Overview and Use of SIMC

Dave Gaskell
NPS Collaboration Meeting
July 17-18, 2022

1. SIMC overview
2. Basics of using SIMC
What is SIMC?

SIMC is the standard Hall C Monte Carlo for coincidence reactions (similar to MCEEP) → written in FORTRAN (now gfortran compatible …)

Features:

→ Optics (COSY) and “aperture checking” Monte Carlos of spectrometers
   \([HMS, SOS, SHMS, HRS's, BigCal, …]\)
→ Includes radiative effects, multiple scattering, ionization energy loss, particle decay
→ Simple prescriptions available for FSIs, Coulomb Corrections, etc.

Reactions implemented:

1. Elastic and quasielastic → \(H(e,e'p), A(e,e'p)\)
2. Exclusive pion production
   → \(H(e,e'\pi^+)n, A(e,e'\pi^{+/-})\) [quasifree or coherent]
3. Kaon electroproduction → \(H(e,e'K^+)\Lambda, \Sigma, A(e,e'K^{+/-})\)
4. \(H(e,e'\pi^{+/-})X, D(e,e'\pi^{+/-})X\) [semi-inclusive]
5. \(H(e,e'K^{+/-})X, D(e,e'K^{+/-})X\) [semi-inclusive]
6. \(H(e,e'\rho \rightarrow \pi^+\pi^-)p, D(e,e'\rho \rightarrow \pi^+\pi^-)\) [diffractive \(\rho\)]
7. \(H(e,e'\pi^0)p, D(e,e'\pi^0)p/n, (e,e'\pi^0)X\)  

Peter’s updates underway
What SIMC is NOT...

SIMC is NOT a full detector response simulation a la GEANT

SIMC does NOT simulate a large class of processes simultaneously to generate backgrounds (like Pythia for example)

SIMC is not a generic event generator \(\rightarrow\) processes are generated over a limited phase space with the specific purpose of being “thrown” into a spectrometer

SIMC is not hard to modify \(\rightarrow\) if you want it to do something else (different cross section model, add new spectrometer, etc.) it is pretty easy to update + I can help
Overview

• Initialization
  – Choose reaction, final state (if appropriate)
  – Disable/enable implementation of (or correction for) raster, eloss …

• Event generation
  – Select vertex based on target size, position, raster size, beam spot size
  – Determine energy, angle generation that will populate 100% of the acceptance (accounting for radiation, energy loss, …)

• Physics Processes
  – Event-by-event multiple scattering, radiative corrections, particle decay, coulomb corrections

• Acceptance
  – Can apply geometric cuts or spectrometer model. Default spec. models include target/spec. offsets, model of magnetic elements, apertures at front, back, middle of magnets, collimators, detector active area

• Event Reconstruction
  – Tracks are fitted in the focal plane and reconstructed to the target. Apply (average) energy loss, fast raster corrections. Calculate physics quantities for Ntuple/Root tree
Example: Exclusive Pion Electroproduction

Initialize limits:

Generate vertex:

Generate scattering kinematics:

H(e,e’π): Generate \( \theta_e, \phi_e, \theta_\pi, \phi_\pi, p_e \) – calculate \( p_\pi \)

D(e,e’π): Generate \( \theta_e, \phi_e, \theta_\pi, \phi_\pi, p_e, \theta_N, \phi_N, p_N \) \( \rightarrow \) calculate \( p_\pi \)

Modifications to kinematics:

Follow particles:

- Apply cross section weighting (use model for free proton/neutron in \( \gamma \)-N center of mass)
- Simulate particle decay
  \( \rightarrow \) Decay the meson and follow the decay product(s)

Calculate Normalization Factors:

Bound nucleon momentum/direction
Spectrometer Models (HMS, SHMS, HRS ...)

- Magnetic elements simulated using COSY
- Real apertures
  - collimators, vacuum pipes
  - magnet apertures (front, middle, and back)
- Fiducial cuts for detectors (depends on analysis, trigger)
- Sequential transformations for magnetic elements, continuous drift for field free regions (important for decay)
- Reconstruct track
  - Use smeared position at drift chamber
  - Fit track at focal plane
  - Reconstruct to target using matrix elements fit from MC data
- Can easily disable spectrometer, apply geometric cuts, add new spectrometer/detector (for example, NPS)
Getting SIMC

https://github.com/JeffersonLab/simc_gfortran
SIMC Basics

• Compiling: just type “make” in the main directory
• Important directories
  – infiles → where the input files for your simulation live
  – worksim → SIMC event-by-event output (root trees)
  – outfiles → where the *.hist file lives → need this for normfac
  – util → “helper” applications to generate output
• Helper applications
  – SIMC default output is a FORTRAN binary file
  – Helper applications in util directory convert to root trees or paw ntuple
  – util/ntuple and util/root_tree → must be compiled separately
  – Choose appropriate Makefile (Makefile.rhel9 or Makefile.rhel7)
SIMC: structures $\rightarrow$ derived types

- Long ago – had to convert from “structures” to ”derived types”

```
structure /double_arm/
  structure /arm/ e
    real*8   delta, yptar, xptar, z
  end structure
structure /arm2/ p
  real*8   delta, yptar, xptar, z
end structure
end structure
```

Use as variables, like:
- recon.e.delta
- vertex.p.xptar

```
type arm
  sequence
  real*8   delta, yptar, xptar, z
end type

type arm2
  sequence
  real*8   delta, yptar, xptar, z
end type

type double_arm
  sequence
  type(arm)::e
  type(arm2)::p
end type
```

Use as variables, like:
- recon%e%delta
- vertex%p%xptar
SIMC RHEL9/Alma9 updates: derived types in modules

• First try to transition to Alma9 – all the derived types were broken
• After lots of googling found a helpful web page that described the problem

• Compilers can handle derived types in 2 different ways
  • Name equivalence: two types are equal if they have the same name
  • Index equivalence: each time a type is declared/defined, it’s given a unique index, even if they have the same name!
  • Old compiler used first option, new gfortran uses second
• How to fix/deal with this?
  • Put the derived types in a module and load the module in each relevant subroutine
  • All the derived types that were in “structures.inc” are now in “modules.f”

```
MODULE structureModule
! Define some BASIC record structures and associated parameters
! ... generic cut --> initialized with MAX large and MIN small
type cuttype
  sequence               
  real*8                min, max
end type cuttype
...
```

In each relevant subroutine, added this statement:

```
USE structureModule
```
SIMC Input File

begin parm e_arm_accept
SPhedgenQdeltaTmin = -10.0 ; delta min (SPECTROMETER ACCEPTANCE)
SPhedgenQdeltaTmax = 10.0 ; delta max
SPhedgenQpWeightMax = -90.0 ; yptar min = [TF] / 1000 (rad)
SPhedgenQpWeightMin = -100.0 ; yptar max = [TF] / 1000
SPhedgenWpXptarMin = -100.0 ; xptar min = [TF] / 1000
SPhedgenWpXptarMax = 100.0 ; xptar max = [TF] / 1000
end parm e_arm_accept

begin parm p_arm_accept
SPhedgenQpdeltaTmin = -15.0 ; delta min (SPECTROMETER ACCEPTANCE)
SPhedgenQpdeltaTmax = 30.0 ; delta max
SPhedgenQpWeightMax = -90.0 ; yptar min = [TF] / 1000 (rad)
SPhedgenQpWeightMin = -100.0 ; yptar max = [TF] / 1000
SPhedgenWpXptarMin = -100.0 ; xptar min = [TF] / 1000
SPhedgenWpXptarMax = 100.0 ; xptar max = [TF] / 1000
end parm p_arm_accept

begin parm beamandtargetinfo
gentxwidth = 0.605 (microns) ; beam width - one sigma (cm) (89 microns)
gentxwidth = 0.605 (microns) ; beam width - one sigma (cm) (42 microns)
targkfr1 = 0.1 ; raster pattern: 1 square, 2 circular 3 triangular
targkfr2 = 0.1 ; horizontal size OR inner radius [2]
targkfr2 = 0.1 ; vertical size OR outer radius [2]
targxoffset = 0.0 ; target x offset (cm) : yx = beam right
targyoffset = 0.0 ; target y offset (cm) : yx = up
targzoffset = 0.0 ; target z offset (cm) : z = downstream
targzoffset = 0.0 ; target z offset (cm) : z = downstream
zreal = zonuinal + offset
end parm beamandtargetinfo

These are offsets applied before the call to the single arm montecarlos.
Offsets are in spectrometer coordinate system. Positive xptar offset
means spectrometer is positioned at large xptar (i.e. below target, and
thus pointing above target and giving a negative shift to particle's xptar)

begin parm spec_offset
specxoffset = 0.
specyoffset = 0.
speczoffset = 0.
xptar offset (cm) : x(x)ptar is slope, so
yptar offset (cm) : y(x)ptar is slope, so
zptar offset (cm) : x(z)ptar is slope, so
end parm spec_offset

Jeffrey

begin parm kinematics_main
Ebeam = 10551.0 ; (MeV)
Ebeam = 10551.0 ; beam energy variation (%)
electron_arm = 5 ; 3 = m:4:8,5:8,2:8,4:8,5:8
hadron_arm = 1 ; 3 = m:4:8,5:8,2:8,4:8,5:8
specEcm = 5000.0 ; e cm central momentum (MeV/c)
specktheta = 16.54 ; e arm angle setting (degrees)
specchp = 4170.0 ; p arm central momentum (MeV/c)
specphitheta = 29.37 ; p arm angle setting (degrees)
end parm kinematics_main

begin parm target
 targA = 1.0 ; target A
targZ = 1.0 ; target Z
targmass_amu = 1.007276 ; target mass in amu
targmass_au = 0 ; recoil mass in amu (e(e-A-1 system,pions-A-2)
targthr = 0.67332 ; target density (g/cm3)
targthick = 295.172 ; target thickness (mg/cm2)
targangle = 0 ; target angle (for solid target) (degrees)
targpuricd = 100. ; target purity (%)
targcan = 3 ; 1 = beam can (fpi), 2 = puddling can (mu cpi) 3 = DEF GeV cell
end parm target

begin parm experiment
ngen = 100000 ; POS: # of successes; NEG: # of tries
EXPcharge = 1.0 ; total charge (nC)
doing_phsp = 0 ; if all of the doing.* are
(doing_kao = 0 ; (ONE = TRUE) False, then doing (e,e'p).
doing_plon = 0 ; (ONE = TRUE)
which_plon = 0 ; 0 = 2p1, 1 = 2p1, 2 = 2p1 Delta, 3 = 2p1 Delta
doing_delta = 0 ; pi/2 with Delta final stat
doing_rho = 0 ; diffractive rho production
doing_sme = 0 ; sidis
doing_hplus = 0 ; positive or negative hadrons for sidis
doing_decay = 0 ; 1 = decay ON, 0 = decay OFF
ctau = 780.4 ; decay length (cm)
use_benhar_SF = 0 ; Use Benhar style spectral function
transparency = 0.36 ; Proton transparency, when using benhar SF
end parm experiment
These parameters should not change much in general

→ "using_Eloss" should always be on to simulate ionization energy loss, but "correct_Eloss" should be off since hcana does not apply a correction

→ Don’t change the radiation flags, except maybe "one_tail" → might want to turn off proton radiation for neutral particles
Input File Notes

• Kinematics
  – Units are in MeV
  – No need to account for ionization energy loss – that’s in simulation
  – Include synchrotron energy loss in beam energy (if relevant)
  – If you find offsets to nominal kinematics in your analysis – apply those to the SIMC kinematics → will impact the model cross section calculation
• Make sure to include contraction of target cell at low temperatures in target information
• Target info mixes units, density in g/cm$^3$, but thickness in mg/cm$^2$
• Generation volume: must be larger than nominal acceptance to get correct yield and cross sections. You can easily check in the *.hist file if the “found” values are too close to the generation limits
• Despite the names, “targ%yoffset” and “targ%xoffset” are beam position offsets
Running SIMC and Generating Output

- Create your input file: eep_example.inp
- run SIMC

```bash
[gaskell@spoon] ./simc
Enter the input filename (assumed to be in infiles directory)
eep_example
filename=../infiles/eep_example.inp
Use random seed = 0
E_bind = 0.0000000000000000000 MeV in limits_init (QF only)
********** H(e,e'p) **********
SHMS is detecting electrons (SSA TUNE)
HMS is detecting hadrons
Welcome to Hall C++ at JLab12 - Got to work!
NOTE: Will NOT be calculating Coulomb correction (default for Hydrogen target)
NOTE: Will NOT correct reconstructed data for energy loss
RANLUX LUXURY LEVEL SET BY RLUXXGO = 3 P= 223
RANLUX INITIALIZED BY RLUXXGO FROM DEFAULT SEED
Initial random vector saved to file: outfiles/eep_example_start_random_state.dat

Generating Event 1 ... 0 successes so far - Monitor: 0.0000E+00
Generating Event 1001 ... 35 successes so far - Monitor: 0.1446E+04
Generating Event 2001 ... 66 successes so far - Monitor: 0.1239E+04
Generating Event 3001 ... 99 successes so far - Monitor: 0.1161E+04
Generating Event 4001 ... 122 successes so far - Monitor: 0.1118E+04
Generating Event 5001 ... 148 successes so far - Monitor: 0.1083E+04
Generating Event 6001 ... 173 successes so far - Monitor: 0.1258E+04
Generating Event 7001 ... 197 successes so far - Monitor: 0.1270E+04
Generating Event 8001 ... 221 successes so far - Monitor: 0.1284E+04
Generating Event 9001 ... 246 successes so far - Monitor: 0.1298E+04
Generating Event 10001 ... 272 successes so far - Monitor: 0.1314E+04
Generating Event 11001 ... 298 successes so far - Monitor: 0.1330E+04
Generating Event 12001 ... 324 successes so far - Monitor: 0.1347E+04
Generating Event 13001 ... 350 successes so far - Monitor: 0.1365E+04
Generating Event 14001 ... 376 successes so far - Monitor: 0.1383E+04
Generating Event 15001 ... 402 successes so far - Monitor: 0.1402E+04
Generating Event 16001 ... 428 successes so far - Monitor: 0.1421E+04
Generating Event 17001 ... 455 successes so far - Monitor: 0.1440E+04
Generating Event 18001 ... 481 successes so far - Monitor: 0.1459E+04
Generating Event 19001 ... 508 successes so far - Monitor: 0.1479E+04
Generating Event 20001 ... 534 successes so far - Monitor: 0.1499E+04
Generating Event 21001 ... 560 successes so far - Monitor: 0.1520E+04
Generating Event 22001 ... 586 successes so far - Monitor: 0.1541E+04
Generating Event 23001 ... 612 successes so far - Monitor: 0.1562E+04
Generating Event 24001 ... 638 successes so far - Monitor: 0.1583E+04
Generating Event 25001 ... 664 successes so far - Monitor: 0.1605E+04
Generating Event 26001 ... 690 successes so far - Monitor: 0.1627E+04
Generating Event 27001 ... 716 successes so far - Monitor: 0.1649E+04
Generating Event 28001 ... 742 successes so far - Monitor: 0.1671E+04
Generating Event 29001 ... 768 successes so far - Monitor: 0.1693E+04
Generating Event 30001 ... 794 successes so far - Monitor: 0.1716E+04
```
Running SIMC and Generating Output

- Go to \texttt{util/root\_tree} directory
- \texttt{./make\_root\_tree}
Make some plots and get some yields

Normalized yield by applying “Weight” from tree
→ Includes cross section, radiative corrections, small jacobian
→ What’s this 0.161960E+07 thing?
   → “normfac” from eep_example.hist file
   → Includes target thickness, simulated charge, generation volume

→ Also need to divide by number of events in tree

Result from this simulation (with these cuts) was:
Yield = 542.9 counts/mC
Reconstructed Quantities in SIMC

• Calculation of reconstructed quantities (W, x, Q2, etc.) may not match how these quantities are calculated in hcana
  – Reconstruction was originally intended to match calculations in Hall C 6 GeV fortran analyzer
• Found to be a problem for H(e,e’p) analysis for KaonLT/PionLT → definitions of missing momentum components weren’t even consistent
• Ideally, we should make calculations in simc match hcana
  – In the short term, should use stand-alone scripts to make comparable output
  – Richard Trotta has done this for H(e,e’p) and exclusive Kaon production: https://github.com/trottar/simc_gfortran/tree/fall_2023_kaon_xsects/recon_hcana
Updates for NPS

- Peter B. has done most of the work needed to make SIMC work for NPS
- Adding event generation for **exclusive and semi-inclusive** $\pi^0$ is straightforward $\rightarrow$ same as $\pi^+$ and $\pi^-$
  - Additional complication is decay of $\pi^0$ into 2 photons (extra subroutine)
- DVCS event generation same as exclusive mesons, just changing mass to zero!
- Physics models needed for cross sections:
  - SIDIS $\pi^0 \rightarrow$ average of $\pi^+$ and $\pi^-$
  - Exclusive $\pi^0$
    - Large $W (>2$ GeV): Average of $\pi^+$ and $\pi^-$, removing pion pole contributions
    - Low $W (<2$ GeV): MAID model
- NPS calorimeter can just be an aperture for checking acceptance. Simple resolution smearing can be added later if desired
- I’m working on incorporating Peter’s updates into the main branch on version on github
SIMC Summary

• SIMC is a pretty simple tool in many ways, but powerful
• If full GEANT4 simulations are needed (e.g. for NPS) SIMC can be combined with those simulations
  – SBS is doing this for d(e,e’N) simulations
• Main drawback is that it’s fortran
  – Modern cross section models are usually in C++, etc.
  – Easy work-around is to use look-up tables → Peter has done this for the MAID model of pion electroproduction for example
• Peter has already implemented many of the needed updates, integration underway