CPPsim User Guide

Table of Contents

Nicholas Miller
August 24. 2017

Installation Guide

*Copied and updated from “StartGuide” in /home/Geant4.10.03/Desktop/User_Guides, made by Bobby Johnston

** Note that whenever the symbol “$” appears that means what follows are lines to be put in the terminal. “#” symbol means whatever follows it is a comment and is not to be put in the terminal

Steps to run CPPsim are:
- (0) Install proper OS, such as linux centos
- (1) Install all programs for Gluex (xerces-c, jana, root, etc.) via wiki
- (2) Install Geant4
- (3) Set up environment
- (4) Install CPPsim
- (5) Run CPPsim
- (6) Alter data files
- (7) Run MVA

---

(0) This step is straightforward and will not be discussed. Choose an OS that supports the necessary software (Geant4, root, xerces-c, etc.)

---

(1) To set up all programs, use scripts written by Mark Ito

Main documentation page:
https://halldweb.jlab.org/docs/gluex_build_web/gluex_build_web.html
Simply follow the instructions outlined here:
https://halldweb.jlab.org/docs/gluex_build_web/node34.html

---

(2) Installation of Geant4

There are some preliminary programs that must be installed, such as cmake and qt (for visualization, optional, will not be discussed)

Here is a helpful site:
http://geant4.slac.stanford.edu/MIT2016/HandsOn1/

A second guide is here:
http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/InstallationGuide/html/ch02s03.html

First change directory into wherever you want Geant4 to be installed

```
        $ wget http://geant4.web.cern.ch/geant4/support/source/geant4.10.02.p01.tar.gz
        $ tar xf geant4.*.tar.gz
        $ mkdir build
        $ cd build
        $ cmake -DCMAKE_INSTALL_PREFIX=<tutorial> -
DGEANT4_BUILD_MULTITHREADED=ON -DGEANT4_INSTALL_DATA=ON -
DGEANT4_USE_QT=ON -DGEANT4_USE_OPENGL_X11=ON -
DGEANT4_USE_GDML=ON  -DXERCESC_ROOT_DIR=<set to root directory of
installation (directory contating the include and lib subdirectoreis of Xerces-C++>
../geant4.10.02.p01
```

Examples of this:
$-------------for installing on MENP desktop 11/29/16---

```
cmake -DCMAKE_INSTALL_PREFIX=/home/Novem/Geant4_2 -
DGEANT4_BUILD_MULTITHREADED=ON -DGEANT4_INSTALL_DATA=ON -
DGEANT4_USE_QT=ON -DCMAKE_PREFIX_PATH=/home/Novem/Qt/5.7/gcc_64
-DGEANT4_USE_OPENGL_X11=ON -DGEANT4_USE_GDML=ON  -
DXERCESC_ROOT_DIR=/home/Novem/gluex_install/gluex_top/xerces-c/xerces-c-
3.1.4 ../geant4.10.02.p01
```

$--

$------------for installing on Geant4.10.03 desktop 6/28/17-------------------------------------

```
cmake -DCMAKE_INSTALL_PREFIX=/home/Geant4.10.03/Geant4_3 -
DGEANT4_BUILD_MULTITHREADED=ON -DGEANT4_INSTALL_DATA=ON -
```

```
DGEANT4_USE_QT=ON -
DCMAKE_PREFIX_PATH=/home/Geant4.10.03/Qt/5.7/gcc_64 -
DGEANT4_USE_OPENGL_X11=ON -DGEANT4_USE_GDML=ON  -
DXERCESC_ROOT_DIR=/home/Geant4.10.03/gluex_install/gluex_top/xerces-
c/xerces-c-3.1.4 ../geant4.10.03.p01
```

-

When done properly, last lines should read:

```
-- Configuring done
-- Generating done
-- Build files have been written to: <some-directory>/build
```

Finally, do as below:

```
        $ make -j N  #(N is for number of cores)
        $ make install
```

(In the case below, Geant4 was installed in the Novem user in a folder called Geat4_2. Update the source and export lines accordingly)

```
        $ source /home/Novem/Geant4_2/bin/geant4.sh
        $ export PATH=$PATH:/home/Novem/gluex_install/gluex_top/xerces-
c/xerces-c-3.1.4/bin/
```

You must include the below line to get it to work until you set up your environment properly

```
        $ source <tutorial>/bin/geant4.[c]sh
```

---

(3) To set up the environment, (using bash) create file named .bashrc and include the following:

```
        $ cd ~
        $ gedit .bashrc
```

Below shows my current .bashrc file

```
//
#This sets up the environment variables so that CPPsim can be run properly

#Set ROOSYS variable
export ROOTSYS=/home/Geant4.10.03/gluex_install/gluex_top/root/buildroot

# Setup sim-recon (also sets up root, xerces, HDDS, etc...)
. /home/Geant4.10.03/gluex_install/gluex_top/sim-recon/prod/Linux_CentOS7-x86_64-
gcc4.8.5/setenv.sh

#Below is setup.sh for Gluex_Install
#Begin
export GLUEX_TOP=/home/Geant4.10.03/gluex_install/gluex_top
export BUILD_SCRIPTS=$GLUEX_TOP/build_scripts
source $BUILD_SCRIPTS/gluex_env_version.sh
/home/Geant4.10.03/gluex_install/gluex_top/version.xml
#End

# Setup GEANT4
export G4=/home/Geant4.10.03/Geant4_3
. $G4/bin/geant4.sh
. $G4/share/Geant4-10.3.1/geant4make/geant4make.sh

# Add cmake bin to path variable
export PATH=$PATH:/opt/cmake/bin/

# Change color of terminal
export PS1="\e[0;31m[\u@\h \W]\$ \e[m "

#Add colors to directories when using ls
#eval `dircolors ~/.dir_colors`
alias ls="ls --color=auto"

#Needed for using mcsmear when working on CPPsim
export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/Gluex4.10.03/gluex_install/glu
ex_top/rcdb/rcdb_0.00/cpp/lib

#Add ROOT to source
source /home/Geant4.10.03/gluex_install/gluex_top/root/buildroot/bin/thisroot.sh
```

#Add xerces to path variable
export PATH=$PATH:/home/Novem/gluex_install/gluex_top/xerces-c/xerces-c-3.1.4/bin/

#Set HDDS_HOME
source /home/Geant4.10.03/gluex_install/gluex_top/sim-recon/sim-recon-2.14.0/Linux_CentOS7-x86_64-gcc4.8.5/setenv.sh

---

(4) To install CPPsim, do the following

Go to some directory where you want CPPsim to be installed. The following line downloads CPPsim from the repository at subversion

*Note that you need to type out the web address after "svn co" into the terminal like this: "svn co https://web-address/"

```
        $ svn co
https://halldsvn.jlab.org/repos/trunk/Experiments/PionPolarizability/src/CPPsim


        $ mkdir CPPsim-build #if making a new geometry build, call it something
like 12-x-10-y-geom
        $ cd CPPsim-build
        $ ../CPPsim/run_cmake
        $ make -j8
```

#The next might not be necessary, if you run make and it gives the error:

```
/home/bobby/Geant4_2/CPP_Simulations/CPPsim/src/root_fudge.cc:27:43: error: no
'Element& TMatrixTBase<Element>::NaNValue()' member function declared in class
'TMatrixTBase<Element>'
 Element & TMatrixTBase<Element>::NaNValue()
```

#then do the following:

```
        $ cd CPPsim/src
        $ mv root_fudge.cc root_fudge.cc.unused
```

---

(5) To run CPPsim, do the following:

*Note that this is a test for CPPsim. Usually the simulation will be run using .dat files

First, you have to build gen_2mu. It does not automatically get built when installing CPPsim. This command is to build it is:

```
$ cd $HALLD_HOME/src/programs/Simulation/gen_2mu
$ scons -u install

$ cd ~/CPPSIM/CPPsim-build
$ mkdir tmp
```

# this is an example of making a working directory. Can also name based off of the types of simulation you are running i.e. theta limits and energy limits

```
$ cd tmp
$ cp ../../CPPsim/control.in .
```

*Note that the following is not necessary if you comment out the GEOMFILE line in control.in. If this line is not commented out, then it will assume there is a GDML file in the directory. For the purposes of this test that is fine, but when running the actual simulation it is more convenient to comment it out, since it will find the GDML file itself and the user won't have to generate a new one each time.

After this you have to make the GDML file from a HDDS geometry and put it in tmp

```
$ cd $HDDS_HOME/$BMS_OSNAME/src
$ root -l hddsroot.C
root [1] gGeoManager->Export("main_HDDS.gdml")
```

Go back to tmp and copy the GDML file into the build directory

```
$ cd ~/CPPSIM/CPPsim-build/tmp
$ cp $HDDS_HOME/$BMS_OSNAME/src/main_HDDS.gdml .
```

The control.in file is used to control the simulation. You can change particle type, number of triggers, etc here.
We can also generate events using the event generator gen_2mu, see below:

```
        $ gen_2mu -N 1000 #This generates muons
        $ gen_2mu -pions -N 1000
#This generates "pions" - not physically accurate
        $ ../CPPsim
```

The above will run CPPsim. When done it will output a file CPPsim.hddm.
To run CPP sim in interactive mode, use $ ../CPPsim -i
If there is a vis.mac macro in the directory, a viewer will open up (using the flag “-i” will not work if there is no vis.mac macro in the directory)

---

(6) We must then smear out the output file and run hd_root on it

```
        $ mcsmear CPPsim.hddm
# Runs mcsmear on CPPsim.hddm and outputs the file CPPsim_smeared.hddm
```

*Note that you need to type out the web address after “svn co” into the terminal like this: “svn co https://web-address/”

```
# only once
        $ cd ../..
        $ svn co
https://halldsvn.jlab.org/repos/trunk/Experiments/PionPolarizability/src/plugins/cppmva



#This will set up CPPMVA which is needed for MVA
        $ cd cppmva/
        $ scons install
# ONLY NEED TO DO ONCE

        $ cd ../CPPsim-build/tmp
        $ hd_root -PPLUGINS=cppmva CPPsim_smeared.hddm

#This will output hd_root.root
#We can view details at this point by putting it through root
        $ root -l hd_root.root
        root[1]$ cppmva->Print()
        root[2]$ .ls
```

```
root[3]$ cppmva->Draw("Nfcal_hits")
root[4]$ cppmva->Draw("Nfmwpc1")
```

---

(7) If all is proper now, we can run a MVA on the root file.

#First copy in trainMVA.C file to working directory

```
$ cd cppmva
$ cp trainMVA.C ../../<working directory>
$ gedit trainMVA.C
```

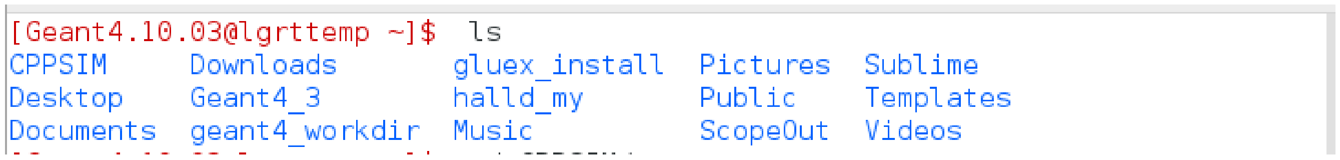
####edit trainMVA.C so that it references the proper files#################

```
$ root -l -q -b trainMVA.c
$ ln -s CPPMVA_out.root TMVA.root
$ root -l hd_root.root
root[1]$ TMVA::TMVAGui()
```

Computer Setup (Geant4.10.03 user)

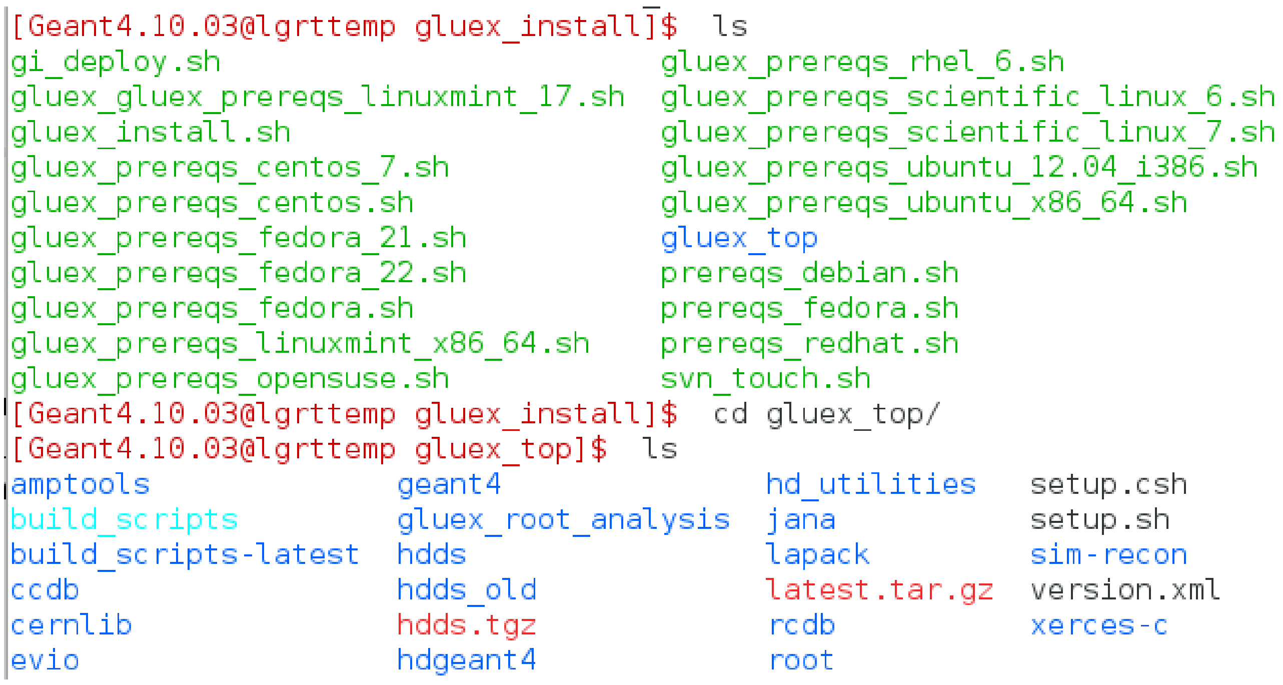
Note: This is assuming that you are on the Geant4.10.03 user. This document was also made in August 2017 so it may be outdated.

Here is what the home directory looks like:



```
[Geant4.10.03@lgrttemp ~]$  ls
CPPSIM     Downloads       gluex_install  Pictures   Sublime
Desktop    Geant4_3        halld_my       Public     Templates
Documents  geant4_workdir  Music          ScopeOut   Videos
```

In ~/gluex_install lies all the packages necessary to run the CPPsim simulation. They are in found ~/gluex_install/gluex_top. Both of these are in the picture below. These include Xerces-C, CERNLIB, ROOT, EVIO, CCDB, RCDB, JANA, HDDS, and sim-recon.



```
[Geant4.10.03@lgrttemp gluex_install]$  ls
gi_deploy.sh                         gluex_prereqs_rhel_6.sh
gluex_gluex_prereqs_linuxmint_17.sh  gluex_prereqs_scientific_linux_6.sh
gluex_install.sh                     gluex_prereqs_scientific_linux_7.sh
gluex_prereqs_centos_7.sh            gluex_prereqs_ubuntu_12.04_i386.sh
gluex_prereqs_centos.sh              gluex_prereqs_ubuntu_x86_64.sh
gluex_prereqs_fedora_21.sh           gluex_top
gluex_prereqs_fedora_22.sh           prereqs_debian.sh
gluex_prereqs_fedora.sh              prereqs_fedora.sh
gluex_prereqs_linuxmint_x86_64.sh    prereqs_redhat.sh
gluex_prereqs_opensuse.sh            svn_touch.sh
[Geant4.10.03@lgrttemp gluex_install]$  cd gluex_top/
[Geant4.10.03@lgrttemp gluex_top]$  ls
amptools               geant4               hd_utilities    setup.csh
build_scripts          gluex_root_analysis  jana            setup.sh
build_scripts-latest   hdds                 lapack          sim-recon
ccdb                   hdds_old             latest.tar.gz   version.xml
cernlib                hdds.tgz             rcdb            xerces-c
evio                   hdgeant4             root
```

In the hdds directory (~/gluex_install/gluex_top/hdds) lies the files for the geometry. One can also get to this directory, in any directory, by doing:

$ cd $HDDS_HOME

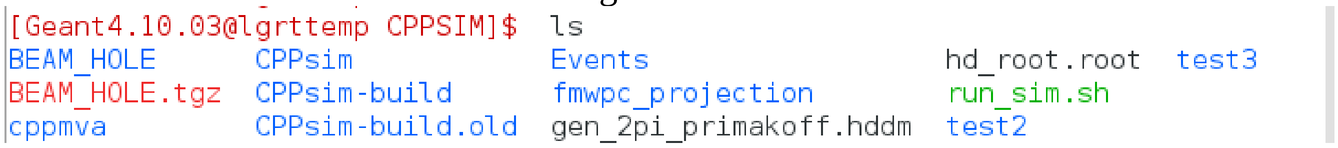
The “$” sign before HDDS_HOME means that it is a variable, with HDDS_HOME = /home/Geant4.10.03/gluex_install/gluex_top/hdds

IMPORTANT: $HDDS_HOME MUST POINT TO THE hdds DIRECTORY, FOUND IN ~/gluex_install/gluex_top/hdds. IF IT DOES NOT, THE PROGRAM CANNOT FIND THE GEOMETRY FILES AND WILL NOT RUN THE SIMULATION.

In $HDDS_HOME:

- cpp_HDDS.xml – Where everything is located with respect to the beamline
- ForwardMWPC_HDDS.xml – Where everything is defined within the mother volume

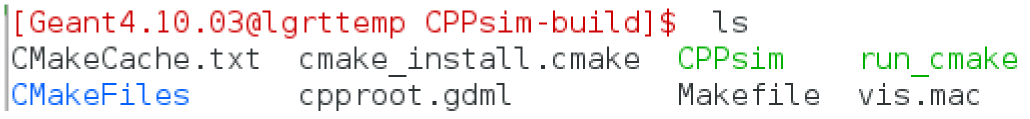
In ~/CPPSIM one will find the following sub-directories:



```
[Geant4.10.03@lgrttemp CPPSIM]$  ls
BEAM_HOLE      CPPsim            Events                 hd_root.root  test3
BEAM_HOLE.tgz  CPPsim-build      fmwpc_projection       run_sim.sh
cppmva         CPPsim-build.old  gen_2pi_primakoff.hddm  test2
```

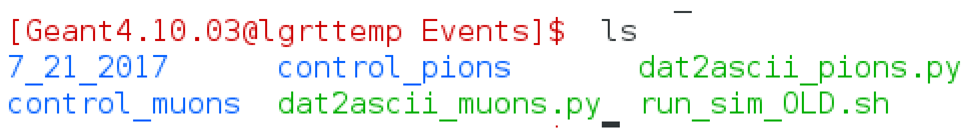
In ~/CPPSIM/CPPsim are the source files used to run the simulation.

In ~/CPPSIM/CPPsim-build, shown below, the CPPsim executable (in green) is located. As well as the run_cmake executable to build the simulation.



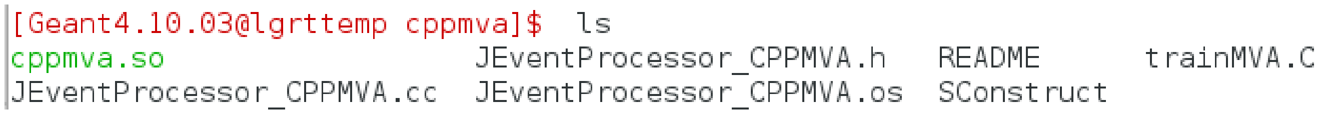
```
[Geant4.10.03@lgrttemp CPPsim-build]$  ls
CMakeCache.txt  cmake_install.cmake  CPPsim    run_cmake
CMakeFiles      cpproot.gdml         Makefile  vis.mac
```

In ~/CPPSIM/Events, shown below, there are two folders called control_muons and control_pions. In each of these folders, there are control.in files for their respective particle. These are used when a user runs the simulation.



```
[Geant4.10.03@lgrttemp Events]$  ls
7_21_2017      control_pions      dat2ascii_pions.py
control_muons  dat2ascii_muons.py  run_sim_OLD.sh
```

The folder /7_21_2017 is an example of a folder that holds event files for the simulation. In this folder there are two .dat files, each for muons and pions that the simulation used to generate root files used for analysis (check section Running CPPsim)



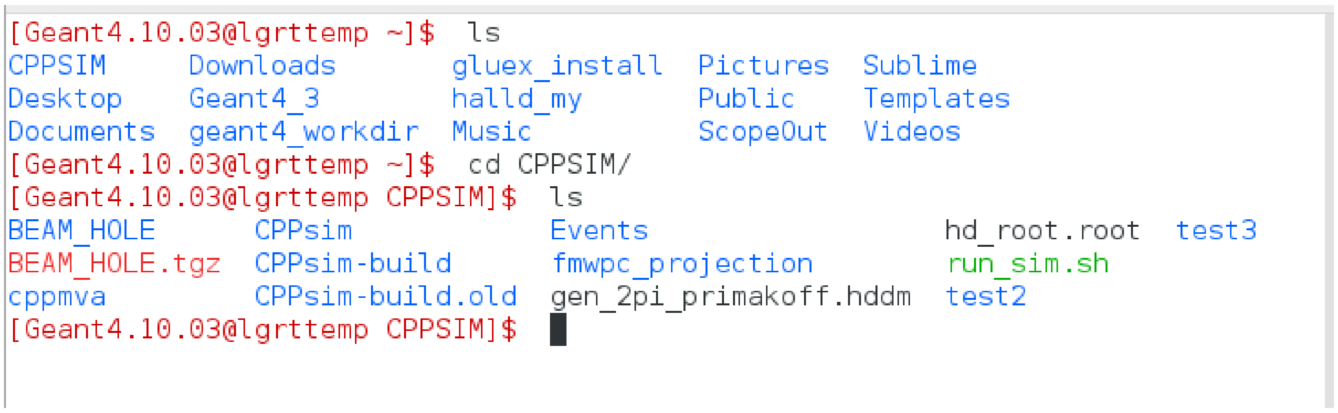
```
[Geant4.10.03@lgrttemp cppmva]$  ls
cppmva.so                    JEventProcessor_CPPMVA.h   README      trainMVA.C
JEventProcessor_CPPMVA.cc    JEventProcessor_CPPMVA.os  SConstruct
```

In ~/CPPSIM/cppmva (pictured above) is the multivariate analysis plugin. In this folder is trainMVA.C, the file which ROOT uses to perform the MVA.

In ~/CPPSIM/fmwpc_projection is the projection plugin to project the position of the particle and track it.

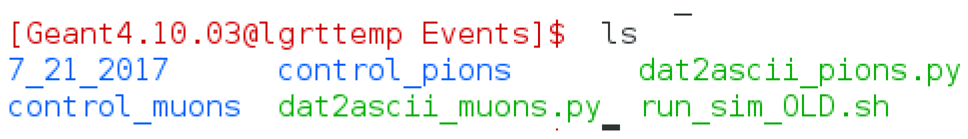
Back in the home directory, ~/Geant4_3 is where Geant4 was installed. It contains the Geant source files, as well as the examples Geant4 comes with. You can look online to find how to run these examples.

Running CPPsim



```
[Geant4.10.03@lgrttemp ~]$  ls
CPPSIM     Downloads       gluex_install  Pictures  Sublime
Desktop    Geant4_3        halld_my       Public    Templates
Documents  geant4_workdir  Music          ScopeOut  Videos
[Geant4.10.03@lgrttemp ~]$  cd CPPSIM/
[Geant4.10.03@lgrttemp CPPSIM]$  ls
BEAM_HOLE      CPPsim           Events                   hd_root.root  test3
BEAM_HOLE.tgz  CPPsim-build     fmwpc_projection         run_sim.sh
cppmva         CPPsim-build.old gen_2pi_primakoff.hddm   test2
[Geant4.10.03@lgrttemp CPPSIM]$
```

In the CPPsim directory (~/CPPsim as shown above), one can find all necessary components to run the simulation.



```
[Geant4.10.03@lgrttemp Events]$  ls
7_21_2017      control_pions        dat2ascii_pions.py
control_muons  dat2ascii_muons.py   run_sim_OLD.sh
```

One should put the .dat files received for a run in /Events (as shown above). An example of this is the folder /7_21_2017 where the muon and pion .dat files are located from a run made on July 21, 2017. In /Events, there is also the control_pions file and the control_muons file where the control.in files are located for their respective particle. The scripts to convert .dat files to .ascii files are also found here. (dat2ascii_muons.py and dat2ascii_pions.py)

Looking at a “control.in” file located in ~/CPPSIM/Events/control_pions (or control_muons), the input file should be “gen_2mu.hddm” for muons and “gen_2pi.hddm” for pions in order to match with the run_sim.sh script. If the input file is different, either the run script or control.in files must be changed. Make sure the output files in control.in also match the ones in run_sim.sh – as of right now they are both “CPPsim_pions.hddm” and “CPPsim_muons.hddm”.

The “BGGATE” and “BGRATE” lines in control.in should be un-commented (take out the “c”) if the user would like to add noise. The dead radius and dead width are defined at the bottom as FMWPC_DEAD_RADIUS and FMWPC_DEAD_WIDTH. Also, GEOMOPT and GEOMFILE linse should be commented out unless the geometry file is in the same directory that the simulation is being run in.

The “run_sim.sh” file found in ~/CPPSIM needs to be updated for the conditions of each run. Changing of file names or directory locations may be needed (they are commented in the script).This file assumes the .dat files are located in some directory in Events (~/CPPSIM/Events/some-directory). This script goes through each of the .dat files and converts them to ascii and then to hddm. It runs JANA (before CPPsim) and CPPsim. It smears the files, runs the cppmva plugin and runs trainMVA.C. As of right now the script assumes you’re in some directory in CPPSIM (~/CPPSIM/some-directory).

Now to actually run the simulation:
Assuming you’re in some-directory within CPPSIM (~/CPPSIM/some-directory)

```
$ ../run_scripts.sh
```

After everything runs, the important files will be the hd_root_muons.root and hd_root_pions.root. To open them in ROOT do:

```
$ root -l hd_root_muons.root
```

To look at the trees and entries using MVA do (in ROOT):

```
root [1] cppmva→Print()
```

To draw the hits for one of the branches do:

```
root [2] cppmva→Draw(“Nfmwpc”)                #Or any of the branch names
```

To get the GUI for the multivariate analysis:

```
root [3] TMVA::TMVAGui()
```

The important plots will be the signal efficiency plots (5b).

Possible Future Issues

This section is dedicated to issues I ran into when installing and running the simulation which were not already addressed above.

---

(1) ROOT is too updated

For whatever reason, the newer versions of ROOT are not compatible with cppmva (they took out the C++ file which is needed to run it). If ROOT needs to be installed, install ROOT version 6.06.08.

The version of ROOT that comes with the GlueX packages may be too new. To check which version of ROOT you have, you can type in the directory:

```
$ which root
```

If ROOT is re-installed, everything else needs to be recompiled (all the GlueX packages).

Make sure the ROOTSYS variable is set to the build directory where /bin is.

---

(2) hdds is on the wrong branch

One should make sure that they're on the master branch in hdds. This will assure that the geometry is updated (at least to what David/Ilya are doing). To do this do:

```
$ cd $HDDS_HOME
$ git status
```

If it says "On branch master" then everything is fine. If not do:

```
$ git checkout master
$ git pull
$ scons install
```

One can do "git branch" to see available branches. Also after this, one must go to ~/CPPSIM/cppmva and do "scons install" to update the MVA. Also go to ~/CPPSIM/fmwpc_tracking and do “scons install” to update it as well.

---

(3) No ROOT support for GDML

This error will pop up when ROOT was installed without support for GDML. To fix this you have to uninstall ROOT and reinstall it with the flag "-Dgdml=ON" and "-Dminuit2=ON" when you run cmake. It will look like this in the terminal:

```
$ cmake -Dgdml=ON -Dminuit2=ON ../root-6.08.06
```

If you do this, look at **(1)**. You need to make sure everything else is recompiled, as well as properly set the ROOTSYS variable.