Integrating the Rivet analysis tool into EPOS 4

Johannes JAHAN[1,2,*], *Klaus* WERNER[2,**], and *Damien* VINTACHE[2]

[1]Department of Physics - University of Houston, Houston, TX 77204, USA
[2]Subatech, Nantes University - IN2P3/CNRS - IMT Atlantique, Nantes, FR

Abstract. EPOS 4 is the last version of the high-energy collision event generator EPOS, released publicly in 2022. It was delivered with improvements on several aspects, whether about the theoretical bases on which it relies, how they are handled technically, or regarding user's interface and data compatibility. This last point is especially important, as part of a commitment to provide the widest possible use. In this regard, a new output data format have been implemented, based on the HepMC standard libraries. This feature enables in particular the analysis of EPOS simulations with Rivet , an analysis and validation toolkit for Monte Carlo event generators, with recent major upgrades on concerning heavy-ion analysis methods. In order to take advantage of this, the use of Rivet has been implemented directly in the EPOS analysis machinery, ensuring an easy and fast solution for comparison with experimental data, beneficial for both developers and users. We will hence present here the details of this implementation and the results obtained thanks to it.

1 Introduction

Event generators (EGs) are codes engineered to compute models in order to simulate collisions on an event-by-event basis, using Monte-Carlo sampling techniques. They occupy a key-role in the field of high-energy physics, as they offer the advantage of offering a perfect knowledge of the whole history of simulated events, although indeed biased by a dependence on the parametrization of the underlying model(s) they are based on. EGs are used on the experimental side, contingently with detector simulation, to help designing new hardware and facilities, testing analysis and charactirizing systematic errors. Most of all, they bridge the gap between theory and experiment as being employed for data interpretation and model validation [1].

For this reason, user's interface, data compatibility and especially compliance to standard formats used by the community is primordial for an EG, to ensure the widest use possible. It has been an important aspect of the development of EPOS4 [2], the latest version of EPOS released publicly in Oct. 2022 [3]. Hence, after describing summarily in section 2 the main physics features of EPOS, with a focus on the new developments achieved for EPOS4, we will explain in section 3 how it has been adapted to take advantage of the features of the Rivet analysis toolkit [4]. We will then confront results of similar analyses obtained through the

[*]e-mail: jahan.johannes@gmail.com
[**]e-mail: werner@subatech.in2p3.fr

EPOS analysis framework and through Rivet , as a cross-check validation for EPOS analyses, and discuss possible discrepancies. Finally, we will consider different outlooks and potential improvements for this new EPOS+Rivet framework.

2 EPOS4

EPOS[1] is a Monte-Carlo multi-purpose EG for high-energy scatterings. It has been developed to simulate any type of collisions, based the same formalism, from $e^+ + e^-$ to $A + A$ systems.

Initial conditions within EPOS are simulated using the Parton-based Gribov-Regge Theory [5], a S-matrix-based multiple scattering approach including perturbative QCD calculations for hard processes. Important improvements have notably been achieved in EPOS4 regarding the coherent treatment of factorization and saturation, by including a dynamical saturation scale [2]. The partons cascades created through initial conditions are then mapped to color-flux strings exchanged between collision remnants, as shown on Figure 1. When the density of strings is too high, which happens in high-multiplicity hadronic collisions for instance, strings can not decay independently from each other, a so-called core-corona procedure is applied [6].

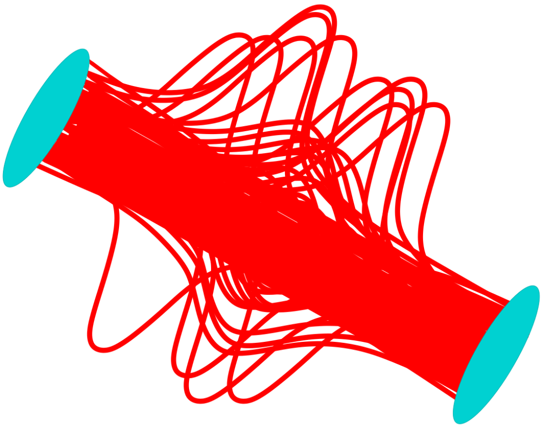

Figure 1. Schematic view of a collision with multiple strings (in red) exchanged between projectile and target remnants (in blue) [7].

The string segments in the high-density region will form the core, which will evolve according to 3+1D viscous relativistic hydrodynamics [8, 9], while the high-p_T segments will escape to form the corona, evolving and breaking according to a relativistic string model [10, 11]. Dynamical effects of strings segment's energy loss while escaping the core are also taken into account [12]. The only equation of state originally available for the hydrodynamic evolution contains a simple crossover between deconfined matter and hadronic phase, with conservation of the B, Q and S conserved charges currents [8]. A new feature of EPOS4 is the possibility to select an equation of state from the BEST collaboration, including a 1st order phase transition and a critical point which location can be chosen [13, 14].

[1] *Energy conservation + Parallel scattering + factOrization + Saturation*

Once the core have evolved and cooled down to a given critical value of energy density ϵ_H, it is hadronised using a microcanonical procedure newly implemented in EPOS4 [15], which plays an important role to reproduce canonical suppression of heavy hadrons in small systems. At last, all hadrons formed from both core and corona (via string fragmentation in the latter case) can re-interact through hadronic cascades, modeled with UrQMD as an hadronic afterburner [16].

3 EPOS4 + RIVET

Any new version of an event generator requires intensive testing, in order to find the most adequate tuning of parameters that enables to reproduce results for the widest possible range of energies and system sizes. For EPOS4, the applicability lies from p+p collisions with collision energies $\sqrt{s} = 0.2 - 14$ TeV, to Au+Au/Pb+Pb collisions with collision energies from $\sqrt{s_{NN}} \approx 20$ GeV/A to $\sqrt{s_{NN}} = 5$ TeV/A, covering almost the entire diagram displayed in Figure 2. Thus, testing a new parametrization across all the systems and energies is highly time consuming in terms of computational resources: we estimate having utilized around 210 million hours of CPU-time in 2022, the year of the public release of EPOS4.

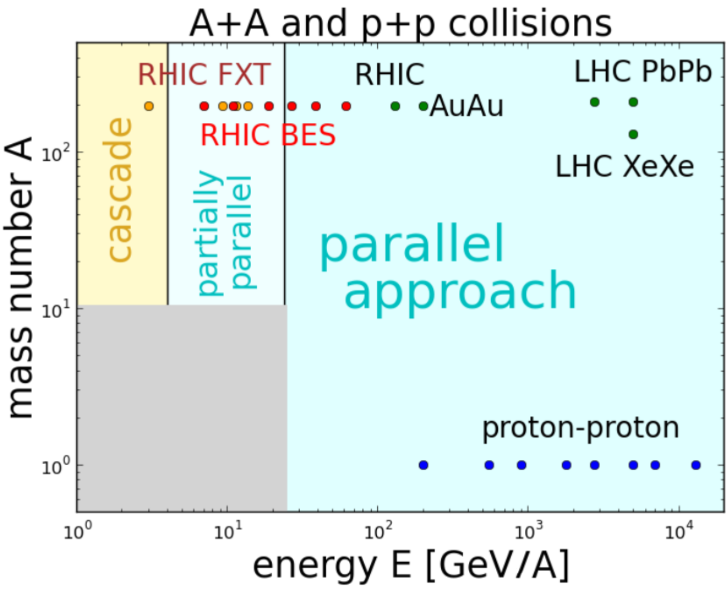

Figure 2. System-size vs. energy range diagram of all current high-energy hadronic scattering experiments and theoretical formalism used to describe them [17].

Such tedious task hence motivates the use of model-to-data comparison tool, namely Rivet . Although EPOS includes its own on-stream analysis framework, the choice to use Rivet relies on the fact that the analyses it contains are usually submitted by the experimental collaborations themselves, thus as close as possible to the ones used to obtain the data published in scientific papers. For us developers, it gives a way to cross-check the analyses we have already implemented through EPOS4 on-stream analysis framework, and allows us to complete this set of analyses thanks to an extensible catalogue. At the same time, it offers to users the option of using an independent analysis tool which is standardized and easy to handle.

3.1 Rivet

Rivet (which stands for *Robust Independent Validation for Experiment and Theory*) is a system for validation of MC event generators, based on a C++ framework for analysis algorithms [18]. The purpose of Rivet is to offer a simple and standardized tool for comparison between EGs simulations and experimental data, as well as ensuring analysis conservation for experimental collaborations.

In Rivet , each analysis reflects a single publication, containing thus the code to run all, or part of all the analyses which results are presented in the publication. Each of them is provided with the corresponding set of experimental data, connected and synchronised with HepData when available [19]. Rivet is based on the standard HepMC library for input data from simulations, contains libraries of generator-independent event analysis methods and even comes with a dedicated histogramming an plotting tool, YODA [20].

3.2 HepMC output format with EPOS4

The first step to make EPOS4 compatible with the use of Rivet is to produce the standard data format it reads, namely HepMC (for "*High-energy physics Monte-Carlo*"). HepMC a package of object-oriented C++ libraries for event record in high-energy physics Monte-Carlo generators and simulations [21]. With EPOS4, one can produce ASCII output files using either HepMC2 (based on version 2.6.9 [22]) or HepMC3 (based on version 3.2.6 [23]).

Consequently, the code and user's interface of EPOS4 have been modified to leave the choice of activating or not the production of HepMC recording, as well as selecting other features. One of the main questions when dealing with big amounts of data, which are necessary when comparing simulations with experimental results, is how to save memory space. The challenge is to find the good balance between recording enough information in the simulated event records to capture all the necessary information needed for analyses, and trying to avoid recording useless information which would take space for nothing. To give a concrete example, 10k complete events of Au+Au collisions at $\sqrt{s_{NN}} = 39$ GeV/A, simulated with EPOS4 under uncompressed HepMC 2 format, occupy 2.1GB of memory space. A significant comparison to experimental results, however, usually requires several millions of events.

The amount of information that is needed from recorded simulated events depends on the type of analysis which is run. For instance, a simple charged particle multiplicity analysis requires only the list of final-state particle created in the collision, while in the case of quarkonia studies, one needs to record the decay products of any quarkonium, to be able to reconstruct it. Hence, we give in EPOS4 the possibility to choose between several HepMC recording modes.

1. "`final_state`" mode, recording only the final-state particles of each event, in addition to the beam particles, is usually enough for simple analyses like charged multiplicity distributions.

2. "`decays`" mode, recording all particles with a lifetime longer than a given value τ_{decay}, as well as their decay cascade (with τ_{decay} a parameter which can be specified by the user), is useful for most analyses of identified particles that need correction from feed-down contributions.

3. "`before_hacas`" mode, recording the distribution of particles before hadronic cascade when activating the simulation of hadronic cascades with UrQMD.
4. "`without_hacas`" mode, recording the distribution of particles in the alternative scenario where the simulation of hadronic cascades wouldn't be activated (using the "`without_hacas`" mode), when it is.
5. "`full`" mode, recording the entirety of the simulated events, although only considering meaningful physical particles, ignoring thus theoretical objects such as string segments.

In the event of a user interested in studying specific hadronic species, one can also specify the identifiers of given species which decay cascade will then be recorded, even if they don't match the recording conditions defined in the recording mode being used.

Another feature is the possibility to apply a rapidity boost to the system, which can be useful when simulating collisions for asymmetric systems for instance. More details regarding the HepMC output in EPOS4 are given in the online documentation [24].

3.3 Implementing Rivet into EPOS analysis framework

Thanks to the production of HepMC output files, it is now possible to run Rivet analyses on EPOS4 simulations. We have thus decided to push this functionality further, in order to help us optimizing the tuning process of the model. We have integrated Rivet directly into the pre-existing EPOS analysis framework, meaning that any user can now call Rivet analyses from EPOS configuration files, like any other regular EPOS analysis, and results will be added to the usual EPOS analysis output files (called **.histo** files). Hence, although one can still use Rivet the regular way (by producing at first a HepMC file, and then executing in a second time Rivet to run an analysis on it), this 2-steps process can now be done all at once from a single execution of EPOS4.

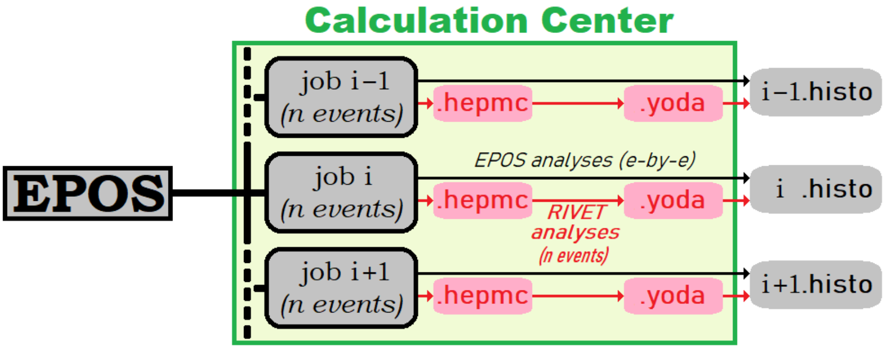

Figure 3. Schematic representation of an analysis workflow for multiple EPOS jobs running in parallel.

To do so, we have developed a Python wrapper which takes information regarding possible Rivet analyses to run from the configuration file, run those analyses and then convert the histograms obtained from the YODA format returned as an output by Rivet into the format used in EPOS **.histo** files. The way this process is integrated into the EPOS analysis framework is displayed schematically in Figure 3, for a scenario where several jobs, simulating

n events with EPOS, are working in parallel. For the i^{th} job, on-stream EPOS analyses are run on an event-by-event basis, with the corresponding obtained histograms being updated at every iteration in the **.histo** file. Once all n events of this job have been simulated, the HepMC file produced that contains those events is fed into Rivet by the Python wrapper to run the analyses requested by the user. The histograms obtained and stored in the resulting YODA files (1 per analysis) are then converted and stored into the corresponding **.histo** file of this same i^{th} job by the Python wrapper. At last, the temporary YODA files are deleted, as well as the HepMC files (both represented in red on Figure 3 for this reason), if the user have not specified they wanted to save them. This way, one can actually run Rivet analyses on-the-fly without having to save temporary files which can take up significant memory space.

4 Analysis Results & Comparisons

In this section, we compare similar analyses run both with the EPOS4 on-stream analysis framework and with Rivet , for $p + p$ collisions at collision energy $\sqrt{s}$ = 7 TeV. The aim is to cross-check the validity of EPOS analyses, thus making sure the parametrization is based on trustworthy results, although the ones shown here only reflect a small part of the whole scan in systems and energies achieved in that extent.

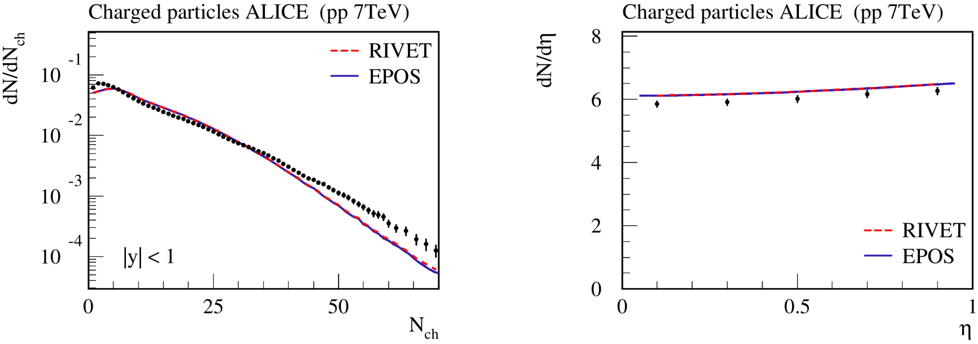

Figure 4. Results from identical analyses run with EPOS4 on-stream analysis framework and Rivet , on p + p simulations at $\sqrt{s}$ = 7 TeV, compared with ALICE data displayed in black dots, for: (a) charged particle multiplicity distribution, (b) charged particle pseudorapidity distribution [25].

The first checks shown here concern general quantities charactirizing the events analyzed. On Figure 4, one observes that multiplicity and pseudorapidity distributions for charged particles are identical for both analysis tools, confirming that the analyses run with EPOS4 are correct (as we assume the Rivet analyses are supposedly the correct ones).

On Figure 5 (a), one can see that no D^* are found through the Rivet analysis. This is due to the fact that those very short-lived resonances are not recorded in HepMC files because they have a $\tau_{D^*} < \tau_{decay}$, which is set by default to 10^{-19}s. It illustrates thus why the fact to let the user select specific hadronic species they want to record, despite the established trigger, is important for some analyses. On Figure 5 (b), one can observe a discrepancy between EPOS4 and Rivet analyses regarding charged Ξ production at low-p_T. It is an important finding, as we do not understand its cause and will hence have to investigate in details the differences between the methods used in the 2 different frameworks.

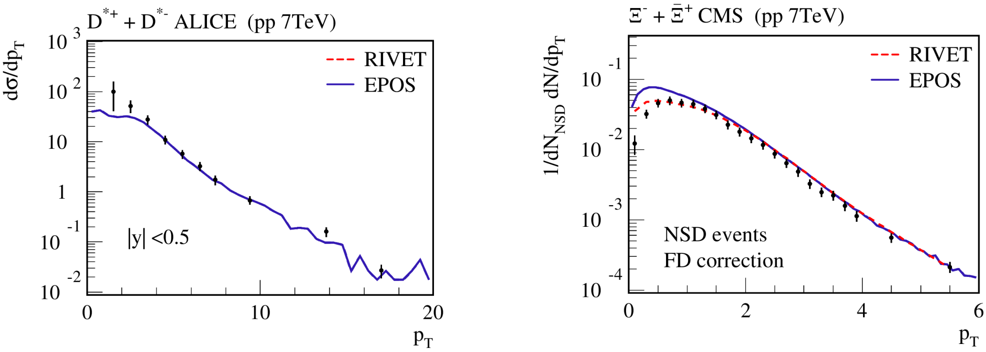

Figure 5. Same as Figure 4, for: (a) charged D^* meson p_T-differential distribution [26], (b) charged $\Xi + \overline{\Xi}$ p_T spectrum compared with CMS data [27].

5 Summary & Outlook

To summarise, we have added the possibility to produce HepMC event record files with EPOS4 as part of the efforts towards improving its compatibility for its public release. This new feature not only enables to use the Rivet package to analyze EPOS4 simulations, but we have also included the possibility to call Rivet analyses directly through the EPOS on-stream analysis framework. The goal of such implementation is not only to simplify the experience of the user, but also to help developers in the process of scanning all different collision systems and energies to find the best set of parameters to tune the EG.

Thanks to this implementation, Rivet analyses are run on the fly, like any other analysis from the EPOS4 analysis framework, reducing the number of steps to be executed by the user. It helps to save memory space too, since it allows to get rid of the HepMC file containing the recorded events, and we allow the user to chose which specific plots they want to save from the Rivet analysis. However, the ability to run Rivet analyses for heavy-ion is more tedious, since they usually require an extra step for centrality calibration. To deal with it in EPOS4, we pre-save centrality calibration files, assuming that any newly generated events should follow approximately the same distribution. If it is not the case, the user would then need to run again the analyses using a newly generated centrality calibration file, which makes thus losing the advantage of combining steps when calling Rivet analyses through EPOS4.

However, this new feature could be further improved by using FIFO pipe to run Rivet , in order to save the memory space used by HepMC files when running jobs, even if only temporarily. Finally, despite the efforts to develop all the tools necessary for heavy-ion analyses such as centrality determination [28], the analysis coverage in Rivet for such systems is too limited to make it a complete and efficient tool for model parametrization.

References

[1] J.M. Campbell, M. Diefenthaler, T.J. Hobbs, S. Höche, J. Isaacson, F. Kling, S. Mrenna, R. et al. (2022), arXiv:2203.111110 [hep-ph]

[2] K. Werner (2023), arXiv:2301.12517 [hep-ph]

[3] EPOS4 – https://klaus.pages.in2p3.fr/epos4/

[4] C. Bierlich et al., SciPost Phys. **8**, 026 (2020), arXiv:1912.05451 [hep-ph]

[5] H.J. Drescher, M. Hladik, S. Ostapchenko, T. Pierog, K. Werner, Phys. Rept. **350**, 93 (2001), arXiv:hep-ph/0007198

[6] K. Werner, Phys. Rev. Lett. **98**, 152301 (2007), arXiv:0704.1270 [nucl-th]

[7] K. Werner, **MPI in event generators**, in *Joliot-Curie School* (October 2018)

[8] K. Werner, I. Karpenko, T. Pierog, M. Bleicher, K. Mikhailov, Phys. Rev. C **82**, 044904 (2010), arXiv:1004.0805 [nucl-th]

[9] I. Karpenko, P. Huovinen, M. Bleicher, Comput. Phys. Commun. **185**, 3016 (2014), arXiv:1312.4160 [nucl-th]

[10] K. Werner, Phys. Rept. **232**, 87 (1993)

[11] H. Drescher, PhD Thesis, Université de Nantes (1999), http://hal.in2p3.fr/in2p3-00021346

[12] K. Werner, I. Karpenko, M. Bleicher, T. Pierog, S. Porteboeuf-Houssais, Phys. Rev. C **85**, 064907 (2012), arXiv:1203.5704 [nucl-th]

[13] M. Stefaniak, K. Werner, J. Jahan, H.P. Zbroszczyk, Phys. Rev. C **108**, 014905 (2023), arXiv:2209.12979 [hep-ph]

[14] P. Parotto, M. Bluhm, D. Mroczek, M. Nahrgang, J. Noronha-Hostler, K. Rajagopal, C. Ratti, T. Schäfer, M. Stephanov, Phys. Rev. C **101**, 034901 (2020), arXiv:1805.05249 [hep-ph]

[15] K. Werner (2023), arXiv:2306.10277 [hep-ph]

[16] M. Bleicher et al., J. Phys. **G25**, 1859 (1999), arXiv:hep-ph/9909407

[17] K. Werner, **EPOS4 - An overview**, in *38th Winter Workshop on Nuclear Dynamics* (February 2023)

[18] RIVET – https://rivet.hepforge.org/

[19] HepData – https://www.hepdata.net/

[20] YODA – https://yoda.hepforge.org/

[21] HepMC – http://hepmc.web.cern.ch/hepmc/

[22] L.G..L.S. M. Dobbs, J.B. Hansen, ***HepMC 2: A C++ event record for Monte-Carlo generators*** (2010), http://www.t2.ucsd.edu/twiki2/pub/HEPProjects/HepMCReference/HepMC2_user_manual.pdf

[23] A. Buckley, P. Ilten, D. Konstantinov, L. Lönnblad, J. Monk, W. Pokorski, T. Przedzinski, A. Verbytskyi, Comput. Phys. Commun. **260**, 107310 (2021), arXiv:1912.08005 [hep-ph]

[24] EPOS4/Manuals – https://klaus.pages.in2p3.fr/epos4/code/manual

[25] K. Aamodt et al. (ALICE), Eur. Phys. J. C **68**, 345 (2010), arXiv:1004.3514 [hep-ex]

[26] B. Abelev et al. (ALICE), JHEP **01**, 128 (2012), arXiv:1111.1553 [hep-ex]

[27] V. Khachatryan et al. (CMS), JHEP **05**, 064 (2011), arXiv:1102.4282 [hep-ex]

[28] C. Bierlich et al., Eur. Phys. J. C **80**, 485 (2020), arXiv:2001.10737 [hep-ex]