STAR Note 419

MEVSIM: A Monte Carlo Event Generator for STAR

R. L. Ray

Department of Physics
The University of Texas at Austin, Austin, Texas 78712

R. S. Longacre

Physics Department, Brookhaven National Laboratory, Upton, NY 11973

Abstract

A fast, simple to use Monte Carlo based event generator is presented which is intended to facilitate STAR simulation studies and the development of STAR analysis software. This new event generator provides a fast, convenient means for producing large numbers of uncorrelated A+A collision events which can be used for a variety of applications in STAR, including quality assurance evaluation of event reconstruction software, determination of detector acceptances and tracking efficiencies, physics analysis of event-by-event global variables, studies of strange, rare and exotic particle reconstruction, and so on. The user may select the number of events, the particle types, the multiplicities, the one-body momentum space distributions and the detector acceptance ranges. The various algorithms used in the code and its capabilities are explained. Additional user information is also discussed. The computer code implementation exists in both standalone and STAF package forms. The latter version of the code is called MEVSIM.

1 Introduction

When carrying out STAR detector simulations and physics analysis studies one often needs many events of a similar type that can either be readily selected or tightly controlled by the user to meet the specific needs of the simulation project. For example, in evaluating the performance of the event reconstruction software it is beneficial to use events which have the same particle multiplicities and distributions from event-to-event in order to help identify anomalous changes in the tracking performance. Events with flat distributions in rapidity facilitate the determination of detector acceptances and tracking efficiencies. Events with enhanced numbers of multi-strange particles and other rare particles facilitate the study of the corresponding reconstruction and physics analysis software. When using standard events generators such as HIJING, VENUS, RQMD, VNI, etc. the user has little control over the simulated particle production other than to select the beam

energy and colliding nuclei, impact parameter, and perhaps the settings of some physics process control switches. Furthermore, most of these codes require significant amounts of cpu-time. For many STAR simulation studies these sophisticated models are not really necessary and unless the user is interested in the inherent particle correlations which a given model may produce, a simple Monte Carlo particle generator will generally suffice.

In response to this need we have developed a standalone Fortran 77 computer code and a parallel STAR software library STAF package, called MEVSIM. The code is designed to provide a fast, convenient means for producing large numbers of uncorrelated A+A collision events which can be used for a variety of applications in STAR, including quality assurance evaluation of event reconstruction software, determination of detector acceptances and tracking efficiencies, physics analysis of event-by-event global variables, studies of strange and rare particle reconstruction, and so on. The user may select the detector acceptance ranges, the number of events, the particle types, the multiplicities, and the one-body momentum space distributions with respect to transverse momentum (p_T), rapidity (y) and azimuthal angle (ϕ) from a menu of models.

The particle multiplicities can either be fixed for each event or allowed to randomly vary from event-to-event according to Poisson statistics. The parameters of the one-body momentum space distributions (*e.g.* slope parameter, rapidity width, radial expansion velocity) can either be fixed or randomly varied from event-to-event according to a Gaussian distribution in order to simulate dynamical fluctuations. In addition the user may specify a reaction plane angle and include anisotropic flow. The overall multiplicities for all particle types in each event can also be scaled by a common, multiplicative factor in order to simulate trigger fluctuations. In addition to the usual list of particle types in GEANT, the code also includes a number of short lived resonances, several of which have explicit mass distributions. The STAF package version, MEVSIM, includes routines which decay these resonances using known branching ratios and relativistic decay kinematics [1]; the resulting parent-daughter relationships are recorded in the output and the decay products are then passed on to GEANT. Resonances are not decayed in the standalone version however. Finally, the events produced are assumed to be in the A+A center-of-momentum frame which for symmetric beam species at RHIC coincides with the lab frame.

In the following sections we describe the random sampling methods used in the code, the one-body momentum space distribution models, the acceptance cuts for application to STAR, the implementation of resonances, trigger fluctuation simulations, reaction plane and anisotropic flow simulations, the standalone code and related files, the STAF package MEVSIM, and example input files.

2 Random Sampling Methods

The method used in the code to randomly sample a one-dimensional probability distribution is described first, followed by an explanation of the random selection - random rejection algorithm which is used to populate momentum space according to the specified one-body distributions.

In order to randomly sample a one-dimensional probability distribution $P(x)$ throughout the domain $x_1 \leq x \leq x_2$ one first obtains the integral,

$$F(x) = \int_{x_1}^{x} P(x')dx' \ , \tag{1}$$

normalized such that $F(x_2) = 1$. By finding the value of x such that $F(x) = f$, where f is a randomly selected number between 0 and 1, a random collection of x values can be obtained. A histogram of these values of x will be statistically consistent with the probability distribution $P(x)$. This method is much faster than the usual random selection - random rejection method since cpu-time is not wasted on trial values of x which end up being discarded. In the code the above is implemented by integrating $P(x)$ on a fixed mesh, randomly selecting f and interpolating the inverse F function on an uneven mesh using a Lagrange interpolation routine.

The preceding method is used to sample the reaction plane angle, the overall multiplicity rescaling factor, the particle multiplicities, the parameters of the one-body momentum space distribution model, the anisotropic flow parameters, and the resonance masses. Generally these follow Gaussian probability distributions,

$$P(x, x_{\rm mean}, \sigma) = N \exp\left[-(x - x_{\rm mean})^2/2\sigma^2\right] \ , \tag{2}$$

where N is a normalization constant and the user specifies the mean value, σ and number of σs (domain) over which x is to be sampled. The particle multiplicities follow a Poisson distribution,

$$P(x, x_{\rm mean}) = \frac{x_{\rm mean}^{x} e^{-x_{\rm mean}}}{x!} \tag{3}$$

with mean value $x_{\rm mean}$ and standard deviation $\sqrt{x_{\rm mean}}$. The user specifies the mean multiplicity and the ($\pm$) number of standard deviations about the mean over which to sample. The resonance masses follow approximate Breit-Wigner distributions which are listed in Sec. 5. In order for each of these quantities to fluctuate the user must input positive values for both the standard deviation and the ($\pm$) number of standard deviations over which to sample.

The momentum space distributions, being two-dimensional in p_T and y, are sampled by the usual random selection - random rejection method. Furthermore, since the anisotropic flow parameters depend on p_T and y, it was more convenient to also sample the azimuthal (ϕ) distribution by the random selection - random rejection method, rather than use the preceding integration-interpolation method.

The arbitrary, two-dimensional function $g(u,v)$, defined in the domain $u_1 \leq u \leq u_2$ and $v_1 \leq v \leq v_2$, is randomly sampled as follows. The maximum value of g ($g_{\rm max}$) in the domain is determined. Values of u and v in the domain are randomly selected and the quantity $g(u,v)/g_{\rm max}$ is computed. A random number from 0 to 1 is selected and if it is $\leq g(u,v)/g_{\rm max}$ (where g must be non-negative which is the case for particle densities), then the randomly selected coordinates u and v are retained; if not, these coordinates are rejected and another trial set of u and v is selected. Such methods are reliable but can suffer from poor cpu performance if the function $g(u,v)$ is strongly peaked within its domain. For RHIC collisions the p_T and y distributions are not expected to have large spikes and we expect this sampling method to perform well. However, as noted in Sec. 7, care should be exercised when including p_T and y dependence in the anisotropic flow in order to avoid poor cpu performance.

3 Momentum Distribution Models

For all momentum space distribution models used in the code the (p_T, y) and azimuthal (ϕ) dependences are factored. The various models for the (p_T, y) dependences are described in this section, while the ϕ dependence models are explained in Sec. 7.

A total of six (p_T, y) distribution models can be selected; the same model must be applied to each particle ID type in a run. The models include the following:

1. Factorized p_T and y dependence [2],

$$\frac{d^2N}{dp_T dy} = Ap_T \mathrm{e}^{-m_T/T} \mathrm{e}^{-y^2/2\sigma_y^2} \tag{4}$$

   where A is a nomalization constant, $m_T = \sqrt{m^2 + p_T^2}$, m is the particle mass, T is the m_T slope parameter or "temperature," and σ_y is the rapidity distribution width.

2. Non-expanding spherical thermal source (Pratt) [3],

$$\frac{d^2N}{dp_T dy} = Ap_T E \mathrm{e}^{-E/T} \tag{5}$$

   where $E = m_T \cosh(y)$ is the total energy.

3. Non-expanding spherical thermal source (Bertsch) [4],

$$\frac{d^2N}{dp_T dy} = \frac{Ap_T E}{\mathrm{e}^{E/T} - 1}. \tag{6}$$

4. Spherically expanding, thermally equilibrated source (Pratt) [5],

$$\frac{d^2N}{dp_T dy} = Ap_T E \mathrm{e}^{-\gamma E/T} \times \left[\frac{\sinh(y_p)}{y_p} + \frac{T}{\gamma E}\left(\frac{\sinh(y_p)}{y_p} - \cosh(y_p)\right)\right] \quad (7)$$

where $y_p = \gamma v p/T$, $\gamma = 1/\sqrt{1-v^2}$, v is the radial expansion velocity parameter, and $p = \sqrt{E^2 - m^2}$ is the total momentum.

5. Arbitrary, factorized p_T and y dependence loaded bin-by-bin,

$$\frac{d^2N}{dp_T dy} = Af(p_T)g(y) \quad (8)$$

where f and g are input histograms.

6. Arbitrary, two-dimensional function of p_T and y loaded bin-by-bin,

$$\frac{d^2N}{dp_T dy} = Ah(p_T, y) \quad (9)$$

where h is a two-dimensional histogram.

4 Kinematic Acceptance

The generated particles are confined within specified kinematic boundaries. The acceptance cuts are designed to be applicable to the STAR geometry. The acceptances are defined by lower and upper values for p_T, pseudorapidity (η) and azimuthal angle (ϕ) (between 0° and 360°).

5 Resonances

The available particle types include those listed in GEANT [6] plus a number of additional particles and resonances which are listed in Table I along with their assumed mass and width parameters. For most of these the widths are sufficiently broad to warrant including the mass fluctuations explicitly in the generated particle list. The mass distribution models assumed in the code are described in this section. Since the mass and width parameters depend somewhat on the model, the parameters listed in Table I may not agree exactly with similar values tabulated elsewhere, such as in Ref. [7]. In the code the resonance masses are sampled after the p_T, y and ϕ distributions have been determined. Hence, in the final output particle list the p_T, y, ϕ and mass distributions will be exactly as specified by the user. However, because the masses are changed after p_T and y are fixed,

the pseudorapidity values will change and the pseudorapidity distributions will be altered by the mass fluctuations.

The following resonance shapes (un-normalized) were assumed:

1. ρ-meson [8]

$$W(M) = \frac{q}{(M - M_\rho)^2 + \Gamma_\rho^2/4} \ , \tag{10}$$

where q is the c.m. momentum in the final $\pi - \pi$ system, and M_ρ and Γ_ρ are listed in Table I.

2. ω-meson [9]

$$W(M) = \frac{M_\omega^2 \Gamma_\omega}{(M_\omega^2 - M^2)^2 + M^2 \Gamma_\omega^2(s)} \ , \tag{11}$$

where $\Gamma_\omega(s) = \Gamma_\omega \, (M/M_\omega)^3$ and M_ω and Γ_ω are given in Table I.

3. ϕ-meson [10]

$$W(M) = \frac{\Gamma_\phi^2/4}{(M - M_\phi)^2 + \Gamma_\phi^2/4} \left(\frac{q}{q_0} \frac{E_0}{E} \right)^3 \ , \tag{12}$$

where q is the c.m. momentum in the final $K - K$ system, $E = \sqrt{q^2 + M_K^2}$, M_K is the kaon mass, q_0 and E_0 are evaluated for $M = M_\phi$, and M_ϕ and Γ_ϕ are given in Table I.

4. Δ-resonance [11]

$$W(M) = \frac{4\pi}{\mu q} \frac{\Gamma(q)^2}{(M - M_\Delta)^2 + \Gamma(q)^2/4} \ , \tag{13}$$

$$\Gamma(q) = \frac{2(qR)^3 \Gamma_\Delta}{1 + (qR)^2} \ , \tag{14}$$

where $qR = 0.81 q/m_\pi$, q is the c.m. momentum in the final $\pi - N$ system, and μ is the reduced, total energy of the $\pi - N$ system given by,

$$\mu = \frac{\epsilon_\pi \epsilon_N}{\epsilon_\pi + \epsilon_N} \ , \epsilon_\pi = \sqrt{q^2 + m_\pi^2} \ , \epsilon_N = \sqrt{q^2 + M_N^2} \ , \tag{15}$$

where m_π and M_N are the pion and nucleon masses, respectively, and M_Δ and Γ_Δ are given in Table I.

5. K^* resonances [12]

$$W(M) = \frac{\Gamma(q)^2 M_{K^*}^2}{(M^2 - M_{K^*}^2)^2 + \Gamma(q)^2 M_{K^*}^2} \ , \tag{16}$$

$$\Gamma(q) = \frac{2\Gamma_{K^*}(q/q_0)^3}{1 + (q/q_0)^2} \ , \tag{17}$$

where q is the c.m. momentum in the final $\pi - K$ system, q_0 is evaluated for $M = M_{K^*}$, and M_{K^*} and Γ_{K^*} are given in Table I.

For the STAF version of MEVSIM the user may request that the resonances be decayed according to preset branching ratios from ISAJET [1] using relativistic kinematics, or according to an arbitrary user specified decay scheme. For the latter application the interested user may refer to the file `decay.dat` in the MEVSIM package area of the STAR software library and the ISAJET writeup [1] for further information. The user may also choose to decay some, but not all resonances listed in the input file. If no resonances are decayed then the final output list of particles will be the same as that from the standalone version. If, for example, ρ-meson decay is selected, then all charge states (*i.e.* ρ^+, ρ^- and ρ^0) will be decayed if present in the input file.

6 Trigger Fluctuations

In the STAR A+A event stream, for which the Level 0 triggers are based on integrated yields in the CTB and MWC, the actual particle multiplicities would fluctuate from event-to-event, even if the trigger acceptance settings could correspond to an ideal δ-function. To simulate this effect in the data stream we included in the code the ability to rescale the multiplicities of all particle types in the event by a common multiplicative factor, f. This factor is selected randomly for each event according to a Gaussian distribution which the user specifies.

Ideally, the nominal mean multiplicities for each particle type, $N_{\rm pid}$, should be multiplied by f and the resulting Poisson distribution with mean value $fN_{\rm pid}$ and standard deviation $\sqrt{fN_{\rm pid}}$ sampled to yield the actual particle multiplicities for the event. However, it turns out to be much more straightforward in the code and to have less impact on the cpu demand, to instead, sample the nominal multiplicity distributions first and then scale the results to get the actual particle multiplicities for the event. Using this method the particle multiplicity distributions will have the correct mean value of $fN_{\rm pid}$ but the incorrect (non-Poisson) standard deviation of $f\sqrt{N_{\rm pid}}$ which is in error by the factor $\sqrt{f}$. It is anticipated that f will be within a few percent of 1.0 such that the actual, non-Poisson multiplicity distributions will only err by a few percent.

7 Reaction Plane and Anisotropic Flow

The detection and study of directed flow, elliptical flow and perhaps higher modes of particle flow in non-central A+A collision events at RHIC are of great interest to the STAR collaboration. These phenomena can be simulated with this generator. The user may select from among four options for the reaction plane:

1. Ignore the reaction plane, events are azimuthally invariant, *i.e.* no flow.

2. Assume a fixed reaction plane angle; useful for testing analysis code and debugging.

3. Assume a randomly fluctuating reaction plane angle which follows a Gaussian distribution; useful for evaluating reaction plane measurement resolution effects.

4. Assume a random reaction plane angle, uniformly distributed between 0° and 360°, which is the way the data will be acquired in the experiment.

The model assumed for the anisotropic flow is from Ref. [13], where

$$E\frac{d^3N}{dp^3} = \frac{1}{2\pi p_T}\frac{d^2N}{dp_T dy}\left[1 + \sum_{n=1}^{6} 2V_n(p_T, y)\cos[n(\phi - \Psi_R)]\right] , \tag{18}$$

where Ψ_R is the reaction plane angle, ϕ is the azimuthal angle of the outgoing particles, and the parameters $V_n(p_T, y)$ are given by [14],

$$V_n(p_T, y) = (V_{n1} + V_{n2}p_T^2)\exp\left(-V_{n3}(y - y_{\rm cm})^2\right) , \ n = \text{even} \tag{19}$$

$$V_n(p_T, y) = (V_{n1} + V_{n2}p_T)\text{sign}(y - y_{\rm cm})\left(V_{n3} + V_{n4}\,|y - y_{\rm cm}|^3\right), n = \text{odd} \tag{20}$$

where sign(y) is ± 1 for $y > 0$ or $y < 0$, respectively, and $y_{\rm cm}$ is the c.m. rapidity which is 0 for symmetric A+A collisions. Up to six Fourier components are permitted; directed flow ($n = 1$) and elliptical flow ($n = 2$) are most often used.

The user should be aware that careless use of the $V_n(p_T, y)$ parametrization could have negative impact on the code's performance. Large values of $V_{n,i}$ (i = 1,2,3,4) and/or large p_T and y acceptance ranges can lead to very large, unrealistic anisotropy, particularly near the edges of the acceptance. This can result in "spikey" ϕ dependences that will cause the random rejection process to perform inefficiently. The $V_n(p_T, y)$ parametrization should therefore be used with caution and the user should verify that the "flow" remains within reasonable bounds throughout the (p_T, y) acceptance range.

8 Standalone Version

The source code, executable, documentation and example input files are available on the RHIC computing facility disks under `~ray/evgen/multi_gen` in the files `multiplicity_gen.f` (source), `Parameter_values.inc`, `common_dist_bin.inc`, and `common_facfac.inc` (include files), `bin/rcf/multiplicity_gen.exe` (executable binary for `rcf.rhic.bnl.gov`), `README` and `multiplicity_gen.doc` (documentation), `mult_gen_venus-like.in` and `mult_gen_vini-like.in`. The latter two files are example input for the standalone version of the code and have been designed to reproduce the major particle distributions predicted by VENUS and VNI for 200 A·GeV Au+Au central collisions, respectively. The input parameters are defined and explained in `multiplicity_gen.doc`, the relevant portion of which is reproduced in Appendix A. The `README` file contains information necessary for

running the code. Although resonances may be included, they are not decayed in the standalone version.

The standalone code produces a log file, `mult_gen.log`, containing diagnostic information and error messages if any. For normal, successful runs this file should be empty. An output file, `mult_gen.out`, is produced which includes a lengthy header containing the input parameters plus several computed quantities along with the list of particles. All events are accumulated in this one file, each being separated by event header lines. This output file format follows the GSTAR text format convention described in Ref. [15]. An additional, optional output file, `mult_gen.kin`, can be activated which will write out just the list of particles with only their GEANT particle ID type, mass and kinematic information included. This is useful for analyses with PAW since this ASCII text file can be easily loaded into PAW-*ntuples*.

9 STAF Package Version - MEVSIM

The STAF version of `multiplicity_gen.f` and related files are contained in the STAR software library package under `$STAR/pams/gen/mev`. The source code is contained in the files `mevsim.F` (main program), `multiplicity_gen.F` (subroutines and functions), `hepmass.F` (HEP particle masses), `idtran.F` (GEANT to HEP particle ID translator), `iddhep.F` (ISAJET to HEP particle ID translator), `idisa.F` (HEP to ISAJET particle ID translator), `isaprt.F` (ISAJET particle decay routines), `multi_gen.inc`, `headpss.inc` and `decay.dat` (data file for particle and resonance decay information). A documentation file is in the `/doc` subdirectory. The VENUS- and VNI-like input files are in `mult_gen_venus-like.in` and `mult_gen_vini-like.in` which are essentially the same as the corresponding files for the standalone version but differ in the first few lines of input and omit the starting random number generator seed. A description of the input file for MEVSIM is provided in Appendix B. Instructions for running MEVSIM in STAF are given in Appendix C. However, due to the volatility of STAR's computing hardware and software infrastructure, these steps may or may not work in the future and should instead be regarded as general guidelines.

MEVSIM outputs the particles into the STAF table called '`particle`' which can be saved as an external `xdf` file. Header information about the event is stored in the first row of the '`particle`' table. This includes the following: (1) impact parameter, (2) reaction plane angle, (3) event generator code number where VENUS-like 200 A·GeV central Au+Au MEVSIM events are assigned code number 62 and VNI-like events are assigned number 80, (4) c.m. energy per nucleon-nucleon collision pair, (5) colliding system A(west).A(east), (6) run number, (7) event number, (8) date and (9) time of run. Recent I/O software has been developed which allows MEVSIM output (as well as that from any other STAR event generator) to be saved as PAW-*ntuples*. A log file, `mult_gen.log`, is also written out which contains input data summaries, run configuration information, and error messages, if any.

10 Summary

A fast, convenient Monte Carlo event generator, MEVSIM, has been developed for STAR simulations. With this code the user has complete control over the particle types, the multiplicities and the momentum space distributions of the particles in the simulated events. Events generated by MEVSIM are already in daily use as part of the event reconstruction quality assurance program in STAR. MEVSIM events can also be used in many other types of simulation and physics analysis projects in STAR for years to come. A standalone version and a STAF package version are both available for STAR users.

Table I: Resonances included in MEVSIM.

Resonance	ID	Mass (GeV)	Width (GeV)
ρ^+	151	0.783	0.177
ρ^-	152	0.783	0.177
ρ^0	153	0.783	0.177
ω	154	0.782	0.0084
η'	155	0.95750	0.000208
ϕ	156	1.0194	0.00442
J/ψ	157	3.09693	0.000068
Δ^-	158	1.232	0.071
Δ^0	159	1.232	0.071
Δ^+	160	1.232	0.071
Δ^{++}	161	1.232	0.071
K^{*+}	162	0.89183	0.0498
K^{*-}	163	0.89183	0.0498
K^{*0}	164	0.89610	0.0505

A Appendix

This appendix contains a description of the input for the standalone version of the Monte Carlo event generator, `multiplicity_gen.f`

```
(1) n_events - Selected number of events in run. Can be anything
               .ge. 1.
(2) n_pid_type - Number of particle ID types to include in the
                 particle list. e.g. pi(+) and pi(-) are counted
                 separately.  The limit is set by parameter npid
                 in the accompanying include file 'Parameter_values.inc'
                 and is presently set at 30.
(3) model_type - equals 1,2,3,4,5 or 6 so far.  See comments in
                 Function dNdpty to see what is calculated.
                 The models included are:
               = 1, Factorized mt exponential, Gaussian rapidity model
               = 2, Pratt non-expanding, spherical thermal source model
               = 3, Bertsch non-expanding spherical thermal source model
               = 4, Pratt spherically expanding, thermally equilibrated
                    source model.
               = 5, Factorized pt and eta distributions input bin-by-bin.
               = 6, Fully 2D pt,eta distributions input bin-by-bin.
                    NOTE: model_type = 1-4 are functions of (pt,y)
```

```
                          model_type = 5,6 are functions of (pt,eta)
 (4) reac_plane_cntrl - Can be either 1,2,3 or 4 where:
                      = 1 to ignore reaction plane and anisotropic flow,
                          all distributions will be azimuthally symm.
                      = 2 to use a fixed reaction plane angle for all
                          events in the run.
                      = 3 to assume a randomly varying reaction plane
                          angle for each event as determined by a
                          Gaussian distribution.
                      = 4 to assume a randomly varying reaction plane
                          for each event in the run as determined by
                          a uniform distribution from 0 to 360 deg.
 (5) PSIr_mean, PSIr_stdev - Reaction plane angle mean and Gaussian
                             std.dev. (both are in degrees) for cases
                             with reac_plane_cntrl = 2 (use mean value)
                             and 3.  Note: these are read in regardless
                             of the value of reac_plane_cntrl.
 (6) MultFac_mean, MultFac_stdev - Overall multiplicity scaling factor
                                   for all PID types; mean and std.dev.;
                                   for trigger fluctuations event-to-evt.
 (7) pt_cut_min,pt_cut_max - Range of transverse momentum in GeV/c.
 (8) eta_cut_min,eta_cut_max - Pseudorapidity range
 (9) phi_cut_min,phi_cut_max - Azimuthal angular range in degrees.
(10) n_stdev_mult - Number of standard deviations about the mean value
                    of multiplicity to include in the random event-to-
                    event selection process.  The maximum number of
                    steps that can be covered is determined by
                    parameter n_mult_max_steps in the accompanying
                    include file 'Parameter_values.inc' which is
                    presently set at 1000, but the true upper limit for
                    this is n_mult_max_steps - 1 = 999.
(11) n_stdev_temp - Same, except for the "Temperature" parameter.
(12) n_stdev_sigma- Same, except for the rapidity width parameter.
(13) n_stdev_expvel - Same, except for the expansion velocity parameter.
(14) n_stdev_PSIr   - Same, except for the reaction plane angle
(15) n_stdev_Vn     - Same, except for the anisotropy coefficients, Vn.
(16) n_stdev_MultFac - Same, except for the multiplicity scaling factor.
(17) n_integ_pts - Number of mesh points to use in the random model
                   parameter selection process.  The upper limit is
                   set by parameter nmax_integ in the accompanying
                   include file 'Parameter_values.inc' which is presently
                   set at 100, but the true upper limit for n_integ_pts
                   is nmax_integ - 1 = 99.
```

```
   (18) n_scan_pts  - Number of mesh points to use to scan the (pt,y)
                      dependence of the model distributions looking for
                      the maximum value.  The 2-D grid has
                      n_scan_pts * n_scan_pts points; no limit to size of
                      n_scan_pts.
   (19) irand       - Starting random number seed.

**************************************************************************
    FOR MODEL_TYPE = 1,2,3 or 4:
    Input the following 17 lines for each particle type; repeat these
    set of lines n_pid_type times:

         (a) gpid - Geant Particle ID code number
         (b) mult_mean,mult_variance_control - Mean multiplicity and
                                               variance control where:
             mult_variance_control = 0 for no variance in multiplicity
             mult_variance_control = 1 to allow Poisson distribution for
                                       particle multiplicities for all events.
             Note that a hard limit exists for the maximum possible
             multiplicity for a given particle type per event.  This is
             determined by parameter factorial_max in accompanying include
             file 'common_facfac.inc' and is presently set at 10000.
         (c) Temp_mean, Temp_stdev - Temperature parameter mean (in GeV)
             and standard deviation (Gaussian distribution assumed).
         (d) sigma_mean, sigma_stdev - Rapidity distribution width (sigma)
             parameter mean and standard deviation (Gaussian distribution
             assumed).
         (e) expvel_mean, expvel_stdev - S. Pratt expansion velocity
             (in units of c) mean and standard deviation (Gaussian
             distribution assumed).
         (f) Vn_mean(k);  k=1,4  - Anisotropic flow parameters, mean values
                                   for Fourier component n=1.
         (g) Vn_stdev(k); k=1,4  - Anisotropic flow parameters, std.dev.
                                   values for Fourier component n=1.

             Repeat the last two lines of input for remaining Fourier
             components n=2,3...6.  Include all 6 sets of parameters
             even if these are not used by the model for Vn(pt,y) (set
             unused parameter means and std.dev. to 0.0).  List 4 values
             on every line, even though for n=even the 4th quantity is
             not used.

**************************************************************************
```

```
    FOR MODEL_TYPE = 5 input the following set of lines for each particle
                       type; repeat these n_pid_type times.

         (a) gpid - Geant Particle ID code number
         (b) mult_mean,mult_variance_control - Mean multiplicity and
                                               variance control where:
             mult_variance_control = 0 for no variance in multiplicity
             mult_variance_control = 1 to allow Poisson distribution for
                                       particle multiplicities for all events.
         (c) pt_start, eta_start - minimum starting values for pt, eta
                                   input for the bin-by-bin distributions.
         (d) n_pt_bins, n_eta_bins - # input pt and eta bins.
         (e) delta_pt, pt_bin - pt bin size and function value, repeat for
                                each pt bin.
         (f) delta_eta, eta_bin - eta bin size and function value, repeat
                                  for each eta bin.
         (g) Vn_mean(k);  k=1,4  - Anisotropic flow parameters, mean values
                                   for Fourier component n=1.
         (h) Vn_stdev(k); k=1,4  - Anisotropic flow parameters, std.dev.
                                   values for Fourier component n=1.

             Repeat the last two lines of input for remaining Fourier
             components n=2,3...6.  Include all 6 sets of parameters
             even if these are not used by the model for Vn(pt,y) (set
             unused parameter means and std.dev. to 0.0).  List 4 values
             on every line, even though for n=even the 4th quantity is
             not used.

         NOTE: The pt, eta ranges must fully include the requested ranges
               in input #4 and 5 above; else the code execution will stop.

         Also, variable bin sizes are permitted for the input distributions.

         Also, this input distribution is used for all events in the run;
         no fluctuations in this "parent" distribution are allowed from
         event-to-event.

**************************************************************************
    FOR MODEL_TYPE = 6 input the following set of lines for each particle
                       type; repeat these n_pid_type times.

         (a) gpid - Geant Particle ID code number
         (b) mult_mean,mult_variance_control - Mean multiplicity and
```

```
                                        variance control where:
    mult_variance_control = 0 for no variance in multiplicity
    mult_variance_control = 1 to allow Poisson distribution for
                              particle multiplicities for all events.
(c) pt_start, eta_start - minimum starting values for pt, eta
                          input for the bin-by-bin distributions.
(d) n_pt_bins, n_eta_bins - # input pt and eta bins.
(e) delta_pt - pt bin size, repeat for each pt bin.
(f) delta_eta - eta bin size, repeat for each eta bin.
(g) i,j,pt_eta_bin(i,j) - read pt (index = i) and eta (index = j)
                          bin numbers and bin value for full 2D space.
(h) Vn_mean(k);  k=1,4  - Anisotropic flow parameters, mean values
                          for Fourier component n=1.
(i) Vn_stdev(k); k=1,4  - Anisotropic flow parameters, std.dev.
                          values for Fourier component n=1.

    Repeat the last two lines of input for remaining Fourier
    components n=2,3...6.  Include all 6 sets of parameters
    even if these are not used by the model for Vn(pt,y) (set
    unused parameter means and std.dev. to 0.0).  List 4 values
    on every line, even though for n=even the 4th quantity is
    not used.

NOTE: The pt, eta ranges must fully include the requested ranges
      in input #4 and 5 above; else the code execution will stop.

Also, variable bin sizes are permitted for the input distributions.

Also, this input distribution is used for all events in the run;
no fluctuations in this "parent" distribution are allowed from
event-to-event.
```

B Appendix

This appendix contains a description of the input for the STAF package version, MEVSIM.

```
(1) VSSHEP(1) - Run number
(2) VSSHEP(2) - Starting number for the events
(3) PSSHEP(3) - Generator code number where '62' is for VENUS-like
```

```
                Au+Au central 200 AGeV collisions and '80' is for
                VNI-like Au+Au central 200 AGeV collisions.
(4) PSSHEP(4) - Energy of nucleon-nucleon colliding pairs in the
                center-of-momentum system (GeV)
(5) PSSHEP(5) - A(west).A(east) for the colliding systems, e.g.
                for Au+Au this is 197.197
(6) CNOF      - Optional decay data file (Y or N), where
                Y means no user defined decay data file to be read in,
                  default decay parameters in code are used.
                N means read in user defined decay data file and
                  overwrite default parameters in code.
(7) CPI0      - pi0 decay option flag (Y or N), where Y means code
                should decay pi0, N means that pi0 are not decayed
                by MEVSIM.
(8,9...)      - Repeat the decay flags for remaining resonances.  Refer
                to mult_gen_venus-like.in for up-to-date list of
                available resonances which can be decayed.
                Resonances not included in this list cannot be decayed
                by MEVSIM.  The decay flag applies to all charge states
                of a resonance, e.g. decay rho+, rho- and rho0 or do not
                decay the rhos, but cannot choose to decay the rho+ and
                not the rho-, for example.

Resume input as in Appendix A, starting with:

(2) n_pid_type - Number of particle ID types to include in the
                 particle list. e.g. pi(+) and pi(-) are counted
                 separately.  The limit is set by parameter npid
                 in the accompanying include file 'Parameter_values.inc'
                 and is presently set at 30.

   etc., through ...

(18) n_scan_pts  - Number of mesh points to use to scan the (pt,y)
                   dependence of the model distributions looking for
                   the maximum value.  The 2-D grid has
                   n_scan_pts * n_scan_pts points; no limit to size of
                   n_scan_pts.

but do not input the random number generator seed.  Then continue with
the input parameters for each selected particle ID type as in
Appendix A.
```

C Appendix

Guidelines for running the STAF package, MEVSIM, are listed here. The following are explicit step-by-step commands to enter starting from your home computer. Given the volatility of STAR's computing hardware and software infrastructure however, there is, unfortunately, no guarantee that these exact steps will always work in the future.

- ssh rsun00.rhic.bnl.gov (Secure shell connection to rsun00, or other RCF processor with working version of STAF)
- stardev (set STAR Library Level to 'dev' or other)
- mkdir testmev (create a working area)
- klog (AFS login, followed by your AFS password)
- cvs co pams/gen/mev (CVS checkout of MEVSIM package)
- cd pams/gen/mev
- makes (compile and link)
- cd ../../..
- cp ~longacre/SL99e/writxdf.kumac . (STAF kumac to run MEVSIM, fill the STAF table '`particle`,' and write out an `xdf` file '`genoutput.xdf`'.)
- cp ~longacre/SL99e/mult_gen.in . (Generic MEVSIM input file, edit for your specific case, filename must remain '`mult_gen.in`.')
- cp ~longacre/SL99e/mess.kumac . (kumac to run after MEVSIM which reads the `xdf` file and produces some plots.)
- (Remove or rename log files and output files from previous runs, if any.)
- staf
- exec writxdf n (where 'n' is the number of events to generate)
- quit

References

[1] H. Baer, F. E. Paige, S. D. Protopopescu and X. Tata, "ISAJET 7.48 - A Monte Carlo Event Generator for pp, $\bar{p}p$, and e^+e^- Reactions," Brookhaven National Laboratory preprint BNL-HET-99/43, eprint archive hep-ph/0001086 (1999), unpublished.

[2] T. J. Humanic, Phys. Rev. C **50**, 2525 (1994).

[3] S. Pratt, Phys. Rev. C **49**, 2722 (1994).

[4] M. Herrmann and G. F. Bertsch, Phys. Rev. C **51**, 328 (1995).

[5] S. Pratt, Phys. Rev. D **33**, 1314 (1986).

[6] GEANT manual, CONS 300-1,2.

[7] Particle Data Group, Phys. Lett. B **239**, 1-516 (1990).

[8] F. E. James and H. L. Kraybill, Phys. Rev. **142**, 896 (1966).

[9] V. M. Aulchenko *et al.,* Phys. Lett. B **186**, 432 (1987).

[10] J. C. Bizot *et al.,* Phys. Lett. B **32**, 416 (1970).

[11] B. H. Bransden and R. G. Moorhouse, *The Pion-Nucleon System*, (Princeton University Press, Princeton, NJ, 1973), pgs. 53-54.

[12] D. Aston *et al.,* Nucl. Phys. B **296**, 493 (1988); Zhangbu Xu, private communication.

[13] A. M. Poskanzer and S. A. Voloshin, Phys. Rev. C **58**, 1671 (1998).

[14] A. M. Poskanzer, private communication.

[15] P. Jacobs and D. Irmscher, "GSTAR: A Geant-based Detector Simulation Chain for STAR," STAR Note 235 (1996).