frontiers | Frontiers in Physics

TYPE Brief Research Report
PUBLISHED 29 October 2024
DOI 10.3389/fphy.2024.1502621

SMASH as an event generator for heavy-ion collisions

Alessandro Sciarra[1] and Hannah Elfner[1,2,3,4]*

[1]Institute for Theoretical Physics, Goethe University, Frankfurt, Germany, [2]Department for hot and dense QCD matter, GSI Helmholtzzentrum für Schwerionenforschung, Darmstadt, Germany, [3]Frankfurt Institute for Advanced Studies, Frankfurt am Main, Germany, [4]Helmholtz Research Academy Hesse (HFHF), GSI Helmholtz Center, Campus Frankfurt, Frankfurt am Main, Germany

Check for updates

OPEN ACCESS

EDITED BY
Andreas Redelbach,
Frankfurt Institute for Advanced Studies,
Germany

REVIEWED BY
Qingfeng Li,
Huzhou University, China
Varese Timóteo,
State University of Campinas, Brazil

*CORRESPONDENCE
Hannah Elfner,
h.elfner@gsi.de

RECEIVED 27 September 2024
ACCEPTED 16 October 2024
PUBLISHED 29 October 2024

CITATION
Sciarra A and Elfner H (2024) SMASH as an event generator for heavy-ion collisions.
Front. Phys. 12:1502621.
doi: 10.3389/fphy.2024.1502621

In this article we present an overview of the SMASH hadronic transport approach that is applied for non-equilibrium dynamics of hadrons in heavy-ion collisions. We will give an overview about the ingredients of the approach and the applications for the dynamical description of heavy-ion collisions and for calculations of fundamental properties of the hadron gas. The main emphasis of the article will be the infrastructure for sustainable software development that we have developed over the last 10 years including extensive unit tests and continuous integration. We will also provide one section about the performance of the code and how it can be analyzed and improved in the future.

KEYWORDS

transport theory, sustainable software, performance, event generator, heavy-ion collisions

1 Introduction

The building blocks of matter and the fundamental forces of nature are summarized in the standard model of particle physics. Besides the electromagnetic and the weak interaction, the strong force is responsible for keeping quarks together. Gluons are the corresponding exchange particles and bind the quarks in pairs and triples such that color neutral objects are formed, the hadrons. Studying hadronic matter under extreme conditions of temperature and density is the main goal of heavy-ion research. Within the collisions of nuclei that are accelerated to almost the speed of light by major accelerators around the world—currently the Large Hadron Collider (LHC) at CERN, the Relativistic Heavy Ion Collider (RHIC) at BNL and the SIS-18 at GSI—the matter is heated and compressed such that the circumstances are similar to the ones only microseconds after the Big Bang or the ones in neutron star mergers [1].

At such high temperatures and/or densities, hadronic matter undergoes a phase transition to the quark-gluon plasma, where hadrons are not bound anymore and quarks and gluons become the active degrees of freedom. To study the properties of this new state of matter as well as the nature of the transition are the main motivations behind heavy-ion research. Experimentally, it is only possible to observe the final state particle distributions of hadrons and electromagnetic probes in the detector. To connect the measurements to fundamental insights, detailed models of the dynamical evolution are indispensable. The hadronic transport approach SMASH (Simulating Many Accelerated Strongly-interacting Hadrons) [2, 3] is one of the general purpose event generators in the field. This microscopic transport approach incorporates the non-equilibrium dynamics of hadrons in a heavy-ion collision and is therefore applicable for the full evolution at low beam energies and for the early/late stages at high beam energies.

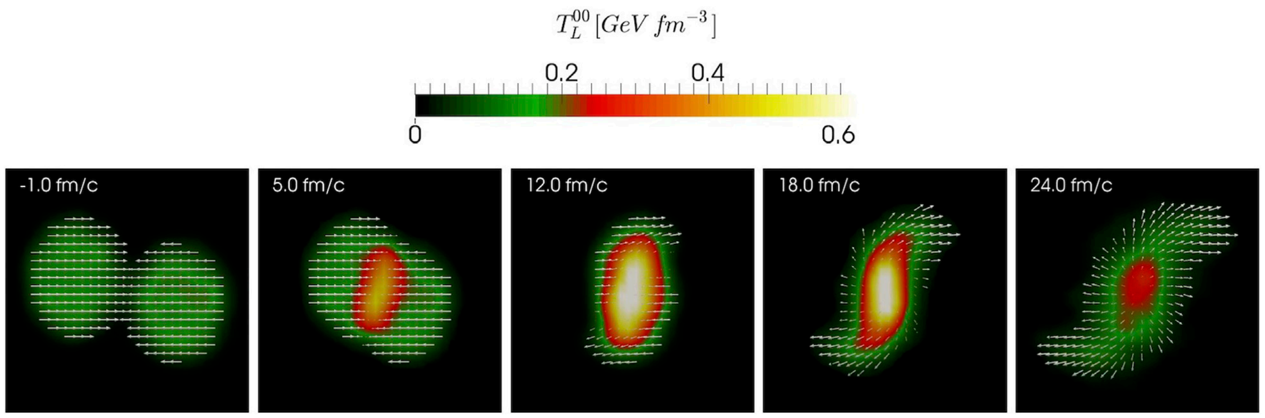

FIGURE 1
Figure taken from [2]. Snapshots of the time evolution of a Au + Au collision at E_{kin} = 0.8A GeV with impact parameter b = 3 fm, N_{test} = 20. The energy density T_L^{00} (background color) and velocity of Landau frame (arrows) are displayed for the baryons in the system. The velocity is proportional to the arrow length and the maximal arrow length corresponds to a velocity of 0.55 c.

Transport approaches have been developed and applied for more than 30 years and many approaches exist [4–11] that are available to the public, at least on request. It is crucial to compare different approximations and frameworks carefully to assess the impact of certain choices on experimental observables. This has been carried out extensively by the Transport Model Evaluation Project [12–14]. This exemplifies why open source codes and transparent development practices are important to allow the user to assess the validity of the approach and understand changes over time, when new features are included. Another aspect is that a modern software development infrastructure allows new students and external collaborators [15–17] to access and work with the code without much overhead.

In this work, the main ingredients and scope of the SMASH transport approach are explained in Section 2. In Section 3 the software development environment is described in detail to showcase an example of transparent development in line with the FAIR4RS rules. Section 4 addresses performance and optimization potential of the code, since it is also part of sustainable software development to use high performance computing resources efficiently. The last Section 5 summarizes the main findings and conclusions.

2 Physics application

SMASH is a relativistic hadronic transport code, that represents an effective solution of the relativistic Boltzmann equation incorporating all well-known hadronic species based on the particle data book from 2018 [18].

$$p^\mu \partial_\mu f_i(x,p) + m_i F^\alpha \partial^p_\alpha f_i(x,p) = C^i_{\text{coll}}$$

where f_i is the distribution function, C^i_{coll} is the collision term, F^α is the force experienced by individual particles and m_i is the particle mass. For high beam energy collisions, the potential term can be neglected $F^\alpha = 0$, while for low beam energy collisions, $F^\alpha = -\partial^\alpha U(x)$ where $U(x)$ is the mean-field potential. The collision term contains all the details of the interactions between the particles. At lower energies the reactions proceed via resonance excitation and decay while at high energies strings are excited and fragment via Pythia [19, 20]. Within SMASH binary collisions are handled via a geometric collision criterion, but there is also the stochastic rates approach available relying on testparticles and interaction probabilities on a spatial grid, which allows for multi-particle reactions.

Figure 1 displays several steps of the time evolution for a low energy heavy ion collision corresponding to the ones studied at SIS-18 at GSI. One can clearly see the hot and dense region that is formed as well as the collective velocity of the system. The collective flow is one of the major observables to constrain transport coefficients and the equation of state of strongly interacting matter.

Figure 2 (left) shows as an example the energy dependence of the proton-proton cross-sections. At low energies, resonance excitations dominate, followed by a regime of soft string excitation and decay while hard processes take over at high energies. The elastic cross-section is fitted to the available experimental data. In Figure 2 (right) the mass dependent width of the $N^*(1440)$ resonance is shown. The width is inversely proportional to the lifetime of the resonance. The mass of the resonance is determined by a relativistic Breit-Wigner distribution in each individual scattering. Dependent on the mass, different decay processes are kinematically possible, therefore the width is changing accordingly. These figures are shown to give an impression of the complexity of modeling all the individual reaction channels of more than 150 hadronic species.

SMASH can be applied in several settings; `Collider` for full collision simulations as shown in Figure 1, `Box` for infinite hadronic matter, `Sphere` for expanding matter as well as `List` where the user can insert any particle list of choice to be evolved in time. Examples of calculations in infinite hadronic matter include the calculation of transport coefficients of the hadron gas at different temperatures and chemical potentials (see, e.g., [22, 23]). The expanding sphere serves as a simplified setup to simulate collision dynamics as has been employed, e.g., in [24]. The listmodus is crucial for calculations where SMASH is employed for the late hadronic rescattering as part of hybrid approaches. The

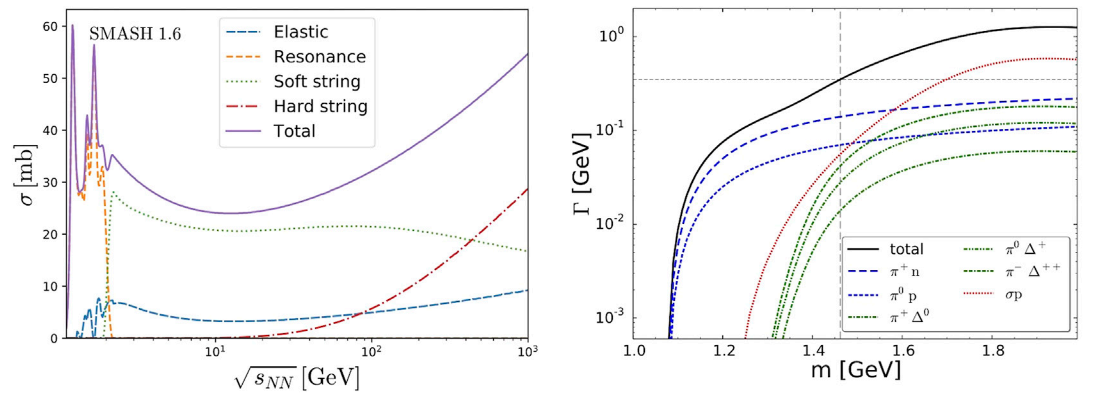

FIGURE 2
Figures taken from [2, 21]. Left: Cross-section for pp reactions as a function of the center-of-mass energy, the contributions of different processes are shown. Right: Mass-dependent width of the *N**(1440) resonance indicating that different branching ratios appear at different thresholds.

standard model for the dynamical evolution of high energy heavy-ion collisions involves a non-equilibrium initial state, a viscous hydrodynamic evolution and hadronic rescattering. SMASH has been applied for this purpose in our own SMASH-vHLLE hybrid[1] approach [25, 26] as well as by other theory groups around the world [27–30]. It is also possible to couple to SMASH as a library to employ certain modules or just the hadron gas properties. The complete particle content and the resonance properties can be changed by exchanging two human readable text file, `particles.txt` and `decaymodes.txt`, which offers a lot of flexibility.

An additional target group of the approach are large experimental collaborations like ALICE, STAR, NA61, CBM and HADES that are running the code to make predictions or compare to their measurements. To ensure that external users can follow updates of the codes and how changes and addition of features affect the physics observables a physics analysis suite is run for each tagged version of the code. The results for each SMASH version are publicly available[2] and the corresponding analysis code is released[3] as well. This test includes inherent tests such as the comparison of collision rates to analytic expectations and detailed balance tests as well as comparisons to experimental data in vacuum (elementary cross-sections and angular distributions) and hot and dense medium (AA collisions at different beam energies). To ensure simple application, the installation of the code is documented in detail and docker/singularity images are provided for each tagged public version. There are also all commonly used output formats supported by SMASH including several stages of OSCAR formats in text and binary versions as well as a ROOT interface. In addition, there is an interface for HepMC output to connect to the RIVET analysis software that is being adopted by the heavy-ion community recently [31].

3 Sustainable software development

SMASH is an open source code developed by one working group, and employed by many theoretical groups and experimental collaborations world-wide. To make such a code sustainable, a few points are important. The software development follows strict rules that ensure the FAIR4RS principles for software development [32] and, hence, sustainability. The released code is publicly available on GitHub and published on Zenodo, where each version gets a DOI associated and the `CITATION.cff` file maintained in the codebase is used for metadata (Findable and Accessible). Internal development of new features and constant improvements happen in a private repository, that is regularly made public, whenever a new version of the software is released (1–2 times per year). For flawless and straightforward usage, every new release is shipped with Docker images that allow any user to immediately run simulations in a dedicated container. Versioning is crucial to avoid misunderstandings and misinterpretation of physics results. Typically, in software development, semantic versioning is used, where the first digit highlights major changes, the second digit indicates backward compatible changes and the third digits is reserved for small bug fixes. SMASH does not fully comply with this versioning scheme, as new physics features are often driving new releases and backward compatibility is not always easy to guarantee. However, a detailed `CHANGELOG` file is kept up-to-date during development and any potentially relevant change for the user is listed and shortly described there. Every new release is provided with both an extensive code documentation as well as a less technical user guide (Reusable).

No large codebase could survive, grow and improve over many years without a rigorous and complete testing strategy or if good practices in software development were not applied as often as possible. Unit, integration and system tests are regularly developed in SMASH to ensure code correctness and over 100 test suites exist. Figure 3 illustrates how software tests are classified and what is usual to be expected in terms of execution time and number of tests. SMASH acceptance tests are developed in separate codebase (smash-analysis) and consists of a whole plethora of physics tests already mentioned in Section 2.

1 https://github.com/smash-transport/smash-vhlle-hybrid

2 http://theory.gsi.de/~smash/analysis_suite/SMASH-3.1/index.html

3 https://github.com/smash-transport/smash-analysis

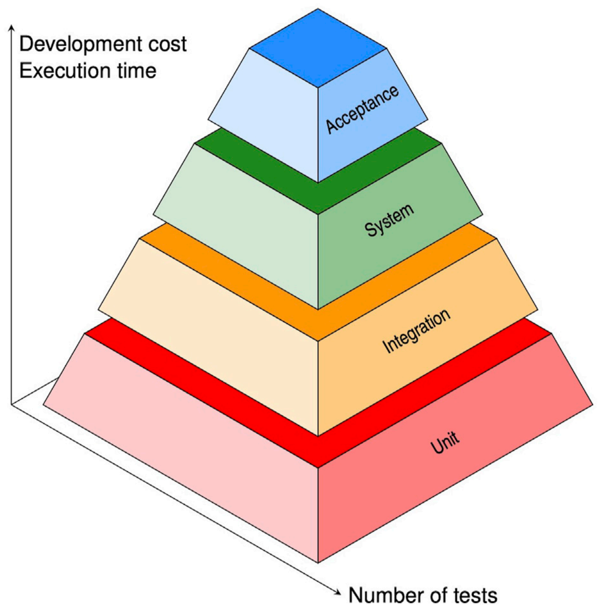

FIGURE 3
The tests pyramid, taken from [33]. Different types of software tests are classified according to some features. The base surface of each layer is proportional to how many of that test type should exist in a codebase and it is inversely proportional to their execution time and development cost. Roughly speaking, unit tests are about testing each unit of the codebase. A unit might be, e.g., a module or sometimes a class/function. Integration tests are meant to test the correct interaction between different units. System tests ensure the correctness of larger (if not all) parts of a codebase. Acceptance tests determine whether the application meets the requirements. In our context, they ensure that physics is correctly described and, e.g., experimental data reproduced as expected.

New developments happen on dedicated Git branches and are merged into the private codebase via pull requests, a very well known GitHub functionality, which offers the possibility to associate given requirements before authorizing the merge. These are a positive code review and the successful run of a given set of actions. The former is thoughtfully done by at least another group member, although often many developers interact exchanging suggestions and improvements. The latter, instead, is an automatic continuous integration system which is set up to build the code on a few common architectures and with different compilers, to ensure proper formatting as well as complete documentation of the code and to check that all unit, functional and system tests pass. If any of these operations fail, the merge of the pull request is denied. Due to their intense computational cost, acceptance tests cannot be run at each pull request, nor would it make sense. A natural point in development to run them is just before each new public release, in order to ensure transparency between versions.

Given the available tests infrastructure as well as the procedure to merge code changes into the codebase, developers are free to experiment and refactor code. This is a key aspect in order to be able to constantly apply clean code [34] principles and, hence, keep the software development sustainable also from the technical point of view of changing existing code and writing new one.

The software is written in C++ and built via CMake and is modular, such that it is useable in parts or fully as a library (Interoperable). Usage of new C++ standards is constantly considered, although the need to offer a reliable product on most machines in the world delays upgrades to most recent language features (at the moment C++17 is used).

4 Performance and optimization

SMASH performance is a core aspect in development and there is lots of ongoing effort trying to find possible improvements. However, "*Programmers waste enormous amounts of time thinking about, or worrying about, the speed of noncritical parts of their programs, and these attempts at efficiency actually have a strong negative impact when debugging and maintenance are considered. We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil. Yet we should not pass up our opportunities in that critical 3%.*" – D. Knuth [35]. This is the reason why optimization tasks are kept as much as possible separated from development and clean-code refactoring. Roughly speaking the code has first to work—tests ensure that—and then, exactly because tests exist, the code can be freely changed looking for better performance. This is only done after having thoroughly profiled the code, typically measuring how many times single functions are called and how much of SMASH runtime is spent executing each of them. In this way, educated guesses about possible bottlenecks can be made and optimization efforts structured.

In a complex codebase, no matter how carefully development is done, accidental pessimizations can occur during development. This is why SMASH is shipped with a standardized set of benchmarks which are run on the same machine before every new release, in order to check how the new version performs with respect to the previous one(s). As different compilers produce different executables in terms of performance, benchmarks are run compiling the code using two of the most common C++ compilers: GNU and LLVM. In Figure 4 the benchmark outcome comparing two given SMASH versions is visualized. Overall the trend is clear: The codebase performance has constantly improved and in the last couple of years SMASH got roughly two times faster in most of the possible setups.

5 Summary and conclusion

SMASH is a general purpose event generator for heavy-ion collisions based on the relativistic Boltzmann equation. The microscopic non-equilibrium dynamics of all hadrons are simulated from initial to final stage or it is employed for the late hadronic rescatterings at high beam energies. The way how sustainable software development is employed is pioneering in the landscape of research software in the field of theoretical heavy-ion reactions. Therefore, the steps that are undertaken to ensure transparency and reproducibility of results are described in detail. The initial performance analysis and first optimization have been shown as well, but there is certainly room for additional work in the future. For example, one can look into multi-threading or parallel computing for CPU intensive settings involving stochastic rates, many testparticles or parallel ensembles.

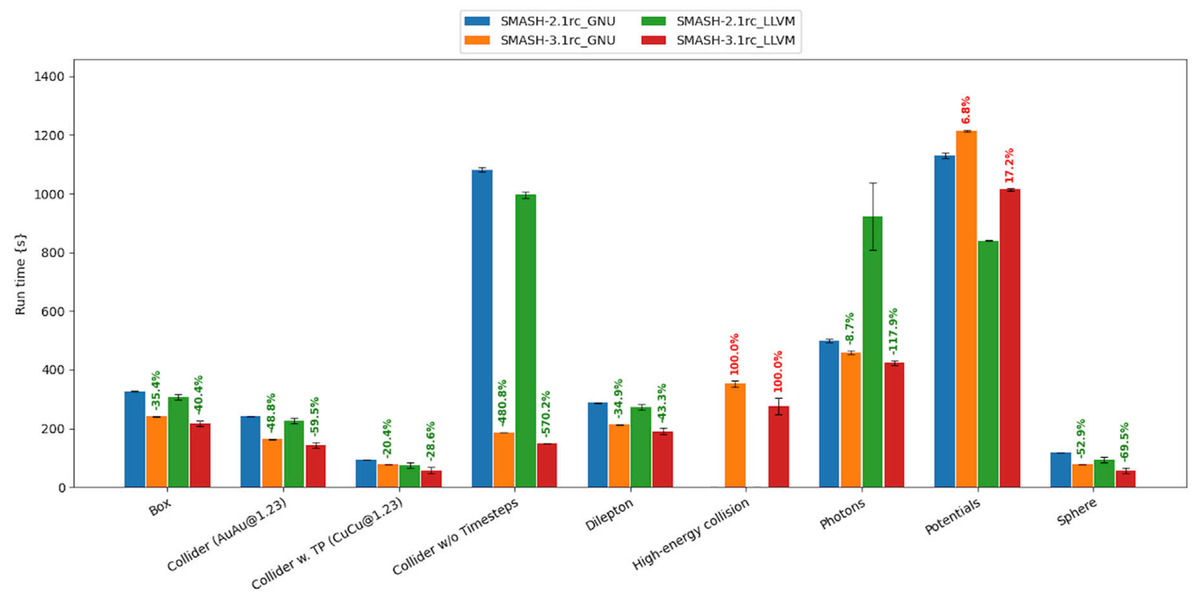

FIGURE 4
Evolution of SMASH performance over four releases. Different groups of bars refer to different SMASH setups. The percentage above the second and fourth bar within each group indicates the relative improvement (green) or loss (red) of performance. Since the "high-energy collision benchmark" has been introduced in SMASH-2.2, 2 bars are missing there. The potentials implementation is known to be sub-optimal and performance has slightly degraded there, because of some code-entanglement which will be cleaned up soon. `Box`, `Collider` and `Sphere` refer to the SMASH default setups already mentioned in Section 2, while the other names refer to different physics systems and/or observables, possibly simulated with different approaches. The `bin/benchmarks` folder in SMASH codebase contains all further details.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: https://github.com/smash-transport/smash/releases/tag/SMASH-3.1.

Author contributions

AS: Conceptualization, Data curation, Investigation, Methodology, Software, Supervision, Validation, Visualization, Writing–original draft, Writing–review and editing. HE: Conceptualization, Funding acquisition, Investigation, Project administration, Resources, Software, Supervision, Visualization, Writing–original draft, Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project number 315477589 – TRR 211.

Acknowledgments

Computational resources have been provided by the Center for Scientific Computing (CSC) at the Goethe-University of Frankfurt and the GreenCube at GSI.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The handling editor AR declared a shared affiliation with the author(s) at the time of review.

Publisher's note

References

1. Abbott BP, Abbott R, Abbott T, Acernese F, Ackley K, Adams C, et al. GW170817: observation of gravitational waves from a binary neutron star inspiral. *Phys Rev Lett* (2017) 119:161101. doi:10.1103/PhysRevLett.119.161101

2. Weil J, Steinberg V, Staudenmaier J, Pang LG, Oliinychenko D, Mohs J, et al. Particle production and equilibrium properties within a new hadron transport approach for heavy-ion collisions. *Phys Rev C* (2016) 94:054905. doi:10.1103/PhysRevC.94.054905

3. Wergieluk A, Weil J, Tindall J, Steinberg V, Staudenmaier J, Sorensen A, et al. smash-transport/smash: SMASH-3.1 (2024). doi:10.5281/zenodo.10707746

4. Hartnack C, Li ZX, Neise L, Peilert G, Rosenhauer A, Sorge H, et al. Quantum molecular dynamics: a microscopic model from unilac to CERN energies. *Nucl Phys A* (1989) 495:303C–320C. doi:10.1016/0375-9474(89)90328-X

5. Bass SA, Belkacem M, Bleicher M, Brandstetter M, Bravina L, Ernst C, et al. Microscopic models for ultrarelativistic heavy ion collisions. *Prog Part Nucl Phys* (1998) 41:255–369. doi:10.1016/S0146-6410(98)00058-1

6. Nara Y, Otuka N, Ohnishi A, Niita K, Chiba S. Study of relativistic nuclear collisions at AGS energies from p + Be to Au + Au with hadronic cascade model. *Phys Rev C* (2000) 61:024901. doi:10.1103/PhysRevC.61.024901

7. Lin ZW, Ko CM, Li BA, Zhang B, Pal S. Multiphase transport model for relativistic heavy ion collisions. *Phys Rev C* (2005) 72:064901. doi:10.1103/PhysRevC.72.064901

8. Bratkovskaya EL, Cassing W, Konchakovski VP, Linnyk O. Parton-hadron-string dynamics at relativistic collider energies. *Nucl Phys A* (2011) 856:162–82. doi:10.1016/j.nuclphysa.2011.03.003

9. Buss O, Gaitanos T, Gallmeister K, van Hees H, Kaskulov M, Lalakulich O, et al. Transport-theoretical description of nuclear reactions. *Phys Rept* (2012) 512:1–124. doi:10.1016/j.physrep.2011.12.001

10. Pierog T, Karpenko I, Katzy JM, Yatsenko E, Werner K, Epos LHC. EPOS LHC: test of collective hadronization with data measured at the CERN Large Hadron Collider. *Phys Rev C* (2015) 92:034906. doi:10.1103/PhysRevC.92.034906

11. Gläßel S, Kireyeu V, Voronyuk V, Aichelin J, Blume C, Bratkovskaya E, et al. Cluster and hypercluster production in relativistic heavy-ion collisions within the parton-hadron-quantum-molecular-dynamics approach. *Phys Rev C* (2022) 105:014908. doi:10.1103/PhysRevC.105.014908

12. Zhang YX, Wang YJ, Colonna M, Danielewicz P, Ono A, Tsang MB, et al. Comparison of heavy-ion transport simulations: collision integral in a box. *Phys Rev C* (2018) 97:034625. doi:10.1103/PhysRevC.97.034625

13. Ono A, Xu J, Colonna M, Danielewicz P, Ko CM, Tsang MB, et al. Comparison of heavy-ion transport simulations: collision integral with pions and resonances in a box. *Phys Rev C* (2019) 100:044617. doi:10.1103/PhysRevC.100.044617

14. Wolter H, Colonna M, Cozma D, Danielewicz P, Ko CM, Kumar R, et al. Transport model comparison studies of intermediate-energy heavy-ion collisions. *Prog Part Nucl Phys* (2022) 125:103962. doi:10.1016/j.ppnp.2022.103962

15. Sorensen A, Koch V. Phase transitions and critical behavior in hadronic transport with a relativistic density functional equation of state. *Phys Rev C* (2021) 104:034904. doi:10.1103/PhysRevC.104.034904

16. Sorensen A, Oliinychenko D, Koch V, McLerran L. Speed of sound and baryon cumulants in heavy-ion collisions. *Phys Rev Lett* (2021) 127:042303. doi:10.1103/PhysRevLett.127.042303

17. Oliinychenko D, Sorensen A, Koch V, McLerran L. Sensitivity of Au+Au collisions to the symmetric nuclear matter equation of state at 2–5 nuclear saturation densities. *Phys Rev C* (2023) 108:034908. doi:10.1103/PhysRevC.108.034908

18. Tanabashi M, Hagiwara K, Hikasa K, Nakamura K, Sumino Y, Takahashi F, et al. Review of particle physics. *Phys Rev D* (2018) 98:030001. doi:10.1103/PhysRevD.98.030001

19. Bierlich C, Chakraborty S, Desai N, Gellersen L, Helenius I, Ilten P, et al. A comprehensive guide to the physics and usage of PYTHIA 8.3. *Scipost Phys Codeb* (2022) 2022:8. doi:10.21468/SciPostPhysCodeb.8

20. Bierlich C, Chakraborty S, Desai N, Gellersen L, Helenius I, Ilten P, et al. Codebase release 8.3 for PYTHIA. *Scipost Phys Codebases* (2022) 8-r8.3–r8.3. doi:10.21468/SciPostPhysCodeb.8-r8.3

21. Mohs J, Ryu S, Elfner H. Particle production via strings and baryon stopping within a hadronic transport approach. *J Phys* (2020) 47:065101. doi:10.1088/1361-6471/ab7bd1

22. Rose JB, Torres-Rincon JM, Schäfer A, Oliinychenko DR, Petersen H. Shear viscosity of a hadron gas and influence of resonance lifetimes on relaxation time. *Phys Rev C* (2018) 97:055204. doi:10.1103/PhysRevC.97.055204

23. Rose JB, Greif M, Hammelmann J, Fotakis JA, Denicol GS, Elfner H, et al. Cross-conductivity: novel transport coefficients to constrain the hadronic degrees of freedom of nuclear matter. *Phys Rev D* (2020) 101:114028. doi:10.1103/PhysRevD.101.114028

24. Dorau P, Rose JB, Pablos D, Elfner H. Jet quenching in the hadron gas: an exploratory study. *Phys Rev C* (2020) 101:035208. doi:10.1103/PhysRevC.101.035208

25. Schäfer A, Karpenko I, Wu XY, Hammelmann J, Elfner H. Particle production in a hybrid approach for a beam energy scan of Au+Au/Pb+Pb collisions between =4.3 GeV and 200.0 GeV.Eur.Phys.J.A58. (2022) 230. doi:10.1140/epja/s10050-022-00872-x

26. Götz N, Elfner H. Temperature and net baryochemical potential dependence of in a hybrid approach. *Phys Rev C* (2022) 106:054904. doi:10.1103/PhysRevC.106.054904

27. Nijs G, van der Schee W, Gürsoy U, Snellings R. Bayesian analysis of heavy ion collisions with the heavy ion computational framework Trajectum. *Phys Rev C* (2021) 103:054909. doi:10.1103/PhysRevC.103.054909

28. Everett D, Ke W, Paquet JF, Vujanovic G, Bass SA, Du L, et al. Multisystem Bayesian constraints on the transport coefficients of QCD matter. *Phys Rev C* (2021) 103:054904. doi:10.1103/PhysRevC.103.054904

29. Everett D, Ke W, Paquet JF, Vujanovic G, Bass S, Du L, et al. Phenomenological constraints on the transport properties of QCD matter with data-driven model averaging. *Phys Rev Lett* (2021) 126:242301. doi:10.1103/PhysRevLett.126.242301

30. Wu XY, Qin GY, Pang LG, Wang XN. D viscous hydrodynamics at finite net baryon density: identified particle spectra, anisotropic flows, and flow fluctuations across energies relevant to the beam-energy scan at RHIC. *Phys Rev* (2022) 105:034909. doi:10.1103/PhysRevC.105.034909

31. Bierlich C, Buckley A, Butterworth J, Gutschow C, Lonnblad L, Procter T, et al. Robust independent validation of experiment and theory: rivet version 4 release note. *SciPost* (2024).

32. Barker M, Chue Hong NP, Katz DS, Lamprecht AL, Martinez-Ortiz C, Psomopoulos F, et al. Introducing the fair principles for research software. *Scientific Data 9* (2022) 622:622. doi:10.1038/s41597-022-01710-x

33. Sciarra A. Clean testing–Good practices in general coding. *GitHub* (2023).

34. Martin RC. Clean code: a handbook of agile software craftsmanship Upper Saddle River, NJ: prentice Hall (2009).

35. Knuth DE. Structured programming with go to statements. *ACM Comput Surv* (1974) 6:261–301. doi:10.1145/356635.356640