A Monte Carlo Muon Generator for Cosmic-Ray Muon Applications

G. Bonomi,[1,2] A. Donzella,[1,2] D. Pagano,[1,2] A. Zenoni,[1,2] G. Zumerle,[3,4] and N. Zurlo[2,5]

[1]*Department of Mechanical and Industrial Engineering, University of Brescia, Italy*
[2]*Istituto Nazionale di Fisica Nucleare (INFN), Pavia, Italy*
[3]*Department of Physics and Astronomy, University of Padova, Padova, Italy*
[4]*Istituto Nazionale di Fisica Nucleare (INFN), Padova, Italy*
[5]*Department of Civil, Environmental, Architectural Engineering and Mathematics, University of Brescia, Italy*

Corresponding author: G. Bonomi
Email: germano.bonomi@cern.ch

Abstract

Cosmic-ray radiation, thanks to its high penetration capability and relative abundance, has been successfully used in scientific research and civil applications for a long time. For example, techniques based on the attenuation of cosmic-ray muons (transmission muography) or on their angular scattering (scattering muography) have been used to study the inner structure of volcanoes, to search for hidden chambers in Egyptian pyramids, inspect nuclear waste containers, and monitor blast furnaces. In addition to these imaging techniques, cosmic-ray muons have also been used for the detector alignment in large experiments in nuclear and elementary particle physics and, more recently, proposed for the alignment and stability monitoring of mechanical structures. Transmission muography applications are sensitive to the angular distribution of cosmic muons, and many applications of scattering muography are also sensitive to their momentum distribution. For these reasons, an accurate simulation of the dependency of the muon flux on momentum and direction is a key requirement for every generation tool targeting such applications. Moreover, as the inspection of large structures requires a large number of cosmic-ray muons, the generator has also to be fast. A new Monte Carlo generator of cosmic-ray muons, called EcoMug (Efficient COsmic MUon Generator) and specifically designed for transmission and scattering applications, is presented. It is a header-only C++11 library, based on a specific parametrization of experimental data, but is easily replaceable by the user with a custom one. Unlike other tools, EcoMug gives the possibility of originating the cosmic-ray muons from different surfaces (plane, cylinder, and half-sphere), while keeping the correct angular and momentum distribution of generated tracks. The main features of EcoMug, its mathematical foundations, and applications to selected study cases are presented.

Keywords: cosmic rays, muon generator, muography
DOI: 10.31526/JAIS.2022.290

1. INTRODUCTION

Even though the first applications of cosmic-ray muons were implemented in the second half of the 20th century, in the last two decades, they became more and more popular. A description of the techniques and an overview of the possible applications can be found in [1] and references therein. To summarize, the applications can be divided into three main categories: transmission muography, scattering muography, and muon metrology. The first one (*transmission muography*, occasionally also referred to as *muon radiography*) relies on the measurement of attenuation of the cosmic-ray flux crossing a given material to infer the composition of the volume under investigation. The second one (*scattering muography* sometimes also referred to as *muon tomography*) exploits the fact that muons are subject to angular scattering when traversing a given object: measuring the overall deviation of the trajectory, it is possible to gain information about the crossed material. The third one (*muon metrology*) is an extension of the technique used for years in particle physics to align detectors with interposed materials, using the fact that, on average, muons are straight lines crossing the detectors themselves.

These applications are different in many ways, for example, for the usage of the detectors and for the type of reconstruction algorithms; nevertheless, they basically all rely on simulation tools for the generation of the cosmic-ray muons and their tracking. Monte Carlo studies are of extreme importance to designing a particular experiment and also to interpreting the results of the analysis. One of the main reasons is the fact that, in general, in real life, the momentum of the cosmic-ray muons cannot be measured while it is obviously known in the MC. While, in many cases, the tracking toolkit is either GEANT4 [2] or FLUKA [3],[1] for the

[1]For experiments dealing with extremely large targets, MUSIC [4] and PUMAS [5] are also popular.

generation of the cosmic-ray muons, instead, there is not such an agreement on the tools to be used. Many research groups use custom-made software, based on specific parametrizations and approximations of the flux of cosmic-ray muons, whereas others rely on dedicated packages. Among them, we can perform the following simplified classification: cosmic-ray air shower (CRAS) generators, simulating the full cascade of secondary particles initiated by primary cosmic rays; parametric generators, using a parametrization of the flux of muons, based either on experimental data or on the results from simulations with CRAS generators; special generators, specifically designed for underground, high altitude, or underwater experiments.

Among the first category, it is worth citing CORSIKA [6], probably the most used, CRY [7], and MCEq [8]. Concerning parametric generators, examples are given by GEMC [9] and CMSCGEN [10]. Among the special generators, the most used are muTeV [11] and MUPAGE [12].

As stated previously, even if some dedicated packages are available, many groups rely on custom-made codes. Concerning the specific requirements of the generators, transmission muography applications are sensitive to the angular distribution of muons, and many applications of scattering muography are also sensitive to their momentum distribution. For these reasons, an accurate simulation of the dependency of the muon flux on momentum and direction is a key requirement for every generation tool targeting such applications. Moreover, as the inspection of large structures often requires a large number of cosmic-ray muons, the generator has also to be fast.

Our research group at the University of Brescia is involved in muon applications for a couple of decades and thus has developed its own cosmic-ray generator. Actually, the first versions of the code were used in nuclear physics experiments such as FINUDA at Laboratori Nazionali di Frascati (LNF), ATHENA, and AEgIS at CERN. It was then improved and applied in two European Projects (MuSteel [13] and MuBlast [14]) and in feasibility studies of new applications (see [15, 16]) gaining new features and capabilities. In 2021, all the developments were finally packed into a new standalone toolkit to make it available for everyone that would feel the need. The new generator has been called EcoMug, an acronym for *An Efficient COsmic MUon Generator*, and it is described in detail in [17]. The code is available at https://github.com/dr4kan/EcoMug under the GPL-3.0 license.

In the following, the mathematical foundations of the toolkit, the capability to generate over a flat sky, a cylinder, or a hemisphere, possible cases of use, and examples on how to integrate EcoMug into Geant4 will be presented.

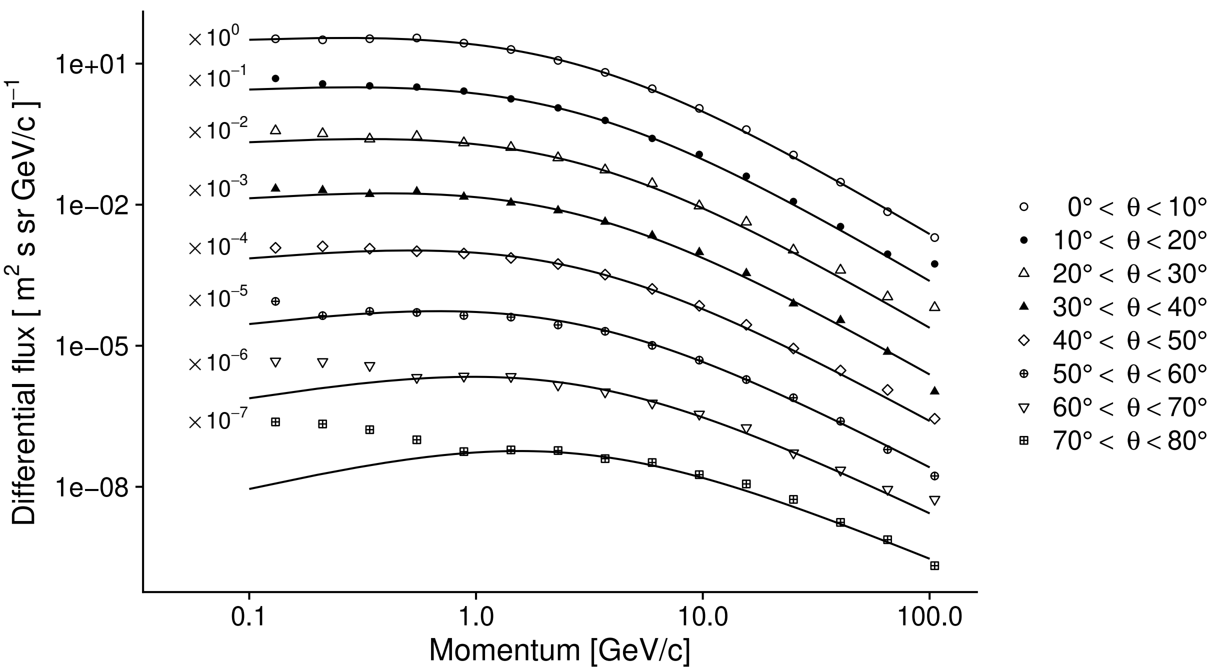

FIGURE 1: Experimental differential flux, as a function of the momentum and for eight zenith angle intervals (data from [18]). Superimposed curves are the predictions from the parametrization used in EcoMug.

2. DATA PARAMETRIZATION

EcoMug is a parametric generator and it is based on the differential flux parametrization of the experimental data collected by the ADAMO detector and reported in [18].[2] Clearly, each generator aims at having the *best* distributions of momentum and arrival

[2]ADAMO was a magnetic spectrometer that had been developed to allow a precise measurement of the spectra of the main cosmic-ray charged components at ground level. The data reported in [18] were collected during a campaign of measurements in 2003 and 2004.

direction of the muons, being as close as possible to the *real* one. This is quite difficult, if not impossible, since the cosmic-ray muon flux depends on many parameters such as the altitude, the atmospheric pressure and temperature, and the solar activity. The choice of the ADAMO detector data in EcoMug was based on the fact that such study reported the flux dependencies on the momentum and on the θ angle with respect to the vertical direction and also because a parametrization of the flux was already available, being worked out in L. Bonechi Ph.D. Thesis [19]. The data, along with the result of the parametrization, are reported in Figure 1. EcoMug generates muons with momentum up to 1000 GeV/c and zenith angles up to 90°. The user should keep in mind that the generator cannot describe accurately the flux outside the range of experimental data. It is also clear from Figure 1 that at low momenta and large zenith angle, the parametrization is not able to reproduce the data behavior. This is most probably due to a significant contribution of electrons in such region, which is not taken into account by the generator.

Assuming the reference system of Figure 2, the ADAMO detector data were parametrized as follows:

$$J \equiv J(t,p,\theta,\phi) = \frac{dN}{dt \cdot dp \cdot \sin\theta \cdot d\theta \cdot d\phi \cdot dS_n} = f(p) \cdot (\cos\theta)^n \cdot \frac{1}{\mathrm{m^2 \cdot s \cdot sr \cdot GeV/c}}, \tag{1}$$

where $f(p)$ and n are functions of p as

$$\begin{aligned} f(p) &= \left[1600 \cdot \left(\frac{p}{p_0} + 2.68\right)^{-3.175} \cdot \left(\frac{p}{p_0}\right)^{0.279}\right], \\ n(p) &= \max\left[0.1,\ 2.856 - 0.655 \cdot \ln\left(\frac{p}{p_0}\right)\right] \end{aligned} \tag{2}$$

with $p_0 = 1\,\mathrm{GeV/c}$ and $p > 0.040\,\mathrm{GeV/c}$.

The J function represents the number of cosmic-ray muons crossing a surface dS_n perpendicular to the muon direction, in an interval of time dt, with momentum between p and $p + dp$, within a $d\Omega$ solid angle, being $d\Omega = \sin\theta\, d\theta\, d\phi$, around the direction ($\theta$, ϕ). It is worth stressing the fact that the presence of the dS_n term is due to the way the data were collected. Indeed, the ADAMO detector measured the flux of cosmic-ray muons at different incoming angles, rotating the apparatus and counting the particles reaching the detector perpendicularly.

Dealing with algorithms in a computing code, it is actually easier and more efficient to use a modified differential function, which we call J', expressing the number of particles crossing a horizontal surface element dS_n in the time dt, with momentum between p and $p + dp$, zenith angle between θ and $\theta + d\theta$, and azimuth angle between ϕ and $\phi + d\phi$:

$$J' = \frac{dN}{dt \cdot dp \cdot d\theta \cdot d\phi \cdot dS_n} = J(t,p,\theta,\phi) \cdot sin\theta = f(p) \cdot (\cos\theta)^n \cdot sin(\theta) \cdot \frac{1}{\mathrm{m^2 \cdot s \cdot sr \cdot GeV/c}}. \tag{3}$$

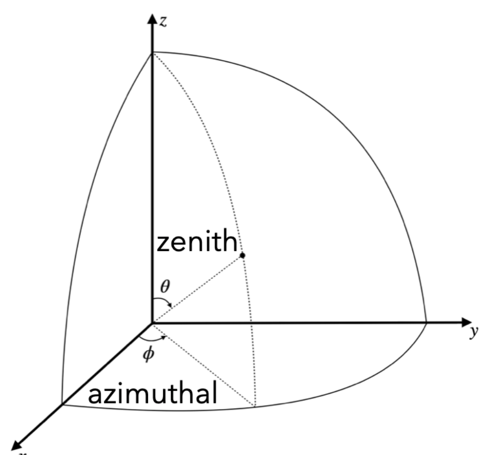

FIGURE 2: Definition of the coordinate system used in EcoMug.

3. GENERATION OVER DIFFERENT SURFACES

All the available cosmic-ray generators, to our knowledge, reproduce the arrival spatial and energy distributions over a flat surface. When dealing with high structures, such as a blast furnace, this is computationally cumbersome, since most of the generated cosmic rays, in the end, do not intercept the object under study and are useless. For this reason, in the MuBlast project [14], we developed an algorithm to generate also over a cylindrical surface. For other applications, such as the AEgIS experiment at CERN, it proved to be very useful to generate over a hemispherical surface. EcoMug [17] incorporated such capabilities and it is indeed able to generate over a horizontal surface (*flat-sky generation*), a cylinder lateral surface (*cylindrical generation*), and a hemispherical surface (*half-spherical generation*). In the following, the three cases will be presented separately.

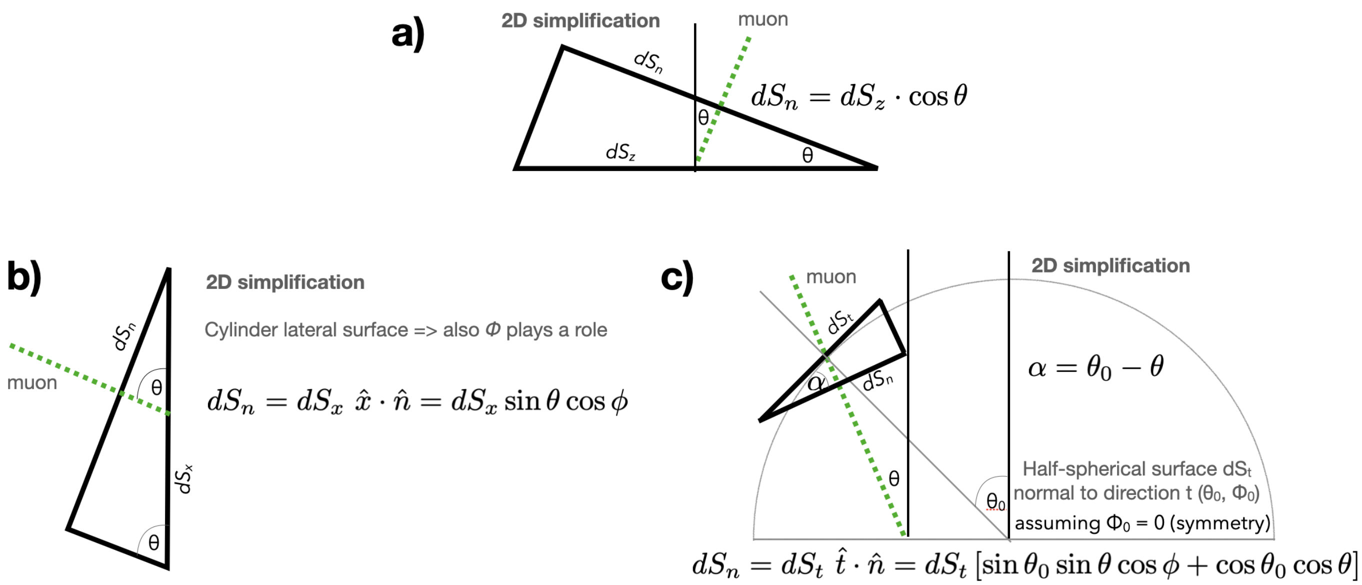

FIGURE 3: Geometrical correlation between the infinitesimal surface dS_n, perpendicular to the muon direction, and a horizontal, vertical, and over a hemisphere infinitesimal surfaces.

3.1. *Generation over a Flat Sky*

Taking into account the relation between an infinitesimal horizontal surface dS_z and dS_n, as represented in Figure 3(a), the J'_z differential function to generate over a flat-sky surface can be parametrized as follows:

$$J'_z \equiv \frac{dN}{dt \cdot dp \cdot d\theta \cdot d\phi \cdot dS_z} = f(p) \cdot (\cos\theta)^{n+1} \cdot \sin\theta \cdot \frac{1}{\mathrm{m}^2 \cdot \mathrm{s} \cdot \mathrm{sr} \cdot \mathrm{GeV/c}}. \tag{4}$$

In the flat-sky generation, the origin position of generated cosmic-ray muons is uniformly distributed over the generation surface, the azimuth angle ϕ is uniformly randomly generated in $[0, 2\pi]$ (or in a subset if requested by the user), and the momentum and zenith angle θ are sampled from the differential flux in equation (4). From a technical point of view, this sampling makes use of both the inverse transform sampling and acceptance-rejection methods [20],[3] as J'_z is not invertible. Indeed, even though J'_z depends on p because of the product $[(p/p_0) + 2.68]^{-3.175} \cdot (p/p_0)^{0.279}$, the first factor is a much quicker changing function than the second one. Therefore, p is initially sampled from $(p + 2.68)^{-3.175}$, using the inverse transform, and then the acceptance-rejection method is applied to the remaining terms of J'_z: in this way, the rejection rate of the method is significantly reduced. This strategy is also applied to the other generation methods.

3.2. *Generation over a Cylinder*

Let us consider a cylinder whose symmetry axis coincides with the z-axis of the reference frame in Figure 2. Let dS_x be a surface element on the cylinder orthogonal to the x-axis and dS_n its projection on a plane perpendicular to the muon direction. Because of the symmetry in ϕ, for a cylindrical surface, we can choose the surface element orthogonal to the x-axis without loss of generality. Taking into account the relation between the infinitesimal vertical surface dS_x and dS_n, as represented in Figure 3(b), the J'_x differential function to generate over a lateral cylinder surface can be parametrized as follows:

$$J'_x \equiv \frac{dN}{dt \cdot dp \cdot d\theta \cdot d\phi \cdot dS_x} = f(p) \cdot (\cos\theta)^{n} (\sin\theta)^2 \cos\phi \cdot \frac{1}{\mathrm{m}^2 \cdot \mathrm{s} \cdot \mathrm{sr} \cdot \mathrm{GeV/c}}. \tag{5}$$

The origin position of the muons is uniformly randomly chosen on the cylindrical generation surface, whereas the momentum, zenith angle θ, and azimuth angle ϕ are sampled from the differential flux J'_x in equation (5). By noting that equation (5) only correlates p and θ, by means of the n term, we can independently generate ϕ, uniformly in $\cos\phi$, and then only sample the other two variables from J'_x, without the $\cos\phi$ term. In this way, the acceptance-rejection method is only applied to the momentum and zenith angle, as in the flat-sky case.

[3] The acceptance-rejection method is an algorithm for drawing a sample from an N-dimension target pdf, by performing a uniformly random sampling of the $(N + 1)$-dimension Cartesian graph and by keeping only those values in the region under the graph of the target pdf.

3.3. Generation over a Hemisphere

Always with reference to the coordinate system in Figure 2, let us consider a half-sphere with the base laying on the x-y plane and centered on the origin. Let dS_t be a generic surface element on a half-sphere, orthogonal to the direction (θ_0, ϕ_0), and dS_n its projection on a plane perpendicular to the muon direction. In analogy to what has been done for the cylindrical case, we can choose $\phi_0 = 0$ without loss of generality, because of the symmetry in ϕ of the half-sphere. Taking into account the relation between the infinitesimal horizontal surface dS_t and dS_n, as represented in Figure 3(c), the J_t' differential function to generate over a hemispherical surface can be parametrized as follows:

$$J_t' \equiv \frac{dN}{dt \cdot dp \cdot d\theta \cdot d\phi \cdot dS_t} = f(p) \cdot (\cos\theta)^n \left[\sin\theta_0 (\sin\theta)^2 \cos\phi + \cos\theta_0 \cos\theta \sin\theta\right] \cdot \frac{1}{\mathrm{m}^2 \cdot \mathrm{s} \cdot \mathrm{sr} \cdot \mathrm{GeV/c}}. \tag{6}$$

As expected, if $\theta_0 = 0$, equation (6) becomes equal to the one for the flat sky (see equation (4)), whereas if $\theta_0 = \pi/2$, it becomes equal to the one for the cylindrical case (see equation (5)).

Equation (6) correlates not only p, θ, and ϕ but also θ_0, which, together with ϕ_0, defines the origin position of the muon on the half-sphere. For this reason, the sampling of all these variables from J_t' involves the use of the acceptance-rejection method in 4D space, resulting in a worsening of the generation speed with respect to the other methods.

4. COMPARISON OF THE GENERATION MODES

The generation methods discussed above, and implemented in EcoMug, are mathematically equivalent, provided that all of them grant the proper coverage of the geometrical acceptance of the detection system. However, depending on the case study, one method could be more effective than the others, with respect to the generation time. A full comparison of the methods in different conditions is described in [17]. In general, if the structure under investigation is a vertical one, in which the height is bigger than the width, the cylindrical generation is surely more appropriate. Assuming two $1,25 \times 2,50\,\mathrm{m}^2$ vertical muon detectors placed at a couple of meters of distance, the cylindrical generation is 4 to 5 times more efficient than the half-spherical one and even more if compared to the flat-sky one. On the other hand, in the case of a telescopic detector consisting of three horizontal detection layers ($1,00 \times 1,00\,\mathrm{m}^2$ placed at a distance of 50 cm), the flat-sky generation results 4 to 5 times more efficient than the half-spherical one, being the cylindrical generation deeply unfavorable for geometrical reasons. When dealing with a more complex detector, consisting of eight sensitive layers arranged in an octagonal configuration, inspired by inner tracking detectors of some experiments at colliders, then the half-spherical generation proved to be the best one. First of all, it is the only one that guarantees full coverage of the geometrical acceptance. Indeed, in general, the cylindrical generation would miss the almost vertical muons, while the flat-sky generation would lack the almost horizontal muons. To compensate for such limits in the geometrical acceptance, one should increase the dimensions of the cylinder or of the flat sky. In any case, the half-spherical generation, being the radius and length of the detector chosen, respectively, equal to 1 m and 4 m, results 3 to 4 times more efficient than the cylindrical or flat-sky generations.

In terms of CPU times, the three generation modes are also different. The time required to generate 1 million muons, using a compiled code with the $-$O3 flag on a 2.6 GHz Intel Core i7 6 core, was on average 1.3 s for the flat-sky generator, 2.6 s for the cylindrical generator, and 4.9 s for the half-sphere generator. It is worthwhile mentioning that since the number of cosmic-rays is large in these Monte Carlo simulations, a very fast and reliable pseudo-random number generator (PRNG) is useful. EcoMug internally uses a class called EMRandom, which is based on the xoroshiro128+ algorithm [21].

5. IMPLEMENTATION IN C++ APPLICATIONS

EcoMug has been designed as a header-only C++11 library. In other words, the full definitions of classes and functions are contained in a single header file. The integration of EcoMug is thus extremely easy in any C++ project. It is enough to include the header file into the source code and point the compiler at the location of the EcoMug.h file.

The use of the library requires the initialization of the *EcoMug* class, the choice of the generation method, and the definition of the size and position of the generation surface, as in Listings 1, 2, and 3.

```
EcoMug gen; // initialization of the class
gen.SetUseSky(); // plane surface generation
gen.SetSkySize({{10., 10.}}); // x and y size of the plane
// (x,y,z) position of the center of the plane
gen.SetSkyCenterPosition({{0., 0., 20.}});
```

Listing 1: EcoMug setup for a flat surface generation.

```
EcoMug gen; // initialization of the class
gen.SetUseCylinder(); // cylindrical surface generation
gen.SetCylinderRadius(10.); // cylinder radius
gen.SetCylinderHeight(30.); // cylinder height
```

```
// (x,y,z) position of the center of the cylinder
gen.SetCylinderCenterPosition({{0., 0., 15.}});
```

Listing 2: EcoMug setup for a cylindrical surface generation.

```
EcoMug gen; // initialization of the class
gen.SetUseHSphere(); // half-spherical surface generation
gen.SetHSphereRadius(30.); // half-sphere radius
// (x,y,z) position of the center of the half-sphere
gen.SetHSphereCenterPosition({{0., 0., 0.}});
```

Listing 3: EcoMug setup for a half-spherical surface generation.

Once the setup of the instance of the *EcoMug* class is done, the generation of a cosmic-ray muon can be implemented as shown here below in Listing 4.

```
// Setup of the instance of the EcoMug class
// as in the example code 1
EcoMug gen;
gen.SetUseSky();
gen.SetSkySize({{10., 10.}});
gen.SetSkyCenterPosition({{0., 0., 20.}});

// The array storing muon generation position
std::array<double, 3> muon_position;

// Loop to generate 1000 cosmic-ray muons
for (auto event = 0; event < 1000; ++event) {
    gen.Generate(); // generate a cosmic-ray muons
    muon_position = gen.GetGenerationPosition();
    double muon_p = gen.GetGenerationMomentum();
    double muon_theta = gen.GetGenerationTheta();
    double muon_phi = gen.GetGenerationPhi();
    double muon_charge = gen.GetCharge();
    ...
}
```

Listing 4: Accessing position, direction, momentum, and charge of generated cosmic-ray muons in EcoMug.

It is worth reminding that angles are in radians, momentum is in GeV/c, while the unit of measure of the position is arbitrary and depends on the choice done in the simulation code. A more detailed description of how to integrate EcoMug in a Geant4 application is presented in Appendix A of [17].

EcoMug also provides the possibility of using a custom function for J, in case the users consider that the default parametrization (see equation (1)) does not provide an accurate description of the differential flux of cosmic-ray muons in the conditions of their experiments. The example code reported here below (Listing 5) shows how to define a custom J and how to pass it to EcoMug, through the method *SetDifferentialFlux*. It is important to note here that the custom J, given the way in which EcoMug handles the generation, must represent the number of cosmic-ray muons crossing a surface dS_n *perpendicular to the muon direction*, in an interval of time dt, with momentum between p and $p + dp$, within a $d\Omega$ solid angle, being $d\Omega = \sin\theta \, d\theta \, d\phi$, around the direction (θ, ϕ).

```
double J(double p, double theta) {
    double A = 0.14*pow(p, -2.7);
    double B = 1. / (1. + 1.1*p*cos(theta)/115.);
    double C = 0.054 / (1. + 1.1*p*cos(theta)/850.);
    return A*(B+C);
}
EcoMug genA; // to generate the cosmic-ray muons according to the EcoMug default parametrization
EcoMug genB; // to generate the cosmic-ray muons according to a custom parametrization
// genB can indeed also be used to generate a background noise

genA.SetUseSky();                           genB.SetUseSky();
genA.SetSkySize({{x, y}});                  genB.SetSkySize({{x, y}});
genA.SetSkyCenterPosition({0., 0., z});     genB.SetSkyCenterPosition({0., 0., z});
genA.SetMinimumMomentum(150);               genB.SetMinimumMomentum(150);

genB.SetDifferentialFlux(&J);
```

```
    for (auto event = 0; event < nevents; ++event) {
        genA.Generate(); // generate from default J
        genB.GenerateFromCustomJ(); // generate from user-defined J
        // the user, here below, can choose to use genA or genB, according to a given fraction,
        // to retrieve the muon data (position, momentum and angular variables)
        ...
}
```

Listing 5: Using a custom J in EcoMug.

In the previous code, the *EcoMug* class is used to generate according to the default J (see equation (1)) and to a Gaisser-like parametrization [22] user-defined J. The use of a custom definition for J requires a function of both momentum and θ to be passed to the generator by means of the method *SetDifferentialFlux*. Afterwards, the user can invoke the generation of a muon, according to the specified J, with the method *GenerateFromCustomJ*. The above example code can actually be used to generate both the signal, the cosmic-ray muons through the default J, and a user-defined background, through the custom J, according to a given user-defined ratio. In some experimental conditions, indeed, it may be useful to also generate a background, for examples coming from electrons or other particles mimicking the muons. If implemented in Geant4 applications, to do that, the user should remind to change the mass of the generated particle using the function *fParticleGun->SetParticleDefinition(...)*.

6. CONCLUSIONS

A new Monte Carlo generator of cosmic-ray muons, called EcoMug, has been presented. EcoMug gives the possibility of originating the cosmic-ray muons from different surfaces (plane, cylinder, and half-sphere) keeping the correct angular and momentum distribution of generated tracks. It is based on a real data parametrization [19] of the ADAMO detector data [18]. For those cases where the proposed parametrization does not provide an accurate description of the differential flux of muons, a custom definition can be used. EcoMug has a fast and optimized code, and it offers the possibility of restricting the angular variables and the momentum of muons at the generation level. EcoMug is a header-only C++11 library, consisting of a single .h file, and it is freely available under the GPL-3.0 license at https://github.com/dr4kan/EcoMug. To summarize, EcoMug allows for a very efficient generation of cosmic-ray muons for those applications requiring high statistics, as typical transmission muography and tomography applications do.

CONFLICTS OF INTEREST

The authors declare that there are no conflicts of interest regarding the publication of this paper.

ACKNOWLEDGMENTS

The authors acknowledge the support from the National Institute of Nuclear Physics (INFN) and from the universities they belong to. Part of the work was developed within the Mu-Blast project, funded by the Research Fund for Coal and Steel of the European Union (RFSR-CT-2014-00027). They wish also to thank Marwa Al Moussawi for demonstrating that the EcoMug code can also be used to generate a user-defined background.

References

[1] G. Bonomi, P. Checchia, M. D'Errico, D. Pagano and G. Saracino, *Applications of cosmic-ray muons*, Progress in Particle and Nuclear Physics **112**, 103768 (2020).
[2] S. Agostinelli et al., *Geant4—A simulation toolkit*, Nucl. Instrum. Methods Phys. Res. **A 506** (3) 250–303 (2003).
[3] T. Böhlen, F. Cerutti et al., *The FLUKA code: Developments and challenges for high energy and medical applications*, Nucl. Data Sheets **120** 211–214 (2014).
[4] V. A. Kudryavtsev, N. J. C. Spooner, and J. E. McMillan, *Simulations of muon-induced neutron flux at large depths underground*, Nucl. Instrum. Methods Phys. Res. **A 505** 683–687 (2003).
[5] Valentin Niess, Anne Barnoud, Cristina Carloganu, and Eve Le Ménédeu, *Backward Monte-Carlo applied to muon transport*, Comput. Phys. Comm. **229** 54, (2018).
[6] D. Heck, T. Pierog, and J. Knapp, *CORSIKA: An air shower simulation program*, Astrophys. Source Code Libr. **ascl–1202** (2012).
[7] C. Hagmann, D. Lange, and D. Wright, *Cosmic-ray shower generator (CRY) for Monte Carlo transport codes*, 2007 IEEE Nuclear Science Symposium Conference Record, IEEE, (2007).
[8] A. Fedynitch, R. Engel, T. K. Gaisser, F. Riehn, and T. Stanev, *Calculation of conventional and prompt lepton fluxes at very high energy*, EPJ Web of Conferences, Vol. **99**, EDP Sciences, 08001 (2015).
[9] GEMC: GEant4 Monte-Carlo, URL https://gemc.jlab.org/gemc/html/index.html
[10] P. Biallass and T. Hebbeker, *Parametrization of the cosmic muon flux for the generator CMSCGEN*, arXiv preprint arXiv:0907.5514 (2009).
[11] G. Battistoni, A. Margiotta, S. Muraro, and M. Sioli, *FLUKA as a new high energy cosmic ray generator*, Nucl. Instrum. Methods Phys. Res. **A 626–627**, S191-S192, (2011).

[12] G. Carminati, M. Bazzotti, A. Margiotta, and M. Spurio, *Atmospheric muons from parametric formulas: A fast generator for neutrino telescopes (MUPAGE)*, Comput. Phys. Comm. **179** (12) 915–923 (2008).

[13] MuSteel project, Research Fund for Coal and Steel RFSR-CT-2010-00033.

[14] MuBlast project, Research found for coal and steel RFSR-CT-2014-00027.

[15] S. Pesente et al., *First results on material identification and imaging with a large-volume muon tomography prototype*, Nucl. Instrum. Methods **A 604** 738 (2009).

[16] G. Bonomi, M. Caccia, A. Donzella, D. Pagano, V. Villa, and A. Zenoni, *Cosmic ray tracking to monitor the stability of historical buildings: a feasibility study*, Meas. Sci. Technol. **30** (4) 045901 (2019).

[17] D. Pagano, G. Bonomi, A. Donzella, A. Zenoni, G. Zumerle, and N. Zurlo, *EcoMug: An Efficient COsmic MUon Generator for cosmic-ray muon applications*, Nucl. Instrum. Methods Phys. Res **A1014**, 165732 (2021).

[18] L. Bonechi, M. Bongi, D. Fedele, M. Grandi, S. Ricciarini, and E. Vannuccini, *Development of the ADAMO detector: test with cosmic rays at different zenith angles*, 29th International Cosmic Ray Conference **Vol. 9**, pp. 283 (2005).

[19] L. Bonechi, *Misure di Raggi Cosmici a Terra con l'Esperimento ADAMO* (Ph.D. thesis), Università degli Studi di Firenze, Dipartimento di Fisica (2004).

[20] G. Casella, C. P. Robert, and M. T. Wells, *Generalized accept-reject sampling schemes*, Institute of Mathematical Statistics Lecture Notes - Monograph Series, Institute of Mathematical Statistics, 2004, pp. 342–347.

[21] S. Villa and D. Blackman, *xoshiro/xoroshiro generators and the PRNG shootout*, URL https://prng.di.unimi.it/

[22] T. K. Gaisser, R. Engel, and E. Resconi, *Cosmic Rays and Particle Physics*, Cambridge University Press, 2016.