Monte Carlo @ Gluex: A Tutorial

Thomas Britton

GlueX Tutorial 2018

Overview

- **Goal: produce a sample of $\omega \rightarrow \pi^+\pi^-\pi^0$ MC**

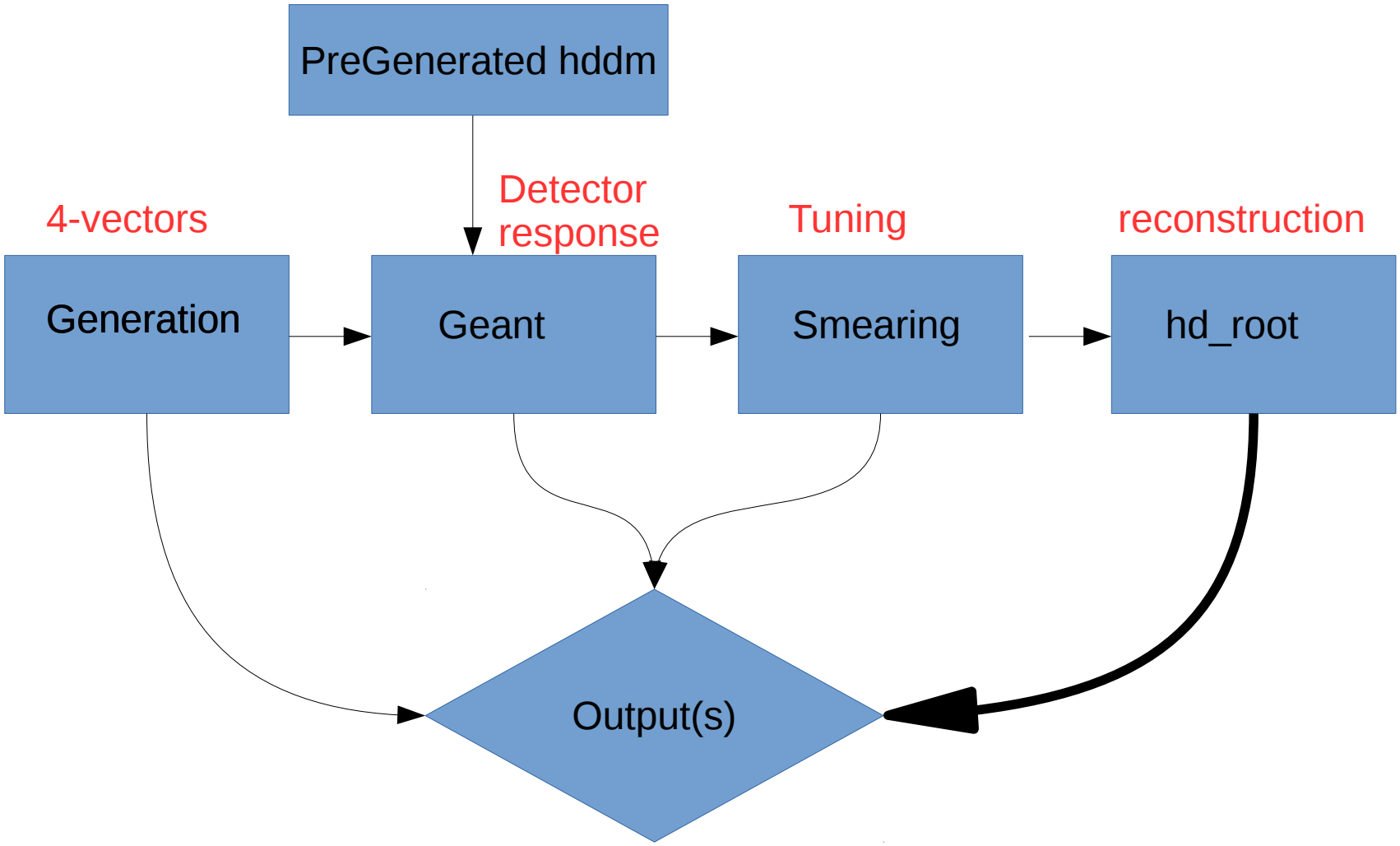
- Tools:
  - **Sim-recon** (previously described in 1a)
  - **MCwrapper** (hd_utilities v1.14)

  - Optional: OSG
    - certificate

Philosophy

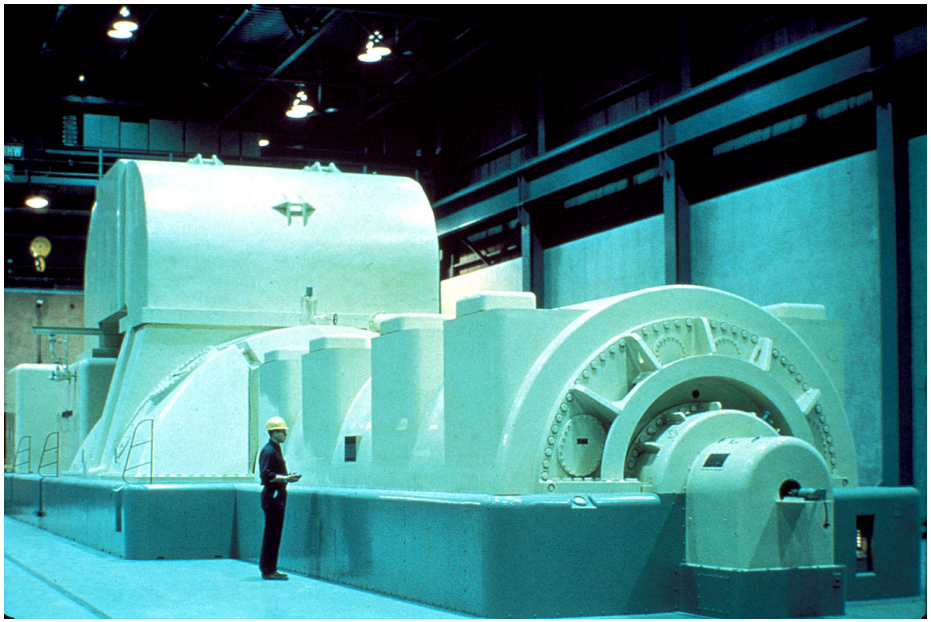
- MCwrapper seeks to be a one-stop-shop for simulation for GlueX/Hall-D. It needs to be able to:
  - Complete the production chain; from generation through (hd_root) plugins
  - Run both geant3 and geant4 easily
  - Provide basic standards of simulation
  - Be customizable for individual studies
  - Utilize various batch systems
  - Provide support for new users

MC Production Line



Generation

- Currently MCwrapper supports the following generators (and more)
  - Pre-made .hddm files
  - Genr8
  - Bggen
  - GenEtaRegge
  - gen_2pi_amp
  - gen_pi0
  - **gen_omega_3pi**
  - gen_2k
  - gen_2pi_primakoff
  - ...



Generation cont.

- Utilizes RCDB calls and the supplied run number to fill out various parameters as desired such as:
  - Radiator thickness
  - Coherent Peak Position
  - Electron beam energy
  - etc

- This takes the leg work out of grabbing the right values for a given run number
- Note: the generator must be able to utilize these values in order for the wrapper to control them

Geant

- Supports both hdgeant and hdgeant 4
- Uses RCDB to fill out many of the settings (cards) for you
- Adds in or removes backgrounds
  - Random trigger
  - Beam Photon background
  - Custom file

MCsmear

- Takes the output of geant and smears it. Tries to make the pure MC look more like data that comes from the actual detector.
- MCwrapper will configure mcsmear to fold in the random trigger backgrounds if desired

hd_root

- Runs your favorite plugin(s) over the smeared hddm file
- This should leave you with root files and/or trees ready for merging and further analysis

ROOT
Data Analysis Framework

Obtaining MCwrapper

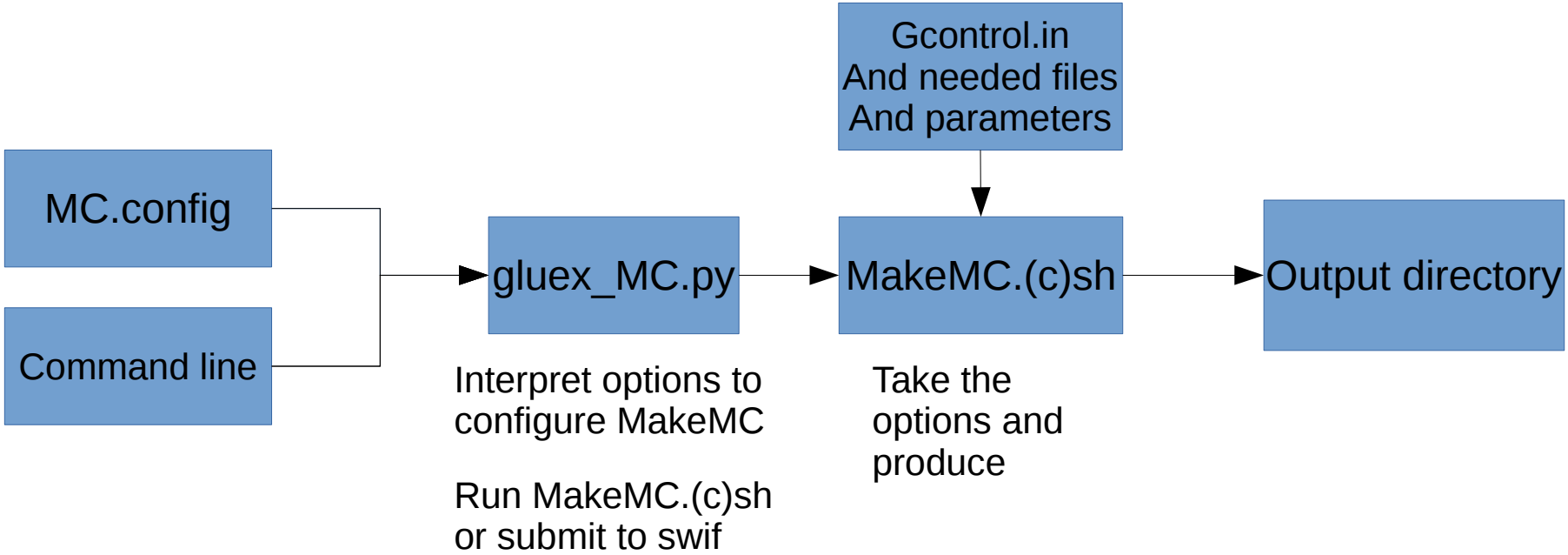
Assuming you completed session 1a...

it is already in your path!

Note: MCwrapper contains no compiled code. Thus it is ready to be used

MCwrapper Flow

It is advised to run outside of MCWRAPPER_CENTRAL



Command Line

- Usage: **gluex_MC.py config_file [Run_Number] [num_events] [all other options]**
- where **[all other options]** are:

variation=%s where %s is a valid jana_calib_context variation string (default is "mc")
per_file=%i where %i is the number of events you want per file/job (default is 10000)
numthreads=%i sets the number of threads to use to %i. Note: will overwrite NCORES set in MC.config
base_file_number=%i where %i is the starting number of the files/jobs (default is 0)
generate=[0/1] where 0 means that the generation step and any subsequent step will not run (default is 1)
geant=[0/1] where 0 means that the geant step and any subsequent step will not run (default is 1)
mcsmear=[0/1] where 0 means that the mcsmear step and any subsequent step will not run (default is 1)
recon=[0/1] where 0 means that the reconstruction step will not run (default is 1)
cleangenerate=[0/1] where 0 means that the generation step will not be cleaned up after use (default is 1)
cleangeant=[0/1] where 0 means that the geant step will not be cleaned up after use (default is 1)
cleanmcsmear=[0/1] where 0 means that the mcsmear step will not be cleaned up after use (default is 1)
cleanrecon=[0/1] where 0 means that the reconstruction step will not run (default is 1)
batch=[0/1/2] where 1/2 means that it will submit to the specified batch system (default is 0)
logdir=[path] will direct the .out and .err files to the specified path in qsub

Example:

gluex_MC.py MC.config 11366 10000 cleanmcsmear=0 batch=1

Config File

All options in one place!

Use full paths!

```
#THESE TWO ARE OPTIONAL IF THE STANDARD RUNNING DOESN'T SUIT YOUR NEEDS
#CUSTOM_MAKEMC=use-this-script-instead
#CUSTOM_GCONTROL=use-this-Gcontrol-instead
#=====================================================================

#VARIATION=mc calibtime=timegoeshere #set your jana calib context here with or without calibtime Default is variation=mc
#RECON_CALIBTIME=timegoeshere

#RUNNING_DIRECTORY=/run/in/this/directory #where the code should run.  This is defaulted to ./.  Use only when NEEDED

#ccdbSQLITEPATH=/your/sqlite/path/ccdb.sqlite  #if you use SQLITE and it is not part of the environment file that gets sourced
#rcdbSQLITEPATH=/your/sqlite/path/rcdb.sqlite  #if you use SQLITE and it is not part of the environment file that gets sourced


#TAG=my-custom-prefix-tag

DATA_OUTPUT_BASE_DIR=OUTPUT-LOCATION #your desired output location

NCORES=4      # Number of CPU threads to use or nodes:node-id:ppn or nodes:ppn depending on your system

GENERATOR=generator-to-use #or you may specifile file:/.../file-to-use.hddm
GENERATOR_CONFIG=config file for generator

#common parameters for generators
#eBEAM_ENERGY=12 #either use rcdb or do not set to pull the value for the chosen run number from the rcdb
#RADIATOR_THICKNESS=50.e-06#either use rcdb or do not set to pull the value for the chosen run number from the rcdb
#COHERENT_PEAK=9 #either use rcdb or do not set to pull the value for the chosen run number from the rcdb
#GEN_MIN_ENERGY=4
#GEN_MAX_ENERGY=12

GEANT_VERSION=3
BKG=None #[None, BeamPhotons, TagOnly, custom e.g bg.hddm:1.8] Can be stacked eg Random+TagOnly:.123 where the :[num] defines BGRATE

#optional additional plugins that will be run along side danarest and hd_root.  This should be a comma separated list (e.g. plugin1,plugin2)
#CUSTOM_PLUGINS= #or file:/.../file-to-use which is a configuration file for jana/hd_root
#==============================================================================
#EVERYTHING BELOW FOR BATCH ONLY

#VERBOSE=True

BATCH_SYSTEM=swif #can be swif or condor or osg or qsub adding :[name] will pass -q [name] into PBS.

#environment file location
ENVIRONMENT_FILE=your-environment-files     #change this to your own environment file

WORKFLOW_NAME=WORKFLOW-NAME #SWIF WORKFLOW NAME
PROJECT = gluex          # http://scicomp.jlab.org/scicomp/#/projects
TRACK= simulation          # https://scicomp.jlab.org/docs/batch_job_tracks

# RESOURCES for swif jobs
DISK=5GB            # Max Disk usage
RAM=5GB            # Max RAM usage
TIMELIMIT=300minutes      # Max walltime.  This may be of the form xx:xx:xx depending on your system
OS=centos7        # Specify CentOS65 machines
```

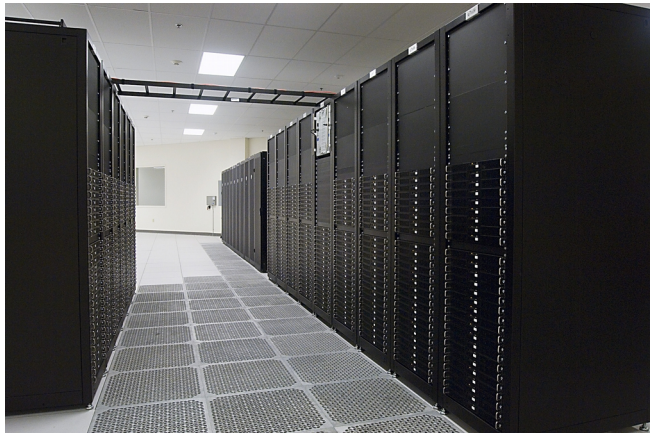
Some of the Most Important Options

- **DATA_OUTPUT_BASE_DIR**
  - This is the full path to the root output directory. Mcwrapper will create the last directory if needed
    - /work/halld2/home/[uname]/OUTPUT/
- **GENERATOR_CONFIG**
  - Most generators contain hooks to make your life easier. Look in the corresponding folder in $MCWRAPPER_CENTRAL/Generators
- **BKG**
  - None
    - Removes backgrounds
  - BeamPhotons
    - Add EM background comprised of a bunch of “beam photons”
  - TagOnly
    - Same as above but only produces tagger accidentals
  - Random
    - Data from the random trigger is fed into the MC at the hit level.
    - Requires **Random:[sim-recon-tag]**

- **CUSTOM_PLUGINS**
  - Either a comma seperated list of plugin names or more powerfully file:/……/jana_config.config (a control file

Generating MC at JLab

- Fill out some parts of the config file MCtutorial_SWIF.config (the [] entries)
  - DATA_OUTPUT_BASE_DIR (full path)
    - It will create the directory if needed
  - GENERATOR_CONFIG
    - The full path to and including gen_omega_3pi.cfg
  - BATCH_SYS
    - Use swif
  - WORKFLOW_NAME
    - Your choice
  - CUSTOM_PLUGINS
    - Use **file:/**[full pth]/jana_config.config . The config file you made previously in session 1b
  - ENV_FILE
    - Use the environment file you made in session 1a



- **gluex_MC.py MCtutorial_SWIF.config 30730 100000 batch=1**

It is recommended to run a small interactive job (without batch=1) before submitting to any batch system

Open Science Grid

- The OSG is a cluster of clusters
  - Institutions can donate computational resources and in exchange use grid resources
- Jobs submitted to the grid get placed on an available node; somewhere...
- Need a grid certificate with a valid Virtual Organization (GlueX is a valid VO)
  - Getting_a_Grid_Certificate
- Need a submit node
  - scosg16 at jlab

Open Science Grid

Make some MC on the grid

- The OSG is essentially condor with some special configurations
  - It is already integrated with MCwrapper. Just be on a submit node (eg scosg16) with BATCH_SYS=OSG in your wrapper config file
    - Note: the output directory etc must be systems mounted on the submit node
- There is an example of an OSG configuration file in the session 2b tutorial folder

Output

DATA_OUTPUT_BASE_DIR/

```
drwxr-sr-x 4 tbritton halld-2  4 Apr  4 14:07 configurations
drwxr-sr-x 2 tbritton halld-2  9 May 25 09:16 hddm
drwxr-sr-x 2 tbritton halld-2 19 May 25 09:16 root
```

- Configurations
  - Geant
    - Control.in (actually used)
  - Generation
    - Generator config file (actually used)
- Hddm
  - *.hddm
- Root
  - *.root
- Logs
  - Logs from batch

NOTE: If a job crashes for whatever reason on the OSG all files in the running directory will be returned to the main directory: DATA_OUTPUT_BASE_DIR

Other Helpful Tips

- Use the **TAG** option in the wrapper config file allows you to insert "non-canonical" text into file names
  - e.g. TAG=BGRate_2
- **Clean up folders** created during interactive jobs that failed
  - E.g. "11366_0". Failure to do so can result in successive runs being unable transfer some output data
- Use the **cleanxxxxx=0** options on the command line to save output
  - e.g. cleanmcsmear=0 will save the smeared hddm files
- Provide a **run range** to generate data in proportion to data taking
  - e.g. gluex_MC.py 11366-11555 100000000 ...

MCwrapper as a Framework

- MCwrapper provides a **framework** for MC production at GlueX, it, by itself, does not generate or simulate directly
- Because MCwrapper contains no compiled code it can be **easily wrapped**. Just templatize the wrapper config and command line
  - Find and replace is your friend!

Conclusion

- MCwrapper is a light weight set of scripts to manage the production of <u>your</u> MC
  - Integrates submission to numerous batch systems including qsub and condor
  - Reduces human errors
  - Guarantees consistency and provides provenance
  - Customizable for a wide variety of use cases

- Now go forth and make some MC!
- For additional help or bug reports contact **tbritton@jlab.org**