Overview of TRIPOLI-5, a Monte Carlo code for HPC

Davide Mancusi, Emeric Brun, Benjamin Dechenaux, Kévin Fröhlicher, Thomas Gonçalves, Alexis Jinaphanh, Mikolaj Adam Kowalski, Coline Larmier, Fausto Malvagi, Grégory Millasseau, et al.

EPJ Nuclear Sci. Technol. **10**, 26 (2024)

https://doi.org/10.1051/epjn/2024028

EPJ Nuclear Sciences & Technologies

Available online at:
https://www.epj-n.org

Status and advances of Monte Carlo codes for particle transport simulation, edited by Andrea Zoia, Cheikh Diop and Cyrille de Saint Jean

REGULAR ARTICLE

OPEN ACCESS

Overview of TRIPOLI-5, a Monte Carlo code for HPC

Davide Mancusi[1], Emeric Brun[2], Benjamin Dechenaux[3], Kévin Fröhlicher[3], Thomas Gonçalves[4], Alexis Jinaphanh[1], Mikolaj Adam Kowalski[1], Coline Larmier[1], Fausto Malvagi[3], Grégory Millasseau[3], Wilfried Monange[3], Odile Petit[1] and Andrea Zoia[1,*]

[1] Université Paris-Saclay, CEA, Service d'Etudes des Réacteurs et de Mathématiques Appliquées, 91191 Gif-sur-Yvette, France
[2] CEA, DES, ISEC, DMRC, Univ. Montpellier, Marcoule, France
[3] Institut de Radioprotection et de Sûreté Nucléaire (IRSN), PSN-RES/SNC/LN, F-92260 Fontenay-aux-Roses, France
[4] Université Paris-Saclay, CEA, Service de Génie Logiciel pour la Simulation, 91191 Gif-sur-Yvette, France

Received: 10 June 2024 / Received in final form: 27 August 2024 / Accepted: 15 November 2024

Abstract. CEA, IRSN and EDF have joined forces and started the development of the TRIPOLI-5® Monte Carlo particle transport code in 2022, with the goal of performing massively parallel simulations on hybrid computing architectures. TRIPOLI-5 benefits from the experience gained from previous investigations conducted on the PATMOS mini-app, concerning the portability of particle-transport algorithms in High Performance Computing environments. Currently, the main focus of TRIPOLI-5 is on reactor physics applications, including multi-physics feedback for stationary and non-stationary configurations. In the long run, TRIPOLI-5 will eventually cover a broader range of applications (encompassing radiation shielding and nuclear instrumentation) and thus supersede the current-generation Monte Carlo codes TRIPOLI-4®, developed at CEA, and MORET6, developed at IRSN. In this paper, we provide an overview of the current status of TRIPOLI-5 and highlight the trends for future developments.

1 Introduction

The main advantage of Monte Carlo simulation for particle transport is the almost complete lack of approximations induced by discretization in either geometry and energy, and the capability of integrating the whole neutron and photon interaction laws provided in nuclear data libraries: these features make Monte Carlo methods the "golden standard" for numerical validation of faster but approximate solutions obtained using deterministic solvers, which rely on a discretization of the phase space. In France, CEA and IRSN develop their own legacy Monte Carlo codes, TRIPOLI-4® [1,2] and MORET6 [3,4], respectively: the former is a general-purpose code, covering radiation shielding and material activation, criticality-safety, reactor physics (including fuel depletion and kinetics), and nuclear instrumentation; the latter is mostly focused on criticality-safety and reactor physics applications.

Over the last decade, there has been a rapidly expanding interest in applying Monte Carlo simulation to reactor physics problems at the scale of the whole core, encompassing stationary configurations, fuel depletion, and kinetics, most often including thermal-hydraulics and thermo-mechanics feedback (see e.g. [5–12]). Such multi-physics simulations typically involve a large number of possibly time-dependent material compositions, as well as spatially continuous temperature, density, and nuclide concentration fields. These applications pose distinct challenges in terms of massive parallelism, memory footprint, and use of heterogeneous, quickly evolving computing architectures (including traditional processors, CPUs, as well as graphical processing units, GPUs). In order to cope with these challenges, several Monte Carlo codes such as Serpent [13], OpenMC [14], MCS [15] or MC21 [16] have been either extensively modified or built from scratch.

In this context, CEA, IRSN and EDF have joined forces in 2022 and started the development of TRIPOLI-5®, a new Monte Carlo code whose primary goal in the short term is to address reactor physics problems, integrating multi-physics feedback for stationary and non-stationary configurations. TRIPOLI-5 builds upon the experience accumulated with PATMOS [17], a Monte Carlo mini-app developed at CEA to explore the portability of particle-transport algorithms in High-Performance Computing (HPC) environments, including hybrid CPU/GPU architectures. A pilot version of TRIPOLI-5 will be released to beta-testers at the end of 2024, as a stepping stone towards the first official release covering stationary reactor problems.

In the short term, the main focus of the development efforts for TRIPOLI-5 will be, on the one hand, on multi-physics feedback, depletion calculations, and reactor kinetics. The target architectures will be hybrid, massively parallel CPU/GPU machines, as well as office workstations for more modest workloads. In the long run, the

* e-mail: andrea.zoia@cea.fr

development of TRIPOLI-5 will be eventually extended to cover a broader range of applications, encompassing nuclear instrumentation and radiation shielding with variance reduction. It is envisaged that TRIPOLI-5 will supersede the current-generation Monte Carlo codes TRIPOLI-4 and MORET6 at that stage.

In this paper, we provide a general overview of the present status of TRIPOLI-5, and an illustration of the major features of the code. This manuscript is structured as follows. In Section 2 we will give a summary of the key results achieved with the PATMOS mini-app. Then, we will discuss the general architecture, the main libraries and the current and future developments of TRIPOLI-5: the treatment of elementary data for particle transport will be described in Section 3, the geometry engines in Section 4, and the overall code architecture in Section 5. Preliminary work on benchmarking will be illustrated in Section 6. Conclusions and perspectives will be finally discussed in Section 7.

2 The legacy of the PATMOS mini-app

As mentioned above, the most urgent goal of the TRIPOLI-5 project is to cover reactor physics applications which lie beyond the reach of current-generation codes.

An important step towards this goal was represented by the development of the PATMOS mini-app [17], a Monte Carlo particle transport code whose goal was to port and adapt the traditional Monte Carlo algorithms to architectures used in high-performance supercomputers. PATMOS used simplified neutron physics; in spite of this, its algorithms were considered to be representative of a production-level Monte Carlo code and made it possible to quickly iterate on architectural solutions without the associated maintenance and quality-assurance burden. Many of the successful architectural choices that were made in PATMOS were carried over to TRIPOLI-5, and will be described in detail in the following sections.

Besides foreshadowing TRIPOLI-5, PATMOS was often used in technical collaborations. For example, we investigated the possibility to offload the computation of cross-section Doppler broadening to GPUs. A study was performed using several different GPU frameworks [18]. It was concluded at that time the generic frameworks such as Kokkos [19] or OpenACC [20] were not mature enough and were vastly outperformed by a refined CUDA implementation [21]. PATMOS was also used as a test application for custom-precision mathematical libraries [22].

The main achievements of PATMOS are massively parallel coupled calculations. The first one, realized in 2018–2019, is a depletion calculation of a modified version of the Hoogenboom–Martin–Petrovic benchmark [23]. We used the MENDEL depletion code for this test [24]. The system consists of 241 fuel assemblies, each containing 264 fuel pins. Each pin was axially discretized in 100 zones, yielding a total number of distinct depletable materials of approximately 6.3×10^6, which is about a factor of 1000 larger than the maximum number of depletable regions that can currently be handled by the legacy TRIPOLI-4 code. Although according to the benchmark specifications the temperature field in the fuel is uniform over the entire core, we have used the SIGMA1 method [25] to compute the cross sections, in order for the calculation to be representative of handling a realistic temperature field. We simulated 25 depletion steps up to a maximum burn-up of $35\,\mathrm{GW\,t_{HN}^{-1}}$, using 10^9 histories per depletion step. The calculation ran on CEA's Tera-1000-2 supercomputer [26] in 2019 and used about 8×10^4 Intel KNL cores for 12 h, for a total CPU time of about $1\,\mathrm{Mh_{CPU}}$. Additional, shorter test runs were made using up to 4×10^5 cores and 1.2×10^6 threads, and showed good scaling properties.

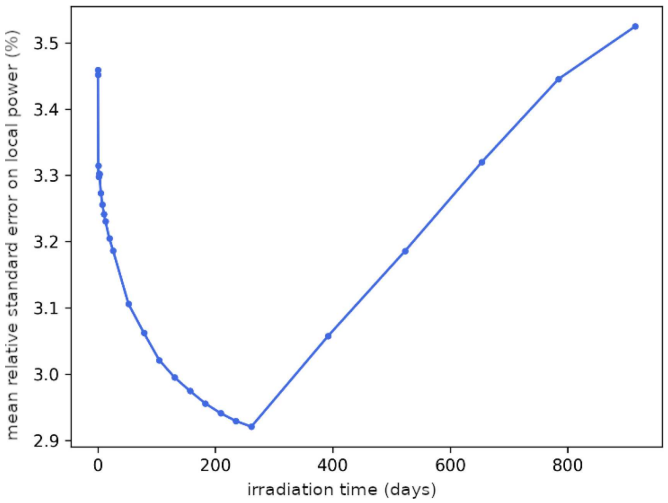

Fig. 1. Mean relative standard error on the local power deposition in the depletable regions, as a function of irradiation time, for the Hoogenboom–Martin–Petrovic depletion calculation performed by PATMOS.

The Hoogenboom–Martin–Petrovic depletion calculation employed four levels of parallelism. At the top level, the simulation used MPI [27] for inter-process communication. The world MPI communicator was split into 11 sub-communicators to perform statistically independent simulations and evaluate confidence intervals on the Monte Carlo results. Within each sub-communicator, each MPI process kept a copy of all the depletion tallies. Within each process, a thread pool managed by OpenMP [28] provided shared-memory parallelism. Finally, a vectorized version of the SIGMA1 method [17] was used to take advantage of the AVX-512 vector units of the KNL.

Figure 1 shows that the statistical uncertainty on the local power deposition (as measured by the relative standard error) was typically of the order of 3%; 96% of the scores had a relative standard error below 8%. Thus, for a typical target uncertainty of 1%, the CPU budget was estimated to be of the order of $10^7\,\mathrm{Mh_{CPU}}$. Overall, this study shows that the PATMOS software architecture is well-suited for large-scale depletion calculations.

More recently, in 2022, PATMOS was used to test the scalability of a large-scale core calculation with thermal-hydraulics feedback. The coupled calculation relied on the services provided by the C3PO platform [29], a Python library dedicated to the development of solver-independent coupling schemes, based on the exchange of

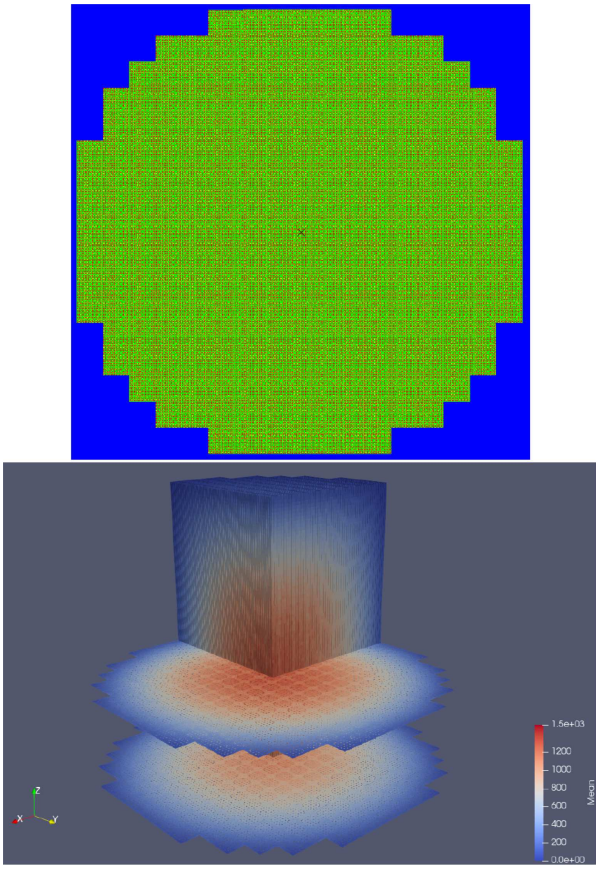

Fig. 2. Top: radial cut of the pressurized water reactor configuration chosen for the demonstration of the multi-physics capabilities of PATMOS. Bottom: power map at stationary conditions, corresponding to an integrated thermal power of 3 GW.

MEDCoupling fields [30]. Thermal-hydraulics was modelled using the THEDI multi-1D simplified solver [31]. The goal of the calculation was to compute the stationary state of a large, regular pressurized water reactor (PWR) core, consisting of 241 fuel assemblies, with an active height of 4 m, and consisting of 17×17 rods. The nominal thermal reactor power was chosen to be 3 GW.

The coupled simulation used an explicit first-order Euler scheme: PATMOS computed the deposited energy on a spatial mesh; these values were fed to the thermal-hydraulics solver, which computed the temperature and material density fields. These were then fed back to PATMOS in order to update the cross sections. A relaxation factor of 0.5 was used on the power distributions computed by PATMOS, to suppress spurious oscillations in the convergence phase. The simulation scheme was thus organized into outer iterations (one exchange of data between the two codes per iteration) and inner iterations (power iteration cycles to accumulate sufficient statistics and internal iterations of the thermal-hydraulics solver to reach the stationary state). Particle tracking was performed using the delta tracking algorithm, in view of the spatially continuous cross sections resulting from the temperature and density fields. The power, fuel temperature, moderator temperature and moderator density fields were discretized on a Cartesian mesh consisting of about 10^7 cells.

A total of 85 coupling iterations between neutron transport and thermal hydraulics were performed; each coupling iteration consisted of 50 cycles of power iteration in neutron transport, followed by a thermal-hydraulics stationary loop. Convergence of the power, temperature and density fields was attained after 8 coupling iterations. The neutron-transport part of the calculation used 2^{14} (16 384) threads, for a grand total of about 200 $\mathrm{kh_{cpu}}$ and 4×10^{11} neutron histories. Figure 2 illustrates the geometry used in the calculation and a color map of the calculated power distribution in the core in stationary conditions. We also performed weak-scaling tests on the coupled calculations, using 2^{10}, 2^{11}, 2^{12}, and 2^{13} threads; we found an efficiency of about 98% at 2^{14} threads, relative to the reference value at 2^{10} threads. This was considered very satisfactory and demonstrated that the PATMOS architecture is suitable for the massively parallel calculation schemes that TRIPOLI-5 aims at.

3 Physics

TRIPOLI-5 can perform k-eigenvalue (power iteration) calculations and fixed-source calculations. More simulation modes will be added in the future.

The TRIPOLI-5 module devoted to elementary data for particle transport is called DELOS. It provides the data model for all the nuclear, atomic, relaxation, and depletion data required for the neutron and photon transport calculation. The data are stored in memory in a format that is suitable for immediate Monte Carlo sampling, whenever relevant. Functions for sampling the distributions are also provided by DELOS.

3.1 Neutrons

In the current version of the code, nuclear data for neutrons are mostly read from ACE files[1]. Files coming from several libraries (ENDF-B, JEFF, and TENDL) and the two processing codes NJOY [33] and GAIA [34] have been tested so far. The ALEXANDRIA TRIPOLI-5 module is responsible for parsing the ACE files and filling the relevant DELOS data structures; it additionally performs coherence checks on the nuclear data and applies some minor fix-ups. If a piece of data does not uphold the required invariants, then the construction of the DELOS object fails and the data cannot be used in TRIPOLI-5. This approach (validation at the boundary) has the advantage of producing a sanitized library, which can be used by TRIPOLI-5 without having to worry about edge cases, inconsistencies, and so on. Once constructed, the DELOS objects are guaranteed to be sane, and are then serialized to disk. Subsequent TRIPOLI-5 runs can then read the data into memory without parsing and checking them.

[1] Probability tables for the unresolved resonance range might optionally come from CALENDF files [32], as discussed below.

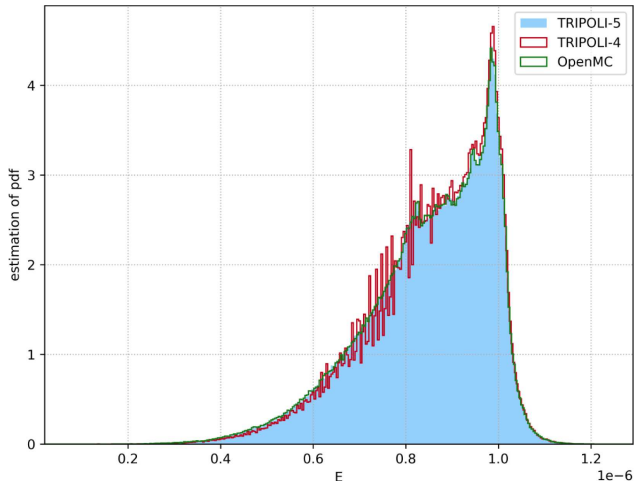

Fig. 3. Comparison between TRIPOLI-5, TRIPOLI-4 and OpenMC for the thermal inelastic scattering law of graphite, for incident neutrons having energy 1 eV, using the JEFF-3.3 nuclear data library. Figure taken from reference [36].

At the time of writing, there is ongoing work to couple TRIPOLI-5 to the nuclear data pre-processing code GALILÉE-1 [35], developed at CEA. GALILÉE-1 is currently used to process ENDF (and, in the near future, GNDS) evaluation files for TRIPOLI-4; in the short term, it will be also adopted to produce DELOS data, as an alternative to the use of ALEXANDRIA.

The fact that Monte Carlo codes rely on different formats for nuclear data (each associated to a distinct processing tool) may lead to subtle but systematic differences when comparing the results obtained for benchmark configurations, as illustrated in the following. In particular, evaluated cross sections in the ENDF format usually have a larger number of points than those produced in the ACE format, which is responsible for slight discrepancies in the interpolated values. Furthermore, in some cases, the format of the data influences the information accessible by the Monte Carlo code. In the case of incoherent elastic scattering in thermal scattering laws, the ACE format does not contain the Debye–Waller factor, contrary to the ENDF format. As a consequence, TRIPOLI-4 can sample the scattering cosine exactly, thanks to the access to the Debye–Waller factor, whereas TRIPOLI-5 (or any other code based on ACE format) must use discrete tables [36].

The sampling routines for neutron-nucleus interaction laws based on nuclear data libraries are basically complete, and include the "free-gas" model, the thermal scattering laws (TSL) for molecular binding or crystal effects (inelastic, coherent elastic and incoherent elastic reactions) [36], and the treatment of the unresolved resonance range (URR) [37] using the probability tables formalism [38]. Currently, two distinct probability tables models are supported: those produced by the PURR module of NJOY (based on Levitt's "ladder" method) [33], and those produced by CALENDF (based on Ribon's Gauss quadrature method) [32].

We have mentioned that the primary target of TRIPOLI-5 is represented by full-scale core calculations with thermal-hydraulics feedback and fuel depletion. In such calculations, it is not feasible to store all the required cross sections in memory at once, because of the large amount of nuclear data that must be loaded to treat all the nuclides present in the system at all the necessary temperatures. Instead of loading pre-tabulated cross sections at many temperatures, one can choose to load the cross sections at one temperature (for example 0 K) and compute the Doppler-broadened cross section at any other temperature on the fly. TRIPOLI-5 uses the SIGMA1 method [25] to perform the numerical integration of the Doppler integral. This results in much longer computation times, typically of the order of a factor of 10, compared to the use of pre-tabulated cross sections [18]; however, the computation of the SIGMA1 integral is a very suitable target for GPU offloading, as detailed in Section 5.2.

The Doppler broadening of the collision kernel relies on the Sampling of the Velocity of the Target nucleus (SVT) model [25], and resonant elastic cross sections of heavy nuclei are dealt with via the Doppler Broadening Rejection Correction (DBRC) method [39], whose implementation had been successfully probed in TRIPOLI-4 [40]. Furthermore, stochastic temperature mixing is currently being implemented.

Extensive verification tests of DELOS have been conducted on benchmark configurations through code-to-code comparisons involving TRIPOLI-4 and OpenMC [36,37]. For this purpose, nuclear data were processed with NJOY version 16.2 [33]; for TRIPOLI-5 and OpenMC, they were subsequently read from ACE files, whereas for TRIPOLI-4 they were read from ENDF files. The implementations of parsing and sampling for energy-angle distributions have been preliminarily verified using the Kolmogorov-Smirnov statistical test [41], for each nuclide of the JEFF-3.3 nuclear data library [42], over a set of incident neutron energies. Furthermore, we performed an additional set of code-to-code comparisons on single-isotope spherical configurations, resulting from a single-energy and isotropic neutron source, for 562 nuclides without TSL and 31 nuclides with TSL, for a total amount of about 5300 tests (without URR). The neutron flux in the sphere was decomposed over 616 equi-lethargic energy groups and was used as a fiducial quantity. The simulation results were tested for equality using the Holm-Bonferroni statistical test [43] on a family of Student t-tests (one Student test per energy bin). Overall, an excellent agreement was found with respect to OpenMC. A slightly less good (although globally satisfactory) agreement was found with respect to TRIPOLI-4; this is mostly due to the differences in nuclear-data pre-processing. For illustration, an example for graphite is shown in Figure 3.

The impact of URR was separately tested for the 443 nuclides of JEFF-3.3 and the 341 nuclides of ENDF/B-VIII.0 [44] that have URR data in the ACE format. Again, an excellent agreement was found between TRIPOLI-5 and OpenMC, when both codes were using PURR-processed URR data. Some discrepancies were spotted for other combinations of codes and URR models [37]. For illustration, an example concerning a nuclide having URR data is provided in Figure 4.

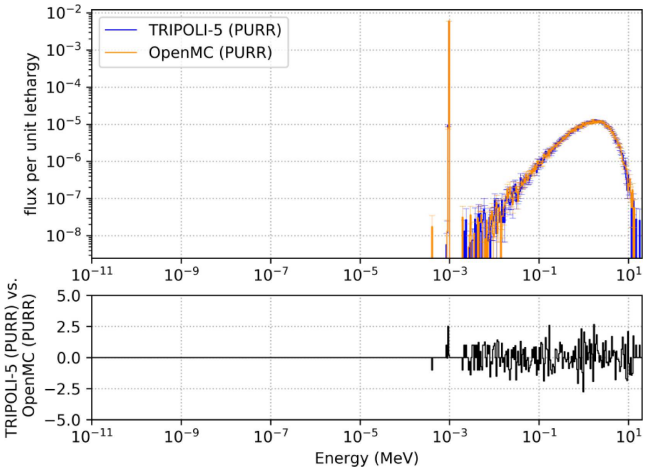

Fig. 4. Comparison of neutron flux per unit lethargy for a sphere containing ^{239}U, corresponding to an isotropic source at 1 keV. Both TRIPOLI-5 and OpenMC have been run using PURR probability tables. Figure taken from reference [37].

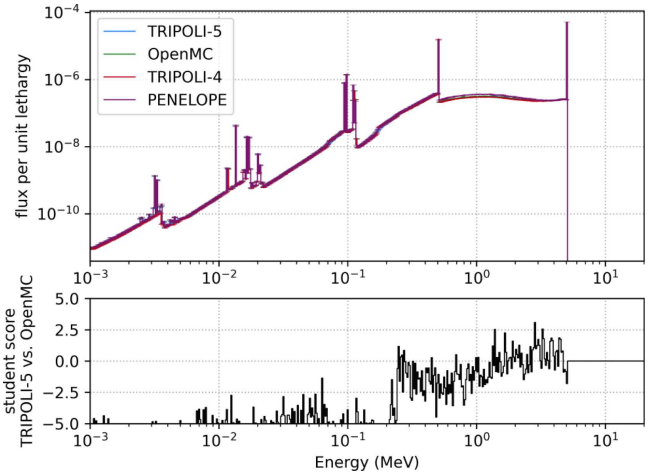

Fig. 5. Comparison of photon flux per unit lethargy for a sphere containing element U, corresponding to an isotropic source at 5 MeV (the thick-target bremsstrahlung model is activated). Slight discrepancies occur between TRIPOLI-5 and OpenMC or PENELOPE, due to stopping power data: OpenMC and PENELOPE compute these data internally, accounting for the electron density of the material, whereas TRIPOLI-5 reads values from the NIST ESTAR database. PENELOPE was modified to ignore delta-ray and triplet production electrons, since these are not included in OpenMC and TRIPOLI-5. Figure taken from reference [36].

3.2 Photons

A preliminary version of photon interactions has been also implemented, including photo-atomic reactions and the thick target bremsstrahlung (TTB) model, and is currently under extensive verification using the same approach as for neutron physics.

The photo-atomic interactions in TRIPOLI-5 are based on the EPICS library [45], but are supplemented by additional data from PENELOPE [46]. Unlike many other reactor physics codes, TRIPOLI-5 uses photon interaction cross sections under linear interpolation (as opposed to more common log–log interpolation). This causes, of course, an increase in memory burden, but it enables to pre-compute the total cross sections, which increases the efficiency of delta-tracking and makes it possible to treat photoelectric capture from each sub-shell as a separate reaction (analogously to neutron inelastic scattering). Furthermore, this choice enables the use of EPICS 2017 and 2023 libraries [47]. The reaction models are largely based on PENELOPE, with several simplifications. The angular distribution of photo-electron ignores relativistic effects, similarly to what is done in Serpent [48]. The Compton scattering model mixes elements from the PENELOPE implementation with the simpler method of EGS [49]. The thick target bremsstrahlung model is based on the Seltzer and Berger data [50] and is implemented via a pseudo-transport simulation, where the electron is not tracked, but its slowing-down is performed by Monte Carlo calculation. Emissions of a photon above a simulation energy threshold are considered as "hard" collisions and are processed explicitly. All other interactions are treated by a continuous slowing-down model. The production of delta-ray electrons is not included. This Monte Carlo model was chosen over the more common pre-tabulation of outgoing energy distribution of e.g. Serpent [48] and OpenMC, as it can be easily extended to include angular straggling, like in TRIPOLI-4, and other physics.

Since the treatment of photo-atomic interactions is less standardized than for neutrons, each Monte Carlo code uses slightly different approximations. While their effect is most often small enough not to be noticeable in practical problems, the differences are sufficiently large to lead to the failure of the Holm-Bonferroni statistical test. Therefore, we have decided to rely mostly on visual inspection of the simulation results. Differences among codes might stem e.g. from the use of the relativistic impulse approximation, the fluorescence model, stopping power data and model, etc. [36]. The elements selected for the verification were taken from the EPICS2014 library [45]. The code-to-code comparison covers TRIPOLI-4, OpenMC and PENELOPE. Similarly as for the case of neutron interactions, the fiducial quantity chosen for the benchmark is the photon flux per unit lethargy resulting from an isotropic and single-energy photon source placed at the center of a sphere: for illustration, an example is displayed in Figure 5.

Work is ongoing to finalize the implementation of photon interactions, covering in particular the production of photons from neutron reactions. Photo-nuclear reactions will be also implemented and tested with respect to TRIPOLI-4 on simple benchmarks [51].

4 Particle tracking and geometry

Particle tracking in TRIPOLI-5 is performed using either surface tracking [52] or delta tracking [53]. The latter has been shown to be highly effective for models having

repetitive patterns, such as lattices [54], or complex media composed of hundred of thousands of cells [55].

In the current version of the code, TRIPOLI-5 relies on two kinds of geometry models. In the wake of TRIPOLI-4, TRIPOLI-5 supports the ROOT geometry package [56]. The model is based on a set of constructive solid geometry (CSG) elements, organized into a hierarchy where each volume is fully contained by another volume. For reactor physics applications, a set of Python templates previously developed for TRIPOLI-4 simplifies the creation of common reactor geometries, based on simpler predefined patterns, including fuel pins, assemblies, and cores [1].

Furthermore, a native CSG model has been developed for TRIPOLI-5: the AGORA library. Similarly to CSG models available in other Monte Carlo codes such as MCNP [57], Serpent [13] or OpenMC [14], AGORA volumes are defined by combining regions, bounded by an elementary surface, using a Boolean algebra of intersection, union, and complement operators. Each volume is then associated to a material. AGORA supports the use of *universes*, i.e. sets of cells that can be used multiple times in the same geometry, and *lattices*, i.e. repetitive patterns of universes. For illustration, a radial cut of the AGORA geometry representing the Hoogenboom–Martin–Petrovic benchmark is provided in Figure 6. We are currently developing the use of a connectivity map, which was shown to considerably accelerate surface tracking [55]. For the visualization of the geometrical models, AGORA relies on an extended version of the T4G tool developed for TRIPOLI-4 [58,59].

The correctness and performance of AGORA are currently under investigation, with comparison to the ROOT geometry model of TRIPOLI-5 and various geometry models available in TRIPOLI-4 and OpenMC. The verification tests are based on the use of equivalent geometry models. In the simplest test, we compare the materials "seen" at a given location by different geometry engines. In more sophisticated tests, we also compare the lengths traveled by a particle in each material region, for a given starting point and direction. The results of these comparisons will be reported in a forthcoming paper.

Future work will focus on providing support for the legacy geometry models of TRIPOLI-4 and MORET.

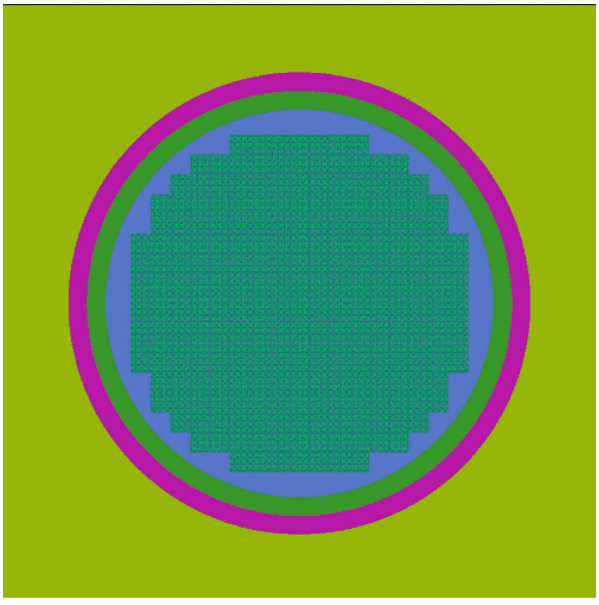

Fig. 6. Radial cut of the AGORA geometry corresponding to the Hoogenboom–Martin–Petrovic benchmark. Visualization is obtained using the extended version of the T4G tool.

5 Software architecture

The experience acquired with the PATMOS mini-app informs most of the architectural choices of TRIPOLI-5.

5.1 Modularity

TRIPOLI-5 is written in C++. It is conceived as a toolkit of libraries that users can combine to perform Monte Carlo particle transport simulations. It sensibly differs in this respect from TRIPOLI-4, which has the structure of a monolithic executable driven by an input file written in an application-specific language. This choice is motivated by the experience gained in coupling both TRIPOLI-4 [12] and PATMOS (see Sect. 2) to external thermal-hydraulics solvers. The development of the TRIPOLI-4 coupling was found to require very invasive code modifications, mostly because TRIPOLI-4 had not been designed with the goal of supporting multiple simulation runs within the same code execution. The necessary developments, which were made in the context of a Ph.D. thesis [60] on a topic branch, were never permanently merged in the development version of TRIPOLI-4, because the integration cost was estimated to be too high. In contrast, the library structure of TRIPOLI-5, with multiple entry points and a mostly stateless[2] workflow, makes the implementation of concentration, density and temperature updates very natural, provided that the code exposes a sufficiently flexible interface. This constraint has been integrated at the design stage.

With these reflections in mind, it was decided that users should always interact with TRIPOLI-5 via Python. This is in sharp contrast with the choices of legacy codes like TRIPOLI-4, MORET or MCNP, which are typically driven by a text-based input file, or even Geant4 [61], which is mostly C++-based. The TRIPOLI-5 approach is similar to the solution adopted by OpenMC, with the important difference that TRIPOLI-5 users directly interact with the simulation kernel. There is no intermediate storage of the input data. The Python code directly calls into the C++ kernel, via pybind11 bindings [62]. Any modification in the Python script is immediately reflected

[2] The code is designed with the goal of reproducibility, which we define as follows: the code execution flow can only depend on the input data (geometry, nuclear data, simulation parameters, etc.), and must not on any previous execution. This is easier to achieve if the code does not store any state; in practice, we store some nuclear data in memory across runs, to avoid expensive disk access. Also, the random-number generators are attached to the particles.

in the simulation. This eliminates a common cause of unintentional code misuse.

A modular structure makes it also easier to handle optional external dependencies. Our general strategy is to try to stay away from "dependency hell" by carefully auditing external dependencies before bringing them into TRIPOLI-5. This is especially a problem on the C++ side, where a modern package manager is still painfully missing. We tend to prefer header-only libraries to compiled libraries, and Python dependencies to C++ dependencies. Small dependencies are downloaded and compiled (if necessary) at TRIPOLI-5 build time, removing the necessity to install them separately. Large dependencies, such as ROOT [56] or MEDCoupling [30], are made optional. Finally, the modular structure of TRIPOLI-5 also makes it simpler to arrange for distributing some of the code libraries to partners, collaborators, and students.

As far as code input/output is concerned, we plan to build as much as possible on commonly used infrastructure. For example, all code results will be accessible as NumPy arrays from Python [63]. TRIPOLI-5 will also provide specific support for use in Jupyter notebooks [64], especially for visualization and post-processing purposes.

5.2 Parallelism

Modern massively parallel computers are commonly characterized by:

- multi-core CPUs with broad vector registers for single-instruction multiple-data (SIMD) operations;
- multiple CPUs per node, often with non-uniform memory access (NUMA);
- hybrid architectures consisting of CPUs and graphical processing units (GPUs);
- fast network connectivity between the compute nodes.

Each of these features grants access to some amount of parallelism, provided that the code is suitable structured to exploit it. The choices made in TRIPOLI-5 are the following:

- For vectorization, we consider portability to be more important than performance. Modern compilers provide some support for automatic vectorization. This is typically relevant in relatively simple situations, where the compiler is able to infer that the same operation is being performed on multiple data. In more complex situations, the detection of opportunities for automatic vectorization is sometimes defeated by incidental factors, such as control flow. It is possible to apply manual vectorization, but this results in less portable code. For these reasons, in TRIPOLI-5 we choose to rely on automatic vectorization, in the hope of reaping larger gains as compilers learn better automatic vectorization heuristics.
- The parallelism exposed by multi-core architectures is exploited using the OpenMP compiler directives for data parallelism [28]. This is relatively easy to do in Monte Carlo transport codes, because the innermost transport loops have an embarrassingly parallel structure.

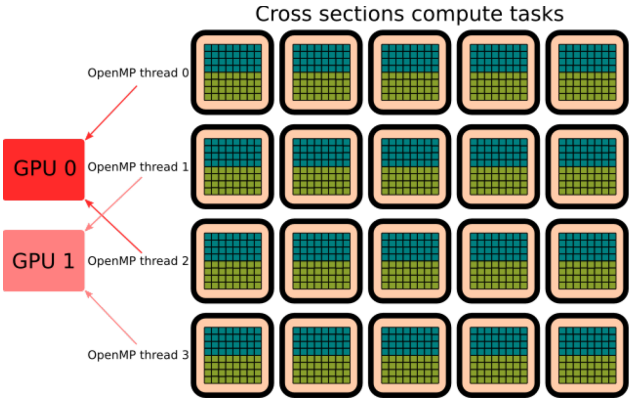

Fig. 7. Multi-GPU scheduling in TRIPOLI-5. In this example, 20 cross sections are offloaded to 2 GPUs.

- TRIPOLI-5 handles both non-uniform memory access and distributed computing with the use of MPI [27]. For example, on nodes with non-uniform memory access, it is generally recommended to spawn one MPI process per memory bank; thus, typical calculations on NUMA nodes employ more than one process per node.

5.3 GPU offloading

The use of GPUs requires a slightly longer discussion. The most straightforward implementation of particle transport Monte Carlo involves two nested loops: the outer loop runs over the particles to be transported, while the inner loop runs over the events encountered by a single particle (flights and collisions). This "history-based" code structure does not lend itself well to the data parallelism paradigm exposed by GPUs [65]. It is possible however to swap the two loops: the outer loop runs over the events for a packet of particles, while the inner loop samples the given event for all the particles in the same packet. This pushes data dependencies "upwards" in the control flow, and embarrassingly parallel loops "downwards". In this "event-based" structure, many parts of the transport algorithm expose some opportunities for data parallelism: for example, cross-section evaluation and geometrical queries are good candidates for GPU offloading. As mentioned in Section 3, the use of the SIGMA1 algorithm for the computation of Doppler-broadened cross sections is a major performance hot spot. Luckily, the SIGMA1 algorithm lends itself well to GPU offloading, since it basically consists of a summation loop. Therefore, the SIGMA1 computation was selected as the first target for a partial GPU port of TRIPOLI-5.

The SIGMA1 loop benefits from parallelism on GPU thanks to a distribution of iterations on GPU threads. TRIPOLI-5 supports the use of multiple GPU cards. Each OpenMP thread is assigned a GPU in a round-robin manner. Figure 7 illustrates the scheduling of computing tasks on GPU. Each large square represents a SIGMA1 cross-section calculation; each calculation is independent of the others and can be asynchronously sent to the GPU (using the scheduling capabilities of the backend and the GPU).

For each cross-section calculation, the internal loop is partitioned in blocks of 32 elements, represented here by the small squares.

It should be remarked that CPU and GPU code are written with distinct paradigms, requiring quite different algorithms, code flow and data structures. At the time of writing, several GPU frameworks attempt to bridge the gap between CPU and GPU programming. Some of them, such as CUDA [21], only target graphics cards developed by a specific vendor (Nvidia for CUDA). Other frameworks, such as Kokkos [19] or OpenACC [20], intend to be applicable to any card vendor, at the possible expense of some performance. At the time of writing, no single framework has emerged as a *de facto* standard for GPU programming; therefore, we do not wish to commit TRIPOLI-5 to a single framework. The CUDA and OpenACC SIGMA1 backends that were developed for PATMOS are in the process of being adapted to TRIPOLI-5; in the near future, a Kokkos backend will also be developed.

5.4 Development process

We conclude this section with a few remarks on the TRIPOLI-5 development process. The core team consists of developers from two research institutes, CEA and IRSN. Most of the collaboration happens on a private GitLab instance. Every contribution to the code is reviewed by one or more fellow developers, before integration. A continuous integration pipeline verifies that the code correctly compiles under different configuration conditions, and that all the tests succeed. The documentation, which is based on Sphinx, is also built at every merge request. Code formatting tools and linters are also part of the automatic verification performed in continuous integration. We believe that automating most of the verification in the process is essential for guaranteeing high code quality, especially in a context where developers are scattered across multiple institutions.

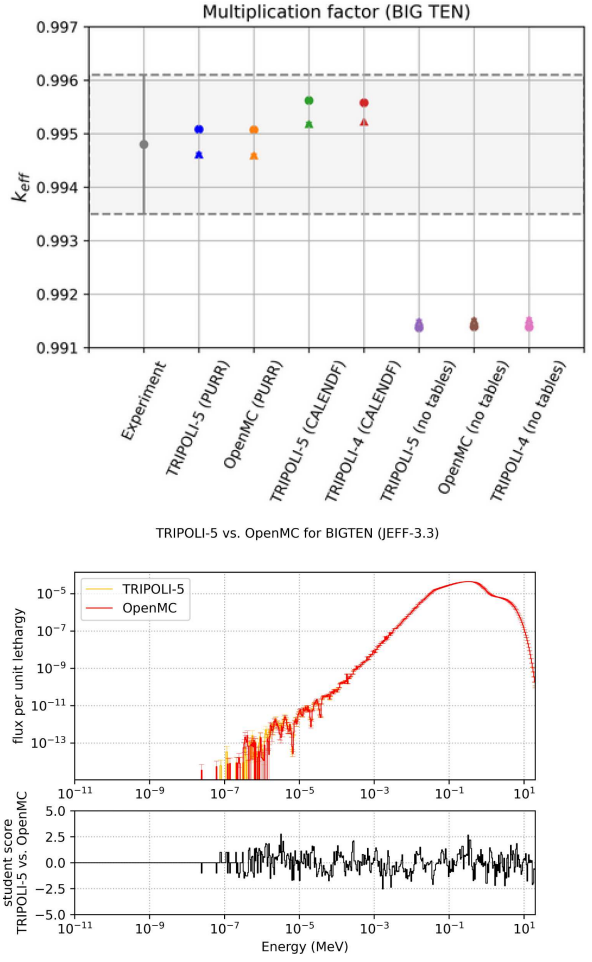

Fig. 8. Top: effective multiplication factor for the Big Ten benchmark, with and without various probability tables. Circles correspond to JEFF-3.3, and triangles to ENDF/B-VIII.0. Bottom: neutron spectra using ENDF/B-VIII.0. Figure adapted from reference [37].

6 Benchmarking

The verification and validation database for TRIPOLI-5 is being progressively enriched. A first set of benchmarks configurations has been drawn from the MCNP and TRIPOLI-4 criticality validation suite [66], which also belong to the ICSBEP collection [67]. This includes Godiva (HEU-MET-FAST-001), Jezebel (PU-MET-FAST-001), IMF03 (IEU-MET-FAST-003), IMF04 (IEU-MET-FAST-004), STACY36 (LEU-SOL-THERM-007, case 36), and Big Ten (IEU-MET-FAST-007-001). The TRIPOLI-5 results are compared to OpenMC and Serpent results. As an illustration, Figure 8 shows a comparison of TRIPOLI-5 and OpenMC for Big Ten, which is a cylindrical core enriched to $\sim 10\%_{\text{weight}}$, surrounded by an annulus of depleted uranium. This benchmark is known to be very sensitive to the unresolved resonance range and thus to the use of probability tables. In this regard, TRIPOLI-5 is today the only code which makes it possible to compare probability tables processed by NJOY (PURR module) [33] and CALENDF [32], coming from the same evaluation. Overall, a satisfactory agreement between TRIPOLI-5 and OpenMC was found. Meaningful discrepancies have been detected and attributed to the use of different probability tables models.

A second set of benchmarks is also under construction, based on frequently used configurations like PWR assembly cases and full-core PWR reactor cases (BEAVRS [68] and Watts Bar [69]). The results of TRIPOLI-5 results are compared to those of OpenMC and Serpent for k_{eff}, the flux of the fundamental k-eigenmode and reaction rates. In the future, the IRPhE benchmarks [70] will also be tackled.

7 Conclusions and future prospects

This paper introduces TRIPOLI-5, the new Monte Carlo particle transport code for reactor physics developed by CEA, IRSN, and EDF. TRIPOLI-5 is a next-generation toolkit library developed with the aim of gracefully scaling

from workstations to massively parallel supercomputers. The code currently implements most of the neutron and photo-atomic physics that is necessary for stationary reactor physics simulations.

In the coming years, the primary goal of TRIPOLI-5 will be to cover applications that are currently inaccessible by codes of the previous generation, such as TRIPOLI-4 and MORET6; this mainly concerns non-stationary (depletion and kinetic) calculations on a full-scale commercial reactor core, taking into account thermal-hydraulics feedback. The software architecture has been specifically designed for flexibility and maintainability. The long-term goal of the project is to replace the codes of the previous generation in other application domains, such as radiation shielding, instrumentation and criticality-safety. This will require the development of a large number of new features, including variance-reduction techniques, adjoint-weighted parameters, perturbations, electromagnetic shower, and charged-particle transport.

Acknowledgments

TRIPOLI-4® and TRIPOLI-5® are registered trademarks of CEA.

Funding

The authors thank EDF for partial financial support.

Conflicts of interest

The authors declare that they have no competing interests to report.

Data availability statement

No data are associated with the paper.

Author contribution statement

All authors contributed to the development and implementation of the technical work described in this manuscript. Writing was coordinated by D. Mancusi and A. Zoia. All authors proof-read the final version of the paper.

References

1. E. Brun, F. Damian, C.M. Diop, E. Dumonteil, F.-X. Hugot, C. Jouanne, Y.K. Lee, F. Malvagi, A. Mazzolo, O. Petit, J.C. Trama, T. Visonneau, A. Zoia, TRIPOLI-4®, CEA, EDF and AREVA reference Monte Carlo code, Ann. Nucl. Energy **82**, 151 (2015)
2. F.-X. Hugot, A. Jinaphanh, C. Jouanne, C. Larmier, Y.K. Lee, D. Mancusi, O. Petit, T. Visonneau, A. Zoia, Overview of the TRIPOLI-4 Monte Carlo code, version 12, EPJ Nucl. Sci. Technol. **10**, 17 (2024)
3. W. Monange, A. Bardelay, New capabilities of the MORET 6 Monte Carlo neutron transport code, in *PHYSOR 2022: International Conference on Physics of Reactors* (ANS – American Nuclear Society, Pittsburg, PA, USA, 2022)
4. W. Monange, New features of the Monte Carlo neutron transport code MORET 6, submitted to EPJ Nucl. Sci. Technol.
5. D. Kelly et al., Analysis of select BEAVRS PWR benchmark cycle 1 results using MC21 and OpenMC, in *PHYSOR2014, Kyoto, Japan* (2014)
6. D.P. Griesheimer, D.C. Carpenter, M.H. Stedry, Practical techniques for large-scale Monte Carlo reactor depletion calculatons, Prog. Nucl. Energy **101**, 409 (2017)
7. T.D.C. Nguyen, H. Lee, S. Choi, D. Lee, MCS/TH1D analysis of VERA whole-core multi-cycle depletion problems, Ann. Nucl. Energy **139**, 107271 (2020)
8. V.H. Sanchez-Espinoza, L. Mercatali, J. Leppänen, E. Hoogenboom, R. Vocka, J. Dufek, The McSAFE project high-performance Monte Carlo based methods for safety demonstration: from proof of concept to industry applications, in *EPJ Web Conf. PHYSOR2020 – International Conference on Physics of Reactors: Transition to a Scalable Nuclear Future* (EDP Sciences, 2021), Vol. 247, p. 06004
9. P.K. Romano, C.J. Josey, A.E. Johnson, J. Liang, Depletion capabilities in the OpenMC Monte Carlo particle transport code, Ann. Nucl. Energy **152**, 107989 (2021)
10. M. García, R. Tuominen, A. Gommlich, D. Ferraro, V. Valtavirta, U. Imke, P. Van Uffelen, L. Mercatali, V. Sanchez-Espinoza, J. Leppänen, S. Kliem, A Serpent2-SUBCHANFLOW-TRANSURANUS coupling for pin-by-pin depletion calculations in light water reactors, Ann. Nucl. Energy **139**, 107213 (2020)
11. D. Ferraro, M. García, V. Valtavirta, U. Imke, R. Tuominen, J. Leppänen, V. Sanchez-Espinoza, Serpent/SUBCHANFLOW pin-by-pin coupled transient calculations for a PWR minicore, Ann. Nucl. Energy **137**, 107090 (2020)
12. D. Mancusi, M. Faucher, A. Zoia, Monte Carlo simulations of the SPERT III E-core transient experiments, Eur. Phys. J. Plus **137**, 127 (2022)
13. J. Leppänen, M. Pusa, T. Viitanen, V. Valtavirta, T. Kaltiaisenaho, The Serpent Monte Carlo code: Status, development and applications in 2013, Ann. Nucl. Energy **82**, 142 (2015)
14. P.K. Romano, N.E. Horelik, B.R. Herman, A.G. Nelson, B. Forget, K. Smith, OpenMC: A state-of-the-art Monte Carlo code for research and development, Ann. Nucl. Energy **82**, 90 (2015)
15. H. Lee, W. Kim, P. Zhang, M. Lemaire, A. Khassenov, J. Yu, Y. Jo, J. Park, D. Lee, MCS – A Monte Carlo particle transport code for large-scale power reactor analysis, Ann. Nucl. Energy **139**, 107276 (2020)
16. D.P. Griesheimer, D.F. Gill, B.R. Nease, T.M. Sutton, M.H. Stedry, P.S. Dobreff, D.C. Carpenter, T.H. Trumbull, E. Caro, H. Joo, D.L. Millman, MC21 v.6.0 – A continuous-energy Monte Carlo particle transport code with integrated reactor feedback capabilities, Ann. Nucl. Energy **82**, 29 (2015)
17. E. Brun, S. Chauveau, F. Malvagi, PATMOS: A prototype Monte Carlo transport code to test high performance architectures, in *International Conference on Mathematics & Computational Methods Applied to Nuclear Science & Engineering, Jeju, Korea* (2017)
18. T. Chang, Evaluation of Programming Models for Manycore and/or Heterogeneous Architectures for Monte Carlo Neutron Transport Codes, Institut Polytechnique de Paris, 2020
19. H. Carter Edwards, C.R. Trott, D. Sunderland, Kokkos: Enabling manycore performance portability through polymorphic memory access patterns, J. Parallel Distrib. Comput. **74**, 3202 (2014)

20. S. Chandrasekaran and G. Juckeland, editors, *OpenACC for Programmers: Concepts and Strategies* (Addison-Wesley, Boston)
21. S. Cook, *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs* (Elsevier, MK, Amsterdam, Boston, 2013)
22. E. Brun, D. Defour, P. De Oliveira Castro, M. Istoan, D. Mancusi, E. Petit, A. Vaquet, A study of the effects and benefits of custom-precision mathematical libraries for HPC codes, IEEE Trans. Emerg. Top. Comput. **9**, 1467 (2021)
23. J. Eduard Hoogenboom, W.R. Martin, B. Petrovic, Monte Carlo performance benchmark for detailed power density calculation in a full size reactor core benchmark specifications, in *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2011)* (CiteSeer, Rio de Janeiro, RJ, Brazil, 2011)
24. S. Lahaye, A. Anne, R. Baron, T.-D. Huynh, A. Tsilanizara, New bateman equation solvers in MENDEL version 3.1, in *International Conference on Nuclear Criticality Safety (ICNC)* (2023)
25. D.E. Cullen, C.R. Weisbin, Exact Doppler broadening of tabulated cross sections, Nucl. Sci. Eng. **60**, 199 (1976)
26. TOP500 website, https://www.top500.org/system/179412 (visited on May 31, 2024)
27. Message Passing Interface Forum, MPI: A message-passing interface standard version 4.1, 2023
28. R. Chandra, editor, *Parallel Programming in OpenMP* (Morgan Kaufmann Publishers, San Francisco, CA, 2001)
29. C. Patricot, C3PO (Collaborative Code Coupling PlatfOrm), https://github.com/code-coupling/c3po
30. SALOME website, http://www.salome-platform.org (visited on April 14, 2021)
31. C. Patricot, THEDI: A multi-1D two-phase flow solver for neutronic codes, in *ICAPP 2019 – International Congress on Advances in Nuclear Power Plants, Juan-les-Pins, France* (2019)
32. J.-C. Sublet, P. Ribon, M. Coste-Delclaux, CALENDF/2010: User manual, Technical report CEA-R-6277, CEA, France, 2011
33. D.W. Muir et al., The NJOY nuclear data processing system, version 2016, Technical report LA-UR-17-20093, Los Alamos National Laboratory, USA, 2019
34. G. Ferran, W. Haeck, M. Gonin, Development progress of the GAIA nuclear data processing software, Nucl. Data Sheets **118**, 491 (2014)
35. M. Coste-Delclaux, C. Jouanne, C. Mounier, GALILÉE-1: Verification and processing system for evaluated data, in *SNA+MC2024, Joint International Conference on Supercomputing in Nuclear Applications + Monte Carlo, Paris, France* (2024)
36. C. Larmier, M.A. Kowalski, A. Jinaphanh, V. Franchini, D.Q.D. Nguyen, F. Malvagi, D. Mancusi, A. Zoia, Preliminary code-to-code comparisons for the implementation of neutron and photon physics in the new Monte Carlo transport code TRIPOLI-5®, in *M&C 2023 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, Niagara Falls, ON, Canada* (2023)
37. C. Montecchio, C. Larmier, D. Mancusi, A. Zoia, Benchmarking of probability tables with TRIPOLI-5®, EPJ Web of Conferences **302**, 07005 (2024)
38. T.M. Sutton and F.B. Brown, *Implementation of the Probability Table Method in a Continuous-Energy Monte Carlo Code System* (United States, 1998)
39. W. Rothenstein, Proof of the formula for the ideal gas scattering kernel for nuclides with strongly energy dependent scattering cross sections, Ann. Nucl. Energy **31**, 9 (2004)
40. A. Zoia, E. Brun, C. Jouanne, F. Malvagi, Doppler broadening of neutron elastic scattering kernel in TRIPOLI-4®, Nucl. Energy **54**, 218 (2013)
41. J.L. Hodges, The significance probability of the SMIRNOV two-sample test, Eur. Phys. J. A **3**, 469 (1958)
42. A.J.M. Plompen, O. Cabellos, C. De Saint Jean, M. Fleming, A. Algora, M. Angelone, P. Archier, E. Bauge, O. Bersillon, A. Blokhin, F. Cantargi, A. Chebboubi, C. Diez, H. Duarte, E. Dupont, J. Dyrda, B. Erasmus, L. Fiorito, U. Fischer, D. Flammini, D. Foligno, M.R. Gilbert, J.R. Granada, W. Haeck, F.-J. Hambsch, P. Helgesson, S. Hilaire, I. Hill, M. Hursin, R. Ichou, R. Jacqmin, B. Jansky, C. Jouanne, M.A. Kellett, D.H. Kim, H.I. Kim, I. Kodeli, A. J. Koning, A. Konobeyev, A. Yu, S. Kopecky, B. Kos, A. Krása, L.C. Leal, N. Leclaire, P. Leconte, Y.O. Lee, H. Leeb, O. Litaize, M. Majerle, J.I. Márquez Damián, F. Michel-Sendis, R.W. Mills, B. Morillon, G. Noguère, M. Pecchia, S. Pelloni, P. Pereslavtsev, R.J. Perry, D. Rochman, A. Röhrmoser, P. Romain, P. Romojaro, D. Roubtsov, P. Sauvan, P. Schillebeeckx, K.H. Schmidt, O. Serot, S. Simakov, I. Sirakov, H. Sjöstrand, A. Stankovskiy, J.C. Sublet, P. Tamagno, A. Trkov, S. van der Marck, F. Álvarez-Velarde, R. Villari, T.C. Ware, K. Yokoyama, G. Žerovnik, The joint evaluated fission and fusion nuclear data library, JEFF-3.3, Eur. Phys. J. A **56**, 181 (2020)
43. S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Stat. **6**, 65 (1979)
44. D.A. Brown, M.B. Chadwick, R. Capote, A.C. Kahler, A. Trkov, M.W. Herman, A.A. Sonzogni, Y. Danon, A.D. Carlson, M. Dunn, D.L. Smith, G.M. Hale, G. Arbanas, R. Arcilla, C.R. Bates, B. Beck, B. Becker, F. Brown, R.J. Casperson, J. Conlin, D.E. Cullen, M.A. Descalle, R. Firestone, T. Gaines, K.H. Guber, A.I. Hawari, J. Holmes, T.D. Johnson, T. Kawano, B.C. Kiedrowski, A.J. Koning, S. Kopecky, L. Leal, J.P. Lestone, C. Lubitz, J.I. Márquez Damián, C.M. Mattoon, E.A. McCutchan, S. Mughabghab, P. Navratil, D. Neudecker, G.P.A. Nobre, G. Noguere, M. Paris, M.T. Pigni, A.J. Plompen, B. Pritychenko, V.G. Pronyaev, D. Roubtsov, D. Rochman, P. Romano, P. Schillebeeckx, S. Simakov, M. Sin, I. Sirakov, B. Sleaford, V. Sobes, E.S. Soukhovitskii, I. Stetcu, P. Talou, I. Thompson, S. van der Marck, L. Welser-Sherrill, D. Wiarda, M. White, J.L. Wormald, R.Q. Wright, M. Zerkle, G. Žerovnik, Y. Zhu, ENDF/B-VIII.0: The 8th major release of the nuclear reaction data library with CIELO-project cross sections, new standards and thermal scattering data, Nucl. Data Sheets **148**, 1 (2018)
45. D.E. Cullen, EPICS2014: Electron photon interaction cross sections, Technical report IAEANDS-218, International Atomic Energy Agency, 2015
46. J. Baró, J. Sempau, J.M. Fernández-Varea, F. Salvat, PENELOPE: An algorithm for Monte Carlo simulation of the penetration and energy loss of electrons and positrons in matter, Nucl. Instrum. Methods Phys. Res. Sect. B Beam Interact. Mater. At. **100**, 31 (1995)

47. D.E. Cullen, EPICS2023: August 2023 status report, Technical report IAEA-NDS-0242, International Atomic Energy Agency, 2023
48. T. Kaltiaisenaho, Photon transport physics in Serpent 2 Monte Carlo code, Comput. Phys. Commun. **252**, 107143 (2020)
49. Y. Namito, S. Ban, H. Hirayama, Implementation of the Doppler broadening of a Compton-scattered photon into the EGS4 code, Nucl. Instrum. Methods Phys. Res. Sect. Accel. Spectrometers Detect. Assoc. Equip. **349**, 489 (1994)
50. S.M. Seltzer, M.J. Berger, Bremsstrahlung spectra from electron interactions with screened atomic nuclei and orbital electrons, Nucl. Instrum. Methods Phys. Res. Sect. B Beam Interact. Mater. At. **12**, 95 (1985)
51. T.K. Tuyet, A. Jinaphanh, C. Jouanne, F. Gérardin, S. Lemaire, A. Zoia, Comparison of the TRIPOLI-4®, DIANE, MCNP6 Monte Carlo codes on the Barber & George benchmark for photonuclear reactions, Nucl. Sci. Eng. **198**, 319 (2024)
52. I. Lux, L. Koblinger, *Monte Carlo Particle Transport Methods: Neutron and Photon Calculations*, 1st edn. (CRC Press, 2018)
53. E. Woodcock, T. Murphy, P. Hemmings, S. Longworth, Techniques used in the GEM code for Monte Carlo neutronics calculations in reactors and other systems of complex geometry, in *Proc. Conf. Applications of Computing Methods to Reactor Problems* (Argonne National Laboratory, 1965), Vol. 557
54. J. Leppänen, Performance of Woodcock delta-tracking in lattice physics applications using the Serpent Monte Carlo reactor physics burnup calculation code, Ann. Nucl. Energy **37**, 715 (2010)
55. H. Belanger, C. Larmier, D. Mancusi, A. Zoia, Optimization of particle tracking methods for stochastic media, in *PHYSOR2022, International Conference on the Physics of Reactors, Pittsburgh, USA* (2022)
56. R. Brun, F. Rademakers, ROOT – An object oriented data analysis framework, Nucl. Instrum. Methods Phys. Res. Sect. Accel. Spectrometers Detect. Assoc. Equip. **389**, 81 (1997)
57. J. Kulesza et al., MCNP® code version 6.3.0 theory & user manual. Technical report LA-UR-22-30006, Los Alamos National Laboratory, USA, 2022
58. F.-X. Hugot, Y.-K. Lee, A New prototype display tool for the Monte Carlo particle transport code TRIPOLI-4, Prog. Nucl. Sci. Technol. **2**, 851 (2011)
59. Y.-K. Lee and F.-X. Hugot, TRIPOLI-4 Monte Carlo code verification and validation using T4G tool, in *ICONE 31, 31st Int. Conf. on Nuclear Engineering, Prague, Czech Republic* (2024)
60. M. Faucher, Coupling between Monte Carlo neutron transport and thermal-hydraulics for the simulation of transients due to reactivity insertions, Ph.D. dissertation, Université Paris-Saclay, Paris, 2019
61. J. Allison, K. Amako, J. Apostolakis, P. Arce, M. Asai, T. Aso, E. Bagli, A. Bagulya, S. Banerjee, G.J.N.I. Barrand, B.R. Beck, Recent developments in Geant4, Nucl. Instrum. Methods Phys. Res. Sect. Accel. Spectrometers Detect. Assoc. Equip. **835**, 186 (2016)
62. W. Jakob, J. Rhinelander, D. Moldovan, PYBIND11 – seamless operability between C++11 and python, https://github.com/pybind/pybind11 (2017)
63. C.R. Harris, K. Jarrod Millman, S.J. Van Der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N.J. Smith, R. Kern, M. Picus, S. Hoyer, M.H. Van Kerkwijk, M. Brett, A. Haldane, J.F. Del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, T.E. Oliphant, Array programming with NumPy, Nature **585**, 357 (2020)
64. B.E. Granger and F. Pérez, Jupyter: Thinking and storytelling with code and data, Comput. Sci. Eng. **23**, 7 (2021)
65. P.K. Romano, A.R. Siegel, Limits on the efficiency of event-based algorithms for Monte Carlo neutron transport, Nucl. Eng. Technol. **49**, 1165 (2017)
66. E. Brun, A. Zoia, J.-C. Trama, S. Lahaye, Y. Nagaya, Inter-code comparison of TRIPOLI and MVP on the MCNP criticality validation suite, in *ICNC-2015, International Conference on Nuclear Criticality Safety, Charlotte, USA* (2015)
67. Nuclear Energy Agency, *International Criticality Safety Benchmark Evaluation Project Handbook* (OECD), https://doi.org/10.1787/110ba6fc-en
68. N.E. Horelik, B.R Herman, B. Forget, K. Smith, Benchmark for Evaluation and Validation of Reactor Simulations (BEAVRS), v1.0.1, in *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C), Sun Valley, ID, USA* (2013)
69. T. Albagami, P. Rouxelin, A. Abarca, D. Holler, L. Moloko, M. Avramova, K. Ivanov, A. Godfrey, S. Palmtag, TVA watts bar unit 1 multi-physics multi-cycle depletion benchmark version 2.3.3, Technical report NEA/EGMPEBV/DOC, The Organization for Economic Cooperation and Development (OECD) Nuclear Energy Agency (NEA), 2022
70. Nuclear Energy Agency, *International Reactor Physics Evaluation Project Handbook* (OECD), https://doi.org/10.1787/8d549c0f-en

Cite this article as: Davide Mancusi, Emeric Brun, Benjamin Dechenaux, Kévin Fröhlicher, Thomas Gonçalves, Alexis Jinaphanh, Mikolaj Adam Kowalski, Coline Larmier, Fausto Malvagi, Grégory Millasseau, Wilfried Monange, Odile Petit, and Andrea Zoia. Overview of TRIPOLI-5, a Monte Carlo code for HPC, EPJ Nuclear Sci. Technol. **10**, 26 (2024)