The PHOBOS Glauber Monte Carlo

B.Alver[2], M.Baker[1], C.Loizides[2], P.Steinberg[1]

[1]*Brookhaven National Laboratory, Upton, NY 11973, USA*
[2]*Massachusetts Institute of Technology, Cambridge, MA 02139, USA*

"Glauber" models are used to calculate geometric quantities in the initial state of heavy ion collisions, such as impact parameter, number of participating nucleons and initial eccentricity. The four RHIC experiments have different methods for Glauber Model calculations, leading to similar results for various geometric observables. In this document, we describe an implementation of the Monte Carlo based Glauber Model calculation used by the PHOBOS experiment. The assumptions that go in the calculation are described. A user's guide is provided for running various calculations.

I. INTRODUCTION

In heavy-ion collisions, initial geometric quantities such as impact parameter and shape of the collision region cannot be directly determined experimentally. However, it is possible to relate the number of observed particles and number of spectator neutrons to the centrality of the collision. Using the percentile centrality of a collision, the initial geometric configuration can be estimated with models of the contents of a typical nucleus.

These models fall in two main classes. (For a recent review, see Ref. [1].) In the so called "optical" Glauber calculations, a smooth matter density is assumed, typically described by a Fermi distribution in the radial direction and uniform over solid angle. In the Monte Carlo based models, individual nucleons are stochastically distributed event-by-event and collision properties are calculated by averaging over multiple events. As discussed in Ref. [1] and Ref. [2], these two type of models lead to mostly similar results for simple quantities such as the number of participating nucleons (N_{part}) and impact parameters (b), but give different results in quantities where event-by-event fluctuations are significant, such as participant frame eccentricity (ϵ_{part}).

In this paper, we discuss in detail the Monte Carlo Glauber calculation implemented by PHOBOS. In section II, the method is outlined and the assumptions that go into the calculation are introduced. In section III, we discuss the implementation and the tutorial functions provided.

II. THE MODEL

The Monte Carlo Glauber Model calculation is performed in two steps. At first, the nucleon positions in each nucleus are stochastically determined. Then, the two nuclei are "collided", assuming the nucleons travel in a straight line along the beam axis (eikonal approximation) such that nucleons are tagged as wounded (participating) or spectator.

A. Makeup of Nuclei

The position of each nucleon in the nucleus is determined according to a probability density function. In a quantum mechanical picture, the probability density function can be thought of as the single-particle probability density and the position as the result of a position measurement. In the determination of the nucleon positions in a given nucleus, it is possible to require a minimum inter-nucleon separation (d_{min}) between the centers of the nucleons.

The probability distribution is typically taken to be uniform in azimuthal and polar angles. The radial probability function is modeled from nuclear charge densities extracted in low-energy electron scattering experiments [3]. The nuclear charge density is usually parameterized by a Fermi distribution with three parameters:

$$\rho(r) = \rho_0 \frac{1 + w(r/R)^2}{1 + exp(\frac{r-R}{a})} , \qquad (1)$$

where ρ_0 is the nucleon density, R is the nuclear radius, a is the skin depth and w corresponds to deviations from a spherical shape. The overall normalization (ρ_0) is not relevant for this calculation. Values of the other parameters used for different nuclei are listed in Table I.

Two exceptions are the deuteron (^{2}H) and sulfur (^{32}S) nuclei. For sulfur, a three parameter Gaussian form is used:

$$\rho(r) = \rho_0 \frac{1 + w(r/R)^2}{1 + exp(\frac{r^2-R^2}{a^2})} . \qquad (2)$$

The values of R, a and w for sulfur are also given in Table I. For deuteron, three options are supported:

1. The three parameter Fermi distribution can be used, with the values given in Table I.

2. The Hulthén form can be used:

$$\rho(r) = \rho_0 \left(\frac{e^{-ar} + e^{-br}}{r} \right)^2 , \qquad (3)$$

where $a = 0.457$ fm^{-1} and $b = 2.35$ fm^{-1} [4, 5].

3. The proton can be randomly placed using the Hulthén form given above and the neutron can be placed opposite to it.

Nucleus	R [fm]	a [fm]	w [fm]
^{2}H	0.01	0.5882	0
^{16}O	2.608	0.513	-0.51
^{28}Si	3.34	0.580	-0.233
^{32}S	2.54	2.191	0.16
^{40}Ca	3.766	0.586	-0.161
^{58}Ni	4.309	0.517	-0.1308
^{62}Cu	4.2	0.596	0
^{186}W	6.58	0.480	0
^{197}Au	6.38	0.535	0
^{207}Pb^a	6.62	0.546	0
^{238}U	6.81	0.6	0

[a]These values are also used for ^{208}Pb for which Fermi parameters are not available. It has been noted that Bessel-Fourier coefficients for the two nuclei are similar [3].

TABLE I: Nuclear charge density parameters for different nuclei, taken from Ref. [3].

It should be noted that the 3rd option was used in PHOBOS analyses.

B. Collision Process

The impact parameter of the collision is chosen randomly from a distribution $dN/db \propto b$ up to some large maximum b_{max} with $b_{max} \simeq 20\,\text{fm} > 2R_A$. The centers of the nuclei are calculated and shifted to $(-b/2, 0, 0)$ and $(b/2, 0, 0)$ [1]. It is assumed that the nucleons move along a straight trajectory along the beam axis. (The longitudinal coordinate does not play a role in the calculation.)

The inelastic nucleon-nucleon cross section (σ_{NN}), which is only a function of the collision energy is extracted from p+p collisions. At the top RHIC energy of $\sqrt{s_{NN}} = 200$ GeV, $\sigma_{NN} = 42$ mb, while at the LHC it is expected to be around $\sigma_{NN} = 72$ mb (with large uncertainty from the unknown elastic cross section). The "ball diameter" is defined as:

$$D = \sqrt{\sigma_{NN}/\pi}. \tag{4}$$

Two nucleons from different nuclei are assumed to collide if their relative transverse distance is less than the ball diameter. If no such nucleon–nucleon collision is registered for any pair of nucleons, then no nucleus–nucleus collision occurred. Counters for determination of the total (geometric) cross section are updated accordingly.

---

[1] Throughout the paper, the reaction plane, defined by the impact parameter and the beam direction, is given by the x- and z-axes, while the transverse plane is given by the x- and y-axes.
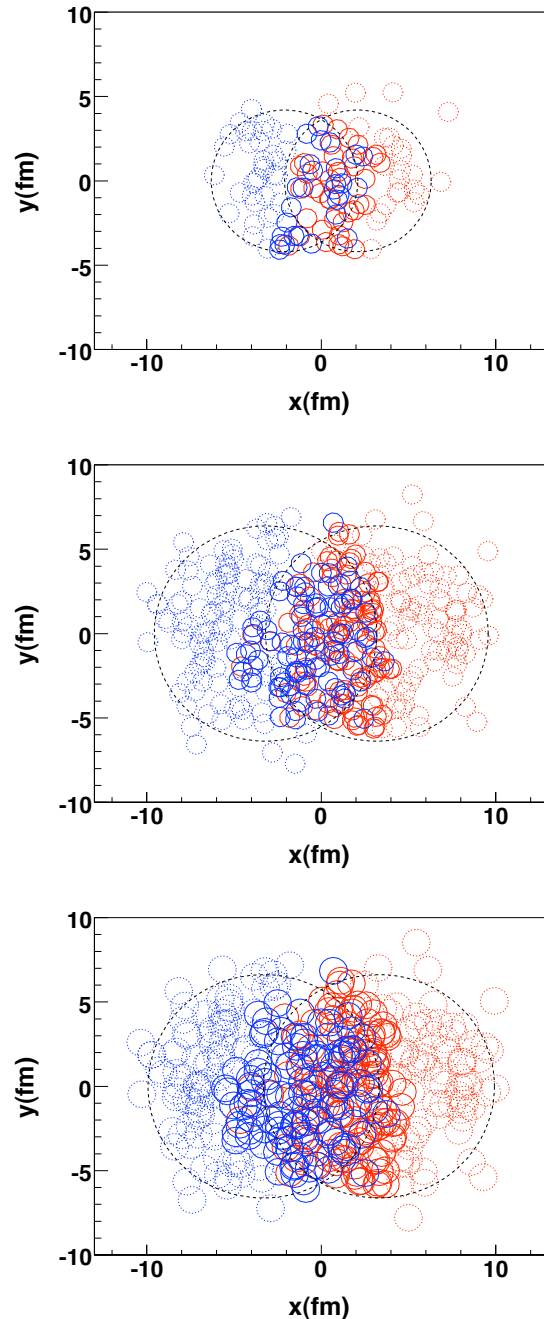

FIG. 1: Typical events for Cu+Cu (top panel), Au+Au (middle panel), and Pb+Pb (lower panel) collisions, the first two performed at RHIC energies and the latter at the LHC. Wounded nucleons (participants) are indicated as solid circles, while spectators are dotted circles.

III. USERS' GUIDE

The PHOBOS Glauber MC code works within the ROOT framework (ROOT 4.00/08 or higher [6]). The code is contained in the macro

runglauber_vX.Y.C [7] (Latest version is 1.0.). Three classes, `TGlauNucleon`, `TGlauNucleus` and `TGlauberMC` and two example functions `runAndSaveNtuple()` and `runAndSaveNucleons()` are defined in the provided macro. While the functionality is essentially complete for known applications of the Glauber approach, users are encouraged to write their own functions to access results of the Glauber simulation or to modify the code:

- `TGlauNucleon` is used to store information about a single nucleon. The stored quantities are the position of the nucleon, the number of binary collisions that the nucleon has had and which nucleus the nucleon is in, "A" or "B". For every simulated event, the user can obtain an array containing all nucleons (via `TGlauberMC::GetNucleons()`).

- `TGlauNucleus` is used to generate and store information about a single nucleus. The user is not expected to interact with this class.

- `TGlauberMC` is the main steering class used to generate events and calculate event-by-event quantities such as the number of participating nucleons.

The steering class `TGlauberMC` has one constructor

```
TGlauberMC::TGlauberMC(Text_t* NA,
                       Text_t* NB,
                       Double_t xsect)
```

where `NA` and `NB` are the names of the colliding nuclei and `xsect` is the nucleon-nucleon cross section given in mb. The defined nuclei names are: "p", "d", "dhh", "dh", "O", "Si", "S", "Ca", "Cu", "W", "Pb", "Au", "Ni" and "U" (see Table I). For deuteron, the names "d", "dhh" and "dh" correspond to the three options described in section II A respectively. Units are generally given in fm for distances, while in mb for cross sections.

A. Running the Code

To generate Au+Au collisions at $\sqrt{s_{NN}} = 200$ GeV ($\sqrt{s_{\mathrm{NN}}} = 42$ mb) one would construct a `TGlauberMC` object by issuing the commands:

```
root [0] .L runglauber_X.Y.C+
root [1] TGlauberMC glauber("Au","Au",42);
```

where the first ROOT command compiles, links and loads the compiled macro [2] including the Glauber code as explained in chapter 2 of the ROOT users' guide.

Events can be generated interactively using the two functions

―――――

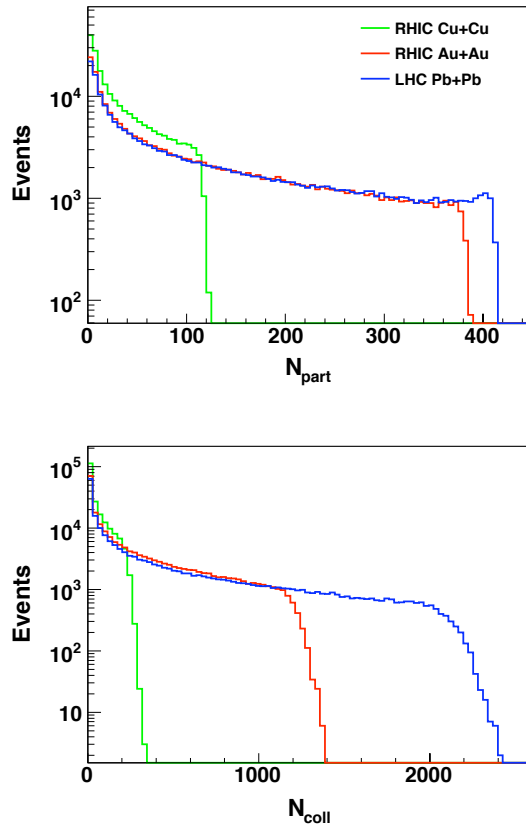[2] Note that you must replace X.Y with the current version number of the code, for example 1.1.

FIG. 2: Distributions of N_{part} and N_{coll} for 10k events for Cu+Cu and Au+Au at RHIC, and Pb+Pb at the LHC.

- `TGlauberMC::NextEvent(Double_t bgen)`, which is used to run an event at a specified impact parameter, or over a range of impact parameters (if `bgen=-1`, the default value) as described in section II B).

- `TGlauberMC::Run(Int_t nevents)` which is used to run a large event sample by invoking `NextEvent` many times.

Other important public member functions are:

- `TGlauberMC::SetMinDistance(Double_t d)`, which is used to set minimum nucleon seperation within a nucleus, d_{min} (default is 0.4 fm)

- `TGlauberMC::SetBmin(Double_t bmin)` and `TGlauberMC::SetBmax(Double_t bmax)`, which can be used to set the range of impact parameter values generated in `Run()`.

- `TGlauberMC::GetTotXSect()` which returns the total nucleus-nucleus cross section, calculated when the function `Run()` is called.

- `TGlauberMC::Draw()` which draws the current event in the current pad.

B. Example functions

Two example functions are provided to demonstrate how to run the model.

`runAndSaveNtuple()` generates a number of Monte Carlo events and saves some event-by-event quantities. It takes as parameters, the number of events to be generated, the collision system, the nucleon-nucleon cross section, the minimum separation distance and the output file name. It creates and stores an ntuple in the output file with the following event-by-event quantities:

- `Npart`: Number of participating nucleons.

- `Ncoll`: Number of binary collisions.

- `B`: Generated impact parameter.

- `MeanX`: Mean of x for wounded nucleons, $\{x\}$.

- `MeanY`: Mean of y for wounded nucleons, $\{y\}$.

- `MeanX2`: Mean of x^2 for wounded nucleons, $\{x^2\}$.

- `MeanY2`: Mean of y^2 for wounded nucleons, $\{y^2\}$.

- `MeanXY`: Mean of xy for wounded nucleons, $\{xy\}$.

- `VarX`: Variance of x for wounded nucleons, σ_x^2.

- `VarY`: Variance of y for wounded nucleons, σ_y^2.

- `VarXY`: Covariance of x and y for wounded nucleons, $\sigma_{xy} \equiv \{xy\} - \{x\}\{y\}$.

- `MeanXSystem`: Mean of x for all nucleons.

- `MeanYSystem`: Mean of y for all nucleons.

- `MeanXA`: Mean of x for nucleons in nucleus A.

- `MeanYA`: Mean of y for nucleons in nucleus A.

- `MeanXB`: Mean of x for nucleons in nucleus B.

- `MeanYB`: Mean of y for nucleons in nucleus B.

It is important to note that for each of these event-by-event quantities a "getter" function is implemented providing the users the option to write their own event loop (using `TGlauberMC::NextEvent()`.

The function `runAndSaveNucleons()` generates a number of Monte Carlo events and saves an array of `TGlauNucleon` objects for each event. It is also possible to use this function to print out the values stored in the nucleons by setting the verbosity parameter. The function takes as parameters the number of events to be generated, the collision system, the nucleon-nucleon cross section, the minimum separation distance, the verbosity flag and the output file name.
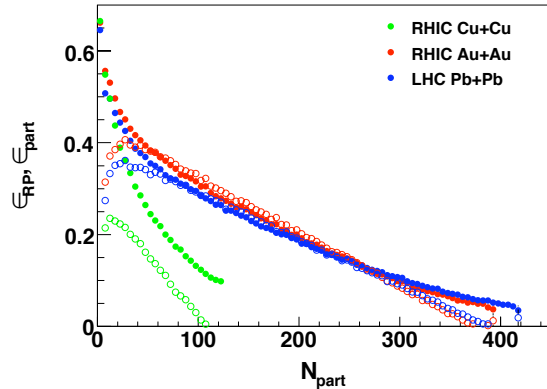

FIG. 3: ϵ_{RP} (open symbols) and ϵ_{part} (closed symbols) as a function of N_{part} for Cu+Cu and Au+Au collisions at RHIC and Pb+Pb collisions at the LHC.

C. Sample Results

As an example application of this code, 10k events were generated for Cu+Cu and Au+Au at RHIC energies ($\sqrt{s_{\mathrm{NN}}} = 42$ mb), and Pb+Pb at LHC beam energy ($\sqrt{s_{\mathrm{NN}}} = 72$ mb) using the `runAndSaveNtuple()` function. The resulting ntuples were used to plot the distributions of N_{part} and N_{coll}, shown in Fig. 2. Using the event-by-event quantities, one can construct combinations of moments like [2]:

- Reaction-plane eccentricity ϵ_{RP}

$$\epsilon_{\mathrm{RP}} = \frac{\mathrm{VarY} - \mathrm{VarX}}{\mathrm{VarY} + \mathrm{VarX}} \quad (5)$$

- Participant eccentricity ϵ_{part}

$$\epsilon_{\mathrm{part}} = \frac{\sqrt{(\mathrm{VarY} - \mathrm{VarX})^2 + 4\mathrm{VarXY}^2}}{\mathrm{VarY} + \mathrm{VarX}} \quad (6)$$

which are shown in Fig. 3 for the different systems.

IV. CONCLUSION

This work has described the PHOBOS implementation of the "Glauber Model" commonly used by heavy ion physics experiments to study the initial state configurations of nuclear matter. The code, accessible online, can be used within user code or in a standalone mode allowing analysis of various distributions (e.g. N_{part}, N_{coll}, b, ϵ_{part}). The authors welcome comments on the code and suggestions on how to make it more useful to both experimentalists and theorists.

---

[1] M.L. Miller, K. Reygers, S.J. Sanders, and P. Steinberg (2007), arXiv:nucl-ex/0701025.

[2] B.Alver *et al.* [PHOBOS Collaboration], Phys. Rev. C **77** (2008) 014906, arXiv:0711.3724 [nucl-ex].

[3] H. De Vries, C.W. De Jager, and C. De Vries, Atomic Data and Nuclear Data Tables **36** 495 (1987).

[4] L. Hulthen and M. Sugawara, Handbuch der Physik **39** 1 (1957).

[5] S.S. Adler *et al.* [PHENIX Collaboration], Phys. Rev. Lett. **91** 072303 (2003).

[6] See `http://root.cern.ch` for installation files and documentation.

[7] See the TGlauberMC page on HepForge (`http://www.hepforge.org/downloads/tglaubermc`) for the most recent TGlauberMC release.