How to use MC generators

Tomasz Golan
University of Wroclaw

NuSTEC, Fermilab 2017

Useful links

- NuWro User Guide
- How to use NuWro
- GENIE Manual
- GENIE Developer Manual

Running NuWro

Params.txt

To run NuWro use the following command:

```
./bin/nuwro   [-i input parameters file] \
              [-o output root file] \
              [-p ''parameter name 1 = value 1''] \
              [-p ''parameter name 2 = value 2''] ...
```

NuWro uses by default the params.txt file located in "nuwro" directory. If the file does not exist, the one from "nuwro/data" folder is loaded. If both files are missing or some of the parameters are not set in the file, default values are used.

NuWro saves by default the event tree into the eventsout.root file. Cross sections are saved by default into the eventsout.root.txt file (it will be discussed later).

Events vs test events

Test events are used to calculate cross section.
They are not saved! It is very fast. Usually, 10^6 test events is enough.

```
number_of_test_events = unsigned int
```

A number of events saved in the output file is set by the parameter:

```
number_of_events = unsigned int
```

Single neutrino flavor beam

beam_type = 0
beam_particle = PDG $(\pm 12, \pm 14, \pm 16)$
beam_energy = E $\rightarrow$ mono-energetic beam
beam_energy = E_{min} E_{max} $\rightarrow$ uniform beam
beam_energy = E_{min} E_{max} a_0 a_1 ... a_n

beam with energy range from E_{min} to E_{max}, $a_i / \sum_j^n a_j$ gives a probability the energy will be drawn from $(i * \varepsilon, \ (i+1) * \varepsilon)$ interval, where $\varepsilon = (E_{max} - E_{min})/n$

Example: beam_energy = 1000 2000 1 2 3 4

10% $\rightarrow E_\nu$ from 1000 1250, 20% $\rightarrow E_\nu$ from 1250 1500 ...

Mixed neutrino flavor beam

beam_type = 1

beam_content = n_1 x_1% be_1

beam_content += n_2 x_2% be_2 ...

n_i → *PDG*, x_i → *fraction of this kind of neutrino*

be_i → *like beam_energy*

Example:

```
beam_content = 12 75% 1000
beam_content += -12 20% 1000 2000
beam_content += 14 5% 1000 1500 1 5 10 15 5 1
```

75% of mono-energetic electron neutrinos

20% of electron anti-neutrinos with uniformly distributed energy ...

Predefined beams

```
@beam/beamfile.txt
```

Predefined beams are located in "nuwro/data/beam" directory.

Setting up a target

Single nucleus

target_type = 0	
nucleus_p = *unsigned int*	→ a number of protons
nucleus_n = *unsigned int*	→ a number of neutrons
nucleus_E_b = *double*	→ a binding potential
nucleus_kf = *double*	→ Fermi momentum
nucleus_target = 0 - 5	→ nucleus model

0 - free nucleon, 1 - Fermi gas, 2 - local Fermi gas

Note, in local Fermi gas k_F and E_B are calculated from the density profile.

Composed target

```
target_type = 1
```

target_content = p_1 n_1 f_1x [E_{B1} k_{F1} NT_1]

target_content += p_2 n_2 f_2x [E_{B2} k_{F2} NT_2] ...

p_i → *number of protons*, n_i → *number of neutrons*

f_i → *number of i-th kind of nucleus in the target*

E_{Bi}, k_{Fi}, NT_i → *binding energy, Fermi momentum, nucleus_target*

Example (C_2H_6O):

```
target_content = 6 6 2x
target_content += 1 0 6x
target_content += 8 8 1x
```

Setting up a target

Predefined targets

```
@target/targetfile.txt
```

Predefined beams are located in "nuwro/data/target" directory.

Turn on/off channels

Channels

dyn_qel_cc = 0,1	→ quasi-elastic charge current
dyn_qel_nc = 0,1	→ elastic neutral current
dyn_res_cc = 0,1	→ resonance pion production CC
dyn_res_nc = 0,1	→ RES NC
dyn_dis_cc = 0,1	→ deep inelastic scattering CC
dyn_dis_nc = 0,1	→ DIS NC
dyn_coh_cc = 0,1	→ coherent pion production CC
dyn_coh_nc = 0,1	→ COH NC
dyn_mec_cc = 0,1	→ meson exchange current CC
dyn_mec_nc = 0,1	→ MEC NC

Some other parameters

Beam direction - default (0,0,1)

beam_direction = x y x

Electromagnetic form factors parameterizations

qel_vector_ff_set = 1 - 6

Axial form factors parameterizations

qel_axial_ff_set = 1 - 4

Axial mass (CC)

qel_cc_axial_mass = M_A

Spectral function

sf_method = 0 - 2

The model for meson exchange current

mec_kind = 1 - 4

see user guide for details and the full list of parameters

Analyzing NuWro's output

event1.h

class event : public TObject

`flags flag;`	qel, res, nc, cc ...
`vector <particle> in;`	incoming particles
`vector <particle> out;`	particles before FSI
`vector <particle> post;`	particles after FSI

Predefined functions

`vect q();`	four-momentum transfer
`double q2();`	four-momentum transfer squared
`double W();`	invariant mass
`int nof (int PDG);`	#particles with PDG before FSI
`int fof (int PDG);`	#particles with PDG after FSI

and many more... see src/event1.h for details

particle.h

CLASS PARTICLE : PUBLIC VECT

`double E();`	total energy
`double Ek();`	kinetic energy
`double mass();`	mass
`double momentum();`	momentum (value)
`vec p();`	momentum as a vector
`vect& p4();`	four-momentum

and many more... see src/particle.h for details

CLASS VEC

`double x, y, z;`	coordinates
`double length();`	vector length
`vec operator+ (vec a, vec b);`	and other operations

CLASS VECT

```
double t, x, y, z; ...
```

and many more... see src/vec.h and src/vect.h for details

Consider charge current scattering of a mono-energetic muon neutrino beam ($E_\nu = 1$ GeV) on carbon.

1. Create an empty file in nuwro directory (*run1.txt*)
2. Set up the parameters (in *run1.txt*):

beam_type = 0	*mono-energetic beam*
beam_particle = 14	*muon neutrino*
beam_energy = 1000	$E_\nu = 1000$ *MeV*
@target/C.txt	*predefined carbon*
dyn_qel_cc = 1	*QEL CC*
dyn_qel_nc = 0	*EL NC*
dyn_res_cc = 1	*RES CC*
dyn_res_nc = 0	*RES NC*
dyn_dis_cc = 1	*DIS CC*
dyn_dis_nc = 0	*DIS NC*
dyn_coh_cc = 1	*COH CC*
dyn_coh_nc = 0	*COH NC*
dyn_mec_cc = 1	*MEC CC*
dyn_mec_nc = 0	*MEC NC*

3. Run NuWro:

./bin/nuwro -i run1.txt -o run1.root

4. You will get two files:

a) run1.root with the events tree

b) run1.root.txt with total cross sections in cm^2

5. To analyze the ROOT file use:

./bin/myroot

1. Load a ROOT file:

TFile f = new TFile ("run1.root")*

2. Set up an pointer to event tree:

TTree t = (TTree*)f→Get("treeout")*

3. Draw some simple distributions:

t→Draw("in[0].E()") — *neutrino energy*

t→Draw("in[1].Ek()") — *primary nucleon kinetic energy*

t→Draw("out[0].p().z") — p_z *of the outgoing lepton*

4. Add extra conditions:

4a. π^+ momentum distribution after FSI

t→Draw("post.momentum()", "post.pdg == 211")

4b. Q^2 distributions for events with single π^0:

t→Draw("-q2()", "fof(111) == 1 && fof(211)+fof(-211) == 0")

Script example

1. Create the *script1.C* file:

```
TFile *f;
TTree *t;

void setFile (const char* input){
  f = new TFile(input);
  t = (TTree*)f->Get("treeout");
}

void leptonEnergy (){
  t->Draw("out[0].E()");
}

void pi0cosine (){
 t->Draw("post.p().z/post.momentum()", "post.pdg == 111");
}
```

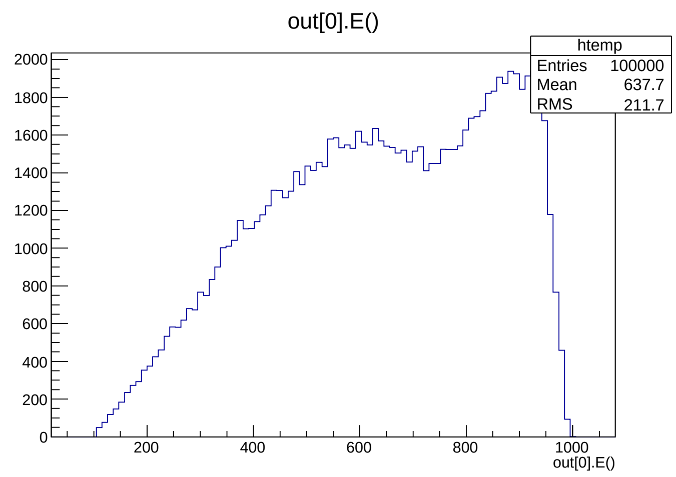
2. Usage:

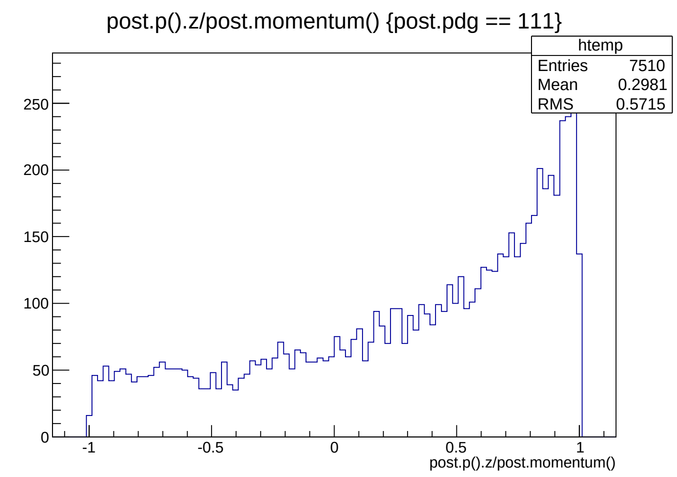
.L script1.C

setFile("run1.root")

leptonEnergy()

pi0cosine()

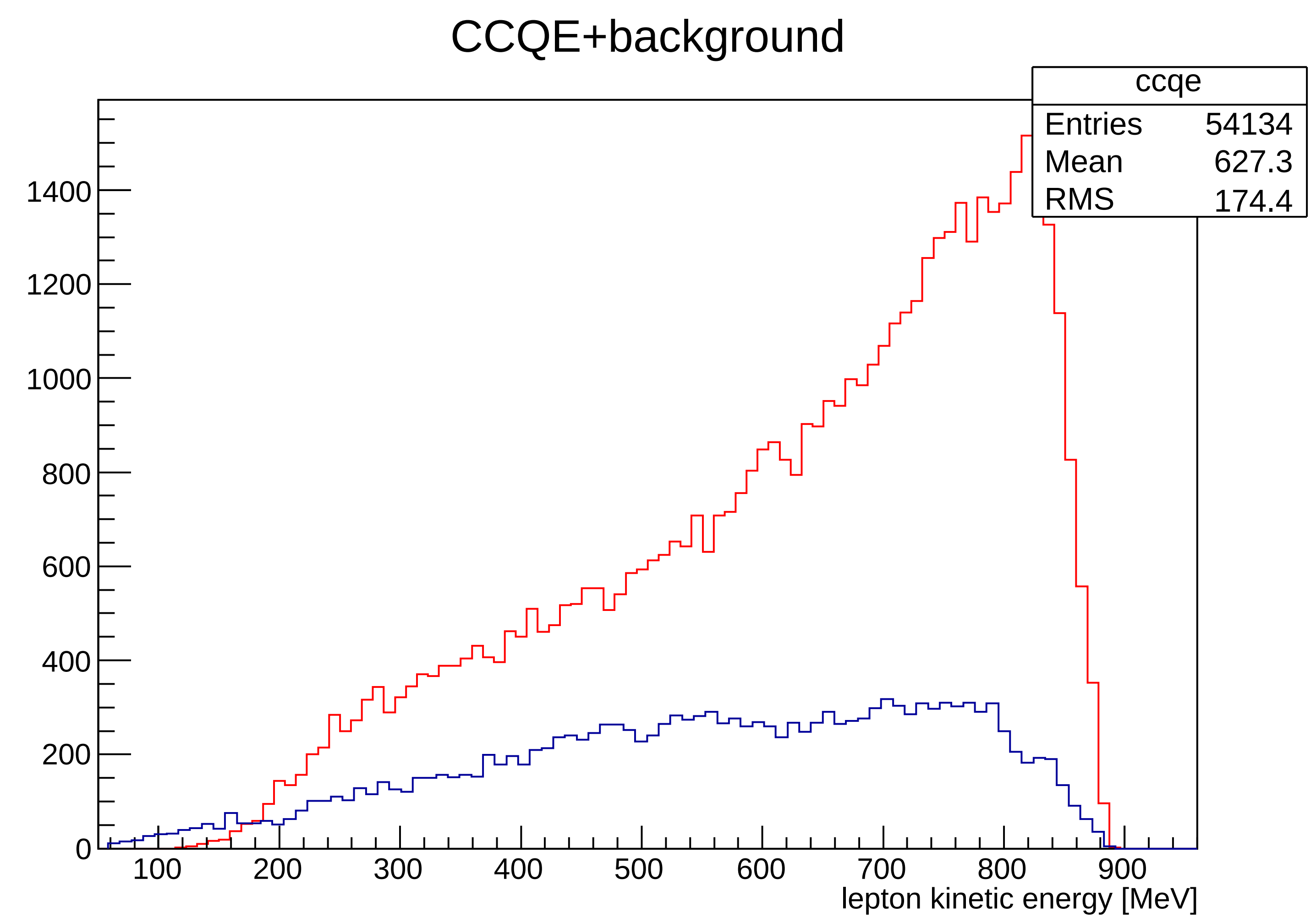




```
void firstPlot (const char* input){
  TFile *f = new TFile(input);
  TTree *t = (TTree*)f->Get("treeout");

  //create "ccqe" and "background" histograms with some cuts
  //goff -> do not create autocanvas

  t->Draw("out[0].Ek() >> ccqe", "flag->qel", "goff");
  t->Draw("out[0].Ek() >> background", "!flag->qel \
          && fof(211)+fof(111)+fof(-211)==0", "goff");

  TCanvas *c = new TCanvas;
  ccqe->SetLineColor(kRed); ccqe->SetTitle("CCQE+background");
  ccqe->SetXTitle("lepton kinetic energy [MeV]");

  ccqe->Draw();
  bkg->Draw("same"); //"same" -> on the same plot

  gSystem->ProcessEvents();
  TImage *img = TImage::Create();
  img->FromPad(c);
  img->WriteImage("first_plot.png");
}
```

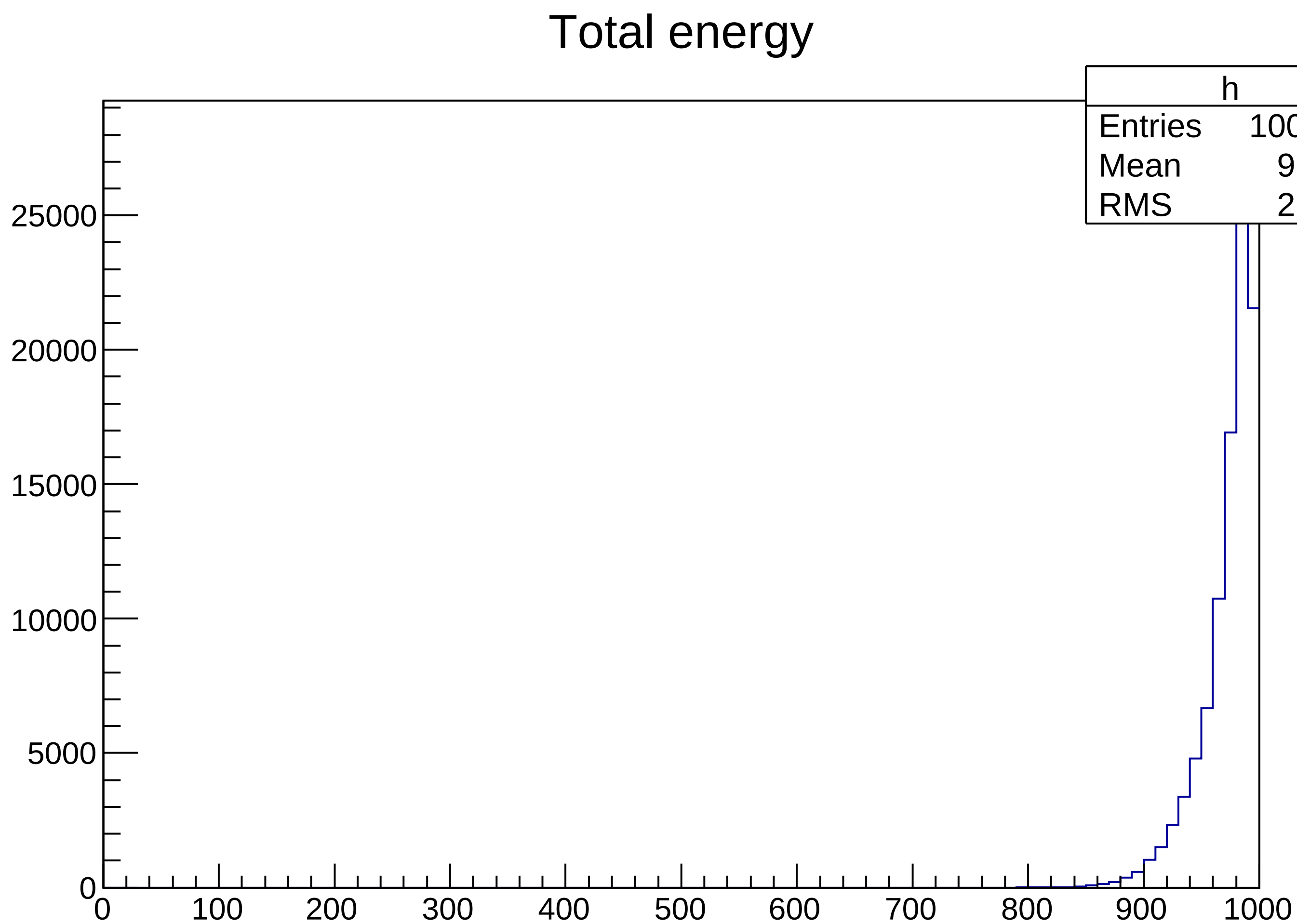
Plot example

The result of the above script:



```
void eventByEvent (const char* input){
  TFile *f = new TFile(input);
  TTree *t = (TTree*)f->Get("treeout");
  //create a pointer to event
  event *e   = new event();
  t->SetBranchAddress("e",&e);

  TH1D* h = new TH1D("h", "Total energy", 100, 0, 1000);

  for (int i = 0; i < t->GetEntries(); i++){
    t->GetEntry(i);

    double E = 0;
    for (int k = 0; k < e->post.size(); k++)
      if (e->post[k].nucleon())
        E += e->post[k].Ek();
      else
        E += e->post[k].E();

    h->Fill(E);
  }
  h->Draw();
}
```

The result of the above script:



Cross section

In NuWro each event is accepted with the probability proportional to the cross section (in a tree each event is equally weighted).

$$\left.\frac{d\sigma}{dx}\right|_{x=x_0} \to \frac{N(x = x_0 \pm \Delta x/2)}{N_{total}} \frac{\sigma_{total}}{\Delta x}$$

The table with cross sections (per nucleon) is saved into the *eventout.root.txt* file:

Channel	*#events*	*Fraction*	*Cross section [cm^2]*	
0	*54134*	*0.54134*	*5.71421e-39*	*(qel cc)*
1	*0*	*0*	*0*	*(qel nc)*
2	*33534*	*0.335339*	*3.53972e-39*	*(res cc)*
3	*0*	*0*	*0*	*(res nc)*
4	*48*	*0.000480844*	*5.07563e-42*	*(dis cc)*
5	*0*	*0*	*0*	*(dis nc)*

...

To read cross section from the *txt* file you can use the following function:

```
double xsec (const char* input)
{
  double temp, res = 0;
  ifstream Input (input);

  getline (Input,string());
  while(Input)
  {
    for (int k = 0; k < 4; k++)
      Input>>temp;
    res+=temp;
  }
  Input.close();
  return res;
}
```

Example

Now we try to figure it out how M_A affects the shape and the normalization of the cross section

1. Prepare the *ccqe_par.txt* file (like *run1.txt* but only QEL CC is on)
2. Prepare a bash script (*ccqe.sh*):

```
#!/bin/sh
for i in $(seq 1000 100 1300)
do
  ./bin/nuwro -i ccqe_par.txt -o ccqe$i.root -p "qel_cc_axial_mass = $i"
done
```

You will get 4 ROOT files: ccqe1000.root, ccqe1100.root, ...

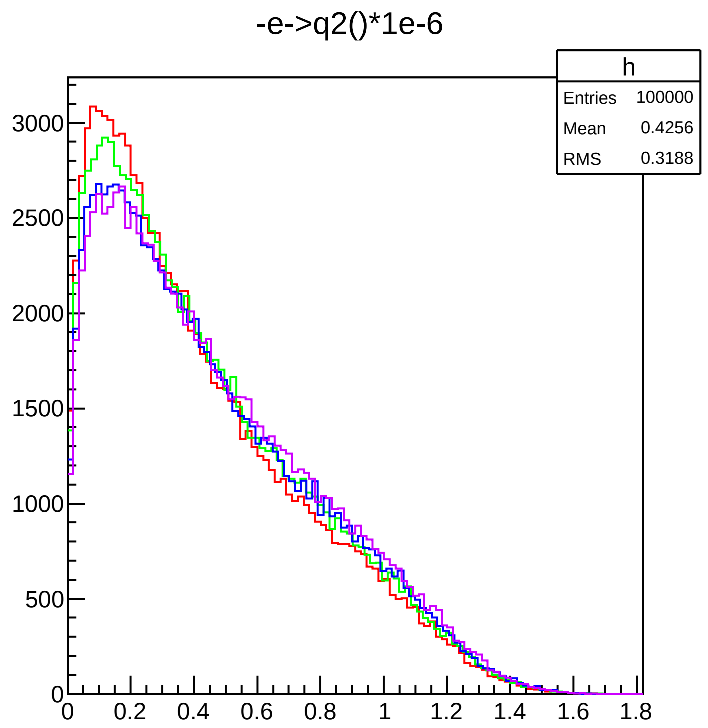
Example

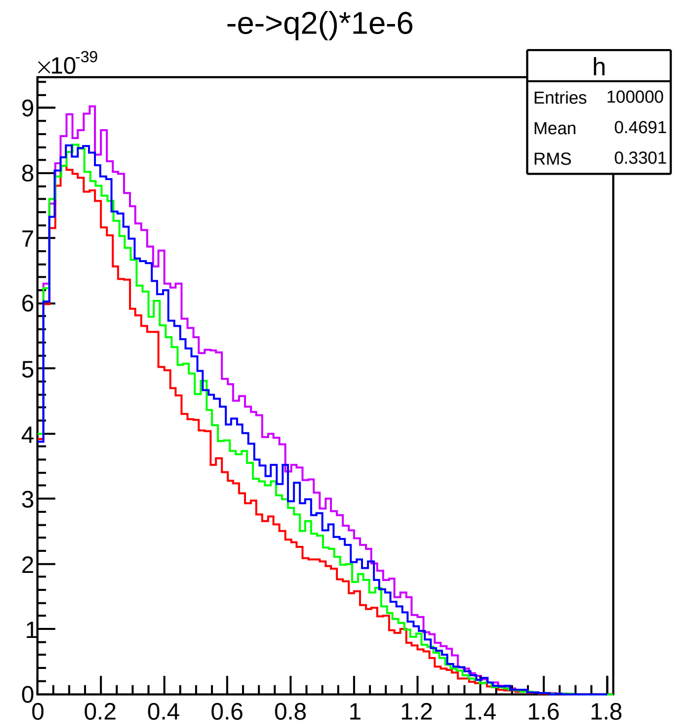
3. Prepare a function for the extraction of a histogram from a ROOT file, for example:

```
TH1F* ccqe_q2 (const char* input){
  TFile *f = new TFile(input);
  TTree *t = (TTree*)f->Get("treeout");
  t->Draw("-e->q2()*1e-6 >> h","","goff");
  TH1F *res = new TH1F(*h);
  return res;
}
```

```
void ccqe_run(){
  TH1F* h1000 = ccqe_q2("ccqe1000.root");
  TH1F* h1100 = ccqe_q2("ccqe1100.root");
  TH1F* h1200 = ccqe_q2("ccqe1200.root");
  TH1F* h1300 = ccqe_q2("ccqe1300.root");

  TCanvas *c = new TCanvas; c -> Divide(2,1); c -> cd(1);

  h1000->SetLineColor(kRed); h1000->Draw();
  h1100->SetLineColor(kGreen); h1100->Draw("same");
  h1200->SetLineColor(kBlue); h1200->Draw("same");
  h1300->SetLineColor(kViolet); h1300->Draw("same");

  c->cd(2);
  double factor = 1.0 / h1000->GetBinWidth(0) / h1000->GetEntries();

  TH1F* h1000n = new TH1F(*h1000 * xsec("ccqe1000.root.txt") * factor);
  TH1F* h1100n = new TH1F(*h1100 * xsec("ccqe1100.root.txt") * factor);
  TH1F* h1200n = new TH1F(*h1200 * xsec("ccqe1200.root.txt") * factor);
  TH1F* h1300n = new TH1F(*h1300 * xsec("ccqe1300.root.txt") * factor);

  h1300n->Draw(); h1000n->Draw("same");
  h1100n->Draw("same"); h1200n->Draw("same");
}
```

The result of the above script:





Running GENIE

GENIE cross section splines

- Calculating cross sections is time consuming
- Do it once and use for all of your event generations
- Use **G**ENIE **M**a**k**e **Spl**ine (**gmkspl**) to make splines:

```
gmkspl -p neutrino_code
       <-t target_codes, -f geometry>
       [-n nknots] [-e max_energy]
       [<--output-cross-sections | -o> xml_file]
       [--input-cross-sections xml_file]
       [--seed rnd_seed_num]
       [--event-generator-list list_name]
       [--message-thresholds xml_file ]
```

gmkspl example

- Generate cross section spline for charged current quasi-elastic muon neutrino scattering off carbon (up to $E_\nu = 2$ GeV with $\Delta E_\nu = 20$ MeV)

```
gmkspl -p 14                              #neutrino PDG
       -e 2                               #max ν energy
       -n 100                             #no. of knots
       -t 1000060120                      #target PDG
       --event-generator-list CCQE #channels list
```

- Target PDG code for nucleus will usually look like:

$$100ZZZNNN0$$

where ZZZ (NNN) is the number of protons (nucleons)

Event Generator List

$GENIE/config/EventGeneratorListAssembler.xml

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

- Event Generator List is defined by number of algorithms and the list of algorithms
- By default 13 algorithms are used

Event Generator

$GENIE/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>
```

- Each Event Generator is defined by modules related to each step of the process
- One can change steps and run w/o compiling

$GENIE/config/UserPhysicsOptions.xml

- There is a lot of model parameters
- They all can be changed in the UserPhysicsOptions.xml file
- Lets check parameters for QEL

```
$ grep --ignore-case qel $GENIE/config/UserPhysicsOptions.xml

<param type="double" name="QEL-Ma"> 0.990 </param>
<param type="double" name="QEL-Mv"> 0.840 </param>
<param type="double" name="QEL-FA0"> -1.2670 </param>
<param type="alg" name="XSecModel@genie::EventGenerator/QEL-CC">
                    genie::LwlynSmithQELCCPXSec/Default </param>
<param type="alg" name="XSecModel@genie::EventGenerator/QEL-NC">
                          genie::AhrensNCELPXSec/Default </param>
<param type="alg" name="XSecModel@genie::EventGenerator/QEL-EM">
                          genie::RosenbluthPXSec/Default </param>
<param type="alg" name="XSecModel@genie::EventGenerator/QEL-CC-CHARM">
                        genie::KovalenkoQELCharmPXSec/Default </param>
<param type="alg" name="XSecModel@genie::EventGenerator/QEL-CC-LAMBDA">
                             genie::PaisQELLambdaPXSec/Default </param>
```

Example xsec spline

- Generate cross section spline for charged current quasi-elastic muon neutrino scattering off carbon (up to $E_\nu = 2$ GeV with $\Delta E_\nu = 20$ MeV)

```
<spline name = "genie::LwlynSmithQELCCPXSec/Default/nu:14;
                tgt:1000060120;
                N:2112;
                proc:Weak[CC],QES;"
                nknots="100">
  <knot> <E>    0.01000 </E> <xsec>               0 </xsec> </knot>
  <knot> <E>   0.030175 </E> <xsec>               0 </xsec> </knot>
  <knot> <E>   0.050351 </E> <xsec>               0 </xsec> </knot>
  <knot> <E>   0.070526 </E> <xsec>               0 </xsec> </knot>
  <knot> <E>   0.090701 </E> <xsec>               0 </xsec> </knot>
  <knot> <E>    0.11088 </E> <xsec> 9.012505665e-15 </xsec> </knot>
  <knot> <E>    0.11434 </E> <xsec> 1.069790095e-13 </xsec> </knot>
  ...
  <knot> <E>     1.8806 </E> <xsec> 1.463880705e-10 </xsec> </knot>
  <knot> <E>     1.9394 </E> <xsec> 1.460427099e-10 </xsec> </knot>
  <knot> <E>          2 </E> <xsec> 1.456919360e-10 </xsec> </knot>
</spline>
```

Pre-calculated splines

Do not bully your CPU!

- You should know how to calculate splines (or at least that it is possible)
- However, whenever it is possible you should use pre-calculated splines:
  - https://www.hepforge.org/archive/genie/data/
  - http://www.genie-mc.org/

GENIE event generation

- Splines contain total cross section as a function of neutrino energy
- It is time to generate some events

GENIE **Ev**ent **Gen**eration

```
gevgen -p neutrino_pdg -t target_pdg -e energy -n nev
       [-h] [-r run#] [-f flux] [-w]
       [-seed random_number_seed]
       [--cross-section xml_file]
       [--event-generator-list list_name]
       [--message-thresholds xml_file]
       [--unphysical-event-mask mask]
       [--event-record-print-level level]
       [--mc-job-status-refresh-rate rate]
       [--cache-file root_file]
```

gevgen example

- Generate 1000 events for charged current quasi-elastic muon neutrino scattering off carbon for $E_\nu \in (1, 2)$

```
gevgen -p 14                                    #neutrino PDG
       -e 1,2                                   #ν energy range
       -n 1000                                  #no. of events
       -t 1000060120                            #target PDG
       --event-generator-list CCQE              #channels list
       --cross-section xsec_splines.xml         #splines file
```

GHepRecords

$$\nu_\mu + n \rightarrow \mu^- + p$$

```
|---------------------------------------------------------------------------------------------------------------------|
|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 |   1.000 |   1.000 |   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 |   0.008 |  -0.195 |   0.291 |   0.930 | **0.940 | M = 0.861
|   3 |           C11 |   2 | 1000060110 |   1 |  -1 |   7 |   7 |  -0.008 |   0.195 |  -0.291 |  10.249 |  10.243 |
|   4 |           mu- |   1 |         13 |   0 |  -1 |  -1 |  -1 |   0.057 |  -0.433 |   0.284 |   0.532 |   0.106 | P = (-0.110,0.831,-0.545)
|   5 |        proton |  14 |       2212 |   2 |  -1 |   6 |   6 |  -0.049 |   0.238 |   1.007 |   1.398 |   0.938 | FSI = 1
|   6 |        proton |   1 |       2212 |   5 |  -1 |  -1 |  -1 |  -0.048 |   0.231 |   0.974 |   1.373 |   0.938 |
|   7 |      HadrBlob |  15 | 2000000002 |   3 |  -1 |  -1 |  -1 |  -0.008 |   0.195 |  -0.291 |  10.249 | **0.000 | M = 10.243
|   8 |      NucBindE |   1 | 2000000101 |  -1 |  -1 |  -1 |  -1 |  -0.002 |   0.008 |   0.033 |   0.025 | **0.000 | M = -0.023
|---------------------------------------------------------------------------------------------------------------------|
|       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) =        5.6379e-38 cm^2  | dsig(Q2;E)/dQ2 =          9.4113e-39 cm^2/GeV^2 | Weight =                   1 |
|---------------------------------------------------------------------------------------------------------------------|
```

GHep status: $GENIE/src/GHEP/GHepStatus.h

Intranuke fate: $GENIE/src/HadronTransport/INukeHadroFates.h

```
------------------------------------------------------------------------------------------------
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    : no set
 |--> probe 4P     : (E =           1, Px =           0, Py =           0, Pz =           1)
 |--> target 4P    : (E =      11.179, Px =           0, Py =           0, Pz =           0)
 |--> nucleon 4P   : (E =    0.929627, Px =  0.00795907, Py =   -0.194854, Pz =    0.291161)
[-] [Process-Info]
 |--> Interaction : Weak[CC]
 |--> Scattering  : QES
[-] [Kinematics]
 |--> *Selected* Bjorken x = 0.777039
 |--> *Selected* Inelasticity y = 0.488052
 |--> *Selected* Momentum transfer Q2 (>0) = 0.484258
 |--> *Selected* Hadronic invariant mass W = 0.93827
[-] [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]
------------------------------------------------------------------------------------------------
```

Analyzing GENIE's output

```
gevdump -f filename [-n n1[,n2]]
```

gevdump prints out GENIE GHEP event records

Examples:

- `gevdump -f gntp.0.ghep.root` *#print all events*
- `gevdump -f gntp.0.ghep.root -n 10` *#print 10th event*
- `gevdump -f gntp.0.ghep.root -n 0,10` *#first 10 events*

Converting GHEP file


```
gntpc -i input_file_name
      -f format_of_output_file
      [-o output_file_name]
      [-v format_version_number]
      [-c copy_MC_job_metadata?]
      [-n number_of_events_to_convert]
```

gntpc converts GENIE GHEP file to other format

Examples:

- `gntpc -i gntp.0.ghep.root -f gst` *#convert GHEP file to gntp.0.gst.root*
- `gntpc -i gntp.0.ghep.root -f rootracker -n 100 -o small.gtrac.root` *#convert 100 events to rootracker format*

Spline to ROOT


```
gspl2root -f input_xml_file
          -p neutrino_pdg_code
          -t target_pdg_code
          [-e maximum_energy ]
          [-o output_root_file ] [-w]
```

gspl2root converts XML xsec splines to ROOT file

Examples:

- `gspl2root -p 14 -t 1000060120 -f xsec_splines.xml -o xsec.root` *#create ROOT file with cross section*
- `gspl2root -p 14 -t 1000060120 -f xsec_splines.xml -o xsec.root -w` *#and save plots to ps file*

Interactive GENIE

```
$ genie                                                                    #run GENIE interactive mode
genie [1] using namespace genie                                            #just for convenience
genie [2] TFile *myFile = new TFile ("gntp.0.ghep.root")                   #open ROOT file
genie [3] TTree *myTree = myFile->Get("gtree")                             #get yout tree
genie [4] myTree->GetEntries()                                             #check number of entries in the tree
(const Long64_t)100
genie [5] NtpMCEventRecord* myEventRecord = new NtpMCEventRecord()         #create you event record pointer
genie [6] myTree->SetBranchAddress("gmcrec", &myEventRecord)               #connect it with gmcrec
genie [7] myTree->GetEntry(0)                                              #get first entry
genie [8] myEventRecord->PrintToStream(cout)                               #print the entry
genie [9] myEventRecord->event->XSec()                                     #print cross section
(const double)1.45242998226155670e-10
genie [10] myEventRecord->event->Summary()->Kine()->Q2(true)               #print Q2
(const double)2.10357281770150784e-01
```

Interactive GENIE

```
$ genie                                                    #run GENIE interactive mode
genie [1] using namespace genie                            #just for convenience
genie [2] TFile *myFile = new TFile ("gntp.0.ghep.root")   #open ROOT file
genie [3] TTree *myTree = myFile->Get("gtree")             #get yout tree
genie [4] myTree->GetEntries()                             #check number of entries in the tree
(const Long64_t)100
genie [5] NtpMCEventRecord* myEventRecord = new NtpMCEventRecord()   #create you event record pointer
genie [6] myTree->SetBranchAddress("gmcrec", &myEventRecord)         #connect it with gmcrec
genie [7] myTree->GetEntry(0)                              #get first entry
genie [8] myEventRecord->PrintToStream(cout)               #print the entry
genie [9] myEventRecord->event->XSec()                     #print cross section
(const double)1.45242998226155670e-10
genie [10] myEventRecord->event->Summary()->Kine()->Q2(true)   #print Q2
(const double)2.10357281770150784e-01
```

Have you noticed?

Units convention

- Theorists units: $\hbar = c = 1$
- but also $GeV = 1$
- Let see how "long" is centimeter:

$$\begin{aligned} \hbar &\approx 197 \text{ MeV} \cdot \text{fm/c} = 0.197 \text{ GeV} \cdot \text{fm/c} \\ 1 &\approx 0.197 \text{ fm} = 0.197 \cdot 10^{-13} \text{ cm} \\ 1 \text{ cm} &\approx 5.07 \cdot 10^{13} \end{aligned}$$

- Thus the cross section in cm^2:

```
genie [11] myEventRecord->event->XSec() / (5.07e+13 * 5.07e+13)
(const double)5.65040121635002092e-38
```

GENIE loop example


```
void genieLoop (const char *ghepFile) // main program
{
  TFile *myFile = new TFile (ghepFile); // open GHEP file
  TTree *myTree = myFile->Get ("gtree"); // get tree
  // initialize event record and connect it to gmcrec
  NtpMCEventRecord* myEventRecord = new NtpMCEventRecord();
  myTree->SetBranchAddress ("gmcrec", &myEventRecord);

  const int nBins = 50; // number of bins in the histogram
  const double minValue = 0.0; // mininum value
  const double maxValue = 2.0; // maximum value

  // create a ROOT histogram
  TH1D *myHistogram = new TH1D ("myHistogram", "myHistogramTitle",
                                nBins, minValue, maxValue);
...
```

GENIE loop example

```
  // number of entries in the tree
  const unsigned int nEvents =  myTree->GetEntries();

  for (unsigned int i = 0; i < nEvents; i++) // events loop
  {
    myTree->GetEntry (i); // get i-th event

    // pointer to the event
    const EventRecord *myEvent = myEventRecord->event;

    // check if the event matches you signal definition
    if (!isSignal (myEvent)) continue;
    // check if the event passed your cuts
    if (!passedCuts (myEvent)) continue;

    // get your kinematics variable
    const double myVariable = getVariable (myEvent);

    myHistogram->Fill (myVariable); // fill your histogram
  }

  // note: this is not differential cross section
  // do you know what is missing?
  myHistogram->Draw();  //draw your histogram
}
```

GENIE loop example


```
// returns 100 * nPiP + 10 * nPiM + nPi0
// e.g. 123 means 1pi+, 2pi-, 3pi0 in the final state
int pionCode (const EventRecord *eventRecord)
{
  return 100 * eventRecord->NEntries(kPdgPiP, kIStStableFinalState) +
          10 * eventRecord->NEntries(kPdgPiM, kIStStableFinalState) +
           1 * eventRecord->NEntries(kPdgPi0, kIStStableFinalState);
}

// returns true if event matches the signal definition
bool isSignal (const EventRecord *eventRecord)
{
  // examples:
  //return (pionCode (eventRecord) == 100); // 1pi+ in the final state
  //return (pionCode (eventRecord) ==  10); // 1pi- in the final state
  //return (pionCode (eventRecord) ==   1); // 1pi0 in the final state
  //return eventRecord->Summary()->ProcInfo().IsQuasiElastic(); // true QEL
  //return eventRecord->Summary()->ProcInfo().IsWeakCC(); // charged current

  return true;
}
```

GENIE loop example

```
// returns true if event passed all cuts
bool passedCuts (const EventRecord *eventRecord)
{
  // examples:
  // W < 1.6 GeV
  //return eventRecord->Summary()->Kine().W (true) < 1.6;
  // lepton energy > 100 MeV
  //return eventRecord->FinalStatePrimaryLepton()->P4()->Energy() > 0.1;

  return true;
}

// returns chosen kinematics variable
double getVariable (const EventRecord *eventRecord)
{
  // examples:
  // Q2
  //return eventRecord->Summary()->Kine().Q2 (true);
  // invariant mass
  //return eventRecord->Summary()->Kine().W (true);
  // lepton energy
  //return eventRecord->FinalStatePrimaryLepton()->P4()->Energy();

  return 0.0;
}
```

GENIE loop example

```
// extract total cross section
double getTotalCrossSection (const char* ghepFile)
{
  // using: N_qel = sigma_qel / sigma_total * N_total

  ...

  double xSecQel = 0.0; // sum of all cc qel events xsec
  int nQelEvents = 0;   // number of cc qel events

  for (unsigned int i = 0; i < nEvents; i++) // events loop
  {

    ...

    // require CC QEL
    if (!myEvent->Summary()->ProcInfo().IsQuasiElastic()) continue;
    if (!myEvent->Summary()->ProcInfo().IsWeakCC()) continue;

    // add event cross section and increase the counter
    xSecQel += myEvent->XSec() / (units::cm2);
    nQelEvents++;
  }

  // sigma_qel = xSecQel / nQelEvents
  // sigma_total = sigma_qel * N_total / N_qel

  return xSecQel / nQelEvents / nQelEvents * nEvents;
}
```

Useful links

- NuWro User Guide
- How to use NuWro
- GENIE Manual
- GENIE Developer Manual