsciiDoc, which is a mostly literate markup language that compiles to HTML and PDF) and modify it and submit pull requests on GitHub:
    - https://github.com/GENIEMC/GENIEDevelopersManual

Framework Internals

- We won't discuss the framework internals here. In general, you don't need to know much about them to use GENIE and develop and tune physics models.
  - There are a couple of organizing patterns we need to understand. We'll go over those in this school.
- If you are really interested in framework internals, see
  - Section Five of the NIM*
  - http://projects-docdb.fnal.gov/cgi-bin/ShowDocument?docid=2922

*Andreopoulos, C. and Bell, A. and Bhattacharya, D. and Cavanna, F. and Dobson, J. and others. "The GENIE Neutrino Monte Carlo Generator". Nucl.Instrum.Meth. A614. 87-104. 2010.

Installation

- There are instructions in the Physics and Users manual.
- Additionally, there are instructions on the web:
  - http://genie.hepforge.org/installation.html
- The only “tricky” parts are third-party libraries and compiler and OS support. Generally, GCC on *nix systems will work.

Installation

- If you download the GENIE source code, you will find installation scripts for the third-party libraries in:
  - $GENIE/src/scripts/build/ext
- Also, for LINUX ONLY*, there is a full installation script hosted on our UNOFFICIAL scratch project site on GitHub:
  - https://github.com/GENIEMC/lamp

```
$ git clone https://github.com/GENIEMC/lamp.git

$ ./rub_the_lamp.sh    # no arguments to see help

$ ./rub_the_lamp.sh -p 6  # <- this is probably what you want
```

- Note that "lamp" uses slightly different scripts to install the third party packages than those included with GENIE. (A to-do for us is to unify these.)
- ROOT takes a while to build, but installation is otherwise fairly quick. The total time is less than one hour.

*For Macs, see: http://home.fnal.gov/~perdue/GENIEDevelopersManual.html

Short Side Note

- If you installed GENIE via `lamp` last week you may have built ROOT without gdml enabled. (It is a library I didn't know was needed.)
- Goto $ROOT: Add `--enable-gdml` to the 'config.status' file. `make distclean`, then `./configure `cat config.status`` and `make`.
- Then goto $GENIE and `make clean && make`.
- If you installed after May 11 or so, the scripts included this library in the configure.

Installation

- Installation on Linux systems with recent versions of GCC is very straightforward.
- We do not exercise builds on Windows very often and don't really support it.
- Installation on a newer Mac (or older Mac with "newer", 5.x+ Xcode), is tricky because Apple has stopped supporting the GNU toolchain and GENIE does not build using the version of `clang` that comes with Xcode 5.x+. (The fix for `clang` is possibly understood, but not implemented yet).
  - One fix is to install GCC yourself. (You will need gfortran also.) See the following link and check for the subsection on building on OSX for ideas on how to do this (we plan on a more official fix this summer):
    - http://home.fnal.gov/~perdue/GENIEDevelopersManual.html#_building_genie

Running GENIE

- Three basic modes:
  - Generate “interaction splines.” These are files that store the total cross section as a function of channel and neutrino energy on different targets.
    - Computing the total cross section is computationally expensive, but it doesn’t change event to event, so do it once for all the materials in your experiment and store the results.
  - Generate interactions.
  - Run validation apps.

gmkspl

- GENIE Make Spline
- Example:

```
$ gmkspl -p 14,-14 -t 1000060120 -o ccqe_carbon_splines.xml \
--event-generator-list CCQE
```

- Set the projectile (PDG code), the target (PDG Ion Code), the output, and the event generator list.
- See '$GENIE/config/EventGeneratorListAssembler.xml' for a list of valid event generator lists.
  - You will almost always want the default, but you may want to produce a narrower set of channels for specific studies.
- Event generators themselves are defined in '$GENIE/config/UserPhysicsOptions.xml'. (You will see this file a lot!)

EventGeneratorListAssembler.xml

Previous slide...

```
<param_set name="CCQE">
   <param type="int" name="NGenerators"> 1                                  </param>
   <param type="alg" name="Generator-0"> genie::EventGenerator/QEL-CC </param>
</param_set>
```

Defined in EventGenerator.xml

UserPhysicsOptions.xml

```
 <!--
  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  Specify which cross section model is to be used by each GENIE
  event generation thread. The parameter name is build as:
  "XSecModel@[name of thread]"
  -->
<param type="alg" name="XSecModel@genie::EventGenerator/QEL-CC">
  genie::LwlynSmithQELCCPXSec/Default
</param>
```

To drill down further, see 'config/EventGenerator.xml'

```
<param_set name="QEL-CC">
   <param type="string" name="VldContext"> </param>
   <param type="int"    name="NModules">   12                                          </param>
   <param type="alg"    name="Module-0">   genie::InitialStateAppender/Default         </param>
   <param type="alg"    name="Module-1">   genie::VertexGenerator/Default              </param>
   <param type="alg"    name="Module-2">   genie::FermiMover/Default                   </param>
   <param type="alg"    name="Module-3">   genie::QELKinematicsGenerator/CC-Default    </param>
   <param type="alg"    name="Module-4">   genie::QELPrimaryLeptonGenerator/Default    </param>
   <param type="alg"    name="Module-5">   genie::QELHadronicSystemGenerator/Default   </param>
   <param type="alg"    name="Module-6">   genie::PauliBlocker/Default                 </param>
   <param type="alg"    name="Module-7">   genie::UnstableParticleDecayer/BeforeHadronTransport </param>
   <param type="alg"    name="Module-8">   genie::NucDeExcitationSim/Default          </param>
   <param type="alg"    name="Module-9">   genie::HadronTransporter/Default           </param>
   <param type="alg"    name="Module-10">  genie::NucBindEnergyAggregator/Default     </param>
   <param type="alg"    name="Module-11">  genie::UnstableParticleDecayer/AfterHadronTransport  </param>
   <param type="alg"    name="ILstGen">    genie::QELInteractionListGenerator/CC-Default  </param>
</param_set>
```

When running a `QEL-CC` event, these are the modules we step through. Note that this is specified in text, not in code and may be changed without re-compiling (and even on-the-fly during a run).

Where do we find them?

You can often find a "lookup table" in 'master_config.xml':

```
<genie_config>

   <!-- ****** CONFIGURATION FOR EVENT GENERATION MODULES ****** -->
   <config alg="genie::EventGenerator">              EventGenerator.xml              </config>
   <config alg="genie::FermiMover">                  FermiMover.xml                  </config>
   <config alg="genie::HadronTransporter">           HadronTransporter.xml           </config>
   ...
   <config alg="genie::QELKinematicsGenerator">      QELKinematicsGenerator.xml      </config>
```

And extra information if you follow the table (here 'QELKinematicsGenerator.xml', etc.).

Typically though, these files also correspond to class names by convention:

```
$ ls -1 src/QEL/Q*.{h,cxx}
src/QEL/QELHadronicSystemGenerator.cxx
src/QEL/QELHadronicSystemGenerator.h
src/QEL/QELInteractionListGenerator.cxx
src/QEL/QELInteractionListGenerator.h
src/QEL/QELKinematicsGenerator.cxx
src/QEL/QELKinematicsGenerator.h
src/QEL/QELPrimaryLeptonGenerator.cxx
src/QEL/QELPrimaryLeptonGenerator.h
```

Example Cross Section Spline:

In general, the spline you need for your experiment will use both neutrinos and antineutrinos and have many different models and targets.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!-- generated by genie::XSecSplineList::SaveSplineList() -->

<genie_xsec_spline_list version="2.00" uselog="1">

<spline
  name="genie: LwlynSmithQELCCPXSec/Default/nu:-14;
  tgt:1000060120;
  N:2212;
  proc:Weak[CC],QES;"
  nknots="44">
<knot> <E>    0.01000 </E> <xsec>           0 </xsec> </knot>
<knot> <E>   0.030466 </E> <xsec>           0 </xsec> </knot>
<knot> <E>   0.050932 </E> <xsec>           0 </xsec> </knot>
<knot> <E>   0.071399 </E> <xsec>           0 </xsec> </knot>
<knot> <E>   0.091865 </E> <xsec>           0 </xsec> </knot>
<knot> <E>    0.11233 </E> <xsec>           0 </xsec> </knot>
<knot> <E>     0.1427 </E> <xsec> 8.487371629e-13 </xsec> </knot>
<knot> <E>    0.18129 </E> <xsec> 4.103713443e-12 </xsec> </knot>
<knot> <E>     0.2303 </E> <xsec> 8.367696679e-12 </xsec> </knot>
<knot> <E>    0.29257 </E> <xsec> 1.290439962e-11 </xsec> </knot>
<knot> <E>    0.37168 </E> <xsec> 1.807792243e-11 </xsec> </knot>
...
```

Target (Carbon)

Cross Section Algorithm

Neutrino Flavor (PDG Code)

Energy (GeV)

Cross Section (GeV^{-1})

- Algorithm and class names follow the convention of naming the file after the class, so they are easy to find:

```
$ cd $GENIE/src

$ find . -name "*LwlynSmithQELCCPXSec*"

./LlewellynSmith/LwlynSmithQELCCPXSec.cxx

./LlewellynSmith/LwlynSmithQELCCPXSec.h
```

When running find and grep, you may want to mask certain directories (e.g., `svn` with `find . -name '*file*' | grep -v svn`, etc.).

More on Splines

- Generating splines takes a significant amount of CPU time (recall we need a computation over dozens of energy positions, dozens of models, and dozens of targets).
- You can find some useful pre-built splines here:
  - https://www.hepforge.org/archive/genie/data/2.8.0/
  - And, at Fermilab, read:
    - https://cdcvs.fnal.gov/redmine/projects/genie/wiki/UPS_products_genie_phyopt_and_genie_xsec

Other Spline Utilities/Comments

- See the Physics and User's Manual for details.
  - `gspladd` – merge spline files.
  - `gspl2root` – convert XML cross section files in a ROOT file.
- Currently, the full cross section is loaded at the beginning of a run. There is some work on-going for making lazy-loading possible, but it is not available yet.

More on Configuration

- Recall the list of modules that appeared in EventGenerator.xml for QEL-CC.
- How do we configure the associated parameters?
- Example: How to turn off intranuclear rescattering? Go to '$GENIE/src':

```
$ find . -name '*HadronTransporter*'

./EVGModules/HadronTransporter.cxx

./EVGModules/HadronTransporter.h
```

LoadConfig

```
void HadronTransporter::LoadConfig(void)
{
  AlgConfigPool * confp = AlgConfigPool::Instance();
  const Registry * gc = confp->GlobalParameterList();

  fHadTranspModel = 0;
  fEnabled = fConfig->GetBoolDef("enable", gc->GetBool("HadronTransp-Enable"));

  LOG("HadTransp", pDEBUG)
       << "Hadron transport was " << ((fEnabled) ? "" : "not ") << " enabled";
  if(fEnabled) {
     RgAlg hadtransp_model =
        fConfig->GetAlgDef("model", gc->GetAlg("HadronTransp-Model"));
     LOG("HadTransp", pDEBUG)
          << "Loading the hadron transport model: " << hadtransp_model;

     // Note: this relies on the fact that if a missing registry entry was
     // found above then it would be filled up by the default choice so that
     // when the registry is looked-up again the same key would point to the
     // correct value. Probably it would be better to convert the RgAlg to AlgId
     // and query the AlgFactory directly.
     fHadTranspModel =
         dynamic_cast<const EventRecordVisitorI *> (this->SubAlg("model"));
     assert(fHadTranspModel);
  }
}
```

We see this pattern of reading from the AlgConfigPool and Registry quite frequently.

We search $GENIE/config:

```
$ grep HadronTransp-Model *
```

```
HadronTransporter.xml:model alg Yes GPL HadronTransp-Model

UserPhysicsOptions.xml: <param type="alg" name="HadronTransp-
Model"> genie::HAIntranuke/Default </param>
```

Drilling down in 'UserPhysicsOptions.xml':

```
<!--
 Intranuclear rescattering
 Use the HadronTransportEnable option to toggle intranuclear rescattering on/off.
 Also, set the preferred hadron transport model.
 Options include:
  - genie::HAIntranuke/Default
  - genie::HNIntranuke/Default
 -->
 <param type="bool" name="HadronTransp-Enable"> true </param>
 <param type="alg"  name="HadronTransp-Model">  genie::HAIntranuke/Default  </param>
```

All we need to do then is set the enable flag to `false`.

- Another example: M_A for CCQE.
- Recall the cross section algorithm is:
  - LwlynSmithQELCCPXSec
  - Go to '$GENIE/src':

```
$ find . -name '*LwlynSmithQELCCPXSec*'

./LlewellynSmith/LwlynSmithQELCCPXSec.cxx

./LlewellynSmith/LwlynSmithQELCCPXSec.h
```

LoadConfig

```
void LwlynSmithQELCCPXSec::LoadConfig(void)
{
  AlgConfigPool * confp = AlgConfigPool::Instance();
  const Registry * gc = confp->GlobalParameterList();

  double thc = fConfig->GetDoubleDef(
                                  "CabbiboAngle", gc->GetDouble("CabbiboAngle"));
  fCos8c2 = TMath::Power(TMath::Cos(thc), 2);

   // load QEL form factors model
  fFormFactorsModel = dynamic_cast<const QELFormFactorsModelI *> (
                                                  this->SubAlg("FormFactorsAlg"));
  assert(fFormFactorsModel);
  fFormFactors.SetModel(fFormFactorsModel); // <-- attach algorithm

   // load XSec Integrator
  fXSecIntegrator =
      dynamic_cast<const XSecIntegratorI *> (this->SubAlg("XSec-Integrator"));
  assert(fXSecIntegrator);
}
```

If we grep the config directory for "FormFactorsAlg," we find:

```
LwlynSmithQELCCPXSec.xml:     <param type="alg"  name="FormFactorsAlg">  genie::LwlynSmithFFCC/Default </param>
LwlynSmithQELCCPXSec.xml:     <param type="alg"  name="FormFactorsAlg">  genie::LwlynSmithFFCC/Dipole  </param>
LwlynSmithQELCCPXSec.xml:     <param type="alg"  name="FormFactorsAlg">  genie::LwlynSmithFFCC/BBA2003 </param>
LwlynSmithQELCCPXSec.xml:     <param type="alg"  name="FormFactorsAlg">  genie::LwlynSmithFFCC/BBA2005 </param>
```

Then, searching src for "LwlynSmithFFCC," we find:

```
./LlewellynSmith/LwlynSmithFFCC.cxx
./LlewellynSmith/LwlynSmithFFCC.h
```

There, we find the computations are done in "LwlynSmithFF".

LoadConfig

```
void LwlynSmithFF::LoadConfig(void)
{
// Load configuration data from its configuration Registry (or global defaults)
// to private data members

  fElFFModel = 0;

  AlgConfigPool * confp = AlgConfigPool::Instance();
  const Registry * gc = confp->GlobalParameterList();

  // load elastic form factors model
  RgAlg form_factors_model = fConfig->GetAlgDef(
                 "ElasticFormFactorsModel", gc->GetAlg("ElasticFormFactorsModel"));
  fElFFModel =  dynamic_cast<const ELFormFactorsModelI *> (
                                          this->SubAlg("ElasticFormFactorsModel"));
  assert(fElFFModel);

  fELFF.SetModel(fElFFModel);

  // axial mass and Fa(q2=0)
  fMa  = fConfig->GetDoubleDef("Ma",  gc->GetDouble("QEL-Ma"));  // Axial mass
  fFA0 = fConfig->GetDoubleDef("FA0", gc->GetDouble("QEL-FA0")); // FA(q2=0)

  fMa2 = TMath::Power(fMa,2);

  // anomalous magnetic moments
  fMuP = fConfig->GetDoubleDef("MuP", gc->GetDouble("AnomMagnMoment-P"));
  fMuN = fConfig->GetDoubleDef("MuN", gc->GetDouble("AnomMagnMoment-N"));

  // weinberg angle
  double thw = fConfig->GetDoubleDef(
                           "WeinbergAngle", gc->GetDouble("WeinbergAngle"));
  fSin28w = TMath::Power(TMath::Sin(thw), 2);
}
```

Finally, going back to the config directory, we grep for `QEL-Ma`...

In 'UserPhysicsOptions.xml':

```
<!--
 Axial and vector masses for quasi-elastic scattering
 -->
 <param type="double" name="QEL-Ma"> 0.990 </param>
 <param type="double" name="QEL-Mv"> 0.840 </param>
```

Whew!

- It can sometimes take a bit of `find`ing and `grep`ing to get what you are after, but everything is there and organized this way for a reason.

  - Note though that we could have found this quickly by searching UserPhysicsOptions for "MA" and "Ma" (case-sensitive).

- For example, suppose you wanted to implement a QE model that used different form factors. Would it make sense for M_A to be located in QELKinematics? Similarly, would you want two copies of a parameter (CC & NC) when there should be only one?

gevgen

- **G**ENIE **Ev**ent **Gen**eration
- Example:

```
$ gevgen -n 10 -p 14 -t 1000080160 -e 1,10 \
-r 100 -f 'x*exp(-x)' --seed 398819 \
--cross-sections $PATH/gxspl-vA-2.8.0.xml
```

- Many other options of course (use a geometry (more on this in a later lecture), weight the events (be careful), specify a generator list, etc.

gevgen

- What happens (in detail) when we run `gevgen`?
- It is very worthwhile walking through with a debugger (e.g., `gdb`). See
  - http://home.fnal.gov/~perdue/GENIEDevelopersManual.html#_running_the_code_the_ccqe_walkthrough
- `gdb` is also very useful for understanding new models under development without resorting to dozens of debug print statements.
- The Developer's Manual (still under construction) also has some useful resources like '.gdbinit' files and basic `gdb` and `valgrind` usage, etc. (clone the repository to get all the files).

Output

- How do we know the code succeeded?
  - GENIE leaves voluminous log files (use '$GENIE/config/Messenger.xml' to control verbosity – the default setting may be too "talkative" for your needs).
  - Under default settings there are pretty-printed GHepRecords we may examine.
  - Failures are (usually clearly) explained:

```
1396638461 NOTICE Rndm : [n] <RandomGen.cxx::SetSeed (92)> : Setting default random number seed at random number generator initialization: 65539
1396638461 NOTICE Rndm : [n] <RandomGen.cxx::SetSeed (92)> : Setting random number seed: 2989819
1396638461 FATAL AppInit : [n] <AppInit.cxx::XSecTable (64)> : Input cross-section file [/minerva/app/users/perdue/GENIE/data/
carbon_oxygen_splines.xml] does not exist!
```

Path-length Scaling

- First GENIE will look at all the interactions it has cross section information for in the spline files and build a total cross section model.

- Then it will look at the geometry and find the largest possible path length (here it is 1 for a single oxygen nucleus).

- Then it will scale the interaction probability so that a neutrino crossing the largest path has an interaction probability of 1.

- This scaling keeps the MC efficient and alters the rate we "use up" neutrinos when we are drawing from flux file representing a certain number of protons on target.

```
1396387997 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::BootstrapXSecSplineSummation (675)> : Summing-up splines to get total cross section for each init state
1396387997 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::BootstrapXSecSplineSummation (683)> : **** Summing xsec splines for init-state = nu-pdg:14;tgt-pdg:1000080160;
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::BootstrapXSecSplineSummation (702)> : Finished summing all interaction xsec splines per initial state
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GetMaxPathLengthList (587)> : Querying the geometry driver to compute the max path-length list
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GetMaxPathLengthList (592)> : Maximum path length list:
[-]
 |---o code: 1000080160 [  O16] -----> path-length = 1
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::ComputeProbScales (719)> : Computing the max. interaction probability (probability scale)
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::ComputeProbScales (810)> : *** Probability scale = 6.78377e-15
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::Configure (487)> : Finished configuring GMCJDriver
```

Flux Neutrinos

- Once we know the probability scale, we begin drawing flux neutrinos and see if they interact (most will not, but thanks to the scaling, it will not require an overly large number of neutrinos to get an interaction).

```
1396387998 NOTICE gevgen : [n] <gEvGen.cxx::GenerateEventsUsingFluxOrTgtMix (328)> :  *** Generating event..... 0
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateEvent (823)> : Generating next event...
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateFluxNeutrino (1022)> : Generating a flux neutrino
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateFluxNeutrino (1036)> :
 [-] Generated flux neutrino:
  |----o PDG-code   : 14
  |----o 4-momentum : (E = 0.327306, px = 0, py = 0, pz = 0.327306) / M^2 = 0 / P = 0.327306
  |----o 4-position : (t = 0, x = 0, y = 0, z = 0)
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateEvent1Try (875)> : Computing interaction probabilities for max. path lengths
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::ComputeInteractionProbabilities (1092)> : Computing relative interaction probabilities for each material
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateEvent1Try (880)> : The no-interaction probability (max. path lengths) is: 97.2556 %
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateEvent1Try (890)> : ** Rejecting current flux neutrino
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateEvent (839)> : Flux neutrino didn't interact - Trying the next one...
```

- Eventually, we get an interaction and the actual channel is drawn from a cross-section-weighted list of all possible reactions (unphysical reactions will have cross section = weight = 0).

```
1396387998 NOTICE GMCJDriver : [n] <GMCJDriver.cxx::GenerateEventKinematics (1210)> : Asking the selected GEVGDriver object to generate an event
1396387998 NOTICE IntSel : [n] <PhysInteractionSelector.cxx::SelectInteraction (98)> :
=============================================================================================
  Selecting an interaction for the given initial state = nu-pdg:14;tgt-pdg:1000080160; at E = 2.57702 GeV
=============================================================================================
1396387998 NOTICE IntSel : [n] <PhysInteractionSelector.cxx::SelectInteraction (100)> : Computing xsecs for all relevant modeled interactions:
1396387998 NOTICE IntSel : [n] <PhysInteractionSelector.cxx::SelectInteraction (170)> :
 |-------------------------------------------------------------------------------------------|
 |                                                    interaction | cross-section (1E-38*cm^2) |
 |-------------------------------------------------------------------------------------------|
 |                              nu:14;tgt:1000080160;N:2112;proc:Weak[CC],QES; |                    7.41357 |
 |                              nu:14;tgt:1000080160;N:2212;proc:Weak[NC],QES; |                    1.24454 |
 |                              nu:14;tgt:1000080160;N:2112;proc:Weak[NC],QES; |                     1.7118 |
 |                         nu:14;tgt:1000080160;N:2212;proc:Weak[CC],RES;res:0; |                     6.1741 |
... etc.
```

GHepRecords

- Discussed "programatically" later. But how do we make sense of the log files?

```
|-------------------------------------------------------------------------------------------------------------|
|GENIE GHEP Event Record [print level:   3]                                                                   |
|-------------------------------------------------------------------------------------------------------------|
| Idx |          Name | Ist |        PDG |   Mother  | Daughter  |      Px |      Py |      Pz |       E |      m  |
|-------------------------------------------------------------------------------------------------------------|
|   0 |         nu_mu |   0 |         14 |  -1 |  -1 |   4 |   4 |   0.000 |   0.000 |   3.000 |   3.000 |   0.000 |
|   1 |           C12 |   0 | 1000060120 |  -1 |  -1 |   2 |   3 |   0.000 |   0.000 |   0.000 |  11.179 |  11.179 |
|   2 |       neutron |  11 |       2112 |   1 |  -1 |   5 |   5 |  -0.054 |   0.160 |  -0.080 |   0.934 | **0.940 | M = 0.915
|   3 |           C11 |   2 | 1000060110 |   1 |  -1 |  12 |  12 |   0.054 |  -0.160 |   0.080 |  10.245 |  10.243 |
|   4 |           mu- |   1 |         13 |   0 |  -1 |  -1 |  -1 |   0.133 |   0.656 |   0.513 |   0.849 |   0.106 | P =
(-0.158,-0.778,-0.608)
|   5 |      HadrSyst |  12 | 2000000001 |   2 |  -1 |   6 |   7 |  -0.188 |  -0.495 |   2.407 |   3.084 | **0.000 | M = 1.854
|   6 |       neutron |  14 |       2112 |   5 |  -1 |   8 |  10 |  -0.670 |  -0.529 |   1.086 |   1.670 |   0.940 | FSI = 7
|   7 |           pi+ |  14 |        211 |   5 |  -1 |  11 |  11 |   0.482 |   0.034 |   1.321 |   1.414 |   0.140 | FSI = 1
|   8 |       neutron |   1 |       2112 |   6 |  -1 |  -1 |  -1 |  -0.014 |  -0.186 |   0.749 |   1.216 |   0.940 |
|   9 |        proton |   1 |       2212 |   6 |  -1 |  -1 |  -1 |  -0.653 |  -0.408 |   0.296 |   1.249 |   0.938 |
|  10 |           pi- |   1 |       -211 |   6 |  -1 |  -1 |  -1 |   0.002 |  -0.057 |   0.024 |   0.153 |   0.140 |
|  11 |           pi+ |   1 |        211 |   7 |  -1 |  -1 |  -1 |   0.482 |   0.034 |   1.321 |   1.414 |   0.140 |
|  12 |      HadrBlob |  15 | 2000000002 |   3 |  -1 |  -1 |  -1 |   0.049 |  -0.038 |   0.096 |   9.297 | **0.000 | M = 9.297
|-------------------------------------------------------------------------------------------------------------|
|       Fin-Init:                                                 |   0.000 |   0.000 |  -0.000 |  -0.000 |         |
|-------------------------------------------------------------------------------------------------------------|
|       Vertex:          nu_mu @ (x =     0.00000 m, y =     0.00000 m, z =     0.00000 m, t =               0 s)  |
|-------------------------------------------------------------------------------------------------------------|
| Err flag [bits:15->0] : 0000000000000000    |  1st set:                                                  none |
| Err mask [bits:15->0] : 1111111111111111    |  Is unphysical:    NO |   Accepted:   YES                                |
|-------------------------------------------------------------------------------------------------------------|
| sig(Ev) =        8.1385e-38 cm^2  | d2sig(x,y;E)/dxdy =        2.4893e-37 cm^2       | Weight =               1 |
|-------------------------------------------------------------------------------------------------------------|
```

GHepRecords

- We also get an interaction summary:

(more on this in a later lecture)

```
------------------------------------------------------------------------------------------------------
GENIE Interaction Summary
------------------------------------------------------------------------------------------------------
[-] [Init-State]
 |--> probe        : PDG-code = 14 (nu_mu)
 |--> nucl. target : Z = 6, A = 12, PDG-Code = 1000060120 (C12)
 |--> hit nucleon  : PDC-Code = 2112 (neutron)
 |--> hit quark    : PDC-Code = 1 (d) [valence]
 |--> probe 4P     : (E =            3, Px =            0, Py =            0, Pz =            3)
 |--> target 4P    : (E =       11.179, Px =            0, Py =            0, Pz =            0)
 |--> nucleon 4P   : (E =     0.933909, Px =   -0.0542517, Py =     0.160284, Pz =   -0.0800058)
[-] [Process-Info]
 |--> Interaction : Weak[CC]
 |--> Scattering  : DIS
[-] [Kinematics]
 |-->  *Running* Hadronic invariant mass W = 1.85428
 |--> *Selected* Bjorken x = 0.435806
 |--> *Selected* Inelasticity y = 0.757872
 |--> *Selected* Momentum transfer Q2 (>0) = 2.00929
 |--> *Selected* Hadronic invariant mass W = 1.85428
[-] [Exclusive Process Info]
 |--> charm prod.  : false
 |--> f/s nucleons : N(p) = 0 N(n) = 0
 |--> f/s pions    : N(pi^0) = 0 N(pi^+) = 0 N(pi^-) = 0
 |--> resonance    : [not set]
------------------------------------------------------------------------------------------------------
```

GHepRecords

Initial State

These specify *ranges*.

```
|----------------------------------------------------------------------------------------------------------------------------------|
|GENIE GHEP Event Record [print level:   3]                                                                                        |
|----------------------------------------------------------------------------------------------------------------------------------|
| Idx |          Name | Ist |        PDG |  Mother  | Daughter  |      Px |      Py |      Pz |       E |      m  |
|----------------------------------------------------------------------------------------------------------------------------------|
|   0 |         nu_mu |   0 |         14 |  -1 |  -1 |   4 |   4 |   0.000 |   0.000 |   3.000 |   3.000 |   0.000 |
|   1 |           C12 |   0 | 1000060120 |  -1 |  -1 |   2 |   3 |   0.000 |   0.000 |   0.000 |  11.179 |  11.179 |
|   2 |       neutron |  11 |       2112 |   1 |  -1 |   5 |   5 |  -0.054 |   0.160 |  -0.080 |   0.934 | **0.940 | M = 0.915
|   3 |           C11 |   2 | 1000060110 |   1 |  -1 |  12 |  12 |   0.054 |  -0.160 |   0.080 |  10.245 |  10.243 |
|   4 |           mu- |   1 |         13 |   0 |  -1 |  -1 |  -1 |   0.133 |   0.656 |   0.513 |   0.849 |   0.106 | P =
(-0.158,-0.778,-0.608)
|   5 |      HadrSyst |  12 | 2000000001 |   2 |  -1 |   6 |   7 |  -0.188 |  -0.495 |   2.407 |   3.084 | **0.000 | M = 1.854
|   6 |       neutron |  14 |       2112 |   5 |  -1 |   8 |  10 |  -0.670 |  -0.529 |   1.086 |   1.670 |   0.940 | FSI = 7
|   7 |           pi+ |  14 |        211 |   5 |  -1 |  11 |  11 |   0.482 |   0.034 |   1.321 |   1.414 |   0.140 | FSI = 1
|   8 |       neutron |   1 |       2112 |   6 |  -1 |  -1 |  -1 |  -0.014 |  -0.186 |   0.749 |   1.216 |   0.940 |
|   9 |        proton |   1 |       2212 |   6 |  -1 |  -1 |  -1 |  -0.653 |  -0.408 |   0.296 |   1.249 |   0.938 |
|  10 |           pi- |   1 |       -211 |   6 |  -1 |  -1 |  -1 |   0.002 |  -0.057 |   0.024 |   0.153 |   0.140 |
|  11 |           pi+ |   1 |        211 |   7 |  -1 |  -1 |  -1 |   0.482 |   0.034 |   1.321 |   1.414 |   0.140 |
|  12 |      HadrBlob |  15 | 2000000002 |   3 |  -1 |  -1 |  -1 |   0.049 |  -0.038 |   0.096 |   9.297 | **0.000 | M = 9.297
|----------------------------------------------------------------------------------------------------------------------------------|
|       Fin-Init:                                                  |   0.000 |   0.000 |  -0.000 |  -0.000 |         |
|----------------------------------------------------------------------------------------------------------------------------------|
|       Vertex:           nu_mu @ (x =     0.00000 m, y =     0.00000 m, z =     0.00000 m, t =                0 s)              |
|----------------------------------------------------------------------------------------------------------------------------------|
| Err flag [bits:15->0] : 0000000000000000    |  1st set:                                                                  none |
| Err mask [bits:15->0] : 1111111111111111    |  Is unphysical:    NO |   Accepted:   YES                                          |
|----------------------------------------------------------------------------------------------------------------------------------|
| sig(Ev) =        8.1385e-38 cm^2  | d2sig(x,y;E)/dxdy =        2.4893e-37 cm^2       | Weight =                1 |
|----------------------------------------------------------------------------------------------------------------------------------|
```

1's "go to GEANT". See 'src/GHEP/GHepStatus.h'

Not enumerated?

GHepRecords

```
typedef enum EGHepStatus {
   kISt Undefined                 = -1,
   kISt InitialState              =  0, /* generator-level initial state */
   kISt StableFinalState          =  1, /* generator-level final state: particles to be tracked by detector-level MC */
   kISt IntermediateState         =  2,
   kISt DecayedState              =  3,
   kISt CorrelatedNucleon         = 10,
   kISt NucleonTarget             = 11,
   kISt DISPreFragmHadronicState  = 12,
   kISt PreDecayResonantState     = 13,
   kISt HadronInTheNucleus        = 14, /* hadrons inside the nucleus: marked for hadron transport modules to act on */
   kISt FinalStateNuclearRemnant  = 15, /* low energy nuclear fragments entering the record as a pseudo-particle */
   kISt NucleonClusterTarget      = 16
}
GHepStatus_t;
```

```
|----------------------------------------------------------------------
|GENIE GHEP Event Record [print level:   3]
|----------------------------------------------------------------------
| Idx |          Name | Ist |        PDG |   Mother   | Daug
|----------------------------------------------------------------------
|   0 |         nu_mu |   0 |         14 |  -1 |  -1 |   4
|   1 |           C12 |   0 | 1000060120 |  -1 |  -1 |   2
|   2 |       neutron |  11 |       2112 |   1 |  -1 |   5
|   3 |           C11 |   2 | 1000060110 |   1 |  -1 |  12
|   4 |           mu- |   1 |         13 |   0 |  -1 |  -1
(-0.158,-0.778,-0.608)
|   5 |      HadrSyst |  12 | 2000000001 |   2 |  -1 |   6
|   6 |       neutron |  14 |       2112 |   5 |  -1 |   8
|   7 |           pi+ |  14 |        211 |   5 |  -1 |  11
|   8 |       neutron |   1 |       2112 |   6 |  -1 |  -1
|   9 |        proton |   1 |       2212 |   6 |  -1 |  -1
|  10 |           pi- |   1 |       -211 |   6 |  -1 |  -1
|  11 |           pi+ |   1 |        211 |   7 |  -1 |  -1
|  12 |      HadrBlob |  15 | 2000000002 |   3 |  -1 |  -1
|----------------------------------------------------------------------
|      Fin-Init:
|----------------------------------------------------------------------
|      Vertex:          nu_mu @ (x =     0.00000 m, y =
|----------------------------------------------------------------------
| Err flag [bits:15->0] : 0000000000000000    |  1st set:
```

Validation Apps

- Compare predictions from GENIE to data.
- There is an extensive "database" of experimental results (stored as ASCII plain-text files) distributed with GENIE, along with citation information.
- More on this in a later lecture.

Conclusions / Review

- Install GENIE using instructions in the Physics and User's Manual or a script package like `lamp`.
- `gmkspl` is the primary app for producing cross section summary splines. These files are used for event selection during event generation.
- `gevgen` creates events.
  - Running either app without arguments will produce a usage guide.
- Both apps feature detailed logs. With default verbosity, you may study the details of interactions during event generation.

Thanks!

Back-up

gmkspl

- What happens when we run `gmkspl`?
- See instructions later in this presentation for a `gdb` walkthrough during event generation.

```
$ gmkspl -p 14,-14 -t 1000060120 -o test.xml --event-generator-list CCCOH
```

- For test runs, it is a good idea to use a single element for your target and limit the species - it can take a very long time to iterate through all the reaction channels and complicated geometries may feature dozens of nuclear targets.