Recent developments in Geant4

J. Allison [a,b], K. Amako [c,a], J. Apostolakis [d], P. Arce [e], M. Asai [f], T. Aso [g], E. Bagli [h], A. Bagulya [i], S. Banerjee [j], G. Barrand [k], B.R. Beck [l], A.G. Bogdanov [m], D. Brandt [n], J.M.C. Brown [o], H. Burkhardt [d], Ph. Canal [j], D. Cano-Ott [p], S. Chauvie [q], K. Cho [r], G.A.P. Cirrone [s], G. Cooperman [t], M.A. Cortés-Giraldo [u], G. Cosmo [d], G. Cuttone [s], G. Depaola [x], L. Desorgher [bt], X. Dong [t], A. Dotti [f], V.D. Elvira [j], G. Folger [d], Z. Francis [v], A. Galoyan [w], L. Garnier [k,1], M. Gayer [d], K.L. Genser [j], V.M. Grichine [i,d], S. Guatelli [y,z], P. Guèye [aa], P. Gumplinger [ab], A.S. Howard [ac], I. Hřivnáčová [ad], S. Hwang [r,ae], S. Incerti [af,ag], A. Ivanchenko [ah,a,d], V.N. Ivanchenko [d,a,ai], F.W. Jones [ab], S.Y. Jun [j], P. Kaitaniemi [aj,ak,2], N. Karakatsanis [al,am,3], M. Karamitrosi [an], M. Kelsey [f], A. Kimura [ao], T. Koi [f], H. Kurashige [ap], A. Lechner [d], S.B. Lee [aq], F. Longo [ar,as], M. Maire [at,a], D. Mancusi [au], A. Mantero [av], E. Mendoza [p], B. Morgan [aw], K. Murakami [c], T. Nikitina [d], L. Pandola [s], P. Paprocki [d,4], J. Perl [f], I. Petrović [ax], M.G. Pia [ay], W. Pokorski [d], J.M. Quesada [u], M. Raine [az], M.A. Reis [ba,bb], A. Ribon [d], A. Ristić Fira [ax], F. Romano [s], G. Russo [s,bc], G. Santin [bd,be], T. Sasaki [c,bf], D. Sawkey [bg], J.I. Shin [aq,5], I.I. Strakovsky [bh], A. Taborda [ba], S. Tanaka [bi], B. Tomé [bj], T. Toshito [bk], H.N. Tran [bl,bm], P.R. Truscott [bn], L. Urban [a], V. Uzhinsky [bo], J.M. Verbeke [l], M. Verderi [bp], B.L. Wendt [bq], H. Wenzel [j], D.H. Wright [f,*], D.M. Wright [l], T. Yamashita [br], J. Yarba [j], H. Yoshida [bs,a]

[a] Geant4 Associates International Ltd., 9 Royd Terrace, Hebden Bridge HX7 7BT, United Kingdom
[b] The University of Manchester, School of Physics and Astronomy, Manchester M13 9PL, United Kingdom
[c] KEK, 1-1 Oho, Tsukuba, Ibaraki 305-0801, Japan
[d] CERN, 1211 Genéve 23, Switzerland
[e] CIEMAT, Medical Applications Unit, Avenida Complutense 40, 28040 Madrid, Spain
[f] SLAC National Accelerator Laboratory, 2575 Sand Hill Road, Menlo Park, CA 94025, USA
[g] National Institute of Technology, Toyama College, 1-2 Ebie Neriya, Imizu, Toyama 9330293, Japan
[h] INFN Sezione di Ferrara, Via Saragat 1, 44122 Ferrara, Italy
[i] Lebedev Physical Institute, Leninskii Pr. 53, Moscow 119991, Russia
[j] Fermi National Accelerator Laboratory, P.O. Box 500, Batavia, IL 60510, USA
[k] IN2P3/LAL, Universite Paris-Sud, Orsay, France
[l] Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA 94550, USA
[m] National Research Nuclear University (Moscow Engineering Physics Institute), Kashirskoe shosse 31, Moscow 115409, Russia
[n] SSW Trading, Am Knick 4, Oststeinbek, Germany
[o] Queen's University Belfast, School of Mathematics and Physics, University Road, Belfast, Northern Ireland BT7 1NN, United Kingdom
[p] CIEMAT, Centro de Investigaciones Energéticas Medioambientales y Tecnólogicas, Avenida Complutense 40, 28040 Madrid, Spain
[q] Sante Croce e Carle Hospital, Via Coppino 26, I-12100 Cuneo, Italy
[r] National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information, 245 Daehak-ro, Yuseonggu, Daejeon 34141, Republic of Korea
[s] Istituto Nazionale di Fisica Nucleare, Laboratori Nazionali del Sud, Via Santa Sofia 62, 95123 Catania, Italy
[t] Northeastern University, College of Computer and Information Science, 202-WVH, Boston, MA 02481, USA
[u] Universidad de Sevilla, Departamento de Física Atómica, Molecular y Nuclear, Apdo. 1065, E-41080 Sevilla, Spain
[v] Universite Saint Joseph, Department de Physique, Beirut, Lebanon
[w] Veksler and Baldin Laboratory of High Energy Physics, Joint Institute for Nuclear Research, Joliot-Curie 6, Dubna, Moscow region 141980, Russia
[x] Universidad Nacional de Córdoba – FaMAF, Medina Allende s/n., Córdoba, Argentina
[y] University of Wollongong, Centre for Medical Radiation Physics, Northfields Avenue, Wollongong NSW 2522, Australia
[z] Illawarra Health and Medical Research Institute, Northfields Avenue, Wollongong NSW 2522, Australia
[aa] Hampton University, Physics Department, 100 E. Queen Street, Hampton, VA 23668, USA
[ab] TRIUMF, 4004 Wesbrook Mall, Vancouver, Canada BC V6T 2A3
[ac] ETH Zürich, IPP, Otto-Stern-Weg 5, 8093 Zürich, Switzerland
[ad] Institut de Physique Nucléaire, Université Paris-Sud, CNRS-IN2P3, 15 rue Georges, Clemenceau, 91406 Orsay Cedex, France
[ae] University of Science and Technology, 217 Gajeong-ro Yuseonggu, Daejeon, Republic of Korea
[af] CNRS-IN2P3, CENBG, UMR 5797, F-33170 Gradignan, France
[ag] Université Bordeaux, CENBG, UMR 5797, F-33170 Gradignan, France
[ah] Centre d'Etudes Nucléaires de Bordeaux Gradignan, 19 Chemin du Solarium, 33175 Gradignan, France
[ai] Ecoanalytica, Moscow 119899, Russia

[aj] *CEA/Saclay, Centre de Saclay, Irfu/SPhN, F-91191 Gif-sur-Yvette, France*
[ak] *Helsinki Institute of Physics, P.O. Box 64, Gustaf Hällström Street 2, FI-00014, University of Helsinki, Finland*
[al] *National Technical University of Athens, School of Electrical and Computer Engineering, 9 Iroon Polytechniou Street, Zografos, Athens 15780, Greece*
[am] *Johns Hopkins University, School of Medicine, 601 N. Caroline Street, Baltimore, MD 21287, USA*
[an] *Notre Dame University, Notre Dame Radiation Laboratory, Notre Dame, IN 46556, USA*
[ao] *Ashikaga Institute of Technology, Omae 268-1, Ashikaga, Tochigi 326-8558, Japan*
[ap] *Kobe University, 1-1 Rokkodai, Nada, Kobe 657, Japan*
[aq] *National Cancer Center, 323 Insa-ro, Ilsandong-gu, Goyang-Si, Gyeonggi-do 10408, Republic of Korea*
[ar] *University of Trieste, Department of Physics, Via Valerio 2, 34127 Trieste, Italy*
[as] *INFN, Trieste, Via Valerio 2, 34127 Trieste, Italy*
[at] *IN2P3/LAPP, 74941 Annecy-le-vieux, France*
[au] *CEA, Centre de Saclay, DEN/DANS/DM2S/SERMA/LTSD, 91191 Gif-sur-Yvette CEDEX, France*
[av] *SWHARD srl, Via greto di Cornigliano 6R, Genova, Italy*
[aw] *University of Warwick, Department of Physics, Gibbet Hill Road, Coventry CV4 7AL, United Kingdom*
[ax] *University of Belgrade, Vinča Institute of Nuclear Sciences, P.O. Box 522, 11001 Belgrade, Serbia*
[ay] *INFN Sezione di Genova, Via Dodecaneso 33, 16136 Genova, Italy*
[az] *CEA, DAM, DIF, F-91297 Arpajon, France*
[ba] *C2TN, Instituto Superior Técnico, Universidade de Lisboa, Campus Tecnológico e Nuclear, EN10 km139.7, 2685-066 Bobadela LRS, Portugal*
[bb] *IEQUALTECS, Lda, Rua Dr. Francisco Sá Carneiro, 36, 2500-065 S. Gregório CLD, Portugal*
[bc] *IBFM-CNR, Contrada Pietrapollastra, Pisciotta, 90020 Cefalu, Italy*
[bd] *European Space Agency/ESTEC, Keplerlaan 1, 2201AZ Noordwijk, The Netherlands*
[be] *RHEA System BV, Jonckerweg 18, 2201DZ, Noordwijk, The Netherlands*
[bf] *SOKENDAI, 1560-35, Hayama, Kanagawa 240-0193, Japan*
[bg] *Varian Medical Systems, 3120 Hansen Way, Palo Alto, CA 94304, USA*
[bh] *The George Washington University, Department of Physics, Washington, DC 20052, USA*
[bi] *Ritsumeikan University, College of Information Science and Engineering, Noji-higashi 1-1-1, Kusatsu, Shiga 525-8577, Japan*
[bj] *LIP-Lisboa/IST, Av. Elias Garcia, 14-1, 1000-149 Lisboa, Portugal*
[bk] *Nagoya Proton Therapy Center, 1-1-1 Hirate-cho, Kita-ku, Nagoya 462-8508, Japan*
[bl] *Ton Duc Thang University, Division of Nuclear Physics, 19 Nguyen Huu Tho Street, Tan Phong Ward, District 7, Ho Chi Minh City, Vietnam*
[bm] *Ton Duc Thang University, Faculty of Applied Sciences, 19 Nguyen Huu Tho Street, Tan Phong Ward, District 7, Ho Chi Minh City, Vietnam*
[bn] *Kallisto Consultancy Ltd., Farnborough, Hampshire GU14 9AJ, United Kingdom*
[bo] *Laboratory of Information Technologies, Joint Institute for Nuclear Research, Joliot-Curie 6, Dubna, Moscow region 141980, Russia*
[bp] *Université Paris-Saclay, Laboratoire Leprince-Ringuet, Ecole Polytechnique, CNRS/IN2P3, 91128 Palaiseau, France*
[bq] *Idaho State University, 921 South 8th Avenue, Pocatello, ID 83209, USA*
[br] *Hyogo Ion Beam Medical Center, Tatsuno, Japan*
[bs] *Naruto University of Education, Takashima, Naruto-shi, Japan*
[bt] *Institut de Radiophysique, Rue du Grand-Pré 1 CH-1007, Lausanne, Switzerland*

A R T I C L E   I N F O

A B S T R A C T

Geant4 is a software toolkit for the simulation of the passage of particles through matter. It is used by a large number of experiments and projects in a variety of application domains, including high energy physics, astrophysics and space science, medical physics and radiation protection. Over the past several years, major changes have been made to the toolkit in order to accommodate the needs of these user communities, and to efficiently exploit the growth of computing power made available by advances in technology. The adaptation of Geant4 to multithreading, advances in physics, detector modeling and visualization, extensions to the toolkit, including biasing and reverse Monte Carlo, and tools for physics and release validation are discussed here.

1. The evolution of Geant4

A major trend in modern experimental science is the increased reliance upon simulation to design complex detectors and interpret the data they produce. Indeed, simulation has become mission-critical in fields such as high energy physics and space science. Another trend is the rapid increase in computing power due to faster processors, multi-core architectures and distributed computing. At the same time, advances in memory technology have not kept pace and the amount of memory available per CPU cycle has decreased. These trends have led to a re-examination of some of the basic assumptions of simulation computing, and to the evolution of the Geant4 simulation toolkit.

The toolkit approach has enabled Geant4 to serve a wide variety of user communities and to change as users' needs change. Now 16 years since its initial public release, Geant4 continues to be the simulation engine of choice for high energy physics experiments at the LHC. ESA and NASA have used and continue to use Geant4 in the design of spacecraft and the estimation of radiation doses received by astronauts and electronic components. It is also used extensively in medical physics applications such as particle beam

therapy, microdosimetry and radioprotection. The basic extensibility of the toolkit has facilitated its expansion into new user domains, such as biochemistry, material science and non-destructive scanning.

Common to nearly all these domains, but especially true for high energy physics, is the demand for increasingly detailed geometries and more sophisticated physical models. This in turn drives the need for more CPU cycles, and the relative decrease of memory drives the need for more efficient memory management.

It became clear that GEANT4 could meet these challenges by adopting the multithreading approach and exploiting the multi-core architectures that have now become commonplace. While significant effort went into the implementation of multithreading, the object-oriented design of the toolkit made the changes much less intrusive than might have been expected. Section 2 of this paper will discuss the details and performance of this implementation.

The remainder of the paper will deal with other improvements in the toolkit since the last GEANT4 general paper [1]. Section 3 covers improvements in kernel functionalities. Among these are new tracking and scoring capabilities, improved geometry models which have resulted in faster, more versatile experimental representations, and improved visualization techniques which provide users with more powerful ways to view them. Section 4 provides updates in electromagnetic and hadronic physics modeling with discussions of recently introduced models, improvements in existing models and physics lists, and results from comparisons to data. Extensions of the toolkit, including a new generic biasing framework, reverse Monte Carlo, native analysis capability, and improved examples, are covered in Section 5. Section 6 describes the extensive improvements in testing and validation. Modern web interfaces and standard testing tools have made it easier for users and developers alike to evaluate the performance of GEANT4. The adoption of modern build tools addressed the need for flexible configuration and support on various computing platforms, as well as the ever-increasing number of data files needed by physics models. This paper concludes in Section 7 with a brief summary of GEANT4 progress and a discussion of prospects for the next decade.

A primer of terms. A number of terms within GEANT4 have meanings which differ somewhat from general usage. Although defined elsewhere [1], they are reviewed here for convenience.

A *track* is a snapshot of a particle at a particular point along its path. Instances of the class G4Track contain the particle's current energy, momentum, position, time and so on, as well as its mass, charge, lifetime and other quantities.

A *trajectory* is a collection of track snapshots along the particle path.

A *step* consists of the two endpoints which bound the fundamental propagation unit in space or time. The length between the two points is chosen by a combination of transportation and physics processes, and may be limited to a fixed size by the user in cases where small step lengths are desired. An instance of G4Step stores the change in track properties between the two endpoints.

Process has two meanings in GEANT4. In the usual computer science sense, it refers to an instance of an application which is being executed. This is the meaning assumed in the discussion of multithreading. In all other discussions the narrow GEANT4 meaning is assumed: a class implementing a physical or navigational interaction. A GEANT4 process is categorized by when the interaction occurs, either at the end of the step (PostStep) or during the step (AlongStep).

An *event* consists of the decay or interaction of a primary particle and a target, and all subsequent interactions, produced particles and four-vectors. G4Event objects contain primary vertices and particles, and may contain hits, digitizations and trajectories.

A *run* consists of a series of events.

2. Multithreading

2.1. The transition to multithreading

The emergence of multi-core and many-core processors has been a well-established trend in the chip-making industry during the past decade. While this paradigm guarantees the continued increase of CPU performance, it requires some adaptation of existing code in order to better utilize these architectures. In typical GEANT4 simulations the most widespread approach for exploiting parallelism is job or process parallelism. This is the spawning of multiple copies of the same process, and is being used in large-scale production by HEP experiments to exploit today's hardware. However a key challenge for this approach is that the amount of random access memory (RAM) required scales linearly with the number of processes. As the number of cores increases beyond 8 or 16, the amount of RAM may become a limiting factor unless a robust solution for the sharing of memory between processes (or an alternative method) can be adopted in production systems.

This is especially true for co-processor technologies in which a high core count is associated with a relatively limited amount of RAM, as in the Intel Xeon Phi co-processor card model 7120P, which hosts 16 GB of RAM for 61 physical cores.

In GEANT4 an alternative solution was developed, in which multithreaded applications share a substantial part of their data between threads in order to significantly reduce the memory footprint. In this design the memory savings are obtained by sharing among all the threads the key data which are constant during simulation: the geometry description and the tables used by electromagnetic physics processes [2]. Threads are otherwise independent.

In this implementation each thread is responsible for simulating one or more full events, thus implementing event-level parallelism. Measurements demonstrate that this approach scales well with the number of threads. Almost linear scaling was obtained from two up to 60 physical cores, the maximum available on shared memory systems that were available for benchmarking. Additional throughput gains of about 20–30% were obtained by using hyperthreading.

2.2. General design

As a Monte Carlo simulation toolkit, GEANT4 profits from improved throughput via parallelism derived from the independence of modeled events and their computation. Until GEANT4 version 10.0, parallelization was obtained with a simple distribution of inputs: each computation unit (e.g. a core of a node in a cluster) ran a separate instance of GEANT4 that was given a separate set of input events and associated random number seeds.

Given a computer with k cores, the design goal of multi-threaded GEANT4 was to replace k independent instances of a GEANT4 process with a single, equivalent process with k threads using the many-core machine in a memory-efficient, scalable manner. The corresponding methodology involved transforming the code for thread safety and memory footprint reduction [3]. A simplified schema of the multithreading model used is shown in Fig. 1.

Before the parallel section of the simulation begins, the geometry and physics configurations are prepared and the random number engine is initialized in order to generate a random sequence of uniformly distributed numbers. This guarantees reproducibility (see below). Threads compete for the next group of events to be simulated, requesting one or more seeds from the
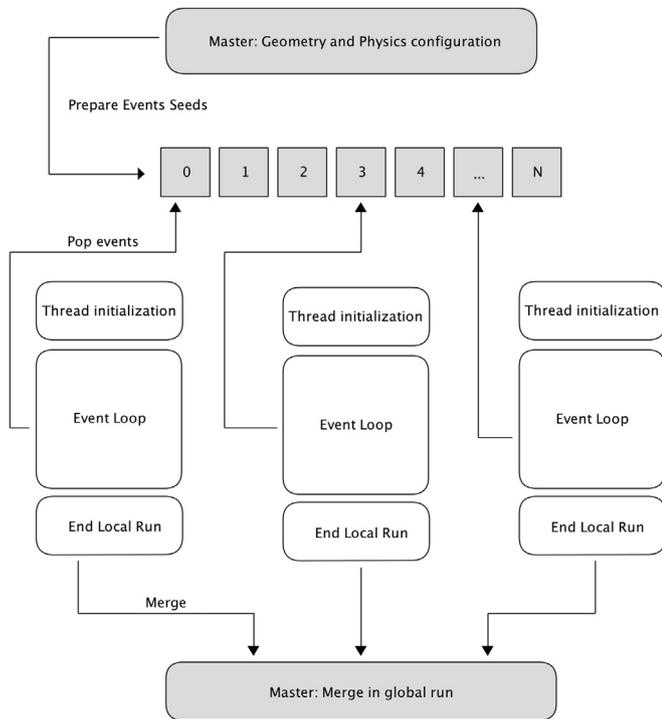
Fig. 1. Simplified description of a GEANT4 multithreaded application: the *master* thread prepares geometry and physics setups for the simulation, and the *worker* threads compete for the next (group of) events to be simulated; otherwise they are independent.

Fig. 2. Sequence diagram of a multithreaded GEANT4 application. The application instantiates one `G4MTRunManager`. When the first run is started one or more worker threads are spawned. The simulation in each worker thread is controlled by the local `G4WorkerRunManager`.

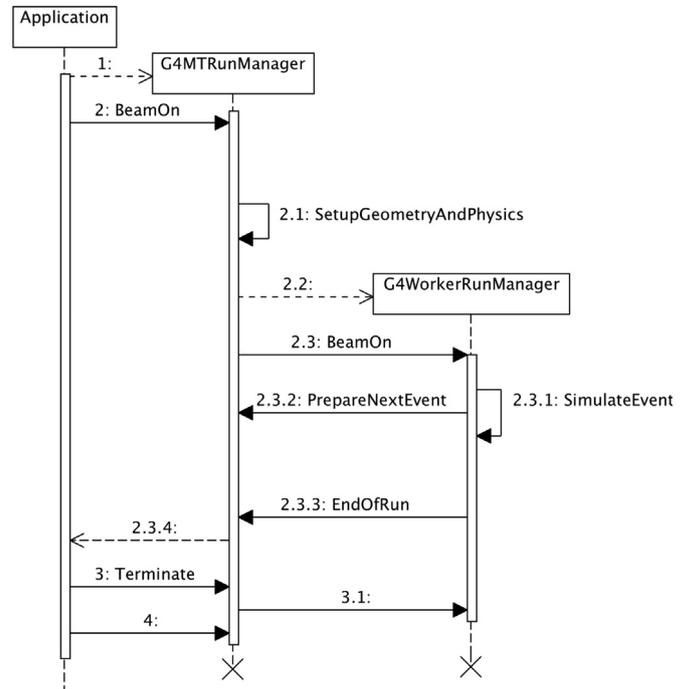shared seeds queue. Output produced by the threads can be reduced at the end of the run. If the application uses the command-line scoring mechanism or histograms from the GEANT4 analysis package, output from these is reduced automatically. User-defined `G4Run` instances can be merged if they implement a `Merge` method.

The multithreading option is based on a *master–worker* model in which one control sequence (the *master*) is responsible for initializing the geometry and physics, and for spawning and controlling *worker* threads. Workers are responsible for the simulation of one or more events. The sequence diagram of a GEANT4 application is shown in Fig. 2 where the main interfaces (`G4MTRunManager` and `G4WorkerRunManager`) and their interactions are shown.

A GEANT4 application is defined by the use of an instance of the `G4RunManager` class or of a user-defined class derived from it. This class defines the main interaction with the user: it provides interfaces to define the *user initializations* (e.g. geometry and physics definitions) and the *user actions* that permit interaction with the simulation kernel and retrieve output information. In particular, `G4RunManager` provides the interface to start the simulation of a run, which is a collection of events. For multithreaded applications a derived class `G4MTRunManager` is used that allows the number of worker threads to be specified. Shared objects, such as the geometry and physics list, are registered by the user to the instance of this class, while the creation of user actions is the responsibility of a new class `G4VUserActionInitialization`. When a new run is requested it is the responsibility of `G4MTRunManager` to start and configure worker threads. Each thread owns an instance of `G4WorkerRunManager` and it shares only the user initialization classes, while it owns a private copy of the user action classes. Workers continue to request events from the master until there are no more events left in the current run. At the end of the run the results collected by threads can be merged in the global run.

The communication between master and workers was implemented with a simple barrier mechanism to synchronize threads, and with the exchange of simple threading messages which currently may be one of:

- *workers start new run* (instruct worker threads to begin the event loop),
- *workers terminate* (instruct workers that the job has concluded, workers should terminate and exit), or
- *wait for workers ready* (master thread is waiting for one or more workers to terminate current event loop, idle workers wait for further messages).

User-defined code can be implemented by specializing key interfaces of certain classes. In particular, the `G4UserWorkerInitialization` class defines the threading model and determines how threads are configured. The main GEANT4 classes relevant to a multithreaded application are depicted in Fig. 3. All interfaces are also available in sequential mode so that user code for a simulation application can be run both in multithreaded or sequential GEANT4 with minimal changes.

2.3. Results

The physics and CPU performance of GEANT4 were measured by comparing the sequential version of the code to a multithreaded equivalent. The results which follow were obtained with a patched version of GEANT4 Version 10.0, and focus on high energy physics applications. This domain was chosen because its complex physics requirements cover the most important use-cases for GEANT4 users:

- high and low energy electromagnetic physics,
- high and low energy hadronic physics,
- tracking in many-volume geometries and
- tracking in electromagnetic fields.

In the near future regular testing will be extended to other user domains such as medicine and space science.
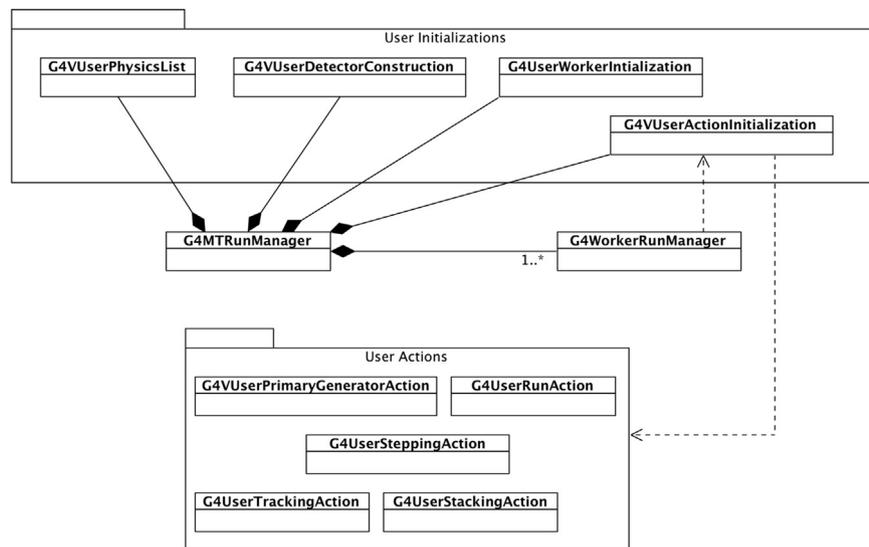
Fig. 3. Class diagram of the main user interfaces [4]. User initializations (e.g. geometry and physics list) are shared among threads and are assigned to the single instance of `G4MTRunManager`, while user actions are created for each thread (via `G4VUserActionInitialization`) and assigned to the thread-private `G4WorkerRunManager`.

So far, two applications have been adapted to multithreading. The first is a test suite (simplified calorimeter) based on a simple geometry setup which allows the study of all types of primary particle species over a very wide energy range [5]. The most interesting particles are electrons and hadrons at high energy because they exercise the majority of physics models used in HEP simulations of full showers. Optional analyses can be performed on the predictions of the simulation in order to study typical HEP quantities. These include both integral quantities like the total energy deposit, and detailed aspects of showers such as the number and type of interactions, and the energy spectra of produced secondaries.

The second application uses the GEANT4 GDML interface [6] to read a realistic geometry description of the CMS experiment at the LHC [7]. No analysis of simulated data is performed, but a large set of materials and geometrical shapes is tested. This application has been used to test physics performance on different hardware setups, including Intel Xeon, ARM, PowerPC and Intel Atom processors, and Intel Xeon Phi co-processors.

2.3.1. Physics equivalence to sequential

It is of course required that the physics calculations are the same in both the multithreaded and sequential versions. Two tests were developed to verify this. The first performs a statistical comparison of physics quantities simulated with the simplified calorimeter testing suite. Typical experimental observables (response, resolution and shower shapes) are compared between multithreaded and sequential versions of the same application. The resulting distributions were confirmed to be statistically equivalent. In this test the RNG engine seeds used in the sequential and multithreading applications were not the same.

A more stringent test compares, event by event, the history of the random number generator (RNG) engine. To guarantee that reproducibility is independent of the number of threads and of the order in which events are simulated, each thread owns a separate instance of the RNG engine, and its status is re-initialized before the simulation of each event with a separate pre-defined seed. The test consists of two steps: a multithreaded application is run and the RNG engine seed recorded at the beginning of each event, together with the status of the engine at the end of the event simulation. The same application is then run in sequential mode, with the RNG engine re-initialized for each event with the seed

from the first run. The engine status at the end of each event should be the same in the sequential and multithreaded versions. It was verified that this occurs for 100% of the cases, except for the test using the radioactive decay module. This remaining discrepancy is being investigated, but it is thought to be due to violations of strict reproducibility – the independence of the results for a particular GEANT4 event from the history of previous events. Extensive checking of the strong reproducibility of GEANT4 physics models and physics lists has significantly reduced the incidence of such issues. Strong reproducibility of all classes used in a GEANT4 application is an important requirement for providing consistent results in multithreaded runs, as results must not depend on the number of workers, or on the assignment of events to workers.

2.3.2. CPU and memory performance

The goal of event-level parallelism with threads is to reduce the memory footprint of parallel applications, while preserving the linear speedup of throughput as a function of the number of physical cores. Additional throughput is also expected in CPU architectures that support hyperthreading adding more workers [8].

Using the GDML geometry of the CMS application, three metrics were studied: the multithreading overhead with the number of threads $k=1$ with respect to a pure sequential application, the linearity of speedup as a function of the number of threads, and the memory reduction with respect to a multi-process approach.

In general, a performance penalty can be expected when comparing a sequential version of an application with the same version running with one worker thread. In GEANT4 this is due to the use of the `thread` keyword that adds an additional indirection when accessing thread-private data. To remove this penalty a careful study of the use of `thread` was carried out: compilation flags were chosen to minimize the impact of thread local storage (TLS) selecting the best model for GEANT4 applications (`initial-exec`). An overhead of (~1%) was measured as shown by the $k=1$ point of Fig. 4.

Fig. 4 shows the speedup linearity obtained with an AMD server (Opteron Processor 6128 running at 2.0 GHz, 4 CPU sockets × 8 cores) as a function of the number of threads, compared to the sequential case. Good linearity was obtained with the CMS geometry simulation. Speedup was linear with efficiencies of more than 95%. This result was confirmed on a number of different architectures: Intel Xeon, Intel Atom and PowerPC processors, ARM technology, and Intel Xeon Phi co-processors.

Reported in Table 1 is a comparison of the performance of different hardware executing the same CMS geometry application with the number of threads equal to the number of logical cores. Differences in performance were due mainly to the relative power of each core and the core multiplicity. The latest generation of Intel Xeon processor showed the best performance, expressed as absolute throughput (events/minute).

Fig. 5 shows relative memory savings as a function of the number of threads for the CMS geometry application. GEANT4 efficiently reduces the memory used by the application when running with k threads $(k > 1)$ compared to k copies of the same application. For example, an application with eight threads requires about half the memory needed for eight clones of the sequential version of the same application. The overhead with one worker thread is expected, since thread-private memory objects are duplicated between worker and master threads. The per-thread memory overhead is at the level of 40–80 MB depending on the application (for the same application described in Table 1 the sequential memory consumption is about 200 MB).

2.4. Further developments

Several improvements and extensions to the multithreading capabilities of GEANT4 are planned for upcoming versions of the code.

With release 10.1 further reductions in the per-thread memory footprint of GEANT4 applications are planned. The most memory-consuming objects in typical simulations have been identified as hadronic cross sections, high precision neutron models, reaction tables of the binary cascade models and the general particle source primary generator; strategies will be implemented to share these objects among threads. In some cases refinement of the design of the modules is being carried out to better match the general multithreading strategy. One goal is to reduce the per-thread memory footprint by a factor of two. This will allow applications to run on co-processors, where memory is of the order of GBs and there are of order 100 threads.

Currently the number of threads is fixed and cannot be modified dynamically. The planned removal of this restriction will achieve better integration with external parallelization frameworks such as Intel Threading Building Blocks (TBB) [9]. A prototype example, `examples/extended/parallel/TBB`, that replaces GEANT4's POSIX threading model with the TBB task-parallelism model, has already been released with GEANT4, but improved and streamlined integration with this library is planned.

For several years now, GEANT4 has provided an example, `examples/extended/parallel/MPI`, that demonstrates integration with Message Passing Interface (MPI) [10]. In version 10.0 this example was migrated to a hybrid approach in which
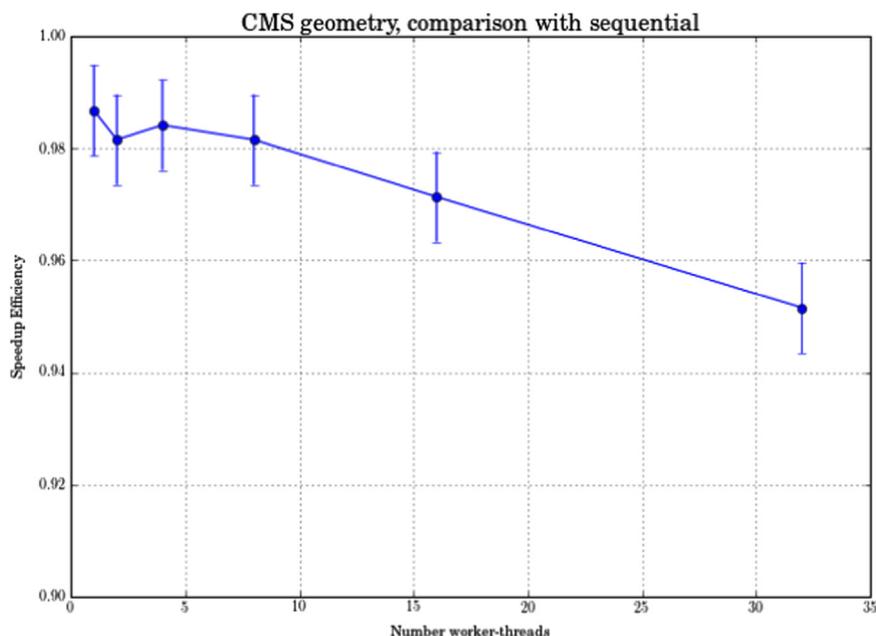
Fig. 4. Speedup efficiency (ratio of throughput of a run with k threads to the throughput of the sequential version), as a function of the number of threads, for the CMS simulation on an AMD-equipped server (Opteron Processor 6128 running at 2.40 Hz, 4 CPU sockets × 8 cores). The multithreading overhead for one thread is only 1%, while the efficiency is greater than 95% for up to the maximum number of threads.

Table 1
Comparison of different hardware when running CMS experiment geometry. Results show throughput (events/minute) per full processor.

Processor type	Throughput (events/min)
Intel Xeon X5650 – 2.67 GHz 6 cores, x2 hyper-threaded (with 12 sequential instances)	320 (324)
Intel Xeon E5-2695 v2 – 2.40 GHz 12 cores, x2 hyper-threaded	535
Intel Atom C2730 – 1.7 GHz 8 cores	74
Exynos 5410 Octa Cortex-A15 1.6 GHz – 4 cores	47
PowerPC A2 – 1.6 GHz 16 cores, x4 hardware threads	119
Intel Xeon Phi 7120P – 1.238 GHz 61 cores, x4 hyper-threaded	334
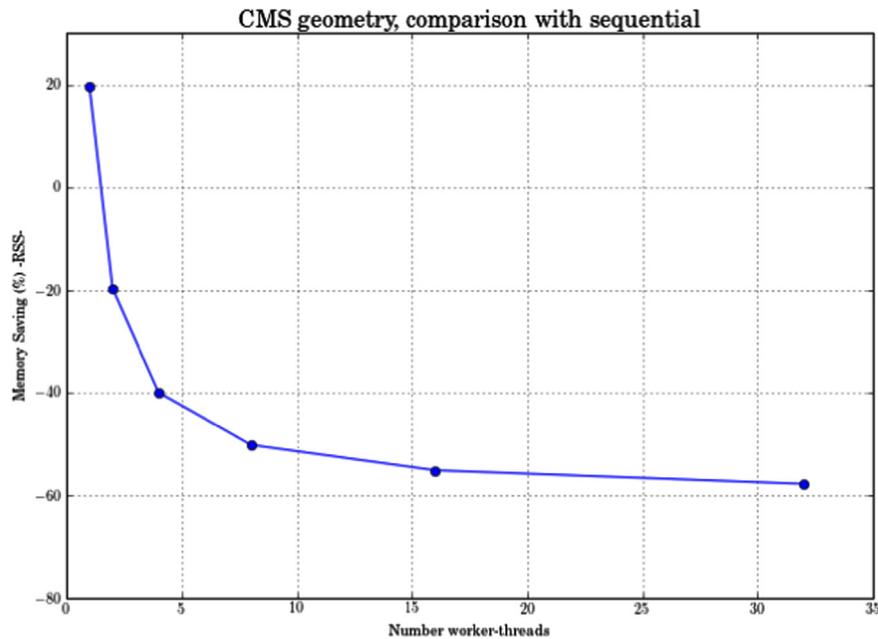
Fig. 5. Relative memory reduction (memory used by a run with k threads with respect to k instances of the sequential version), as a function of the number of threads, for the CMS simulation on an AMD equipped server (Opteron Processor 6128 running at 2.0 GHz, 4 CPU sockets × 8 cores). The memory overhead with one worker thread is due to the duplication of thread-private objects. Already with two worker threads, significant reduction of memory footprint is achieved.

MPI ranks can exploit multithreading to efficiently use memory-distributed systems. Further refinement of the example is planned, with particular attention paid to providing merging of physics results.

3. Kernel functionalities

3.1. Tracking and scoring

3.1.1. Design changes in tracking
The main changes in tracking include easier physics process implementation, new infrastructure for the re-acceleration of charged particles in electric fields, and "reverse Monte Carlo". The problem of re-acceleration is not yet solved and requires further development.

Adjoint or "reverse" Monte Carlo has been available in Geant4 since release 9.3 and modifications to tracking were required to make this possible. The enabling classes have names beginning with `G4Adjoint`. Details of these classes and adjoint Monte Carlo can be found in Section 5.1.

3.1.2. Concrete scorers
In Geant4, a hit is a snapshot of a physical interaction or accumulation of interactions of tracks in a sensitive detector component. "Sensitive" here refers to the ability of the component to collect and record some aspects of the interactions, and to the Geant4 classes which enable this collection. Because of the wide variety of Geant4 applications, only the abstract classes for both detector sensitivity and hit had thus far been provided in the toolkit. A user was therefore required to have the expertise necessary to implement the details of how hits were defined, collected and stored.

To relieve many users of this burden, concrete primitive scorers of physics quantities such as dose and flux have been provided which cover general-use simulations. Flexibly designed base classes allow users to implement their own primitive scorers for use anywhere a sensitive detector needs to be simulated.

Primitive scorers were built upon three classes, `G4Multi-FunctionalDetector`, `G4VPrimitiveScorer` and `G4VSDFilter`. `G4MultiFunctionalDetector` is a concrete class derived from `G4VSensitiveDetector` and attached to the detector component. Primitive scorers were developed on top of the base class `G4VPrimitiveScorer`, and as such represent classes to be registered to the `G4MultiFunctionalDetector`. `G4VSDFilter` is an abstract class for a track filter to be associated with a `G4MultiFunctionalDetector` or a primitive scorer. Concrete track filter classes are also provided. One example is a charged track filter and a particle filter that accept for scoring only charged tracks and a given particle species, respectively.

A primitive scorer creates a `G4THitsMap` object for storing one physics quantity for an event. `G4THitsMap` is a template class for mapping an integer key to a pointer value. Since a physics quantity such as dose is generally accumulated in each cell of a detector component during an event or run, a primitive scorer generates a G4THitsMap<G4double> object that maps a pointer to a `G4double` for a physics quantity, and uses the cell number as the integer key. If a cell has no value, the `G4THitsMap` object has no corresponding entry and the pointer to the physics quantity returns a null. This was done to reduce memory consumption, and to distinguish an unfilled cell from one that has a value of zero. The integer key of the cell is taken from the copy number of the `G4LogicalVolume` of the detector component by default. Geant4 also provides primitive scorers for three-dimensional structured geometry, in which copy numbers are taken at each of the depth levels at which of the logical volumes are nested in the geometric structure. These copy numbers are then serialized into integer keys.

3.1.3. Command-based scoring
Command-based scoring is the easiest way to score primitive physics quantities. It is based on the primitive scorers and navigation in an auxilliary geometry hierarchy ("parallel world", Section 3.2.2). Users are not required to develop code, as interactive commands are provided to set up the scoring mesh.

The scoring mesh consists of a scoring volume in a three-dimensional mesh with a multifunctional detector, primitive scorers

and track filters which may be attached to a primitive scorer. An arbitrary number of primitive scorers can be registered to the scoring mesh.

A given physics quantity, or score, is accumulated in each cell during a run. Interactive commands allow scores to be dumped into a file and written out in CSV format. Other commands allow scores to be visualized as color histograms drawn on the visualized geometry in either a projection or a selected profile.

Because scoring volumes are placed in a parallel world, the scoring volume and the mass volume are allowed to overlap. Command-based scoring can therefore obtain the physics quantity in an arbitrary volume or surface independent of the mass volume. One exception to this is the dose primitive scorer, in which the scoring volumes and their cells must coincide exactly with the mass geometry structure. This is because the mass density of the cell is taken from the mass geometry while the volume of the cell is taken from the scoring geometry.

Most of the command-based scoring is handled in two classes, `G4ScoringManager` and `G4VScoringMesh`. `G4ScoringManger` is the administrative class of command-based scoring. It creates a singleton object which keeps a list of registered scoring meshes, and operates the scoring according to interactive commands and commands from the Geant4 kernel. `G4VScoringMesh` is the base class of scoring meshes which represent scoring geometries. It keeps a list of associated primitive scorers. The `G4VScoringMesh` object creates a series of `G4THitsMap` objects in which each primitive scorer can accumulate physics quantities in a run.

Command-based scoring works in both sequential and multi-threaded modes. In sequential mode, `G4RunManager` creates scoring meshes at the beginning of a run. After each event, the `G4THitsMap` object in `G4VScoringMesh` is updated by adding values in that event to the primitive scorer. In multithreaded mode `G4MTRunManager` creates a master scoring manager. Each worker thread of `G4WorkerRunManager` creates its own local scoring manager with scoring meshes. However, the logical volume of the scoring mesh is shared between master and local scoring managers. The local scoring manager then accumulates physics quantities in the same manner as sequential mode. At the end of a thread, the worker run manager requests the master run manager to merge the scores of the local scoring manager with those of the master scoring manager.

3.2. Detector modeling

3.2.1. Introduction

A key component of Geant4 is the geometry modeler [11], which provides a wide variety of tools and solutions for describing geometry setups from simple to highly complex. Geometrical models of the LHC detectors, for instance, easily reach millions of geometrical elements of different kinds combined together in hierarchical structures. The geometry modeler provides techniques by which memory consumption can be greatly reduced, allowing regular or irregular patterns to be easily replicated, assembled or reflected. This, combined with navigation and optimization algorithms, allows the efficient computation of intersections of simulated tracks with the elements composing any geometry setup.

Recent extensions of the geometry modeler include specialized navigation techniques and optimization algorithms to aid medical simulation studies. This has allowed complex geometrical models of the human body to be developed. Extensions also include parallel navigation and tracking in layered geometries which allow geometry setups to be superimposed on one another with minimal impact on CPU time.

3.2.2. Navigation in geometries

The recent addition featuring "parallel geometries" allows the definition and treatment of more than one independent geometry in parallel for different potential uses, and exploits the existing enhanced interface provided by the navigation system in the Geant4 toolkit [12]. In Geant4 a geometry setup is in general associated with a navigator which is a concrete instance of the `G4Navigator` class. `G4Navigator` was designed such that several instances of it can be created and coexist; each instance can be assigned to a different geometry hierarchy. The primary navigation instance is attached to the "mass world" which is the main geometry hierarchy in which the material of the setup is described; this setup is unique and is used for all physical interactions.

"Parallel world" geometries may be assigned to the additional navigator objects and may be used for example as simple "locators", independent of the mass world, to identify exact positioning in the other geometries of a particular point in the global coordinate system. Each geometry must have an independent root volume (the world volume), which contains a hierarchy of physical volumes. Volumes in one world may overlap volumes in a different world.

Volumes in a parallel world geometry can be associated with the read-out structure of a detector. In shower parameterization studies, for example, the simplified read-out geometry of a calorimeter could overlay its more complex mass geometry. Parallel worlds are also useful in importance biasing and scoring of doses and other radiation measures.

Volumes in a parallel world may have material; these are referred to as the "layered mass geometry". In this case, the material defined in a volume in the parallel world overrides the material defined in the mass world and is used for the calculation of physical interactions. If more than one parallel world is overlaid on the mass world, the parallel worlds are examined, in reverse order of their creation, to see if any volumes with materials are defined in them. Any such volumes found will override the materials in all previously created worlds. Because volumes in the mass geometry always have materials, the material to be used for physical interactions is always uniquely defined.

Layered mass geometry offers an alternative way of describing complicated shapes that would otherwise require massive Boolean operations to combine primitive volumes. Examples include a photo-multiplier system partially dipped in a noble liquid and brachytherapy seeds implanted in the CT-scanned voxels of a patient. A voxel refers to a volume element which represents a value on a three-dimensional grid.

In addition, different parallel worlds may be assigned to different particle types. Thus, in the case of a sampling calorimeter, the mass world could be defined as having only a crude geometry with averaged material, while a parallel world would contain the detailed geometry. The real materials in the detailed parallel world geometry would be associated only with particle types that require accurate tracking, such as muons, while other particle types such as electrons, positrons and gammas would see crude, less complicated geometry for faster simulation.

3.2.3. Navigation in regular geometries

When the voxels in a geometry are numerous and of the same size and shape, a specialized navigator can take advantage of the regularity to deliver faster CPU performance. This is useful in medical physics applications in which there could be millions of identical volumes comprising a 3-D image of a patient.

In this case the navigator can use a regular grid to easily locate the incident particle in the geometry.

In the Geant4 implementation `G4PhantomParameterisation` defines the regular structure of the geometry, using the parameters of dimension, offset and number of voxels in each of three dimensions. `G4RegularNavigation` uses this parameterization to directly locate the voxel visited by tracks in the simulation. An option is provided which allows boundaries between contiguous

voxels to be skipped if they are of the same material, thus significantly reducing the number of tracking steps.

Using this method of navigation, CPU speed improvement factors of three to six have been observed. This factor holds for pure navigation examples (no physics interactions) and for beams of gammas. For the case of electrons or protons most of the time is spent on physics instead of navigation and the speed improvement is substantially reduced. Significant savings in memory consumption (factor of seven) and initialization time (factor of two) were also seen [13].

3.2.4. Exact safety

The "isotropic safety" is the distance to the next volume boundary in any direction. It is calculated by the navigator and is used by the multiple scattering process in two distinct ways. The primary use of the safety is to limit the lateral displacement in order to avoid crossing a boundary within a step. Improved safety values reduce the need for artificial restrictions in electron displacement, and enable it to be better simulated at particular values of the model parameters and production thresholds. The isotropic safety also sometimes influences the step size restriction from multiple scattering. In the default configuration of this process it has an effect only if the safety is larger than the primary restriction (a fraction of the range), in which case the safety is used as the step limit.

The estimation of the isotropic safety was improved for volumes which have several child volumes. Previously only the contents of the current voxel in the optimization voxels structure automatically generated at initialization, and the boundaries of that voxel, were considered. This resulted in a distance which could be significantly underestimated. The new method considers enough voxels to obtain a precise value of the safety, while ensuring that the number of voxels is limited by the running estimate of the safety value.

As a result, an improved estimate of the safety is available. This ensures that the value of the isotropic safety does not depend strongly on the details of the voxelization, on the number of child volumes, or on the presence of volumes in a remote region of the current volume. The extra computation cost was found to be negligible in a complex simulation.

3.2.5. Improved verbosity

The geometry modeler with its current version of the navigator offers an enhanced verbosity system which can be used to help developers and users in debugging problems or to provide closer monitoring of the execution flow and behavior of the navigation system during tracking. A set of UI commands was defined and made available by default in the user application which allows the execution flow to be controlled with five different levels of detail:

```
/geometry/navigator/verbose [level-number]
 – Setting run-time verbosity for the geometry
   navigation
  Command having effect -only- if Geant4 has been
   installed with verbose mode (G4VERBOSE flag) set!
  Level 0: Silent (default)
  Level 1: Display volume positioning and step
   lengths
  Level 2: Display step/safety info on point
   locations
  Level 3: Display minimal state at -every- step
  Level 4: Maximum verbosity (very detailed!)
```

A special UI command

```
/geometry/navigator/check_mode [true/false]
```

was also defined to modify the execution mode of the navigator and perform extra checks for correctness, or to apply stricter and less tolerant conditions to stress precision. This can help in studying and debugging difficult cases by eventually highlighting potential defects in the geometry under consideration.

An additional UI command

```
/geometry/navigator/push_notify [true/false]
```

allows the enabling or disabling of notifications from the navigator for artificial pushes applied by the navigation system along the track direction in case tracks get stuck in particular geometries.

These new tools, in addition to a more rationalized layout of output messages for the information provided by the system, contribute to make the navigation system of GEANT4 more user-friendly and provide powerful means to users and developers to improve the quality of their simulation programs.

3.2.6. Extensions to geometrical primitives

Since GEANT4 release series 8, new geometrical primitives, G4GenericTrap, G4ExtrudedSolid, G4Paraboloid and G4CutTubs, have been part of the toolkit and are shown in Fig. 6.

G4GenericTrap is an arbitrary trapezoid with up to four vertices lying in each of two parallel planes at $-hz$ and $+hz$ perpendicular to the z-axis. Vertices are specified by their x, y coordinates. Points may be identical in order to create shapes with fewer than eight vertices; the only limitation is to have at least one triangle at $+hz$ or $-hz$. The lateral surfaces are not necessarily planar and in that case they are represented by a surface that linearly changes from the edge at $-hz$ to the corresponding edge at $+hz$. This represents a sweeping surface with a twist angle linearly dependent on z, which is different from the twisted solids which have surfaces described by an equation depending on a constant twist angle. In Fig. 6A a G4GenericTrap with eight vertices and a twist is shown; Fig. 6B shows a G4GenericTrap with collapsed vertices and a twist.

G4ExtrudedSolid (Fig. 6C) is a solid obtained by the extrusion of an arbitrary polygon in the defined z sections. Each z section is defined by a z-coordinate, an offset in the x, y-plane and a factor by which to scale the polygon at the given z coordinate. Each section in z of the G4ExtrudedSolid is a scaled version of the same polygon. A second, simplified constructor for the special case of a solid with only two z-sections is also provided.

G4Paraboloid (Fig. 6D) is a solid with a parabolic profile and possible cuts along the z-axis at $+hz$ and $-hz$, with the cut planes perpendicular to the z-axis. To construct the parabolic profile, the following equation is used:

```
Z=a* (x*x+y*y)^2+b
```

with real coefficients a and b; the coefficients are calculated from given radii at $+hz$ and $-hz$.

G4CutTubs (Fig. 6E) is a tube or cylindrical section with cuts applied in z. These cuts are planes defined by a normal vector pointing outside the tube (cylindrical section) and intersect the z-axis at a given point $+hz$ or (and) $-hz$.

An important and rather useful construct for shapes delimited by any kind of complex surface is offered by the G4TessellateSolid class, which allows complex geometrical shapes to be described by approximating their surfaces as a set of planar facets (triangles), with tunable resolution. This technique can be used for importing geometries from CAD systems to generate surface-bounded solids. Recent developments during the implementation of the Unified Solids library [14] provide considerably improved CPU performance, making it possible to use such constructs for very detailed and realistic descriptions of surfaces, while optimally scaling with the number of facets in use. A sample G4TessellateSolid is shown in Fig. 7.

Unified Solids have been available since GEANT4 10.0 as experimental alternatives to the traditional geometrical primitives. The aim is to offer an independent library of such solids to replace the traditional primitives.
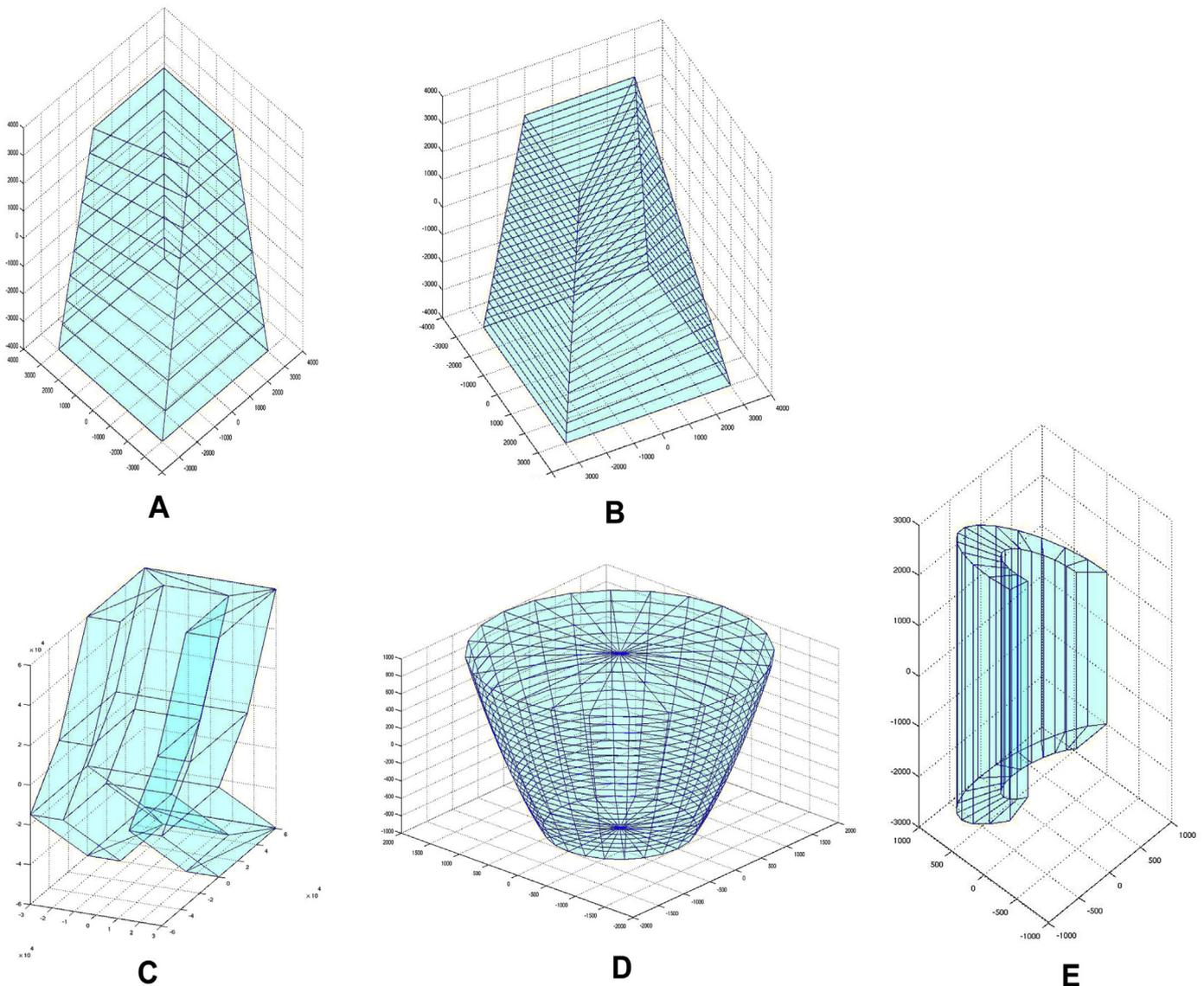
Fig. 6. Recent geometrical primitives added in Geant4: generic trapezoid (A,B), extruded solid (C), parabolic solid (D), and cut tube (E).

Cloning of all geometrical primitives has been possible since release 9.4, with the definition of appropriate copy constructors and assignment operators. This functionality is required when running in multithreaded mode when parameterized geometries are used. All solids also have the ability to compute their own surface area and geometrical volume:

```
G4double G4VSolid::GetSurfaceArea()
G4double G4VSolid::GetCubicVolume().
```

A solid's surface area and geometrical volume are estimated using Monte Carlo sampling methods when it is not possible to compute them with mathematical formulae; in such cases the accuracy can be tuned in case the default does not provide sufficient precision. Computed values are expressed in internal units and are cached for reuse. In a detector setup, these utilities allow for the calculation of the overall mass of the setup for a specific logical volume:

```
G4double
G4LogicalVolume::GetMass(G4bool forced=false,
                 G4bool propagate=true,
                 G4Material* parMaterial=0).
```

The mass of the logical volume tree expressed in internal units is computed from the estimated geometrical volume of each solid and material associated with the logical volume and, by default, its daughters. The returned value is also cached in this case and can be used for successive calls, unless recomputation is forced by providing "true" for the Boolean argument (i.e. in case the geometry setup has changed after the previous call). By setting the "propagate" Boolean flag to "false" only the mass of the current logical volume is returned (with the volume occupied by the daughter volumes subtracted). An optional argument "parMaterial" can be used to specify a custom material for a specific logical volume; the argument is also used internally to consider cases of geometrical parameterization by material.

Since release 8.2 the geometry modeler has provided a tool to precisely identify and flag defects in the geometry setup due to overlapping surfaces of physical volumes. The technique makes use of the ability of each geometrical primitive to randomly generate points lying on its surface and verifying that none of these points is contained in other volumes of the same setup at the same level in the geometry tree. With the release 10 series, this technique is also used when overlap checks are issued as UI
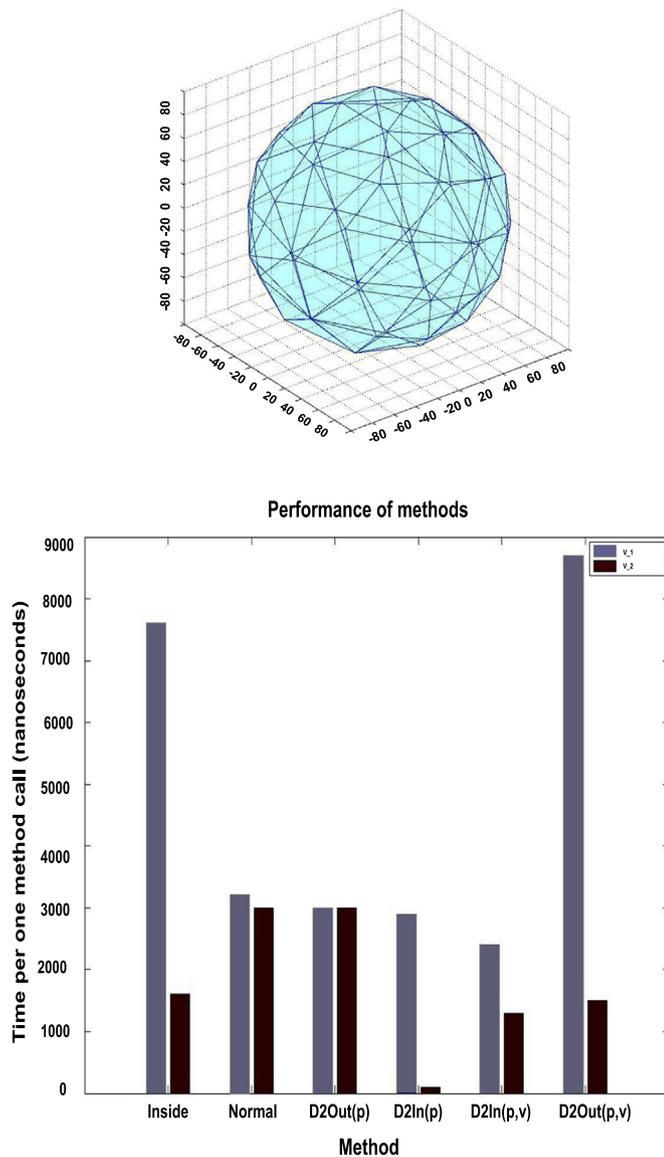
Fig. 7. Relative performance for the tessellated sphere (top), illustrated for each individual method. Method name abbreviations are D2In(p): DistanceToIn(p), D2In(p,v): DistanceToIn(p,v), D2Out(p): DistanceToOut(p), and D2Out(p,v): DistanceToOut(p,v). Light colored bars correspond to GEANT4 9.5.p02 and dark colored bars correspond to GEANT4 9.6.p02. Already for solids composed of a relatively small number of facets (100 as in the case for the sphere), a clear improvement is measured, especially for key methods.

commands, replacing the old method based on sampling over an overlapping grid of lines or cylinders. These UI commands are listed and explained in Section 4.1.11 (Detecting Overlapping Volumes) of the Application Developer Guide [15].

3.2.7. Extensions to propagation in a field

A gravitational field and the ability to create a force for it have been available in the toolkit since release 9.5. Also, the force exerted on the magnetic moment in a gradient B-field is now taken into account for any particle, including neutrals. An equation of motion was added that accounts for the total combined force from magnetic, electric, gravitational and gradient B-fields as well as spin tracking. With this it is possible to simulate the trajectory and spin of ultra-cold neutrons (UCN) and the trapping of neutral hydrogen atoms in a magnetic bottle.

A field can now be registered to a geometrical region, in addition to the global reference frame or to a logical volume, as before.

The mechanism to refine an intersection between a curved trajectory and volume boundaries was revised, making it possible to choose one of three methods or define a user-created method to do this. A new multi-scale "locator" method (the new default), and a locator similar to Brent's method [16] for root-finding, were added as alternatives to the original linear locator. These allow the propagation in fields to cope with difficult trajectories which remain near to but just outside a curved surface. This occurs in typical high energy physics applications which have nearly constant fields along the axis of a tube. The new methods also provide better overall CPU performance than the old default, at the cost of more complex code.

3.2.8. Geometry persistency

Detector geometrical descriptions can be imported and exported from text files according to two different formats: the Geometry Description Markup Language (GDML) [6] based on XML, or in plain ASCII text. GEANT4 provides internal modules which allow the interpretation and conversion of the above formats to and from the internal geometry representation, without the need for C++ programming for the implementation of the various detector description setups.

3.2.8.1. GDML geometry. In version 3 of GDML, the part of GDML I/O which provides the ability to export and import detector geometry descriptions to and from GDML files, was integrated into GEANT4 by means of the GDML module making use of the DOM XML parser provided with the Xerces-C++ [17] software package.

The GEANT4 binding for GDML implements all features supported by the GEANT4 geometry modeler and most of the geometrical entities defined as part of the latest version of the GDML schema. These include all shapes, either CSG or specific solids, and their Boolean combinations. Also included are any combinations of materials, from isotopes to mixtures, and the ability to import definitions compliant with the GEANT4 NIST database.

All types of physical volumes are supported, from placed volumes to replicas and parameterized volumes, including assemblies, divisions and reflections.

GDML supports the modularization of geometry descriptions to multiple GDML files, allowing for rational organization of the modules for complex setups. Verification of the GDML file against the latest version of the schema comes for free thanks to Xerces-C++, with the possibility to turn it on or off in the GEANT4 GDML parser.

Recent additions to the GDML parser enable efficient import/export of tessellated solids, and the treatment of parameterizations for polycones, polyhedra and ellipsoids. Release 10.1 of GEANT4 provides support for the definition, import and export of multi-union structures when making use of the Unified Solids library.

Several examples are provided to illustrate most of the features of the GEANT4 GDML parser:

```
examples/extended/persistency/gdml/G01
examples/extended/persistency/gdml/G02
examples/extended/persistency/gdml/G03
examples/extended/persistency/gdml/G04.
```

Example G01 shows how to write a simple application for importing and exporting GDML files, providing a variety of samples for different kinds of solids, volumes, material descriptions, integration of optical surface parameters, and so on. Example G02 demonstrates how to import/export different geometry setups, including STEP Tools [18] files and structures, integrating them

into a real simulation application. In example G03 it is shown how to define and import extensions to the GDML schema for attributes associated with a logical volume. G04 is a simple example showing how to associate detector sensitivity with a logical volume, making use of the GDML feature for defining auxiliary information.

3.2.8.2. ASCII geometry. The format of the ASCII text file is based on the use of tags: special words at the beginning of each line setting what the line is describing.

With this format the user may describe any of the geometrical objects of Geant4. It is possible to create materials combining elements, materials and detailed isotopic composition. Mixtures of materials can be built by providing the percentage of each material by weight, by volume or by giving the number of atoms. The user may change the default pressure, temperature and state, or set the minimum ionizing energy instead of letting Geant4 calculate it automatically. Instead of explicitly building each material, pre-defined elements or materials from the Geant4 NIST database may be specified.

Most types of Geant4 solids can be described, whether CSG or specific, by including a combination of solids through Boolean operations. Logical volumes can be defined by attaching solids to materials, and color and visualization attributes can be assigned to each one. After building the logical volumes, they can be placed individually or by using replicas, divisions, assemblies or para-meterizations. As it is almost impossible with a scripting language to cover all the possible parameterizations a user may need, only the most common ones are available: linear, circular, square or cubic. If others are needed, it is possible to define them through C++ code and mix them with the rest of the geometry in ASCII format. To place the volumes, rotation matrices can be defined with three different formats providing: values of the three rotation angles about the three axis, the theta and phi angles that define the orientation of the three axes, or the elements of the 3×3 rotation matrix.

To facilitate the definition of a complex geometry, it is possible to use parameters: values that can be assigned to keywords, so that they can be reused later in any part of the geometry. It is also possible to define numerical values through arithmetic expressions. The code automatically assigns a default unit depending on the dimension: mm, degrees, MeV, nanoseconds, g/cm^3, but the user may change it at any place. Comments may be used at any point in the file, using the C++ style of placing two forward slashes before the comment.
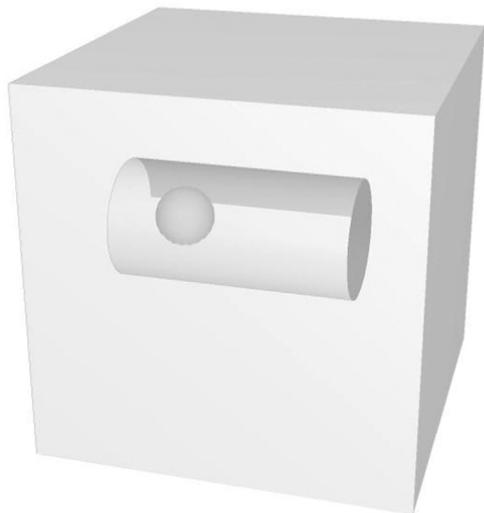

Fig. 8. Geometry setup corresponding to the ASCII specification given in the text.

If the geometry description is long, it may be split into several files, which may be combined by setting a

```
#include
```

tag. It is also possible to combine part of the geometry with C++ code and another part with ASCII format. If the user has a geo-metry already defined in C++, it may be transformed into ASCII format by adding a user action in the code.

The text format is thoroughly checked and clear error messages are provided when necessary. Arithmetic expressions are checked for correctness and the parameters in each tag are compared against expected number and type. An error message results if a line refers to a non-existent element.

An example of the geometry ASCII text format is given here and the resulting picture is shown in Fig. 8:

```
// Define a parameter for later use
:P DIMZ 5.

// Define materials
:ELEM Hydrogen H 1. 1.
:ELEM Oxygen O 8 16.
:ELEM Nitrogen N 7 14.
:MIXT Air 1.214E-03 2
    Nitrogen 0.75
    Oxygen 0.25

// Define rotation matrix
:ROTM R00 0. 0. 0. // unit matrix

// Define volumes and place them
:VOLU world BOX 30. 30. 30. Air

:VOLU "my tube" TUBE 0. 10. $DIMZ*4 G4_WATER
:PLACE "my tube" 1 world R00 0. 0. $DIMZ

:VOLU sphere ORB 5. G4_AIR
:PLACE sphere 1 "my tube" R00 0. 1. 10.
```

An example, `examples/extended/persistency/P03`, is included with the Geant4 distribution to illustrate the use of the ASCII text geometry. Several text geometry files are provided to illustrate the many different geometry construction options. This example also shows how to define a sensitive detector, mix C++ code with ASCII files, extend the format to create a new tag and dump the in-memory geometry to an ASCII file.

3.3. Visualization

Geant4 visualization capabilities [19] have been extended to leverage new user interface technologies, such as Qt [20], and to extend many features in response to user needs. In many cases, visualization solutions that previously required extensive user coding are now provided through rich built-in functionality. Geant4 offers the user many options for visualization drivers, some of which are always installed, others of which require that the user's system include particular external libraries. Available vi-sualization drivers include the basic OpenGL-based [21] drivers (OGLX, OGLWin32 and OGLXm), three OpenGL drivers which are more interactive (Qt, OpenInventor [22] and OIXE) and the file-based drivers HepRApp [23], RayTracer, DAWN [24], VRML [25], gMocren [26] and ASCIITree. Some of these drivers and new fea-tures are detailed below.

3.3.1. Advances in drivers and viewers

The workhorses of the Geant4 visualization system continues to be its OpenGL drivers. Multiple OpenGL drivers are provided because different implementations are required on different operating systems or for different user memory configurations. For example, one flavor of OpenGL driver offers higher refresh speed at the cost of using more memory, while another conserves memory at the cost of speed. The user experience has been simplified so that it is no longer necessary to specify which driver to use (such as /vis/open OGLI or /vis/open OGLSWin32). Instead a single command (/vis/openOGL) may be issued from which Geant4 will select the most appropriate and capable viewer for the user's current system.

Other improvements include speed increases through the streamlining of the set of OpenGL instructions, and the ability to print any OpenGL view to high quality output by exploiting the GL2PS [27] OpenGL to PostScript printing library. OpenGL drivers in X11 and Qt modes allow the user to select ("pick") objects from the GUI in order to interrogate visualized objects, thus obtaining track, hit or geometry information.

Geant4 now supports wrapping an OpenGL viewer within the versatile, highly interactive and platform-independent Qt user interface framework. An example of this is shown in Fig. 9.

This Qt implementation includes GUI functionality to rotate, zoom and translate the view, and to pick visualization objects. A slider lets the user visually "melt away" layers of hierarchical geometries. Movies and EPS output are easily generated. A hierarchical view of the scene's graphical elements allows the user to inspect and modify the color and visibility of each element.

Another hierarchical view provides easy access to the full Geant4 help system.

New features have also been added to the Open Inventor (OI) driver. The availability of OI greatly improved in 2011 when the Coin3D [28] version of these libraries became open-source and freely available. Geant4 made extensive use of the Coin3D classes that extend the original SGI OI library, creating a distinct new "extended" driver OIXE while leaving the basic driver OIX unchanged. Fig. 10 shows an example of the OIXE viewer.

A feature implemented in this area was the ability to save the current view and return to it at any later time in the current or future runs. Views are saved and accumulated in a bookmarks file specified by the user. Each view is tagged with a user-provided or default name, and all views are listed in a scrolling auxiliary window. Clicking on a view name restores the view, or a sequence of views can be walked through via the keyboard's arrow keys. All viewpoint and camera parameters are stored in ASCII form allowing editing or external generation of bookmarks.

As in other OpenGL viewers, object selection from the GUI is supported on trajectories, hits and geometry. The OI driver provides both normal trajectory picking, where all trajectory points are shown, and reduced picking, where only the first and last points are shown. The viewer also supports mouse-over picking, whereby the element data is displayed directly in the viewer window when the mouse pointer is hovered over any object.

The remaining new developments concern moving the camera along a reference path. They are motivated by accelerator and beam line geometries, but may be useful for other large and/or extended structures. The reference path, defined in a piecewise
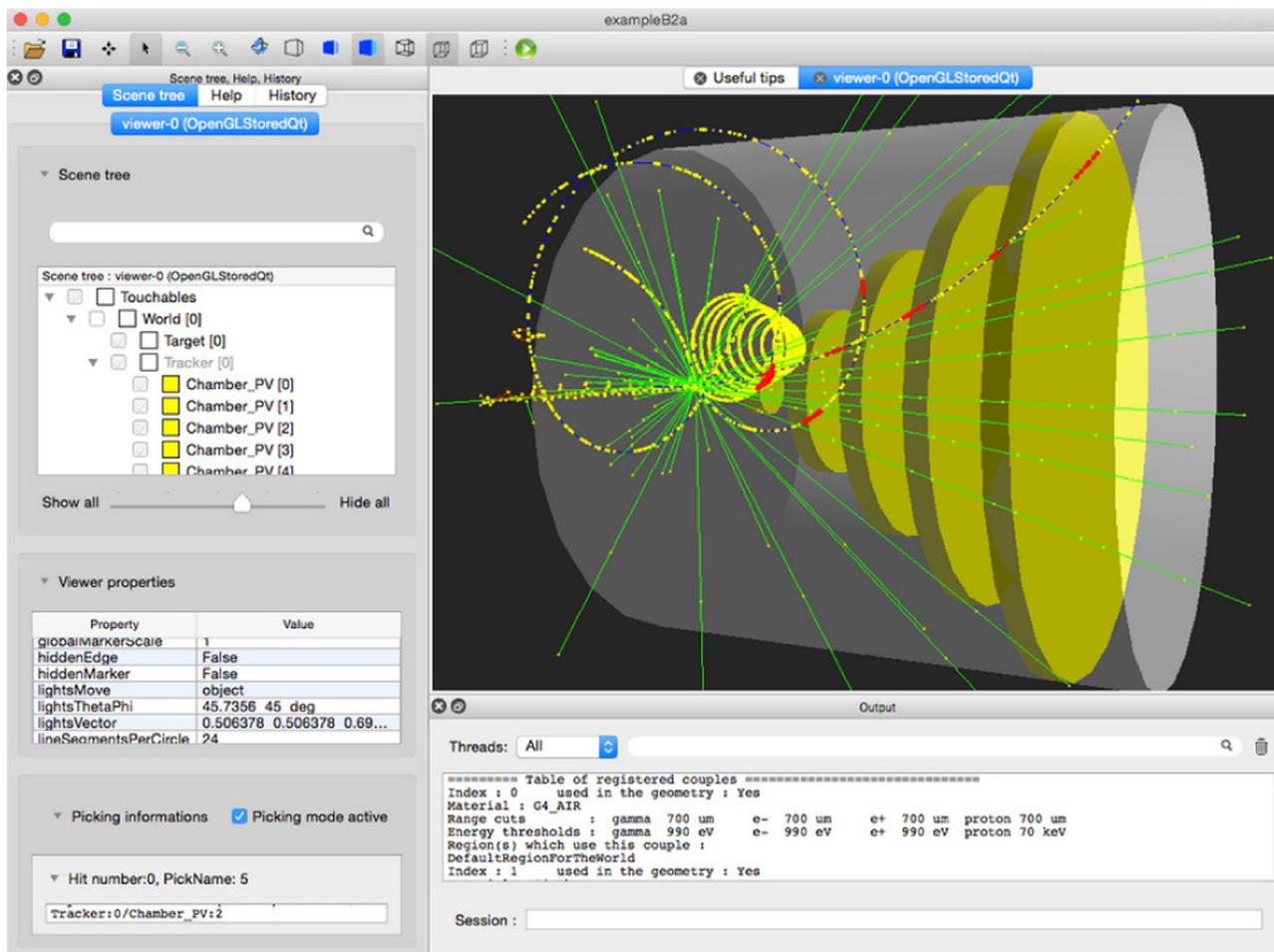

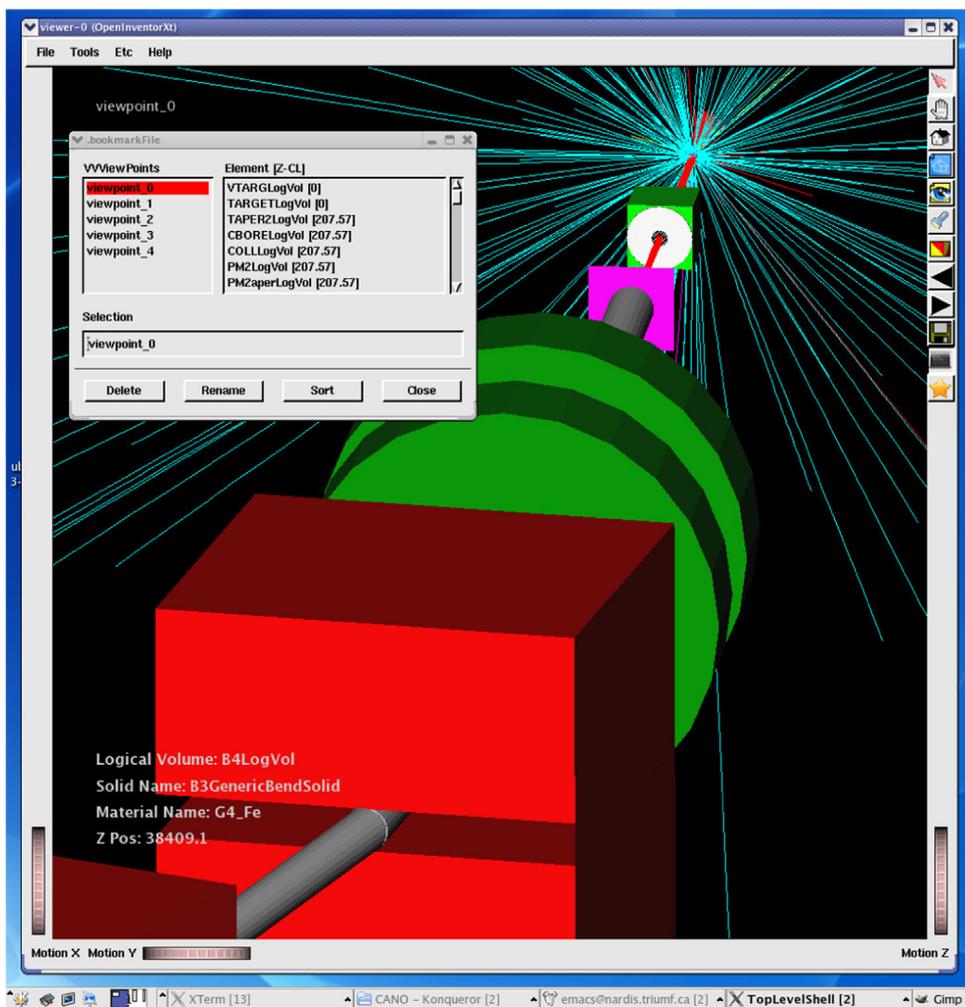**Fig. 9.** Screenshot of OpenGL viewer wrapped in Qt.

Fig. 10. Screenshot of Open Inventor Extended viewer.

linear fashion by an ordered set of points, may be read from a file or copied from any particle trajectory. Once a reference path is established, a navigation panel lists all elements in the geometry, ordered by their distance along the reference path (based on the shortest distance [29] from the element center to the path). The panel may then be used to extract information on the elements or rotate the camera around them.

A reference path animation mode moves the camera continuously along the path, allowing a fly-through giving a particle's-eye view of the geometry. Keyboard controls adjust animation speed and direction and allow adjusting the camera angle to obtain fly-overs and fly-unders.

3.3.2. New features in trajectory modeling and filtering

Many options are now provided for how trajectories should be modeled (how colors or line styles are selected). These improvements have eliminated the most common reason users had to code their own trajectory classes. In addition to the default model, where trajectories were colored by charge, one can now set color or other line properties based on particle ID, particle origin volume, or any other particle attribute that has been loaded into a `G4AttValue`. One can also add entirely new, customized trajectory models. New options make it easy to control whether trajectories are shown as basic lines, lines plus step points or step points alone, and one may also modify step point colors.

Additional new features allow trajectories to be filtered, causing only a specific subset to be drawn. These filtering options match the design of the trajectory modeling options, so that filtering based on charge, particle ID, particle origin volume, or some custom aspect, is possible. Filters may be daisy-chained so that one may show, for example, only the neutrons originating from a particular collimator.

Completing the set of additions to trajectory drawing is the ability to select smooth and rich trajectories. By default, trajectories are represented as a set of line segments connecting particle steps. Because GEANT4's efficient stepping algorithm may require very few steps in some magnetic fields, the default trajectory drawn through a solenoidal field may appear very jagged. The optional smooth trajectory drawing causes additional points to be generated along the particle trajectory so that the visualization is smoother. Rich trajectories concern the amount of additional information with which trajectories and step points are annotated. By default, trajectories have only basic information attached and step points have only position information; thus when one picks on these objects in the various pick-enabled viewing systems (HepRApp, Qt, OI or OpenGL with X11), one discovers only a few pieces of information about the trajectory and no details about the trajectory points. The rich trajectory option enriches this annotation, providing picked trajectories containing many more pieces of information, such as the entire history of geometry volumes traversed. It also adds a wealth of information to step points, such as the type of process that created the step.

3.3.3. Additional new features

Time slicing was added to allow one to produce movies that show the time development of an event. With time slicing enabled, the OpenGL line segments that represent a particle trajectory are annotated with time information. Users can then select an OpenGL view that corresponds to a given time, and a sequence of such views produces the frames of a time development movie. Users can produce these movies in any OpenGL viewer by the appropriate use of GEANT4 command macros. The Qt driver provides a simplified way for users to make such movies.

GEANT4 visualization now has the ability to retain the pointers to previously viewed events, so that after visualizing a set of events, one can go back to the beginning of the set and review the events. When coupled with customized user code that specifies which events should be kept, one can potentially run a very large set of events and then afterwards choose to visualize only those events that satisfied some personal set of trigger conditions.

The following features have also been added:

- parallel worlds, including layered mass worlds, may now be viewed individually or superimposed on the main geometry world;
- magnetic fields may be displayed as a set of arrows indicating local field direction, with arrow lengths proportional to field strength;
- decorations are now supported which allow the user to easily annotate graphical views with text (placed either in 3D coordinates or in the 2D coordinates of the graphics window), run and event number, arrows, axes, rulers, date stamps and logos;
- users may adjust the visibility or appearance of geometry by using the /vis/geometry commands which globally modify the appearance of some set of geometry objects, while the /vis/touchable commands allow control of these objects individually.

3.3.4. Approach to MT

The final set of changes concern GEANT4's migration to multi-threaded (MT) operation. The overall design of visualization remains little-changed for those users running in sequential mode, but significant changes were required to enable visualization from MT mode.

Currently in MT mode, events are only drawn at end of run, that is, once all threads have completed their work. This limitation is scheduled to be removed in release 10.2 by moving part of visualization to its own thread, such that each event is available for drawing as soon as that event is complete.

In MT mode, visualization will properly handle any commands that request drawing of high level graphical objects (geometry volumes, trajectories and decorations such as axes). However, user-supplied code that directly asks the visualization system to draw low level graphical primitives (polygons or polylines) is not supported. This limitation will not likely affect many GEANT4 users, as recent improvements to geometry, trajectory and decoration handling have made such user-supplied code largely unnecessary. Because significant work will be required to remove this limitation, support will come only if there is strong demand for these features in MT mode.

The RayTracer driver has itself been multithreaded to take maximum advantage of MT.

4. Recent developments in physics modeling

4.1. Electromagnetic physics

The GEANT4 set of electromagnetic (EM) physics processes and models [30–32] are used in practically all types of simulation applications including high energy and nuclear physics experiments, beam transport, medical physics, cosmic ray interactions and radiation effects in space. In addition to models for low and high energy EM physics for simulation of radiation effects in media, a sub-library of very low energy models was developed within the framework of the GEANT4-DNA project, with the goal of simulating radiation effects involving physics and chemistry at the sub-cellular level [33].

4.1.1. Unification of EM physics sub-packages

In the early stages of GEANT4, low and high energy electromagnetic processes were developed independently, with the result that these processes could not be used in the same run. To resolve this problem, the interfaces were unified so that the standard, muon, high energy, low energy and DNA EM physics sub-packages [31] now follow the same design.

All GEANT4 physical processes, including transportation, decay, EM, hadronic, optical and others, were implemented via the unique general interface G4VProcess. Three EM process interfaces inherit from it via the intermediate classes G4VContinuousDiscreteProcess or G4VDiscreteProcess [32]:

- G4VEnergyLossProcess, which is active along the step and post step,
- G4VMultipleScattering, which is active along the step and
- G4VEmProcess, which has no energy loss and is active post step and at rest.

These three base classes are responsible for interfacing to the GEANT4 kernel, initializing the electromagnetic physics, managing the energy loss, range and cross sections tables, managing the electromagnetic models, and the built-in biasing options. Each process inherits from one of these base classes, and has one or more physics models. EM physics models were implemented via the G4VEmModel interface. A model is applied for a defined energy range and G4Region, allowing, for example, one model from the low energy and one from the high energy sub-package to be assigned to a process for a given particle type.

Migration to this common design resulted in an improvement of overall CPU performance, and made it possible to provide several helper classes which are useful for a variety of user applications:

- G4EmCalculator: accesses or computes cross section, energy loss, and range;
- G4EmConfigurator: adds extra physics models per particle type, energy, and geometry region;
- G4EmSaturation: adds Birks saturation of visible energy in sensitive detectors;
- G4ElectronIonPair: samples ionization clusters in tracking devices.

These common interfaces enabled the full migration of EM classes to multithreading [34] with only minor modifications of the existing physics model codes. Initialization of the energy loss, stopping power and cross section tables is carried out only once in the master thread at the beginning of simulation and these tables are shared between threads at run time.

Further improvements were made through the factorization of secondary energy and angle sampling. The G4VEmAngularDistribution common interface allows the reuse of angular generator code by models in all EM sub-packages. The implementation of a unified interface for atomic deexcitation, G4VAtomDeexcitation provides the possibility of sampling atomic deexcitation by models from all EM sub-packages.

The consolidation of the EM sub-packages boosts the development of new models, provides new opportunities for the

simulation of complex high energy and low energy effects and enables better validation of EM physics [35].

4.1.2. Gamma models

The basic set of gamma models in the EM physics packages includes models developed for HEP applications [30], models based on the Livermore evaluated data library [36] and a C++ implementation of the Penelope 2008 model [37]. Recent releases of Geant4 have included revised versions of existing models, and the addition of new gamma physics processes and models. The low and high energy models were improved and display similar accuracy in their shared domain of validity [35]. These modifications not only increased model accuracy but increased computational efficiency and enabled sharing of internal physics tables, where possible, in MT mode [34]. New gamma models were added to take into account physics effects not available previously in Geant4 or in other simulation codes.

A new relativistic pair production model, `G4PairProductionRelModel`, was developed for simulations in astrophysics, LHC experiments, and other HEP applications. This model takes into account the Landau–Pomeranchuk–Migdal (LPM) effect [38], which describes the decrease of pair production cross sections at ultra-relativistic energies for dense media [39]. This model is physically accurate only above 100 MeV, as no lower energy corrections are included. It is suggested for use in HEP applications above 80 GeV. The use of the relativistic model is essential for the accurate simulation of new physics at LHC.

Two new gamma conversion models were developed to take into account the effect of gamma ray polarization:

- `G4LivermorePolarizedGammaConversionModel` and
- `G4BoldyshevTripletModel` (to be used in unison with `G4LivermoreNuclearGammaConversionModel`).

The first is responsible for sampling electron–positron pair production by linearly polarized gamma rays with energies above 50 MeV [40], while the second (currently valid only above 100 MeV) simulates the pair production process in the electron field with the emission of an additional recoil electron [41], properly taking into account the so-called "triplet" production total cross section.

The Livermore polarized gamma conversion model is based on the Heitler cross section, where the azimuthal distribution of the pair was obtained by integrating the cross section over energy and polar angles [40].

The Boldyshev triplet model uses Borselino diagrams to calculate the cross sections [42]. Most of the recoil electrons in the Boldyshev model have low energy, with a peak around $8/(T/mc^2)$, expressed in MeV, where T is the gamma energy and m is the electron rest mass. Thus, a model for the cross section was developed including a momentum threshold value of $1mc$, in order to avoid the generation of too many very low energy recoil electrons [43].

Finally, a specialized Compton scattering model `G4LowEPComptonModel` was developed [44,45]. Through the implementation of a theoretical foundation that ensured the conservation of energy and momentum in the relativistic impulse approximation [46], this model implements energy and directional algorithms for both the scattered photon and ejected Compton electron developed from first principles. It was developed to address the limited accuracy of the Compton electron ejection algorithms present in other low energy Compton scattering models that have been observed below 5 MeV [45,47,48]. Fig. 11 shows the comparison of different Geant4 Compton scattering cross sections versus NIST evaluated data [49] calculated with the methodology described in [50]. The `G4LowEPComptonModel` agrees with the reference data to within 1%, the statistical uncertainty of the simulation results. The Penelope and Standard models result in
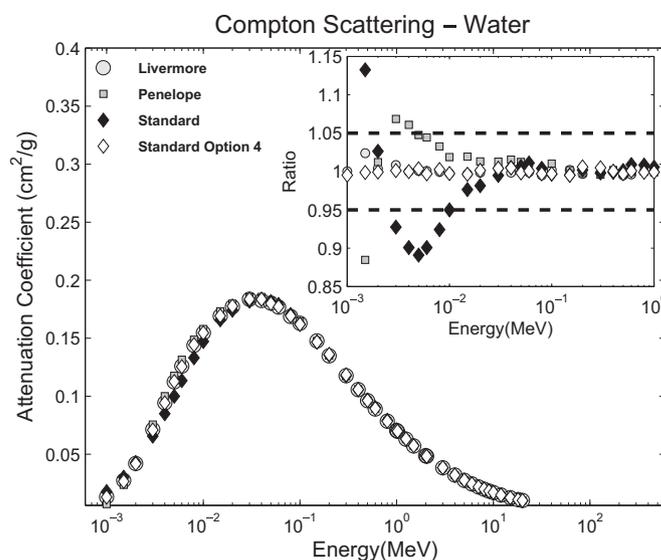
Fig. 11. Compton scattering attenuation coefficient, calculated for different Geant4 models. `G4LowEPComptonModel` is used in the Option4 EM physics configuration. The inset shows the ratio of the coefficient calculated using each alternative Geant4 electromagnetic physics list to the value from NIST XCOM [49]. The dashed lines correspond to a ± 5% difference.

differences up to 10% with respect to the NIST data for energies between 2 and 10 keV. At higher energies, the differences are smaller and are below 1% above 100 keV, corresponding to the statistical uncertainty of the simulation results.

4.1.3. Ionization models

Geant4 offers a range of ionization models for different particle types. These models can be classified as either condensed or discrete. In the condensed approach, the energy loss calculation has a continuous component and a discrete one, discriminated by a given energy threshold. Below this threshold the energy loss is continuous, and above it the energy loss is simulated by the explicit production of secondary electrons [32]. The user does not directly define the threshold because in Geant4 a special method of threshold calculations for different materials used. The user defines a unique *cut in range* [30], whose value is transformed into a kinetic energy threshold per material at initialization time of Geant4. Electrons with this kinetic energy have a mean range in a given material equal to the cut in range and gammas have an absorption length 1/5 of the range cut.

If no value is given in the reference physics lists the default cut value of 0.7 mm is used, providing sufficiently accurate simulation results for many applications. For a specific use-case, *cut in range* values should be optimized per geometry region. It is recommended that this value be defined to be less than the smallest size of geometry volumes in the region.

The *cut in range* approach may be used for other processes besides ionization. These cuts may be defined for gammas, electrons, positrons, and protons, and modified based on particle type and geometry region. However, the cut value cannot be arbitrary. Because Geant4 ionization models usually have an energy range of applicability, there is a lower limit to the electron production threshold. By default the lower limit is 1 keV, but it can be changed by the user. On top of this, any EM model may establish its own lower limit for the threshold. If several models are applied for a given particle type, then the maximum of all limit values from the models is used for the particle. For most ionization models the low limit is equal to the mean ionization potential of a material.

The list of main ionization processes and models following the condensed simulation approach is shown in Table 2.

Table 2
List of GEANT4 ionization processes and models with recommended energy range.

Particle	Process	Model	Energy range
$e-/e+$	G4eIonisation	G4MollerBhabhaModel	10 keV–10 TeV
$e-/e+$		G4PenelopeIonisationModel	0.1 keV–5 GeV
$e-$		G4LivermoreIonisationModel	0.1 keV–1 MeV
All		G4PAIModel	0.1 keV–10 TeV
All		G4PAIPhotModel	0.1 keV–10 TeV
Muons	G4MuIonisation	G4BraggModel	0.1 keV–0.2 MeV
		G4BetheBlochModel	0.2 MeV–1 GeV
		G4MuBetheBlochModel [51]	1 GeV–10 PeV
Hadrons	G4hIonisation	G4BraggModel	1 keV–2 MeV
		G4BetheBlochModel	2 MeV–10 TeV
		G4ICRU73QOModel [52]	5 keV–10 MeV
Ions	G4ionIonisation	G4BraggIonModel	(1 keV–2 MeV)/u
		G4BetheBlochModel	(2 MeV–10 TeV)/u
		G4IonParametrisedLossModel [53]	(1 keV–1 GeV)/u

In the condensed approach, a model of energy loss fluctuations must be used in conjunction with the energy loss model. The G4VEmFluctuationModel interface was developed to accommodate several implementations of fluctuation sampling, and several models derive from it:

- G4UniversalFluctuation – default model applicable to all charged particles based on a previous model [54];
- G4IonFluctuations – for small steps uses G4UniversalFluctuation and for big steps uses a Gaussian width based on a parameterization [55];
- G4PAIModel and G4PAIPhotModel – photo-absorption ionization (PAI) models [56].

PAI models simultaneously provide cross sections and energy loss, and sample energy loss fluctuations. The ionization cross sections of the PAI models derive from gamma absorption cross sections per atomic shell. They are, in general, more accurate and stable versus simulation conditions (cuts, step limits, and energy) than the default model [34,57], but are more computationally expensive because of the internal sampling at each ionization collision. An illustration of simulation performance is shown in Fig. 12 for the test beam data of the ALICE Time Projection Chamber [58,59].

Other studies show that PAI models generally fit the data independently of the step size, while the default model strongly requires the application of extra step limitations.

In the case of thin absorbers, the default model requires at least two particle steps within the defined volume. While having some difficulties for thin layers, the default model provides good physics performance for a variety of applications, in particular for tracking devices (Fig. 13), satisfying the requirements of most HEP experiments.

Recently, alternative ionization processes and models were introduced for specific applications. Contrary to the traditional condensed approach, these processes do not have a continuous energy loss component. They explicitly generate all electrons down to very low energies. They were first developed in the framework of the GEANT4-DNA project (see Section 4.1.10), which aims to model early biological damage induced by ionizing radiation at the DNA scale. The G4DNAIonisation process has different models that apply to electrons, protons and selected ions (H, alpha, alpha+, He, Li, Be, B, C, N, O, Si and Fe) in liquid water [61,62]. Similarly, a specific process, G4MicroElecInelastic, was developed for microelectronics applications, with a corresponding model that applies to electrons, protons and heavy ions in silicon [63,64].

Such models are applicable to a limited energy range and a selected set of materials, and in order to simulate realistic particle
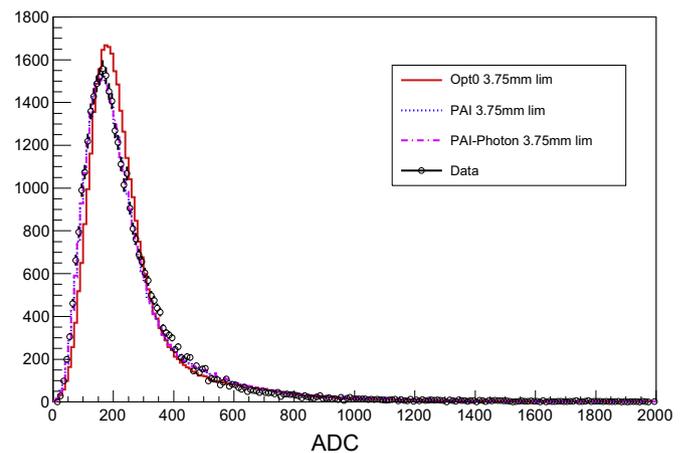
Fig. 12. Proton energy deposition in gas gap in ADC counts for a beam momentum of 3 GeV/c and a gas mixture of Ne–CO_2–N_2. The histogram represents the simulation with a 1 mm cut and a step limit equal to half the gap thickness. The ADC scale for simulation was normalized to the PAI model peak position. The open circles display the data [58,59].
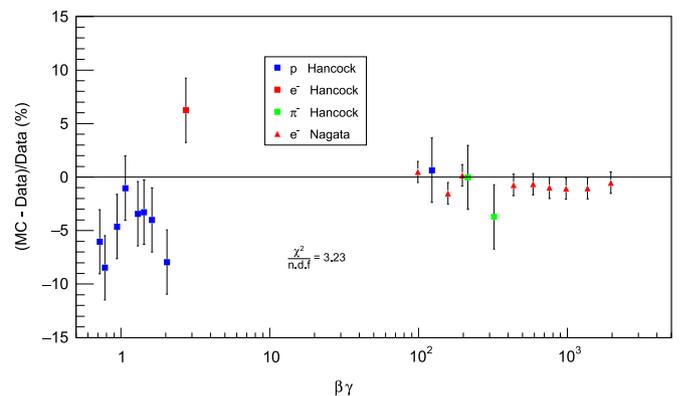
Fig. 13. GEANT4 versus data comparison of the most probable energy deposition in thin layers of silicon (thickness 300 μm Hancock; 1565 μm Nagata). Different beam particles and energies are used from the review [60]. Results are given in percent, and the default EM physics is applied with a cut in range of 100 μm.

transport, it may be necessary to combine them with a continuous ionization process. For this purpose the user may configure, for a given process and particle type, several models for different energy ranges and detector regions [31]. These discrete models produce secondary electrons without the low energy threshold used by continuous ionization models, which could lead to discontinuous transitions between models. To remedy this, the
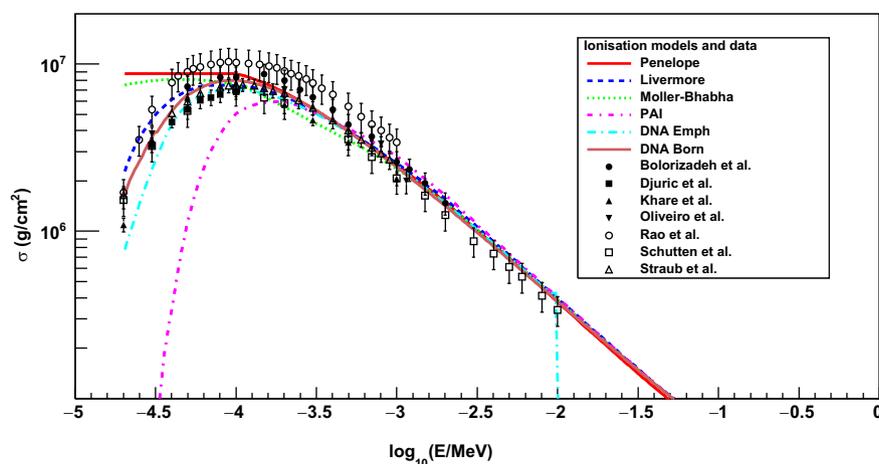
Fig. 14. Total cross section of delta electron production in liquid water as a function of projectile electron energy. Curves correspond to different GEANT4 ionization models, and points correspond to experimental data [62]. The DNA model has an upper validity limit of 1 MeV.

production of secondary electrons below the threshold may be enabled using the G4VSubCutProsessor interface, which works in parallel with the continuous model.

To illustrate this, cross sections of electrons in liquid water are shown in Fig. 14. For the condensed approach models, a delta electron production threshold of 1 keV was used and the total electron cross section was corrected for delta electron production below this threshold.

4.1.4. Multiple and single scattering

At present, the Monte Carlo simulation of charged particle transport in detailed (interaction by interaction) mode is feasible only for projectiles with relatively low energies and for thin targets. In general, the number of elastic interactions of a projectile before being stopped is very large and detailed simulation becomes impractical. The conventional solution for overcoming this difficulty is the implementation of condensed-history simulation schemes, in which the cumulative effect of many elastic scatterings along a given track segment is simulated by using multiple scattering theories such as Molière [65,66], Goudsmit and Saunderson [67] and Lewis [68].

GEANT4 offers a diverse set of multiple scattering and single scattering models [69–72]. Multiple scattering models share the same G4VMscModel interface and were tuned per particle type and application domain. Recently, the possibility of sampling the lateral displacement of a charged particle at a geometry boundary was achieved by moving the sampling of particle scattering from post-step to along-step, before sampling the energy loss and straggling.

Single scattering models sample each elastic collision of a charged particle, resulting in an increased number of steps and increased simulation time in comparison to multiple scattering models. However, single scattering models have several important applications. In particular, they are needed for the simulation of recoils [71,72], which is crucial, for example, for the understanding of single event effects in space electronics. Single scattering models are also needed to perform comparisons and validations of multiple scattering models. Single scattering models are useful for the sampling of charged particle transport in thin layers or low density media, and in the vicinity of boundary crossing between two volumes. In the majority of benchmark results for all particle types, single scattering predictions are closer to reference data than those of multiple scattering.

The choice of multiple scattering model strongly influences the CPU performance of the EM simulation. The new unified design [31] allows different multiple scattering models for different energy ranges to be used within the same simulation. The analysis of all available results from multiple scattering benchmarks [57,70,73] allows establishment of the optimal configuration of multiple and single scattering models in production physics lists.

In default physics lists, the Urban model is used below 100 MeV for electrons and positrons, where this model has a significant advantage in accuracy and CPU speed. In the combined model G4WentzelVIModel, single scattering is applied only for hard scattering, which has a limited cross section, while small angle scattering is sampled as multiple scattering [70]. The G4WentzelVIModel model provides results similar in accuracy to single scattering but it is much more computationally efficient. As such, recent versions of GEANT4 have this combined model set as the default for muon and hadron transport, and for $e^\pm$ above 100 MeV. Validation of multiple scattering models for muons and hadrons are published elsewhere [34,57,70,74].

As an example of benchmark tests carried out, Fig. 15 illustrates the ratios of simulated to measured angular distribution widths taken at the points where the distribution falls to 1/e of the peak value. The measured data taken from literature [75] include a set of different target materials (Be, C, Al, Ti, Cu, Ta, and Au) with an accuracy of about 1%. Using the G4UrbanMscModel of GEANT4 release 10.0, the predicted angular distribution widths are close to the data with a maximum deviation not exceeding 3% for both test cases of 13 and 20 MeV.

4.1.5. Radiation of charged particles

A variety of models to simulate the radiation loss of charged particles are available in the toolkit (Table 3). Significant efforts were made [73] to improve the description of EM shower shapes in order to simulate accurate $H \rightarrow \gamma\gamma$ signals in the LHC detectors [76,77]. High energy EM shower profiles are sensitive to electron/positron bremsstrahlung spectra and angular distributions. All GEANT4 models of bremsstrahlung in the intermediate energy range 1 keV to 1 GeV are based on tables of differential cross sections published by Seltzer and Berger [78]. Evaluated 2-D tables are stored in the EM data set G4LEDATA and are loaded at initialization time. The Ter–Mikaelian suppression of low energy gamma emission due to finite formation length (see [79] and references therein) is taken into account by all models.

For $e^\pm$ above 1 GeV, a relativistic model [73] was developed with an improved treatment of the LPM effect [81]. This was implemented on top of the classical Bethe–Heitler cross section with complete screening. Two types of saturation effects, LPM and formation length, have been combined to limit the number of low energy photons produced. These corrections have a distinct impact
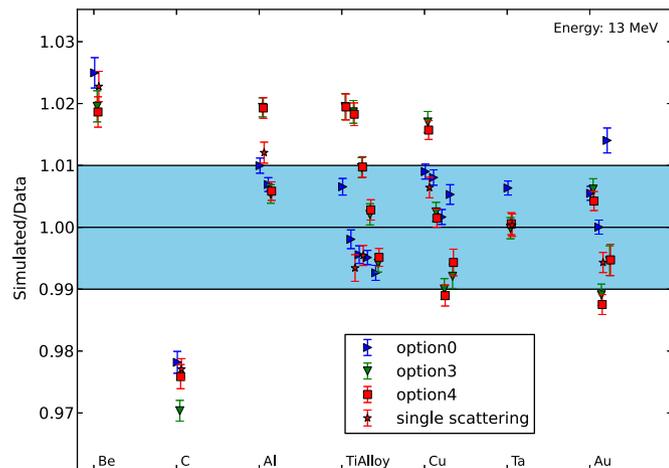
on EM shower shape and fluctuations of energy loss for high energy EM particles, of particular importance in LHC experiments.

Because simulation of radiation losses of muons is also important for LHC experiments, muon bremsstrahlung and pair production models were developed [51]. The effect of catastrophic energy loss by high energy muon bremsstrahlung is well reproduced by simulation and is essential for muon identification. The process of e^+e^- pair production by muons dominates the average energy loss at high energy [51]; proper simulation of the final state requires keeping a detailed 2-D internal table of differential cross sections (Table 3) with a structure chosen to achieve a compromise between memory usage, initialization time, and accuracy [34]. Analysis of CMS test beam data [82] indicates that bremsstrahlung and pair production by pions and protons should be taken into account. This was achieved on top of the muon processes by changing the spin term and the mass of projectile particles [80].

4.1.6. Polarization models

Models for the simulation of linear polarized gamma transport are based on the set of Livermore gamma models: photoelectric effect [83], Rayleigh and Compton scattering [84], and gamma conversion. These models have been part of GEANT4 for a long time. New gamma conversion models briefly described in Section 4.1.2 also take into account linear polarization of a primary gamma. Also the process of positron annihilation was updated, and now takes into account the correlation of photon polarization in the annihilation rest frame.

In parallel, a polarization sub-library was designed to use the standard gamma models [85]. This library allows for the simulation of circularly polarized electrons and positrons in vacuum and

Fig. 15. The MC/data ratio of angular distribution widths measured at the 1/e level for the Urban model of GEANT4 10.0. Results are shown for a 13 MeV beam on the target materials and thicknesses of the electron scattering benchmark [75].

in polarized media. For a complete simulation of a low energy polarimeter setup, all processes relevant to tracking polarized particles through matter, such as spin-dependent Compton scattering, Bhabha–Möller scattering, annihilation, bremsstrahlung and pair production, were implemented. The main application of this library is in the design of positron sources for future linear colliders [86].

4.1.7. High energy models

The processes of gamma conversion to muon pairs [87], and positron annihilation into muons and hadrons [51], were implemented to assist in the design of interaction regions within future linear colliders [88]. Other models were added for the simulation of energy loss of heavy exotic particles, in particular, magnetic monopoles [52]. Because the charges and masses of these objects are not always defined, the new models allow for the flexible definition of the energy ranges over which they are valid, and are not constrained by the lower or upper limits seen in Table 2. An efficient generator for synchrotron radiation by relativistic electrons in magnetic fields was also implemented [89] and recently generalized to synchrotron radiation for any type of long-lived charged particle.

4.1.8. Atomic de-excitation

Atomic de-excitation can be activated in all EM physics lists through the common atomic de-excitation interface `G4VAtomDeexcitation` [31]. Photo-electric effect, Compton scattering, and discrete ionization models provide cross sections of ionization for each atomic shell. The de-excitation code is responsible for sampling the electromagnetic cascade with fluorescence and Auger electron emission, and was implemented using evaluated data [90]. Recently, alternative, more accurate transition energies have become available in GEANT4 10.1 through the addition of a new data set [91].

The ionization cross section model for particle induced X-ray emission (PIXE) is based on the condensed history approach. Specific cross sections can be defined for electrons, protons, and ions. Users can select from different sets of theoretical or empirical shell ionization cross sections [92].

The simulation of K, L, and M X-ray yields demands knowledge of the X-ray production cross sections. These were calculated using the ECPSSR theory, initially developed by Brandt and Lapicki [93] and recently reviewed [94,95]. Computing the X-ray production cross sections from first principles is a time-consuming process due to the numerical double integral of form factor functions needed to obtain the ionization cross sections for each shell or sub-shell (Eq. (23) of [94]), over all possible values of the energy and momentum transfer of the incident particle.

The calculation was expedited through the use either of extensive tables and interpolation methods, or efficient algorithms providing sufficiently good approximations. Algorithms were implemented based on the ECPSSR ionization cross sections for H

Table 3
List of GEANT4 models for simulation of radiation loss with recommended energy ranges. Array size refers to the internal table storing number of primary energy points versus number of secondary energy points.

Particle	Model	Energy range	Array size
$e-/e+$	`G4SeltzerBergerModel` [73]	1 keV–10 GeV	57×32
$e-/e+$	`G4PenelopeBremsstrahlungModel`	1 keV–10 GeV	57×32
$e-$	`G4LivermoreBremsstrahlungModel`	1 keV–10 GeV	31×14
$e-/e+$	`G4eBremsstrahlungRelModel` [73]	1 GeV–10 PeV	
$\mu^\pm$	`G4MuBremsstrahlungModel` [51]	1 GeV–10 PeV	
$\mu^\pm$	`G4MuPairProductionModel` [51]	1 GeV–10 PeV	17×1000
$\pi^\pm, K^\pm, p$	`G4hBremsstrahlungModel` [80]	5 GeV–10 PeV	
$\pi^\pm, K^\pm, p$	`G4hPairProductionModel` [80]	5 GeV–10 PeV	13×1000

and He ions calculated for the K and L shells using the form factor functions for elements with atomic number 6–92 over the energy range of 0.1–100 MeV. In the case of the M shells, the ionization cross sections are given for elements with atomic number 62–92 over the energy range of 0.1–10 MeV. Furthermore, the tables generated to develop the algorithms were obtained by the integration of the form factor functions that describe the process using Lobatto's rule [96], and are also available. The cross sections generated by the algorithms deviate less than 3% from the tabulated values, roughly matching the scatter of empirical data [94]. Further details and considerations of these calculations can be found in [94,95]. Comparisons of simulated and experimental spectra obtained under proton irradiation of several materials are shown in [97,98].

4.1.9. Optical physics

Geant4 can accurately simulate nonlinear scintillators where the light yield is a function of particle type, energy deposition and kinetic energy of the ionizing particle [99]. In scintillators with a linear response, light production is directly proportional to the ionizing energy deposited and the total light produced can be computed as the sum of the light produced during small simulation steps without regard for the kinetic energy of the ionizing particle at each energy-depositing step.

In scintillators with a nonlinear response, the yield during a step is calculated as the difference in yields for hypothetical, totally absorbed particles at the kinetic energies before and after the step. This method correctly models the total light produced by a multiple step ionizing particle track, and accounts for two important cases. In the first case, light is produced correctly for incomplete energy deposition of a charged particle, such as when the particle exits the scintillator volume or when the particle is absorbed in a nuclear reaction. In the second case, the scintillation photon density is larger in the high kinetic energy portion of the track for the usual case where the nonlinear photon yield increases with particle energy. This enables the precision simulation of organic or noble gas scintillators, provided the user supplies the required data inputs.

Two more refinements in the generation of optical photons are that the scintillation process takes into account a finite light emission rise-time, and that the Cerenkov photon origin density along the track segment is no longer constant, but assumes a linear decrease in the mean number of photons generated over the step as the radiating particle slows down.

For the propagation of optical photons, the reflectivity from a metal surface may now be calculated by using a complex index of refraction [100]. Mie scattering was added following the Henyey–Greenstein approximation, with the forward and backward angles treated separately [101]. Surface reflections may be simulated using look-up tables containing measured optical reflectance for a variety of surface treatments [102]. It is possible to define anti-reflective coatings, and transmission of a dichroic filter where the transmission, or conversely the reflection, is dependent on wavelength and incident angle. The optical boundary process also works in situations where the surface is between two volumes, each defined in a different parallel world, thus allowing optical photon propagation for previously impossible geometry implementations.

4.1.10. Geant4-DNA physics models

Geant4 offers a set of physics processes and models [103] to simulate track structure in liquid water, the main component of biological media. These were developed as part of the Geant4-DNA project [104], and extend Geant4 to include the simulation of biological damage by ionizing radiation [105,106].

The first set of discrete processes was delivered in 2007 [61]. Their accuracy was further evaluated and improved [62,107,108]

through the inclusion, for example, of more accurate modeling of electron elastic scattering [109], and additional physical processes for sub-excitation electrons, such as vibrational excitation and molecular attachment [110]. These processes are critical for the modeling of physico-chemical processes in liquid water [111].

A major design upgrade of the software classes was applied in order to allow their combination with other Geant4 EM processes and models, for a coherent modeling of EM interactions [31,112]. Thus, for their simulation applications, users may instantiate a `G4EmDNAPhysics` object from their physics list. This physics constructor contains all required Geant4-DNA physics processes and models for

- electrons, including ionization, excitation, elastic scattering, vibrational excitation and attachment,
- protons and neutral hydrogen, including excitation, ionization, electron capture and stripping,
- alpha particles and their charged states, including excitation, ionization, electron capture and stripping, and
- ionization for Li, Be, B, C, N, O, Si and Fe ions.

Stopping powers and ranges simulated with Geant4-DNA have been compared to international recommendations [113]. These processes can be combined with Geant4 gamma processes. Note that the *Livermore* low energy electromagnetic gamma models are selected by default in the `G4EmDNAPhysics` constructor.

As an illustration,

```
examples/extended/medical/dna/dnaphysics
```

explains how to use this physics constructor. In addition, `examples/extended/medical/dna/microdosimetry` describes how to combine Geant4-DNA and Geant4 standard electromagnetic processes in a simple user application. A variety of applications based on Geant4-DNA processes and models allow the study of elementary energy deposition patterns at the sub-cellular scale. For example, the comparison of dose point kernels [114], S-values [115], radial doses [116], cluster patterns for ions with the same LET [117], the effect of a magnetic field on electron track structures [118], and the modeling of direct DNA damage [119–122] have so far been explored utilizing these tools. They even provide a framework for the future study of non-targeted biological effects [123], extending further the first applications of Geant4 electromagnetic physics at the physics-biology frontier [124–130].

4.1.11. Geant4-DNA physico-chemistry module

Radiation chemistry is the science of the chemical effects of radiation on matter. Simulation tools in this field are relevant to many applications, such as the production of polymers by the irradiation of monomers. However, one of the most studied materials under irradiation is liquid water, because it is used as a coolant in nuclear power plants and its irradiation may create oxidative species that are likely to initiate the corrosion process. Water is also of interest because it is the main component of biological materials.

When biological targets are exposed to radiation, the chemical response can be complex, depending on the composition of the medium as well as on the energy and type of radiation. For example, water radiolysis (dissociation of water by radiation) promotes the creation of oxidative species. These species can either react with one another or with the biological materials, interfering with the normal functioning of one or many cells.

In the context of the Geant4-DNA project, a prototype for simulating radiation chemistry was developed [33,111] and delivered with Geant4 version 10.1. It is now possible to simulate the physical stage, the physico-chemical stage (lasting up to about

1 ps) and the chemical stage (from 1 ps up to 1 μs) of water radiolysis.

The GEANT4-DNA physical processes may in some cases create water molecules which are electronically modified, that is, they may be ionized, excited or have extra electrons in the case of dissociative attachment. The electronic and atomic readjustment of the water molecules can eventually lead to their dissociation. The dissociation of water molecules is taken into account by random selection using user-defined branching ratios [111]. The positioning of the dissociative products is defined by the `G4DNA-WaterDissociationDisplacer` class from qualitative considerations [131]. It is assumed that the dissociation of water molecules is independent of the physical stage of the simulation. This is to say that only electronic modifications undergone by the water molecule are responsible for the dissociative pathways. The branching ratios and positioning of dissociative products may be adjusted by the user if necessary.

Dissociative products can recombine to form new chemical species. To take this recombination into account, a stepping algorithm was developed specifically for managing collisions between GEANT4 tracks. The purpose of this algorithm is to synchronize the transport of tracks during simulation. This means that all tracks are transported over the same time, accounting for chemical reactions that may happen during a given time step. A description of this synchronized stepping, applied to radiation chemistry, is available in [33,132].

This stepping mechanism could, in principle, be applied to applications other than radiation chemistry. A process must first be made compatible with the `G4VITProcess` interface, which contains track information. To run a radio-chemistry simulation, a table of reactions describing the supported reactions and the corresponding reaction rates must be provided.

To simplify the use of the chemistry module, the `G4EmDNA-Chemistry` constructor provides default settings, and three examples, `examples/extended/medical/dna/chem1`, `examples/extended/medical/dna/chem2` and `examples/extended/medical/dna/chem3`, are included. These examples progressively demonstrate how to activate and use the chemistry module from a user application. The module may be used with or without irradiation.

The chemistry module is compatible with the current event-level multithreaded mode of GEANT4. However, the use of the module in this mode with a large number of simulated objects or threads is not recommended because synchronized stepping requires intense memory usage.

For now, the chemistry module allows prediction of the chemical kinetics induced by ionizing radiation. A future goal of the GEANT4-DNA project is to account for DNA damage resulting from irradiation.

4.1.12. Built-in EM biasing options

Four biasing and variance reduction options are available within the EM sub-libraries of GEANT4 [35]:

- cross section biasing, which may be used to study the effects of uncertainties of cross sections;
- forced interaction, implemented for the limited use-case of a thin target;
- splitting, where the interaction of a primary of weight W which would normally produce 1 secondary of weight W, instead produces N secondaries, each with weight W/N, with no modification of the energy of the primary;
- Russian roulette, where secondaries produced by the interaction of a primary particle of weight W are killed with probability $1 − P$, and the survivors' weight is set to W/P; users may define P and the upper energy limit for secondaries for which the method is applied.

These four options are selectable through macro commands or C++ interfaces, and can be applied in user-defined `G4Region`s.

4.1.13. Validation and verification of EM models

Validation of EM physics is performed on several levels. Because EM physics is used in practically all tests and examples, the GEANT4 integrated test system routinely checks all EM physics models. A specific EM validation suite [133] runs on a regular basis for each reference version of GEANT4 (see [34,57,73] and references therein). Dedicated validations of cross sections, stopping powers, and atomic transition energies versus evaluated data and theory are done by GEANT4 developers and different user groups (see, for example, [35,134,135] and Fig. 11). EM physics validation is also performed in various application domains by different user communities, especially by the HEP experiments ATLAS and CMS.

As an example of EM physics validation for HEP applications, the energy resolution of two sampling calorimeters [136,137] versus the cut in range value and GEANT4 version is shown in Fig. 16. This plot illustrates the good agreement of GEANT4 simulation predictions with data, and the stability between GEANT4 versions of simulation results for high energy physics applications.

Further validations come from the medical and space communities, in particular, GATE [138], GAMOS [139], GRAS [140], and TOPAS [141]. There are also many validation results obtained by single user groups. For example, validations for space shielding were done recently in [142] and for therapeutic ion beam simulation in [143].

4.2. Hadronic physics

GEANT4 hadronic physics is loosely defined to cover any reaction which can produce hadrons in its final state. As such, it covers purely hadronic interactions, lepton- and gamma-induced nuclear reactions, and radioactive decay. The interaction is represented as a GEANT4 process which consists of a cross section to determine when the interaction will occur, and a model which determines the final state of the interaction.

Models and cross sections are provided which span an energy range from sub-eV to TeV. Following the toolkit philosophy, more than one model or process is usually offered in any given energy range in order to provide alternative approaches for different applications.

During the last several years, new models and cross sections have been added to the toolkit, while others have been improved and some obsolete models have been removed.
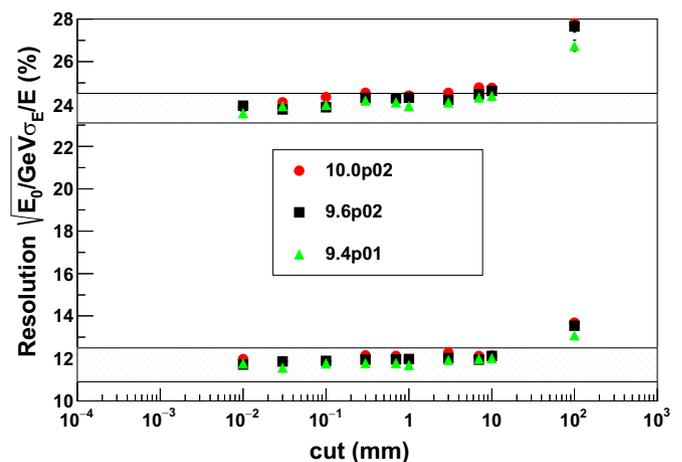

Fig. 16. Energy resolution of two sampling lead/scintillator calorimeters for 10 GeV electrons: squares, circles and triangles indicate GEANT4 simulations for different versions of the toolkit, and each band indicates experimental data with one standard deviation uncertainty [136,137].

4.2.1. Hadronic cross sections

Total, inelastic and elastic cross sections for hadron–nucleus, nucleus–nucleus and antinucleus–nucleus reactions are provided which cover energies up to TeV in some cases. Proton-, neutron- and pion-nucleus cross sections at low to medium energies have been available in GEANT4 since its beginning and their details are covered elsewhere [1]. The focus here is on recent developments in cross sections for other projectile types and at higher energies.

Barashenkov cross sections. The Barashenkov data set describes proton, neutron and charged pion cross sections (total and inelastic) on nuclei [144,145]. The Barashenkov interpolation for the total and inelastic cross sections is essentially based on a quasi-optical model for high energies ($T > 2$ GeV) and on phenomenology, with correction terms of the form $\pi r_o A^{2/3}$, with $r_o \sim 1$ fm. The total, inelastic (and elastic) cross sections were modeled with:

$$\sigma(T, A) = \pi \left[r_o A^{1/3} + \lambda(T, A) \right]^2 f(T) \phi(A)^{\alpha(T)},$$

where λ is the de Broglie length of the projectile in the center of mass system, T is the kinetic energy of the projectile in the lab, A is the atomic weight and $r_o \sim 1$ fm. The functions $f(T)$, $\phi(A)$ and $\alpha(T)$ are series of the form:

$$\sum_i a_i T^{b_i} \quad \text{or} \quad \sum_i a_i A^{b_i}.$$

The general behavior of the optical models is to predict constant cross sections for very high energies. However, experimental data show a moderate relativistic rise of hadron–nucleus cross sections. For this reason the Glauber model was used to describe hadron–nucleus cross sections in the high energy region (above 90 GeV).

Glauber–Gribov extension. The simplified Glauber model cross sections assume Gaussian-distributed, point-like nucleons and are given by [146,147]:

$$\sigma_{tot}^{hA} = 2\pi R^2 \ln \left[1 + \frac{A \sigma_{tot}^{hN}}{2\pi R^2} \right], \quad \sigma_{in}^{hA} = \pi R^2 \ln \left[1 + \frac{A \sigma_{tot}^{hN}}{\pi R^2} \right],$$

$$\sigma_{el}^{hA} = \sigma_{tot}^{hA} - \sigma_{in}^{hA}.$$

Here σ_{tot}^{hA}, σ_{in}^{hA}, and σ_{el}^{hA} are the total, inelastic and elastic cross sections, respectively.

The model is reduced to the selection of σ_{tot}^{hN} and $R(A)$ values. The latest edition of PDG [148] and GEANT4 parameterizations were used for σ_{tot}^{hN}, including the total cross sections of p, $\bar{p}$, n, $\pi^\pm$, $K^\pm$ and Σ^- on protons and neutrons. For known cross sections on protons and neutrons, $A \sigma_{tot}^{hN} = N_p \sigma_{tot}^{hp} + N_n \sigma_{tot}^{hn}$, where N_p and N_n are the number of protons and neutrons in the nucleus. The nuclear radius (the RMS radius of the nucleon Gaussian distribution), is parametrized as $R(A) = r_o A^{\frac{1}{3}} f(A)$, $r_o \sim 1.1$ fm, with $f(A) < 1$ for $A > 21$, and $f(A) > 1$ for the case $3 < A < 21$. Figs. 17 and 18 show the prediction of the Barashenkov and Glauber–Gribov model for total, inelastic and production cross sections of neutrons and protons on a carbon target. The production cross section is defined to be the difference between the inelastic and charge exchange cross sections.

Extraction of CHIPS kaon and hyperon cross sections. The cross sections for kaons and hyperons incident upon nuclei are based on the parameterization by Kossov and Degtyarenko who developed them as part of the CHIPS package [151]. This parameterization was developed using extensive data samples and contains a number of parameters which depend on the type of projectile. With GEANT4 9.6 these cross sections were made independent of the CHIPS package and their interfaces made to conform to the hadronic standard in the toolkit. They are currently used by default in production physics lists such as FTFP_BERT and QGSP_BERT.
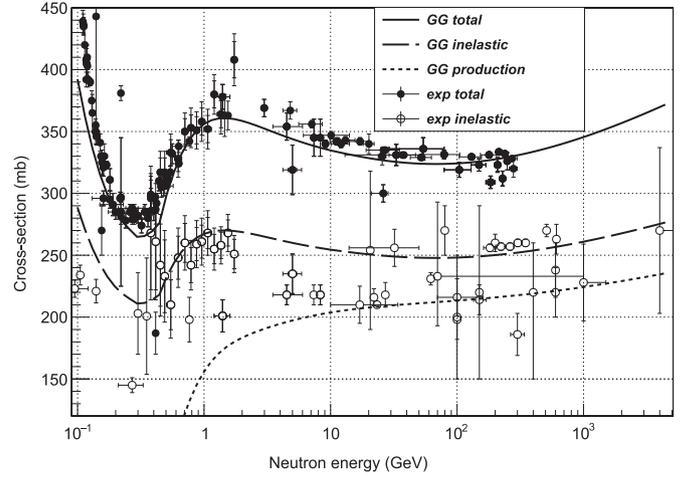
Fig. 17. Total, inelastic and production cross-sections of neutrons on a carbon target in the energy range 10^{-2}–10^3 GeV. Experimental data (open and solid points) from [149,150], lines correspond to the Glauber–Gribov model.
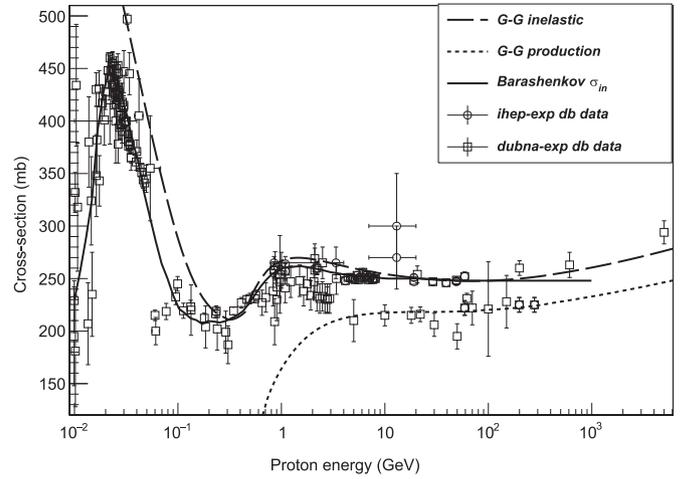
Fig. 18. Inelastic and production cross-sections of protons on a carbon target in the energy range 10^{-2}–10^3 GeV. Experimental data (open points and squares) are from [149,150]. The solid and dashed lines correspond to the Barashenkov and Glauber–Gribov inelastic models, respectively. The dotted line shows the Glauber–Gribov production model.

Antinucleus–nucleus cross sections. Production of anti-nuclei, especially anti-^{4}He, has been observed in nucleus–nucleus and proton–proton collisions by the RHIC and LHC experiments. Contemporary and future experimental studies of anti-nucleus production require a knowledge of anti-nucleus interaction cross sections with matter which are needed to estimate various experimental corrections, especially those due to particle losses which reduce the detected rate. Because only a few measurements of these cross sections exist, they were calculated using the Glauber approach [152–154] and the Monte Carlo averaging method proposed in [155,156].

Two main considerations are used in the calculations: a parameterization of the amplitude of antinucleon–nucleon elastic scattering in the impact parameter representation and a parameterization of one-particle nuclear densities for various nuclei. The Gaussian form from [152,154] was used for the amplitude and for the nuclear density the Woods–Saxon distribution for intermediate and heavy nuclei and the Gaussian form for light nuclei was used, with parameters from the paper [157]. Details of the calculations are presented in [158].

Resulting calculations agree rather well with experimental data on anti-proton interactions with light and heavy target nuclei

$(\chi^2/NoF = 258/112)$ which corresponds to an accuracy of ~8% [158]. Nearly all available experimental data were analyzed to get this result. The predicted antideuteron–nucleus cross sections are in agreement with the corresponding experimental data [159].

Direct application of the Glauber approach in software packages like GEANT4 is ineffective due to the large number of numerical integrations required. To overcome this limitation, a parameterization of calculations [146,147] was used, with expressions for the total and inelastic cross sections as proposed above in the discussion of the Glauber–Gribov extension. Fitting the calculated Glauber cross sections yields the effective nuclear radii presented in the expressions for $\bar{p}A$, $\bar{d}A$, $\bar{t}A$ and $\bar{\alpha}A$ interactions:

$$R_A^{eff} = a\,A^b + c/A^{1/3}.$$

The quantities a, b and c are given in [158].

As a result of these studies, the GEANT4 toolkit can now simulate anti-nucleus interactions with matter for projectiles with momenta between 100 MeV/c and 1 TeV/c per anti-nucleon.

Nucleus–nucleus cross sections. The simulation of nucleus–nucleus interactions and the corresponding cross sections is required by accelerator experiments, cosmic ray studies and medical applications, to name a few domains.

Because nuclei are charged, total and elastic cross sections are infinite due to Coulomb interaction. In reality, they are restricted by the screening of the atomic electrons. This interaction leads to a small-angle scattering which can be ignored in a first approximation. Thus, inelastic cross sections are the most important ones. With increasing energy electromagnetic dissociation (EMD) becomes dominant, especially for the collisions of heavy nuclei. At low and intermediate energies EMD does not play an essential role, while the nuclear break-up and multi-particle productions dominate.

The strong interaction cross sections can be calculated in the Glauber approximation [156,160] at high (> 1 GeV) energies. The description of the cross sections at low and intermediate energies is the challenging component.

A first simple expression was proposed in [161]: $\sigma_{1,2} = \pi(R_1 + R_2 - c)^2$, where R_1 and R_2 are the radii of the two interacting nuclei ($R = r_0\,A^{1/3}$), $r_0 \simeq 1.36$ fm, and $c \sim 0$–1.5 fm, depending on a projectile energy (following [162] and the further refinements of [163] $c \propto (A_1^{-1/3} + A_2^{-1/3})$).

In order to extend the parameterization to the intermediate energy range [164] $\sigma_{AB} = \pi R_{int}^2\,(1 - B/E_{CMS})$ can be used, where R_{int} is composed of two terms, energy dependent and independent, $B = Z_A Z_B e^2/r_C(A^{1/3} + B^{1/3})$ is the Coulomb barrier of the projectile–target system, and E_{CMS} is center-of-mass system energy.

In GEANT4 the "Sihver", "Kox" and "Shen" parameterizations [163–165] are used, with the Shen parameterization recommended for all physics lists.

4.2.2. Hadronic models and processes

Due to the large energy range covered, it is not possible for a single model to describe all the physics encountered in a simulation. A typical sequence of reactions may begin with a high energy hadron–nucleon collision within a nucleus (QCD or parton string model), followed by the propagation of the secondaries of the collision through the nuclear medium (intra-nuclear cascade model), followed by the de-excitation of the remnant nucleus (precompound model) and its evaporation of particles (evaporation and breakup models) until it reaches the ground state. Each of these stages is qualitatively different from the other and will be discussed in turn, highlighting recent developments.

Other reactions covered include nucleus–nucleus interactions (QMD or parton-based models), elastic models (including coherent hadron scattering from nuclei), capture, and lepton- and gamma-nuclear interactions.

Quark–gluon string models. Two models based on quark-parton concepts were implemented in GEANT4, the quark–gluon string (QGS) model [166,167] and the Fritiof (FTF) model. [168,169]. The QGS model is described in [30]. A short description of the FTF model is presented here, but more details are available in the GEANT4 Physics Reference Manual [170].

The FTF model is used in GEANT4 to simulate the following interactions: hadron–nucleus at incident laboratory hadron momenta >3–5 GeV/c, nucleus–nucleus at incident laboratory hadron momenta >2–3 GeV/c/nucleon, antibaryon–nucleus at all energies, and antinucleus–nucleus. Because the model does not include multi-jet production in hadron–nucleon interactions, the upper limit of its validity range is estimated to be 1 TeV/c per hadron.

The modeling of hadron–nucleon interactions in the FTF model includes the simulation of elastic scattering, binary reactions such as $NN \rightarrow N\Delta$ and $\pi N \rightarrow \pi\Delta$, single diffractive and non-diffractive events, and annihilation in anti-baryon–nucleon interactions. Interactions proceed by the production of one or two unstable objects called quark–gluon strings. If only one string is created, the process is called diffraction dissociation.

In the GEANT4 implementation single diffraction dissociation is simulated separately from non-diffractive interactions. A special function which corresponds to a weighted simulation of the diffraction dissociation was introduced to perform this separation. In most other Fritiof-based models this separation is governed by a single parameter, which is not sufficient for a correct description of the large set of experimental data.

Once formed, strings may interact with other nucleons in hadron–nucleus and nucleus–nucleus collisions, producing additional strings. Strings with sufficiently large mass (> 1.2 GeV) may in general have kinks, which are treated as emitted gluons which decay into quark–antiquark pairs. This feature is required in order to reproduce particle multiplicities observed in hadronic interactions at high energies. However, the current FTF implementation does not handle kinks, hence the TeV/c upper limit.

Hadron–nucleon scattering within the model uses the elastic and inelastic cross sections taken from the CHIPS parameterizations [151]. Cross sections for binary reactions and diffraction dissociation were implemented directly in the FTF model as parameterizations of data. Here the cross sections for the unstable objects were taken to be the same as those for stable objects with the same quark content.

Once the unstable objects are produced, the LUND string fragmentation model is used to decay them [171]. The parameters of this model were tuned to experimental data and the available phase space was restricted to take into account low mass string fragmentation. The formation time of hadrons was also applied.

To simulate hadron–nucleus and nucleus–nucleus scattering it is necessary to embed the hadron–nucleon interaction in the nuclear environment. This was done by first assuming a Woods–Saxon parameterization of the one-particle nuclear density for medium and heavy nuclei and a harmonic oscillator shape for light nuclei. Center-of-mass correlations and short-range nucleon–nucleon correlations were taken into account. A simplified Glauber model was used to sample the multiplicity of intra-nuclear collisions. Screening was not considered; estimates and data indicate that it decreases the total hadron–nucleus cross sections by 3–5%, while the inelastic hadron–nucleus cross sections are practically unchanged [172]. Hence any effect on the multiplicity of produced particles is expected to be small.

The number of string objects in non-diffractive interactions is proportional to the number of participating nucleons. Thus, multiplicities in hadron–nucleus and nucleus–nucleus interactions are larger than those in elementary interactions. It is assumed that the reaction creating unstable strings in hadron–nucleus collisions is analogous to that in nucleus–nucleus collisions.

It is known that the Glauber approximation used in this and other string models does not provide enough intra-nuclear collisions for a correct description of nuclear destruction. Traditional cascade models would fulfill this need, except that they produce too many particles. Reggeon theory has been proposed to solve this problem [173], but the detailed calculation required was not appropriate for a reasonably fast computer code. A simplified implementation in GEANT4 assumes [174] that participating nucleons predicted by the Glauber approximation initiate low energy reggeon exchanges in the spectator part of a target nucleus. This reggeon theory inspired model (RTIM) provides the right number of fast nucleons ejected during nuclear destruction and describes secondary particle intra-nuclear cascading [175].

The collective nature of nuclear destruction led to the introduction of a new "Fermi motion" [176,174] simulation which is a refined algorithm for putting involved nucleons on the mass-shell. As shown in Fig. 19, this provides sufficient agreement with experimental data in the transition region, around 3 GeV/c.

When the cascading is finished, the excitation energies of residual nuclei are estimated in the "wounded nucleon" approximation [177]. This allows for a direct coupling of the FTF model to the GEANT4 precompound model and its nuclear fragmentation sub-models. The determination of the particle formation time also allows coupling of the FTF model to the GEANT4 Binary cascade model [178].

Two additional innovations were made in the FTF model, one dealing with low-mass strings and the other making multiplicity corrections in hadron–nucleus and nucleus–nucleus reactions.

All Monte Carlo event generators are challenged with the correct treatment of low mass strings. Such a string is typically handled by first checking that it can decay into two low mass particles. If it can, the decay is simulated; otherwise, the string is converted into a hadron. This step violates energy–momentum conservation and the momenta of all other produced particles must be adjusted to correct for this. In the FTF model all strings with sufficiently large mass are allowed to decay to two particles. As a result, the cross sections of the reactions $\bar{p} + p \rightarrow \bar{n} + n$, $\bar{p} + p \rightarrow \bar{\Lambda} + \Lambda$, and so on, are reproduced. In the case of a two-particle decay, all possible final states are considered, and one is chosen according to its phase space volume. For other final states, standard string fragmentation or direct production of a hadron is possible.

Multiplicity corrections in hadron–nucleus and nucleus–nucleus interactions are necessary when computing the number N_{bin} of intra-nuclear collisions. The distribution of N_{bin} is usually obtained by applying the asymptotic AGK cutting rules [179] to the Glauber amplitude for elastic scattering. These rules must be corrected for finite energies. Because there is no defined prescription for making these corrections, a phenomenological approach, taking into account formation time and using HARP-CDP data [180], was adopted.

Intranuclear cascade models. Three intranuclear cascade models are now offered in GEANT4: Bertini, Binary and INCL++. The extended Bertini cascade [181] is valid for p, n, π, K, Λ, Σ, Ξ, Ω and γ projectiles with incident energies between 0 and 15 GeV. It is also valid for captured μ^-, K^- and Σ^-. Recent extensions allow this model to be used for cascades initiated by high energy muons and electrons. Although this model has its own precompound and deexcitation code, an option exists for using the native GEANT4 precompound and deexcitation modules discussed in the following section.

The Binary cascade [182] simulates p and n-induced cascades below 10 GeV, and π-induced cascades below 1.3 GeV. This is done by propagating hadrons in a smooth nuclear potential, and forming and decaying strong resonances to produce secondaries. The model relies on the native GEANT4 precompound and deexcitation code to handle the post-cascade steps.
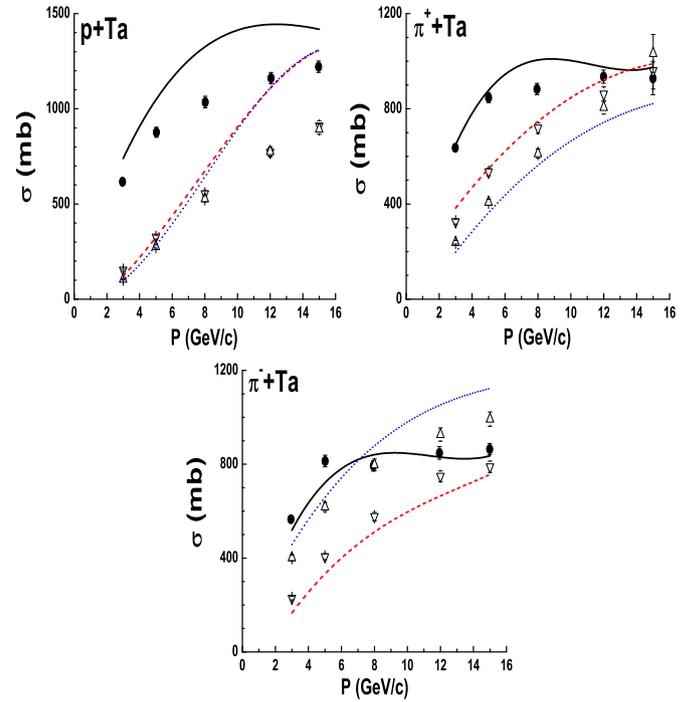
Fig. 19. Inclusive cross sections for p, π^+ and π^- production in pTa, π^+Ta and π^-Ta interactions as a function of projectile hadron momentum. Data from the HARP-CDP group [180] are shown as closed circles for protons and up- and down-triangles for π^+ and π^-, respectively. Lines are FTF model calculations: solid for protons, dashed and short-dashed for π^+ and π^-, respectively.

The Liège Intranuclear Cascade model (INCL) [183] has seen extensive development since its introduction in GEANT4. The original Fortran model was completely redesigned and rewritten in C++ and is now known as INCL++ [184]. It extends the applicability of the legacy version up to ∼15 GeV incident energy, while remaining physics-wise equivalent for nucleon- and pion-induced reactions below 1 GeV. In addition, INCL++ has been extended to handle reactions induced by light ions up to $A = 18$. By default, INCL++ uses the GEANT4 native de-excitation immediately after the cascade stage; it does not include an intermediate pre-equilibrium step. Coupling to the ABLA V3 de-excitation model [185] is also possible.

The precompound model. The native GEANT4 pre-equilibrium model is based on a version of the semi-classical exciton model [186] and is used as the back-end stage of several cascade and quark–gluon string generators. It handles the de-excitation of the remnant nucleus from its formation immediately following a cascade or high energy collision, until it reaches equilibrium. During this time, internal transitions of the pre-compound nuclear system compete with nucleon and light cluster emissions. The passage to the state of statistical equilibrium, which happens when the transition probabilities for increasing and decreasing the exciton number become approximately equal (equilibrium condition), is roughly characterized by an equilibrium number of excitons n_{eq}. In the simulation n_{eq} is a calculated number based on the assumption that the equilibrium condition is met.

Some refinements were introduced recently [187–189], namely more realistic inverse cross section parameterizations and combinatorial factors for particle emission, a phenomenological parameterization of the transition matrix elements, and a more physically consistent condition for the transition to equilibrium, since in certain circumstances this condition is reached well before the previously used rough estimate of n_{eq}.

At the end of the pre-equilibrium stage, the residual nucleus should be left in an equilibrium state, in which the excitation

energy is shared by the entire nuclear system. Such an equilibrated compound nucleus is characterized by its mass, charge and excitation energy with no further memory of the steps which led to its formation.

Nuclear de-excitation models. The final de-excitation of a nucleus to a thermalized state is simulated by several semi-classical models which are responsible for sampling the multiplicity of neutrons, protons, light ions, isotopes, and photons. They are:

- Fermi break-up (FBU) [190],
- statistical multifragmentation [190],
- fission, based on the Bohr–Wheeler semi-classical model [191,192],
- evaporation of nucleons and light fragments, which is handled by models based on either
  - the Weisskopf–Ewing model [193] for fragments up to and including α particles, or
  - the generalized evaporation model (GEM) [194] for the emission of fragments with masses up to ^{28}Mg, and
- G4PhotonEvaporation, which simulates the emission of
  - discrete gammas according to the E1, M1 and E2 transition probabilities taken from the PhotonEvaporation database, which in turn is based on the Evaluated Nuclear Structure Data File (ENSDF) [195], and
  - continuous gammas according to the E1 giant dipole resonance strength distribution.

These models are managed by the G4ExcitationHandler class, in which they may be invoked in complement or sometimes concurrently with each other. Some of them have been thoroughly validated and have undergone continuous improvement in recent years [187,196].

In order to properly describe the double differential cross sections and isotope production data of the IAEA spallation benchmark [197,189], the standard and GEM models were combined to form a hybrid model, and the fission model was improved [198,188,189].

For radiobiological applications it is essential that the FBU model be used by default for the de-excitation of light fragments ($Z < 9$, $A < 17$, taking into account Pauli blocking and all possible decay channels into stable and long-lived fragments. Validations [199,196] triggered many of the refinements to this model.

For proton and ion beam therapy applications, the photon evaporation model, which is critical for the tracking of the Bragg peak from emitted prompt gammas, was improved [200]. The statistical multifragmentation model, responsible for the explosive break-up of heavier hot nuclei ($Z > 8$, $A > 16$, and excitation energy >3 MeV/u), is relevant in simulations of shielding from cosmic radiation and has also been validated [188,199].

Elastic scattering models. Four options exist in GEANT4 for the simulation of elastic hadron scattering from nuclei: the GHEISHA-based and CHIPS-based parameterized models, the Glauber approach, and diffuse diffraction.

The GHEISHA-based models (G4HadronElastic) [201] are valid for all long-lived hadrons at all incident energies. They sample the momentum transfer from a sum of exponentials whose amplitudes are parameterized in Z, A and incident momentum. These models are fast, but significantly overshoot the data at large angles.

The CHIPS-based models (G4ChipsElasticModel) [151] are similar, but sample the momentum transfer from a more complex parameterization which includes details of the nuclear size and potential. Hence, features like diffraction minima are represented. This model is valid for incident protons, neutrons, pions, kaons and anti-protons at all energies.

The G4ElasticHadrNucleusHE model depends on Glauber multiple scattering [202] in a nucleus which is described by its

impact parameter profile. The energy dependence of the scattering is largely determined by a phenomenological combination of hadron–nucleon cross sections. The model is valid for all long-lived hadrons of energies greater than 1 GeV.

The G4DiffuseElastic model [203] uses an optical model of the nucleus and takes into account the diffuse nuclear halo as well as Coulomb effects. It is valid for incident protons, neutrons, pions and kaons of all energies.

The four models are compared to data for 1 GeV protons on silicon in Fig. 20.

Stopping models. The GEANT4 toolkit includes processes to simulate the stopping and capture of hadrons and heavy leptons on nuclei. Previous parameterized models for π^- and K^- capture were replaced by the GEANT4 Bertini intranuclear cascade for negative mesons, baryons and muons, and with the FTF string model for antibaryon capture and annihilation.

The Bertini cascade model handles the capture process by selecting a random location within the nucleus, weighted by the radial nucleon density, to initiate the cascade process. The subsequent cascade is propagated using the same code used for any other hadron–nucleus interaction. In the FTF model, the antibaryon annihilates with a nucleon near the outer "surface" of the nucleus, producing a small number of pions. Those secondaries and the excited nucleus are passed either to the GEANT4 de-excitation model (invoked by G4PrecompoundModel), or to one of the cascade models (Bertini or Binary) for final disposition depending on the energy.

Capture of negative muons on nuclei is also handled using the Bertini model. The muon capture process deals with the atomic capture, the subsequent electromagnetic cascade of the muon down to the lowest orbit, including photon and Auger electron emissions, and the decision about whether the bound muon should decay in that orbit or be captured into the nucleus. If the latter, the Bertini cascade selects a random location within the nucleus (weighted as above), where the μ^- interacts with a proton, producing a neutron, neutrino, and one or more radiative photons. Note that in GEANT4 release 10.0 onward, the radiative cross sections have been set to zero. As above, the neutron is treated as a cascade secondary and processed using the existing code of the Bertini model.

Low energy neutron models. As of GEANT4 10.0, there were three models treating low energy neutrons with high precision: NeutronHP, LEND and NeutronXS. The NeutronHP models, for elastic, inelastic, capture and fission reactions, have been part of GEANT4 for many years [30]. They depend on the GEANT4 Neutron Data Library (G4NDL) for cross sections, branching ratios, final state
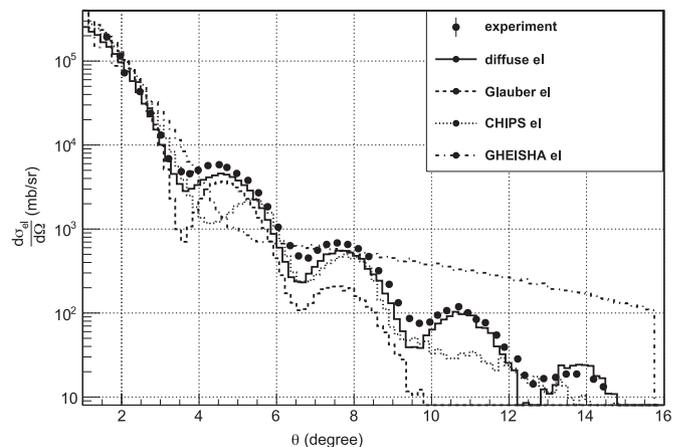

Fig. 20. Differential elastic scattering cross sections for 1 GeV protons on silicon versus polar scattering angle. The histograms represent the diffuse, Glauber, CHIPS and GHEISHA models. The solid circles are experimental data [204].

multiplicities and final state energy and angular distribution parameters. In its original formulation, G4NDL data were drawn from nine different databases, ENDF/B-VI [205], Brond-2.1 [206], CENDL2.2 [207], EFF-3 [208], FENDL/E2.0 [209], JEF2.2 [210], JENDL-FF [211], JENDL-3 [212] and MENDL-2 [213], with the majority coming from the Fusion Evaluated Nuclear Data Library (FENDL). This changed in GEANT4 version 9.5 when G4NDL became solely dependent on US ENDF/B-VI and VII (Evaluated Nuclear Data Files) [205]. Although the other databases are no longer used by G4NDL, they are still available for use with GEANT4.

Many evaluated data libraries, such as ENDF/B-VII.0, JEFF-3.1 and JENDL-4.0, have been converted to the GEANT4 format [214] and can be obtained at the IAEA Nuclear Data Services website [215].

NeutronHP was recently extended to include a new, detailed fission fragment generator (FFG) which was designed to model the complete detectable portion of a fission event. The event is modeled by taking into account mass and momentum conservation. Fission products, from gammas to nuclear fragments are produced with realistic energies. Ternary fission is supported, even though its probability is low, but is currently limited to alpha particles. The FFG is data-based, but designed to accommodate direct physics models. An example of this is symmetric fission and its angular dependencies. This also allows the FFG to fission any isotope, provided that the daughter product yield data are available.

Because NeutronHP provides detailed cross sections in the resonance region and on-the-fly Doppler broadening, the code can be quite slow and for this reason is often not used in physics lists. In order to provide improved CPU performance while retaining part of the NeutronHP precision, the NeutronXS elastic and capture cross sections were developed. These cover neutron energies from sub-eV up to the TeV range. Below 20 MeV the detailed cross section data of the NeutronHP models are binned logarithmically and tabulated. This preserves most of the resonance structure. At all energies the final state for elastic scattering and capture is generated using algorithms from the `G4ChipsElasticModel` and `G4NeutronRadCapture` models. The NeutronXS cross sections yield a roughly four-fold decrease in CPU time for neutron propagation compared to the NeutronHP models and, as of release 10.0, are now standard in most physics lists.

Another alternative to NeutronHP is the set of LEND (Livermore) neutron models. These were designed to be faster than NeutronHP, with more streamlined code, but to provide the same physics. In the LEND models the cross sections are evaluated at a fixed temperature, rather than the on-demand temperature calculations of NeutronHP. This results in a factor five increase in speed. These models use the Livermore online database for their reaction data. It is based in part on ENDF/B-VII and can be obtained from the Lawrence Livermore National Laboratory ftp site [216].

Nucleus–nucleus models. As of release 10.0 there were six GEANT4 models capable of handling nucleus–nucleus collisions: binary light ion, abrasion/ablation, electromagnetic dissociation, QMD, INCL++ and FTF models.

The Binary Light Ion model handles collisions in which either the projectile or the target has mass $A < 13$. Based on the GEANT4 Binary Cascade model [182], it is valid above 80 MeV and below 10 GeV/nucleon.

Operating over a similar energy range, but without limits on the projectile or target masses, the `G4WilsonAbrasion` model, based on NUCFRG2 [217] is faster, but less detailed, than the Binary Light Ion model. It is a geometrical model in which a portion of the target nucleus along the incident path of the projectile is gouged out, forming a forward-going compound nucleus and a residual target. The associated Wilson ablation model is used to de-excite the products of the initial collision.

Also based on NUCFRG2, `G4EMDissociation` is an electromagnetic dissociation model provided to handle the production of nuclear fragments resulting from the exchange of virtual photons between projectile and target nuclei. This model is valid for nuclei of all masses and all energies.

QMD (quantum molecular dynamics) is a native GEANT4 model based on an extension of the classical molecular dynamics model introduced in release 9.1. Each nucleon in the target and projectile nuclei is treated as a Gaussian wave packet which propagates with scattering through the nuclear medium, taking Pauli exclusion into account. The nuclear potential is that of two merging nuclei and its shape is re-calculated at each time step of the collision. Participant–participant scattering is also taken into account. These last two facts combine to make the model rather slow for collisions of heavy nuclei, but the production of nuclear fragments versus energy is well reproduced. The model is valid for all projectile–target combinations and for projectile energies between 100 MeV/nucleon and 10 GeV/nucleon. Since its introduction, the model was made Lorentz covariant and improvements were made in fragment production at relativistic energies.

The INCL++ model, covered above, can also accommodate nucleus–nucleus reactions provided the projectile has a mass below $A = 19$ and an energy between 1 MeV/nucleon and 3 GeV/nucleon. A broad validation campaign on heterogeneous observables has shown that, in spite of the conceptual difficulties, the extended INCL++ model yields predictions in fair agreement with experimental data; it is however crucial to make a suitable choice for the coupling with the statistical de-excitation model.

The FTF model, covered above, is capable of modeling reactions with all combinations of projectile and target mass, with projectile energies in the range 2 GeV/nucleon to about 1 TeV/nucleon. However, validation of this application is still in progress, and collisions of two heavy nuclei are expected to be computationally expensive.

The radioactive decay process. The `G4RadioactiveDecay` process and model handles α, β^-, β^+, isomeric transition (IT) and electron capture (EC) decays, and can be applied to generic ions both in flight and at rest.

Details for each decay or level transition, such as nuclear level energies, branching ratios and reaction Q values, come from the GEANT4 RadioactiveDecay database, which currently contains entries for 2798 nuclides. Details of specific gamma levels used for IT decays are taken from the GEANT4 PhotonEvaporation database. Both the PhotonEvaporation and RadioactiveDecay databases take their data from the Evaluated Nuclear Structure Data File (ENSDF) [195] and have recently been rationalized so that their common nuclear levels have identical values.

Beginning with GEANT4 release 9.6 and continuing through releases currently in preparation, a number of improvements have been made to the radioactive decay package. These include:

- a complete review of the PhotonEvaporation and RadioactiveDecay databases, and updating to the 2013 version of ENSDF,
- the ability to model decays with lifetimes as short as 1 ps,
- decays of observationally stable ground states, that is, those having very long predicted life times, but which have not yet been observed to decay,
- the addition of unique first, second and third forbidden β^- and β^+ decays,
- the default invocation of the atomic relaxation model after IT and EC decays, and
- improved energy conservation for all decay modes.

Gamma- and lepto-nuclear models. Due to the relatively small electromagnetic coupling, gamma- and lepto-nuclear reactions play a small role in high energy physics calorimetry. They are important, though, for nuclear, medium energy and cosmic ray

physics. For this reason GEANT4 models for these reactions were extended and improved.

The `G4PhotoNuclearProcess` is implemented by two models, the Bertini cascade below 3.5 GeV and the quark–gluon-string (QGS) model above 3 GeV. Both models treat the incident gamma as if it were a hadron interacting with a nucleon within the nuclear medium. Nevertheless, below 30 MeV the Bertini model does capture some collective nuclear effects such as the giant dipole resonance.

Both the electro-nuclear and muon-nuclear models (`G4ElectroVDNuclearModel` and `G4MuonVDNuclearModel`) exploit two features of the hybrid electromagnetic hadronic interaction: the factorization of the interaction into separate hadronic and electromagnetic parts and the treatment of the exchanged photon as if it were a hadron. The electromagnetic vertex produces a virtual photon from a two-dimensional cross section table and uses the method of equivalent photons to make the photon real. As in the photo-nuclear case mentioned above, the photon is then treated as a hadron for the remainder of the interaction. For real photons below 10 GeV the Bertini cascade handles the interaction; above 10 GeV the photon is converted to a neutral pion and the interaction proceeds using the FTF string model.

Obsolete models. The first GEANT4 hadronic models, the low energy parameterized (LEP) and high energy parameterized (HEP), were both re-engineered versions of the Fortran code Gheisha [201]. In their original form they were designed and tuned to reproduce high energy shower shapes. They conserved energy on average, but not on an event-by-event basis. With the advent of more theoretically detailed models, both LEP and HEP models were removed from version 10.0 of the toolkit.

Prior to GEANT4 10.0, a number of models and cross section sets dealing with nuclear de-excitation, hadron capture, gamma-nuclear and lepton-nuclear reactions were implemented by the chiral invariant phase space (CHIPS) package. Since version 10.0, most of these reactions have been implemented by other models although some of the cross sections still maintain the original CHIPS coding.

Lastly, the isotope production model, used to count recoil nuclei produced in various reactions, was removed in version 10.0, as all hadronic models now produce recoil nuclei explicitly.

4.3. Physics lists

In GEANT4, physics lists refer to classes which provide the means to collect and organize the particle types, physics models and cross sections required for a particular simulation application. These classes allow physics processes to be registered to the run manager which in turn attaches them to tracks so that they may interact properly with the simulation geometry.

When originally conceived, physics lists were intended to give the user maximum flexibility and responsibility to choose and combine particles, models and cross sections. Developers thus did not provide default physics or specific physics combinations to users, except in certain custom situations. It eventually became clear from user requests that ready-made and validated physics modules were desired which could be easily plugged into user physics lists. This led to the development of several "reference" physics lists which were specialized to provide standard behavior in various application domains. In medical or space applications, for example, detailed atomic physics may be required, while for high energy physics it is not. In high energy applications TeV scale physics is important, but not for medium and low energies.

While the basic, maximally flexible physics list classes are still available and fully documented in the GEANT4 Application Developers Guide [15], the focus here is on prepared, modular physics lists which are organized around builders and constructors.

4.3.1. Constructors

All prepared physics lists in GEANT4 derive from the class `G4VModularPhysicsList` which in turn derives from the base class `G4VUserPhysicsList`. `G4VModularPhysicsList` maintains a vector of physics modules, each of which is an implementation of the `G4VPhysicsConstructor` class. A module, referred to here as a physics constructor, enables the logical grouping of related physics models, cross sections and particles. This allows the most accurate and CPU-appropriate physics models to be applied to given energy ranges and particle types. The `ConstructParticle()` and `ConstructProcess()` methods of `G4VPhysicsConstructor` can be used to instantiate the particle types needed for a given application, and to assign models and cross sections to processes. For example, all pions and kaons could be instantiated in a meson physics constructor, which would also assign two hadronic models, a cascade model at low energies and parton string model at high energies, to the processes pertaining to pions and kaons.

The chosen electromagnetic and hadronic constructors are stored in a vector, which makes it easy to build a new physics list by adding, removing or replacing physics constructors. A large collection of these constructors is included in the GEANT4 toolkit. The electromagnetic constructors are listed in Table 4 and examples for using them [31] are distributed with the release.

4.3.2. Builders

It is convenient to implement the physics constructors with more granular physics components. As an example, consider the physics constructor `G4HadronPhysicsFTFP_BERT`, which implements all inelastic hadronic interactions by using the FTF parton string and Bertini cascade models. It implements the `G4HadronPhysicsFTFP_BERT::ConstructProcess()` method by instantiating and invoking several builder classes, such as `G4FTFPNeutronBuilder`, `G4BertiniPiKBuilder` and `G4HyperonFTFPBuilder`.

Each type of builder has its own class which assigns physics models and cross sections to processes. It is here where the overlap in energy ranges between two models is decided. For an energy E in the overlap region $E_1 < E < E_2$, one of the two models is chosen randomly; the probability to choose the model valid at higher energy is zero at E_1 and one at E_2, increasing linearly with energy. It is also here where models are built from sub-models. More complicated generators, like the FTF parton string model or nuclear de-excitation, are not implemented as a single model, but as a collection of them. This level of detail is not usually of interest to users, hence its encapsulation within the builder classes.

4.3.3. Reference physics lists

As of release 10.0 the toolkit provided nine reference physics lists whose names reflect the combination of models used to describe the hadronic interactions necessary for various applications. Unless otherwise indicated, the electromagnetic interactions in these physics lists are described by the GEANT4 standard EM models. Reference physics lists are extensively and routinely validated.

Perhaps the most-used reference physics list for high energy and space applications is FTFP_BERT. It uses the GEANT4 Bertini cascade for hadron–nucleus interactions from 0 to 5 GeV incident hadron energy, and the FTF parton string model for hadron–nucleus interactions from 4 GeV upwards. The letter P indicates that the GEANT4 precompound mode is used to de-excite the nucleus after the high energy FTF interaction has been completed. The FTFP-Bertini combination forms the backbone of many physics lists.

QGSP_BERT is a popular alternative which replaces the FTF model with the QGS model over the high energy range. Using the

Table 4
List of default and optional GEANT4 EM physics constructor classes. One of several optional EM physics constructors may be chosen by appending its shorthand name, listed in the "Extension" column, to the name of a basic physics list, such as FTFP_BERT_ENV, for example. WVI refers to the Wenzel multiple scattering model as implemented by V. Ivantchenko.

Physics Constructor Name	Application	Extension	Comment
`G4EmStandardPhysics`	HEP		Default (ATLAS)
`G4EmStandardPhysics_option1`		EMV	Simplified (CMS)
`G4EmStandardPhysics_option2`		EMX	Simplified (LHCb)
`G4EmStandardPhysics_option3`	Space & medicine	EMY	Detailed Standard models
`G4EmLivermorePhysics`		LIV	Detailed Livermore models
`G4EmPenelopePhysics`		PEN	Detailed Penelope models
`G4EmStandardPhysics_option4`		EMZ	Combining Best models
`G4EmLivermorePolarizedPhysics`			Polarized models
`G4EmLowEPPhysics`			New low energy models
`G4EmStandardPhysicsWVI`			WVI multiple scattering
`G4EmStandardPhysicsSS`			Single scattering
`G4EmDNAPhysics`	DNA		Default for DNA physics
`G4EmDNAPhysics_option1`	DNA		WVI multiple scattering
`G4OpticalPhysics`	All		Production and transport of optical photons

other intranuclear cascade models, `G4BinaryCascade` or `G4INCLXX` instead of Bertini, produces the FTFP_BIC, QGSP_INCLXX and QGSP_BIC physics lists, of which the latter is often used in medical applications. When high precision neutron propagation is included in a physics list, the letters *HP* are appended to its name, for example FTFP_BERT_HP. Many other physics lists can be built in this way, but only a handful of them are sufficiently tested and validated to qualify as reference lists.

There are also specialized physics lists such as QBBC [218] and Shielding, which are not currently considered to be reference lists, but are often used.

As mentioned in Sections 4.1 and 4.3.1, there are several collections of electromagnetic models besides the standard. Using these in a physics list is made easy by the `G4PhysicsListFactory`. A user need only specify the desired electromagnetic option, and the factory will substitute it for the standard collection in the newly created physics list. Examples of physics lists that may be created using G4PhysicsListFactory are:

- QGSP_BERT_EMV, the QGSP_BERT set of hadronic models with faster but less precise electromagnetic models,
- FTFP_BERT_LIV, the FTFP_BERT set of hadronic models with the Livermore electromagnetic models, or
- QGSP_BIC_DNA, the QGSP_BIC set of hadronic models with the low energy DNA electromagnetic models.

4.4. Results

A critical test of a physics list and the models and cross sections that comprise it is the comparison of its predictions to data from the calorimeters of high energy physics experiments. For this, GEANT4 has relied upon test beam data from the ATLAS [219], CALICE [220], and CMS [221] collaborations. The experimental parameters of interest include the longitudinal, transverse and time distributions of shower energy, the visible deposited energy and energy resolution of the shower, and the relative importance of electromagnetic and hadronic energy as measured by the e/π ratio.

The latter parameter was the first for which good agreement between test beam data and a GEANT4 physics list was obtained [1]. Since then, model improvement guided by thin target validation has resulted in good agreement in almost all the parameters mentioned above. Discussed here are recent comparisons of

predictions from the FTFP_BERT physics list to test beam measurements of the longitudinal and transverse shower shapes, and the shower energy resolution.

In order to perform some of these comparisons a GEANT4 geometry was developed which reproduced the essential details of the calorimeters used to take the data, while omitting other, less critical, yet time-consuming details. Within GEANT4 this is referred to as the simplified calorimeter [222]. Other comparisons were performed within the various experiment collaborations using their own GEANT4 geometry descriptions.

4.4.1. Electromagnetic showers

Significant efforts have been made to improve the simulation of electromagnetic shower shapes in order to describe the details of the $H \rightarrow \gamma\gamma$ signal [223,224] and other reactions. The bremsstrahlung process and the simulation of multiple scattering were reviewed and improved, having been identified as key components in defining shower shapes. Calorimeters are particularly sensitive to the simulation of electron and gamma transport in the MeV energy region. Therefore a large amount of validation and benchmarking was, and continues to be, carried out for medium and low energy electrons and gammas down to about 1 keV. For these validation studies data from numerous thin-target and calorimeter test-beam experiments are used as well as comparisons with other GEANT4 low energy electromagnetic models, such as the Livermore and Penelope sub-packages, which have been recently adapted to a common interface (see Section 4.1.1) with the standard electromagnetic sub-packages [225].

The process of multiple scattering (MSC) of charged particles is a key component of Monte Carlo transport codes. At high energy it defines the deviation of charged particles from ideal tracks, limiting the spatial resolution of detectors. The scattering of low energy electrons defines the energy flow via volume boundaries. This affects the sharing of energy between absorbers and sensitive elements, directly affecting shower shapes. A comprehensive description of recent improvements of the GEANT4 electromagnetic module can be found in [226]. Good agreement was found when GEANT4 predictions were compared with experimental data collected at the LHC [227].

4.4.2. Hadronic showers

To increase the quality of simulations of hadronic showers three main components are needed: a string model at high energies, a cascade model at intermediate energies (from few hundred MeV up to about 10 GeV) and pre-equilibrium and evaporation models at low energies (below a few hundred MeV). For these energy ranges the Fritiof, Bertini and G4Precompound models, respectively, are recommended.

Detector response is an effective test of any model combination. It is defined as the ratio of deposited energy visible to the detector, to the incident beam energy. For the above combination of models (as in the FTFP_BERT physics list), the general agreement between the simulated response and data for hadron-induced showers is at the level of a few percent. Other useful data, such as shower shapes and energy resolution are less precisely described and show agreement at a level of 10–20%.

Fig. 21 shows the comparison between the predictions of GEANT4 simulations with test beam data collected by the ATLAS Collaboration [228]. The response to pion beams is shown as a function of the particle energy for different versions of GEANT4, along with a comparison of the resolutions. Note that no contribution from electronic noise is simulated in this case.

A comparison of Monte Carlo calculations for the lateral (top) and longitudinal (bottom) dimensions of hadronic showers [229] are shown in Fig. 22 as a function of the beam energy for different versions of GEANT4. The detailed validation against experimental data requires the use of highly granular calorimeters such as the ones being designed by the CALICE collaboration. However, preliminary results suggest that GEANT4 hadronic showers are too compact and short. Comparisons with LHC test beam data have shown that a fundamental ingredient for improving the description of the lateral development of showers is the use of intermediate and low energy models that can describe the cascading of hadrons in nuclear matter and the subsequent de-excitation of the wounded nucleus. The longitudinal development of hadron showers mainly depends on the hadronic interactions at higher energies in the forward direction: quasi-elastic scattering and diffraction.

An important effect recently introduced in GEANT4 is the improvement of the neutron capture cross sections and final state generator. Based on the high precision neutron library, it allows for an improved simulation of the time structure and the lateral profile of hadronic showers in neutron-rich materials [230]. Other improvements include a retuned Fritiof model which will be made available in future GEANT4 versions.

5. Toolkit extensions

5.1. Biasing and reverse Monte Carlo

There is a class of applications for which rare events are of interest, and for which standard, or analog, simulation is inefficient, or even impractical. Estimating shielding efficiency in a radioprotection problem is one example: an analog simulation would spend by far most of its time transporting numerous particles and interaction products inside the shield, with only a tiny fraction of particles leaving the shield. In this case, these latter particles are the "rare events" (note that in this context "event" should not be understood as a G4Event). At the opposite extreme would be the simulation of very thin detectors, which is also difficult as only a small fraction of particles may interact inside. Another example is the simulation of single event upsets in an electronic chip within a satellite. Here the volume of interest is very small compared to the overall structure in which the simulation is to be performed. The events of interest (track steps in the
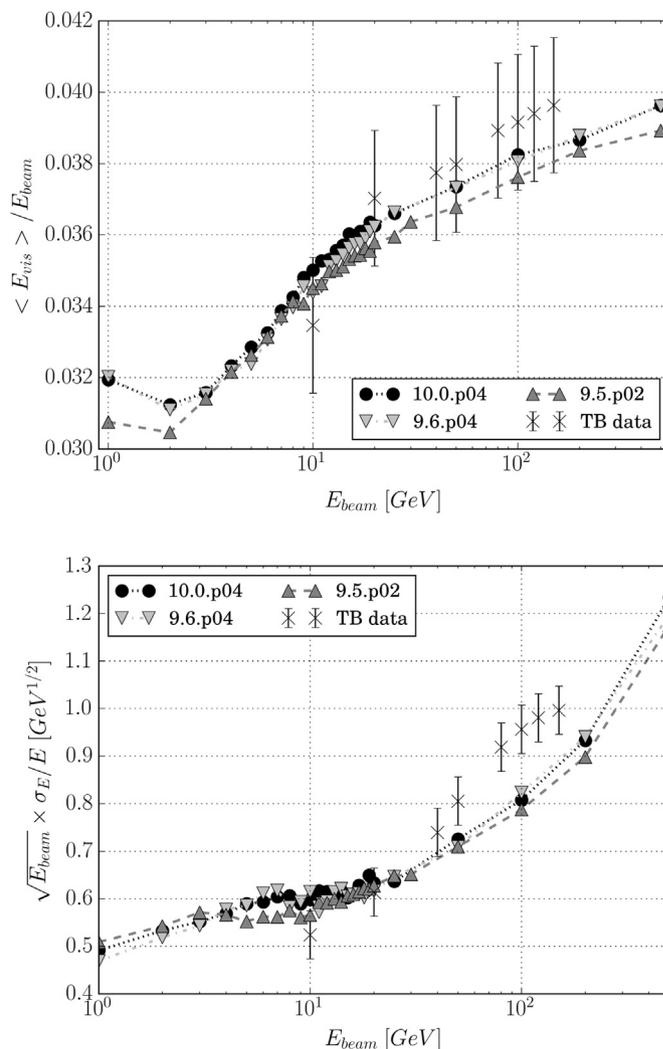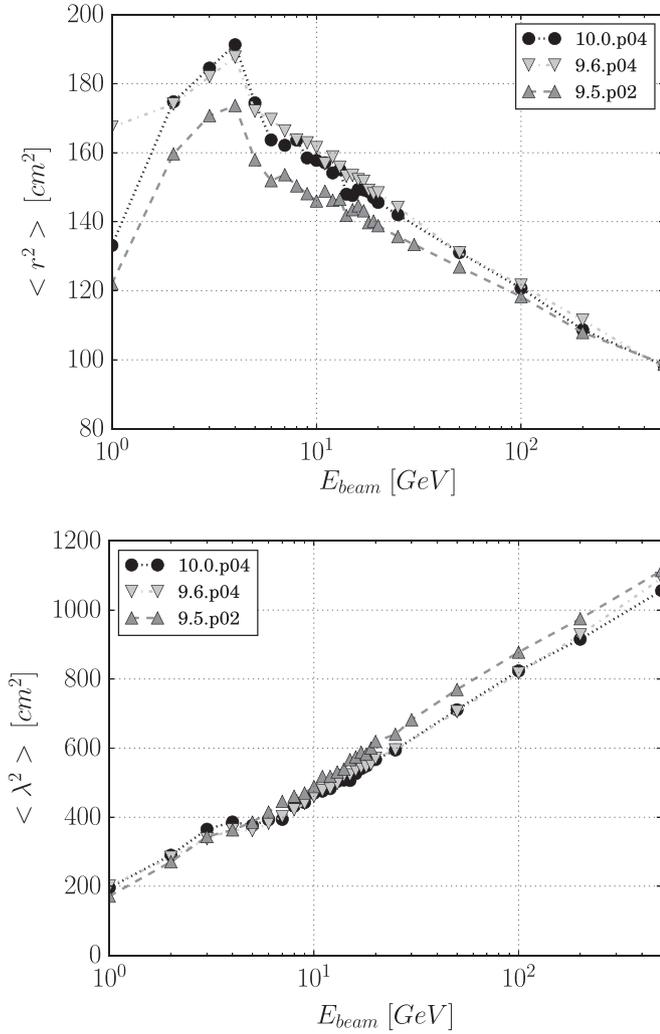

Fig. 21. Comparison of recent GEANT4 versions with test beam data for the response (top) and resolution (bottom) of the copper/liquid argon simplified calorimeter (ATLAS HEC).

small volume) which deposit energy in the chip are rare given that they occur in a fraction of time which is of the order of the ratio of the electronic chip volume to the overall satellite volume.

Event biasing techniques attempt to address these problems by magnifying the occurrence of the rare events. This change in the simulation is reflected in the statistical weight associated with each particle and in the ratio of probabilities in the biased and analog schemes to observe the particle in the current state. This approach allows the simulation efficiency to be boosted with respect to the rare events without sacrificing the physics quality provided by the full, detailed Monte Carlo description.

There is a large variety of biasing techniques, but they rely mostly on two main approaches: splitting and killing (SK) [231,233] and importance sampling (IS) [232,233].

In the first approach, tracks are split (killed) as long as they approach (recede from) the phase space region of interest. The shielding problem can be addressed as follows: to compensate for the natural absorption in the shield material and allow a sizeable number of particles to exit the shield, the flux is artificially regenerated by splitting particles into n identical copies at regular intervals for particles moving forward, that is, on their way toward exiting the shield. At each splitting, the particle weight is divided by n. If the particles move backward, they are randomly killed with probability $1/n$, and their weights are multiplied by n if they

Fig. 22. Comparison of recent GEANT4 versions using the simplified iron/scintillator calorimeter (ATLAS TileCal). Lateral shower shapes (top) and longitudinal shower shapes (bottom) are shown.

applied after the generation stage, at tracking time, weight computation must be handled by the simulation engine, as it will evolve during the particle transport.

The biasing functionalities of GEANT4 are presented here. At the time of the previous GEANT4 general paper [1] there were three main biasing techniques available:

- the Generalized Particle Source (GPS) [15], a versatile set of user commands to generate primary projectiles with various weights, spectra and particle types;
- geometry splitting and weight window [15], a method in which the user specifies weight bounds for each space-energy cell of the geometry, with tracks split or killed according to these bounds; with an extension of the scheme [12] to handle cells in parallel geometries, each of these being assignable to a given particle type;
- hadronic cross section biasing, which allows an overall scale factor to be applied to cross sections, with corresponding corrections to secondary particle weights, for a few relatively rare processes.

Since then the reverse Monte Carlo technique was introduced and work began to extend biasing functionalities using a generic approach. Both of these efforts are discussed below.

5.1.1. Reverse Monte Carlo

Reverse Monte Carlo is a biasing technique in which the primary particles produced by the event generator are tracked backwards in time, acquiring energy along the way, to determine their statistical weight. This technique is especially useful in the simulation of the dose received by a small electronic chip placed inside the large, complex structure of a satellite. The actual particle source consists of highly energetic electrons, and/or protons in the space environment, generally modeled in the simulation as a sphere that envelops the satellite. In this situation the fraction of simulation steps which contribute to the dose in the chip for particles starting from this source is of the order of the ratio of the chip volume to the satellite volume, and hence is very low.

In the reverse Monte Carlo approach particle tracking begins in the vicinity of the volume of interest, using an arbitrary distribution. In the first stage, primary particles are tracked backward in time, with time-reversed physics processes. Such processes and particles are referred to as "adjoint". At each interaction, an adjoint particle acquires energy and may also change its type: an adjoint e^- may, for example, become an adjoint γ if an adjoint photo-electric process takes place. Adjoint particles are tracked back to their extended source. At the end of this tracking the weight of the adjoint particle represents the statistical weight that the full reverse track (from the adjoint source to the external source) would have in a forward simulation. In other words it represents the probability belonging to this specific track from the external source to the adjoint source.

The arbitrary spectrum chosen in the vicinity of the volume of interest leads to a biased spectrum at the source. The statistical weight of a primary particle is then given by the ratio of the actual to the biased distribution values at the source. Note that this weight may be zero, leading to the particle being ignored, if the adjoint particle has an energy which is outside the energy bounds of the source. Once these weights are determined, the second stage begins in which particles are tracked with the usual, time-forward physics processes, beginning from the same space-time point where the adjoint transport started. The weights obtained during the reverse tracking stage allow a proper accounting of their contribution to the dose.

Design. The implementation of reverse Monte Carlo in GEANT4 [234] was designed to reduce as much as possible the

survive. This maintains a high unweighted flux, leading to workable statistics, while the weighted flux is low, as physically expected. In this SK approach, the physics processes are kept unchanged.

In importance sampling, the underlying physical laws are substituted with biased laws that favor the occurrence of rare events. The case of a thin detector volume can be treated as follows: the natural exponential law is replaced by a truncated one, limited to the track path inside the volume in order to guarantee that the collision will occur there. The weight acquired by the particle, or its interaction products, when such a forced collision occurs is the ratio of the two interaction probability law values at the interaction point. Thus the interaction law of a physics process is modified. The final state production law of the process may also be biased.

Biasing can be applied at various levels in a simulation application. Particle generator biasing is a simple case: the analog generator is biased to favor phase space regions or channels of interest. Such generator-level biasing does not necessarily require the weight to be propagated through the simulation stage. The production of rare particles in high energy physics or the generation of specific decay chains are examples of this case, with re-weighting of the rare event being applied at the analysis stage. Such generator-level biasing is of the IS type. When biasing is

modifications of user code required in order to use it. The G4AdjointSimulationManager class should be instantiated in main() and the physics list should be adapted in order to create adjoint particles (electrons, protons and gammas), and to declare their corresponding adjoint electromagnetic processes. An example of such a physics list is given in examples/extended/biasing/ReverseMC01.

During an event in reverse simulation, pairs of adjoint and forward equivalent particles (same type and energy but opposite direction) are generated randomly at the same position on a surface set by the user and containing the small sensitive volume considered in the simulation. This is called the adjoint source. The reverse tracking and computation of the weight of the adjoint particle is controlled by the reverse machinery and no user code is required for the control of the tracking phase. During the forward tracking phase of the equivalent normal particle the usual GEANT4 actions coded by the user are applied in order to compute the various tallies of the simulation.

The analysis part of the user code should be modified to normalize the signals computed during the forward tracking phase to the weight of the equivalent adjoint particle that reaches the external surface. This weight represents the statistical weight that the full reverse track (from the adjoint source to the external source) would have in a forward simulation. If a forward-computed signal is multiplied by the weight of the reverse track and is registered as a function of energy and/or direction it will give the response function of the signal. To normalize it to a given spectrum it has to be further multiplied by a directional differential flux corresponding to this spectrum. The weight, direction, position, kinetic energy, and type of the last adjoint particle that reaches the external source, and that would represent the primary of a forward simulation, can be obtained by public methods of the G4AdjointSimManager class. More details on how to adapt user code to use the reverse Monte Carlo scheme are provided in the Application Developer Guide [15].

Performance. The performance of the reverse Monte Carlo can be evaluated using the execution time and relative precision of the results. A useful figure of merit (*FOM*) is

$$FOM = \frac{1}{R^2 T},$$

(1)

where R is the relative precision reached for a given execution time T, in minutes. For a typical application [234], FOM is 4.9 for a forward method, compared to 7600 for a reverse method. This corresponds to a time speed-up factor of 1250. Such results will of course vary depending on set-up and physics, but it is clear that with this method, large speed-up factors can be achieved without sacrificing precision.

5.1.2. Generic biasing

In an attempt to unify the various forward-tracking biasing techniques, a new scheme was introduced in release 10.0. It aims to provide the user, through a restricted set of base classes, flexible means to handle virtually any type of biasing.

Design. The design relies on two main abstract classes. G4VBiasingOperation represents any type of biasing operation: splitting/killing or physics process modification (of the interaction law, or of the final state generation). The second class, G4VBiasingOperator, is the decision-making entity, and selects the biasing operations to be applied during the current step.

A third, concrete class is G4BiasingProcessInterface which derives from G4VProcess and provides the interface between the tracking and the biasing. At tracking time, G4BiasingProcessInterface checks for a possible G4VBiasingOperator associated with the current volume. If such an operator exists, G4BiasingProcessInterface requests the operator for

biasing operations to be applied. If no operation is returned, G4BiasingProcessInterface continues with standard tracking.

A G4BiasingProcessInterface object can wrap a physics process so that it controls it, and takes over the standard behavior in volumes where biasing is applied. At the beginning of the step the current G4VBiasingOperator may request the G4BiasingProcessInterface to apply an occurrence biasing operation to the physics process. The operator may also request G4BiasingProcessInterface to apply a final state biasing operation when PostStep process actions are invoked. If a G4BiasingProcessInterface object does not wrap a physics process, it is meant for applying splitting/killing biasing operations.

For the case of occurrence biasing, the problem is specified by providing a biased interaction law, represented by the abstract class G4VBiasingInteractionLaw. This is discussed in more detail below.

Occurrence biasing case. GEANT4 transports particles by allowing discrete interaction physics processes to compete in producing the shortest distance to the next interaction. Each distance is sampled according to an exponential law driven by the process "interaction length" (inverse of cross section); the competition makes the sampling equivalent to one that would be made using the total cross section. Occurrence biasing consists in substituting some other interaction law for the analog exponential one. The exponential transform [235] and forced collision are of this type of biasing.

The weight computation relies on the following formalism. For $N(\ell)$ particles present at distance ℓ, and in the asymptotic limit of $N(\ell) \to \infty$, the positive number $-dN$ of these interacting in the next segment $d\ell$ is related to the cross section $\sigma(\ell) \geq 0$ at this position by

$$\frac{-dN}{N(\ell)} = \sigma(\ell)d\ell.$$

(2)

The quantity $\sigma(\ell)d\ell$ is the one-particle probability to undergo an interaction in the segment $[\ell, \ell + d\ell]$. Note that $\sigma(\ell)$ is not assumed to result from physical cross sections per nucleus but rather that it is a simple function of ℓ, referred to here as an "effective cross section". Making $-dN$ proportional to $N(\ell)$ in (2) assumes that tracks are independent of each other. Also, only a dependence on ℓ, and on no previous coordinates, is considered in (2) in order for N and σ to describe the interaction probability in the next segment $d\ell$; this assumes the transport is of Markov nature. Our biasing scheme is based on these two fundamental assumptions.

For a neutral particle in a uniform medium, the physical cross section is independent of ℓ, and the effective cross section is simply $\sigma(\ell) \equiv \sigma_\phi^{\text{neutral}}$, where $\sigma_\phi^{\text{neutral}}$ is the macroscopic cross section in this medium. Its value changes after an elastic interaction, as the projectile energy changed. The case of a charged particle in a uniform medium is more delicate as the particle energy evolves during the transport because of energy loss, or by application of an electric field, for example. The effective cross-section evolves as $\sigma(\ell) \equiv \sigma_\phi^{\text{charged}}(E(\ell))$, where $\sigma_\phi^{\text{charged}}$ is the macroscopic cross section, the complication being that $E(\ell)$ is not a unique function but depends on each particle history.

In a biased scheme, such as the forced collision the effective cross section may evolve with ℓ even for the case of a particle of constant energy.

Integrating (2), leads to

$$N(\ell) = N(0) \cdot \exp\left(-\int_0^\ell \sigma(s)ds\right) = N(0) \cdot P_{NI}(0 \to \ell),$$

(3)

where $P_{NI}(0 \to \ell) \equiv \exp\left(-\int_0^\ell \sigma(s)ds\right)$ is the non-interaction

probability from the origin to distance ℓ, and is a monotonic decreasing function, as it must be for a non-interaction probability.

The Markov nature of Eq. (2) is reflected in $P_{NI}(0 \rightarrow \ell)$: if considering a particle making an initial step $0 \rightarrow \ell_1$, followed by a second step $\ell_1 \rightarrow \ell$, with no interaction, then $P_{NI}(0 \rightarrow \ell_1)P_{NI}(\ell_1 \rightarrow \ell) = P_{NI}(0 \rightarrow \ell)$, and the probability in ℓ does not depend on the previous steps made. This shows that a biased scheme can still follow a competitive approach between processes, whether biased and/or non-biased. The particle flight can be interrupted at any distance, and non-interaction probabilities in both biased and analog schemes will be well-defined.

If $p(\ell)$ is the probability density function of interactions, it is related to the non-interaction probability by:

$$P_{NI}(0 \rightarrow \ell) = 1 - \int_0^\ell p(s)\mathrm{d}s, \tag{4}$$

which conversely leads to

$$p(\ell) = -\frac{\mathrm{d}}{\mathrm{d}\ell}P_{NI}(0 \rightarrow \ell) = \sigma(\ell)P_{NI}(0 \rightarrow \ell). \tag{5}$$

This shows that the probability $p(\ell)\mathrm{d}\ell = P_{NI}((0 \rightarrow \ell)\cdot\sigma(\ell)\mathrm{d}\ell$ that a particle interacts within the segment $\mathrm{d}\ell$ at distance ℓ is the product of the probability that it travels ℓ without interaction, $P_{NI}(0 \rightarrow \ell)$, and the probability $\sigma(\ell)\mathrm{d}\ell$, that it then interacts in the segment $\mathrm{d}\ell$ [236].

$\sigma(\ell)$ can also be expressed using $p(\ell)$ and $P_{NI}(0 \rightarrow \ell)$ using (5)

$$\sigma(\ell) = -\frac{\mathrm{d}}{\mathrm{d}\ell} \log P_{NI}(0 \rightarrow \ell), \tag{6}$$

which is positive, as it must be, as $P_{NI}(0 \rightarrow \ell)$ is decreasing. Providing any of the three functions $\sigma(\ell)$, $p(\ell)$ or $P_{NI}(0 \rightarrow \ell)$ is enough to define the problem.

If several biased independent processes (i), $i = 1 \ldots n$, with effective cross sections $\sigma^{(i)}(\ell)$, contribute to particle interactions, each one is responsible for an interaction amount $-\mathrm{d}N^{(i)}$ in a segment $\mathrm{d}\ell$. These numbers add up, and Eq. (2) becomes

$$\frac{-\sum_i \mathrm{d}N^{(i)}}{N(\ell)} = \sum_i \sigma^{(i)}(\ell)\cdot\mathrm{d}\ell. \tag{7}$$

Eq. (5) keeps the same form, with $\sigma(\ell)$ and $P_{NI}(0 \rightarrow \ell)$ becoming the total effective cross section and non-interaction probability, given by

$$\sigma(\ell) = \sum_i \sigma^{(i)}(\ell), \tag{8}$$

$$P_{NI}(0 \rightarrow \ell) = \prod_i P_{NI}^{(i)}(0 \rightarrow \ell). \tag{9}$$

A scheme has been implemented in which discrete interaction physics processes can be biased independently of each other, each possibly having its analog interaction law replaced by a biased one. The related analog and biased quantities will be denoted by subscripts a, b, or by superscripts (a), (b), respectively.

In a step, any process which looses the competition for the distance to the next interaction ends up with a non-interaction. The biased and analog versions of the same process (i) have different probabilities for this to occur, which is reflected by multiplying the track weight by a non-interaction weight for each process, and which is

$$w_{NI}^{(i)}(0 \rightarrow \ell) \equiv \frac{P_{NI; a}^{(i)}(0 \rightarrow \ell)}{P_{NI; b}^{(i)}(0 \rightarrow \ell)}. \tag{10}$$

For the process which wins the race, the total weight to be applied is

$$w_{\text{total}}^{(i)}(\ell) = \frac{p_a^{(i)}(\ell)\mathrm{d}\ell}{p_b^{(i)}(\ell)\mathrm{d}\ell},$$

$$= w_{NI}^{(i)}(0 \rightarrow \ell)\cdot w_I^{(i)}(\ell), \tag{11}$$

where $w_I^{(i)}(\ell)$ is called the interaction weight, given by

$$w_I^{(i)}(\ell) \equiv \frac{\sigma_a^{(i)}(\ell)}{\sigma_b^{(i)}(\ell)}. \tag{12}$$

Even for this process, it is seen that the non-interaction weight is involved.

In summary, for a track taking a step of length ℓ, each process (i) multiplies the track weight by its non-interaction weight $w_{NI}^{(i)}(\ell)$ [237]; in addition, the process winning the competition for the distance to the next interaction further multiplies the track weight by its interaction weight $w_I^{(i)}(\ell)$.

The design of occurrence biasing relies on the `G4VBiasingInteractionLaw` class which defines an abstract interface to model interaction laws. The $P_{NI}^{(i)}(0 \rightarrow \ell)$ and $\sigma^{(i)}(\ell)$ calculations are to be provided by overriding, respectively, the pure virtual methods

- `G4double ComputeNonInteraction ProbabilityAt(G4double l)` and
- `G4double ComputeEffective CrossSectionAt(G4double l)`,

where `l` is a re-notation of the length ℓ of the step taken.

For a physics process (i) under the control of a `G4BiasingProcessInterface` instance (I), (I) collects the process cross section at the beginning of the step and asks the current biasing operator for a potential occurrence biasing operation. If such an operation is provided, it comes with a `G4VBiasingInteractionLaw` that the operator samples to cause (I) to compete for the next interaction. The process cross section is used to build a version of `G4VBiasingInteractionLaw` implementing a classical exponential law. In the subsequent `AlongStep` stage, all instances (I) provide the non-interaction weight $w_{NI}^{(i)}(0 \rightarrow \ell)$ (Eq. (10)) of their physics process (i) using the method `ComputeNonInteractionProbabilityAt(l)` of the biased and analog laws. In the following `PostStep` stage, if an instance (I) won the next interaction race, it further applies the weight for interaction $w_I^{(i)}(\ell)$ (Eq. (12)) of process (i) using the `ComputeEffectiveCross[1]SectionAt(l)` method of the two laws. This interaction weight multiplies the weight of all the final state tracks (primary or secondaries) issued from the interaction. This final state may be the one of the analog versions of the physics process, or a final-state-biased version of it.

In release 10.0, a few concrete occurrence biasing functionalities were provided. The class `G4BOptnChangeCrossSection` is a biasing operation to change a process cross section. Given this change can be done per process and on a step-by-step basis, it can be used to implement schemes similar to the exponential transform [235].

The class `G4BOptrForceCollision` is a biasing operator that reproduces the forced collision scheme of MCNP [236]. It handles several biasing operations to achieve this: a splitting operation to make a copy of the track entering the volume, an operation to force the primary track to cross the volume without interaction (and no tally) and update its weight accordingly, and an operation to force the collision on the primary clone, using the total interaction cross section collected from the physics processes.

These two schemes have been validated on neutral tracks.

5.2. Error propagation

The track error propagation package serves to propagate one particle together with its error from a given trajectory state (i.e. position and momentum with errors) until a user-defined target is reached (a surface, volume or given track length). Its main use is for the track fitting of simulated or real data to reconstruct the trajectory of a particle that has left several detector signals along its path.

To do its work, this package uses Geant4 geometry and physics, so that the same geometry and magnetic field used in the simulation can be used for the track error propagation. The Geant4 physics equations can also be used, but it is the average trajectory that should be propagated and this must be taken into account. Although the user may use his/her own physics list, a physics list provided in the package is recommended. It has no straggling due to multiple scattering, no fluctuations in the energy loss, no emission of secondary particles and no hadronic processes. This physics list also accommodates backward propagation as well as forward (the direction the track followed to produce the detector signals).

When a track is propagated forward, it loses energy and the energy at the beginning of the step is used to calculate the energy loss. When a track is propagated backward, it gains energy and the energy at the end of the step is used. Thus, depending on propagation direction, quite different values of the energy loss might be obtained. To avoid this, the algorithm uses in both cases the average energy to compute the energy loss.

Propagation is terminated when one of the following targets are met:

- *Surface*: a user-defined surface is reached. The surface does not have to be part of the geometry, as the propagation takes into account both the geometry and the user-defined surfaces at the same time. Currently plane and cylindrical surfaces may be defined, as well as surfaces formed from the combination of the two.
- *Volume*: the surface of one of the logical volumes of the geometry is reached. The user may choose if the track is to stop only when entering the volume, exiting the volume, or in both cases.
- *Track length*: a given track length is reached.

This package was implemented following GEANE [238] from GEANT3, which was based on the equations developed by the EMC collaboration [239].

Users may implement the example provided with the Geant4 distribution, at `examples/extended/errorpropagation`. The geometry in this example simulates a simplified typical high energy physics detector consisting of

- an air-filled beamline,
- an air-filled central detector,
- a copper calorimeter, divided into four sections,
- an aluminium calorimeter, divided into ten sections, and
- an air-filled muon detector.

While the example does not pretend to have a fully realistic geometry, it is sufficient to test the code. Also the volumes were chosen to match those of the example in the GEANE paper so that a detailed comparison of results could be done.

The detector is immersed in a magnetic field with a default value of 1 kG pointing along the negative *z*-axis. This value can be changed by user command. An initially free trajectory state is created to simulate a muon track of 20 GeV along the *x*-axis. This track is propagated until one of the termination targets mentioned

above is reached. An environment variable serves to select the type of target. Another environment variable allows either forward or backward propagation to be selected. The user also has the freedom to choose whether control will be returned to the user after each step, or only after the propagation target has been reached.

5.3. Analysis

5.3.1. Introduction

Analysis tools based on AIDA (Abstract Interfaces for Data Analysis) [240] have been used in Geant4 examples since release 3.0 in December 2000, but until 2010 no analysis code was provided in the Geant4 source. Several AIDA-compliant tools are available, including JAS, iAIDA, Open Scientist Lab and rAIDA [15]. However some of them have not been maintained, some do not implement the AIDA interfaces completely and some are not always easy to install and use.

A new analysis package based on g4tools [241] was added in Geant4 release 9.5 with the aim of providing users with a "lightweight" analysis tool available as part of the Geant4 installation without the need to link to an external analysis package. It consists of the analysis manager classes and also includes the g4tools package.

g4tools provides code to write histograms and ntuples in several formats: ROOT [242], XML 2040 AIDA [240] and CSV (comma-separated values) for ntuples and HBOOK [243]. It is a part of the highly portable inlib and exlib [241] libraries, which also include other facilities like fitting and plotting. These libraries are used to build the ioda application, available on all interactive platforms so far, including iOS and Android, and able to read the file formats written with g4tools.

The analysis classes provide a uniform, user-friendly interface to g4tools and hide from the user the differences between various output technologies. They take care of higher level management of the g4tools objects (files, histograms and ntuples), handle allocation and removal of the objects in memory and provide the methods to access them via indexes. For simplicity of use, all user interface analysis functions are provided within a single class which is seen by the user as `G4AnalysisManager`. Internally, this type is defined using a `typedef` and it can point to one of four type-specific output manager classes: `G4TypeAnalysisManager` where *Type* can be *Csv*, *Root*, *Xml* or *Hbook*.

5.3.2. Histogramming

At present, one-dimensional and two-dimensional histograms are supported. The properties of histograms already created can be changed with the use of dedicated `Set()` functions including a limited set of parameters for histogram plotting.

`G4AnalysisManager` also provides extensions to g4tools suitable for Geant4 applications. Users can choose units and functions which are then automatically applied to filled values, a binning scheme (linear or logarithmic) and an ASCII option which activates the printing of histograms to an ASCII file. The activation option allows the user to activate only selected histograms. When this option is selected, only the histograms marked as activated are returned, filled or saved in a file.

Histograms may also be created and modified interactively or in a macro using the rich set of commands defined in the `G4AnalysisMessenger` class.

5.3.3. Ntuples

Ntuples with *int*, *float* and *double* column types are supported. Depending on the selected output format, more files can be generated when more than one ntuple is defined in a user application. This is the case for XML and CSV, which do not allow writing more

than one ntuple to a file. The ntuple file name is then generated automatically from the base file name and the ntuple name.

5.3.4. Multithreading

Like all other GEANT4 components, the analysis code was adapted for multithreading. In multithreading mode, instances of the analysis manager are internally created on the master and worker threads and data accounting is processed in parallel on worker threads. The migration to multithreading requires no changes in the user's client analysis code. HBOOK output is not supported in multithreading mode.

Histograms produced on worker threads are automatically merged on the call to `Write()` and the result is written to a master file. Ntuples produced on worker threads are written to separate files, the names of which are generated automatically from a base file name, a thread identifier and eventually also an ntuple name. No merging of ntuples is performed in order to avoid an associated time penalty.

5.4. Basic examples

The GEANT4 toolkit includes several fully coded examples which demonstrate the implementation of the user classes required to build a customized simulation. The previous "novice" set of examples, oriented to beginning users, was refactored into "basic" and "extended" example sets in GEANT4 10.0.

The new "basic" examples cover the most typical use-cases of a GEANT4 application while maintaining simplicity and ease of use. They are provided as starting points for new application developers. There are currently five such examples, some of which include several options or sub-examples. The features demonstrated in each example will be presented in the following subsections.

All basic examples have been migrated to multithreading (MT) and no special steps are required to build them in this mode. They will automatically run as MT when they are built on the GEANT4 libraries built with MT mode activated; otherwise they will run in sequential mode. MT mode may be chosen by creating `G4MTRunManager` instead of `G4RunManager` in the `main()` of the example.

Basic examples can be run in interactive mode with visualization, or in batch mode. The most suitable visualization parameters, such as a viewing angle, are selected for each example in order to promote a better understanding of the example scenario and to demonstrate the most useful features of the visualization system. A comprehensive list of visualization features can be found in Chapter 7 of the Application Developer Guide [15].

5.4.1. Example B1

`examples/basic/B1` demonstrates a simple application in which user actions control the accounting of the energy deposit in volumes. The dose in a selected volume is then calculated. The geometry setup is defined with the use of simple placements (`G4PVPlacement`) and the physics is defined with the use of the QBBC pre-packaged physics list. Scoring is implemented directly in the user action classes and an associated "run" object (`B1Run`) is created.

This example also demonstrates several features of the visualization system not shown in other examples, namely the type and appearance of trajectories, the ability to add axes, text, date and time, event number, the GEANT4 logo and the ability to set the visibility of individual volumes.

5.4.2. Example B2

`examples/basic/B2` simulates a simplified fixed target experiment. Two geometry setup options are provided: one using simple placements (`G4PVPlacement`) and one using

parameterized volumes (`G4PVParameterisation`). In addition a global, uniform, transverse magnetic field can be applied using `G4GlobalMagFieldMessenger`. The physics setup is defined using the FTFP_BERT pre-packaged physics list with a step limiter. Scoring is implemented with sensitive detectors and hits.

5.4.3. Example B3

`examples/basic/B3` simulates a simplified positron emission tomography system. The geometry is defined with simple placements and rotations. A modular physics list is used. Primary particles are ^{18}F ions randomly distributed within a volume inside a simulated patient, with an associated radioactive decay process. Scoring is implemented with GEANT4 primitive scorers.

5.4.4. Example B4

`examples/basic/B4` simulates a sampling calorimeter. The geometry is defined using simple placements and replicas (`G4PVReplica`). A global, uniform, transverse magnetic field can be applied using `G4GlobalMagFieldMessenger`. Physics is defined using the FTFP_BERT reference physics list.

Energy deposits and track lengths of the charged particles are recorded event by event in the absorber and gap layers of the calorimeter. This example demonstrates four different scoring methods: user actions, user-developed objects, sensitive detectors and hits, and primitive scorers.

Also demonstrated is the use of GEANT4 analysis tools for accumulating statistics and computing the dispersion of the energy deposit and track lengths of the charged particles. The resulting one-dimensional histograms and an ntuple are saved in an output file, which has ROOT format by default, but can be changed at compilation time to other formats supported by g4tools (AIDA XML, CSV for ntuples, or HBOOK).

5.4.5. Example B5

`examples/basic/B5` simulates a double-arm spectrometer with wire chambers, hodoscopes and calorimeters. The geometry setup, less trivial than in previous examples, is defined using placements, rotations, replicas and a parameterization. In addition a global, uniform, transverse magnetic field can be applied using `G4GlobalMagFieldMessenger`. The physics setup is defined with the FTFP_BERT reference physics list with a step limiter. Scoring within wire chambers, hodoscopes and calorimeters is implemented with sensitive detectors and hits.

`G4GenericMessenger` is used to define user interface commands specific to the user application and GEANT4 analysis tools are used to output scoring quantities in one-dimensional and two-dimensional histograms and an ntuple.

6. Validation

6.1. Release tools

6.1.1. Release process

The software release process is described in the GEANT4 policy document [244]. It has reached a rather mature state in recent years, with few modifications or adaptations made to the tools adopted for development and testing. One adaptation was required following the migration of the source code repository to the subversion (SVN) [245] tool. This involved mainly technical updates to the tools and scripts used to package the software, consistent with most of the principles described in the original release policy.

A new release of the GEANT4 software is usually issued once per year, with a Beta preview of some of the expected new features usually towards the end of the second quarter of the same year.

6.1.2. Software organization

Geant4 software is structured in a directory tree with each category corresponding to a high-level directory. Categories such as geometry, which is comprised of geometry shapes and navigation, and processes, which includes physics interaction models, are recursively subdivided into more and more specialized topics. In general the C++ classes for each topic, such as a model describing a specific type of interaction, are maintained or developed by one or a few developers.

Collaborative software revision management is handled by the SVN tool. During the development process, developers are encouraged to commit frequently changes to the trunk of the SVN repository. When a change within a topic or directory is complete, the developer creates a tag of this code; features of the tag are documented changes in a history file, present in each sub(sub…) directory. A tag may be made of low-, intermediate- or high-level directories. Tags are recorded in the database using SVN commit hooks on the SVN server. The developer can propose such tags for inclusion into future releases using a web interface to the tags database. Proposed tags will then enter the testing cycle.

6.1.3. Testing tools

Within the release process, integration of new or modified code into reference tags is performed. This requires regular nightly testing of new or modified code against the code base on a number of different platforms. A platform is a specific combination of hardware and operating system version, a specific version of compiler, and a set of Geant4-specific build options. This regular testing was migrated from a custom setup to using the CMake tool suite [246].

There is a continuous testing cycle and nightly test runs. Testing on a specific platform is performed using the CTest tool of the CMake tool suite. For each platform, the source code corresponding to the set of tags to be tested is checked out using SVN, CMake is run to configure and compile the code, and CTest runs the tests. More than 80 unit test codes are applied, each testing a specific part of Geant4. In addition most examples and several benchmarks are exercised during testing, bringing the total number of tests run to more than 200. The exact number varies with platform, as optional software required by some tests may not be available on every platform. CMake takes this into account by checking the availability of optional software during configuration, and skipping tests where software is missing. Finally, a summary of test results is uploaded to the CDash web site.

In addition to nightly testing, continuous testing is performed on a small number of platforms using a restricted set of tests. This regularly checks for changes in the status of tags, including also tags proposed for testing, and starts a test run when needed. This restricted testing allows for an early, but limited check of new software giving fast feedback to developers.

6.1.4. Grid tools

The simplified calorimeter is a Geant4 application which uses simplified versions of LHC calorimeters and produces simulation results for beam energies between 1 and 500 GeV. These results are then compared to normalized real data so that the effects of changes between different Geant4 versions or physics models on physics performance may be investigated.

Acquiring accurate simulation results requires running the simplified calorimeter for thousands of events. This, in conjunction with the high energies, makes running the application on developer PCs or even local computing clusters impossible due to time restrictions. This need for computing power was met by the use of resources provided by the worldwide LHC computing grid [247]. Simplified calorimeter jobs are split into smaller fragments and sent for execution in various grid sites around the world.

Results are then collected and pushed to the validation database. The end user, typically a developer, is able to run on-demand analysis tasks through an interactive web application and produce plots. The current grid tools implementation is based on the DIRAC workload management system which is used to distribute the jobs. A system was developed to support different Geant4 tests, examples and applications that are or will be executed on the grid. Geant4 libraries and executables are made available to the sites through the Geant4 CernVM-FS [248] repository. Finally ROOT is used for the results format, merging of split jobs and on-demand analysis.

6.1.5. Computing performance benchmarking and monitoring

Performance evaluation and analysis are essential for monitoring the Geant4 toolkit through its development cycle for expected and unexpected changes in computing performance, and for identifying problems and opportunities for code improvement and optimization. All internal development monthly releases and public releases are profiled with a set of applications that utilize different input event samples, physics lists, and detector configurations. Results from multiple runs benchmarking CPU performance and memory use are compared to those from previous public releases and development reference releases, and posted on a publicly available web-site [249].

In addition to memory footprints and a full summary of call stacks, which includes exclusive and inclusive function path counters, a detailed call graph analysis is made available to Geant4 developers for further analysis. The set of software tools used in the performance evaluation procedure, both in sequential and multithreaded modes, includes FAST [250], IgProf [251] and Open|Speedshop [252]. The scalability of CPU time and memory performance in a multithreaded application is evaluated by measuring event throughput and memory gain as a function of the number of threads for selected event samples.

6.1.6. Quality assurance

Each software release of Geant4 is validated against memory management issues, in particular run-time memory errors (overwrites, memory corruption, etc.) and memory leaks, through use of the Valgrind [253] tool. Verification for run-time memory errors is completely automated as part of the system testing nightly builds, so that results for each test included in the testing suite can be retrieved directly from the testing dashboard and monitored along with the normal development process. Specific sessions of Valgrind runs are executed periodically on a selected set of tests covering a rather wide range of physics configurations to verify the absence of memory leaks during the execution of a simulation run for multiple simulated events; a summary of the results is distributed to developers by the release coordinator, together with hints on where problems may originate. Memory leak checks are usually performed on candidate releases as part of the software release process.

The Valgrind tool is also used for detecting errors in multithreaded simulation applications, by means of its DRD [254] module. DRD allows the identification of potential data-races happening during run-time among threads, when one or more threads try to access the same memory location without proper locking or protection. It also allows easy identification and debugging of lock contention cases, when a test program may get stuck in execution.

Static code analysis is regularly applied at every development release, by instrumenting the entire source code with the Coverity [255] tool. The list of defects identified by the tool are first analysed and then assigned to the responsible developers for further inspection and correction. Since 2010, when this tool was first adopted for quality assurance, almost 3000 defects (most of them

not critical) have been triaged and fixed, with priority given to cases of potentially higher impact.

6.1.7. Distribution

Releases and beta previews, including all data sets, are available from the GEANT4 download page [256] as compressed source code files and in compiled form for Linux, MacOS and Windows. The source code and libraries for Linux are also available from AFS in `/afs/cern.ch/sw/lcg/external/geant4` and in CernVM-FS in `/cvmfs/geant4.cern.ch/geant4`.

6.2. Physics validation tools

6.2.1. Physics validation procedure

The accuracy of the GEANT4 physics models is benchmarked regularly using thin- and thick-target tests. Thin-target tests allow a detailed study of specific observables from a single physics process for a given configuration of projectile particle type, projectile kinetic energy and target material. A large set of published thin-target data, collected over several years, is used roughly once per month to validate each internal development release. These data can also be used for tuning some of the parameters of the physics models. This is particularly true for the hadronic models, many of which are phenomenological.

Thick-target tests are mainly based on test beam setups. These tests allow the assessment of the physics accuracy of GEANT4 simulations in realistic configurations, which involve several physics processes and multiple energy scales. A notable example is the measurement of the properties of electromagnetic and hadronic showers in calorimeter test beam setups which were carried out for the preparation of the LHC experiments, and which are on-going for the Linear Collider detector (CALICE).

Test beam simulations are in general complex and time-consuming, and are therefore repeated by experimentalists only for some GEANT4 public releases. To get more frequent and regular feedback, GEANT4 developers have created a set of simplified test beam setups which are used for regression testing, that is, the comparison of two or more versions of GEANT4 for any observable. When a statistically significant difference is detected, for instance if the hadronic shower becomes wider than that observed in a previous GEANT4 version, the code changes responsible for the difference are investigated.

6.2.2. Physics validation tests

Model development and improvement work is tightly coupled with extensive efforts to validate GEANT4 physics with experimental data. A large collection of tests ranges from validation at the single process level and comparison with thin target experimental data to the validation of complete physics lists and comparison with results from LHC experiments.

In particular, validation with thin target data is crucial in judging the quality of model prediction and model tuning. Thin target validations are done for

- stopping particles ($\bar{p}$, π^-, K^-, Σ^-, Ω),
- low energy data (< 100 MeV) with inclusive n, p production in n, p, γ beams on nuclear targets,
- medium energy data (100 MeV–200 GeV) that includes n, p, π^+ production in p–A or $\pi^\pm$–A interactions,
- high energy data (> 20 GeV) for inclusive hadron production in π or p interactions with nuclear targets, and
- a number of comparisons for simplified yet realistic experimental setups by the GEANT4 team and LHC experimentalists.

6.2.3. Physics validation repository

As the number of regularly performed validation tests increases and the collection of results grows, storing them and making them available to the user community becomes a challenge of its own. It was decided to organize this material in a central repository and to make this data generally and easily available, not only for internal collaboration use, but also for the user community.

The Physics Validation Repository stores data in the form of images with meta-data, or as the raw data points from both simulation and experiment. Meta-data includes descriptions of the tests, lists of references describing the origin of the experimental data, and other parameters which describe the test, such as beam particle type, its energy or momentum, the reaction observed and its secondaries, and the observable that is extracted from the measurement.

The ability to store images allowed the initial population of the database with the available collection of existing test results. The alternative method of storing raw data points allows more inter-activity, and can be used for example, in performing regression testing and model comparisons.

As shown in Fig. 23, the physics validation repository consists of a PostgresSQL database, a Java API and a JSP web application. The PostgresSQL [257] relational database stores collections of tests, such as images, tags, descriptions, references, and images with meta-data or raw data points, both experimental and simulated.

The Java API is based on the data access object (DAO) design pattern, which provides an abstract interface to the database. Using mapping application calls to the persistence layer enables the DAO to provide some specific data operations without exposing details of the database.

The web application [258] is based on the Enterprise Edition (Java EE) [259] of the Java Platform, deployed on a GlassFish Application server [260]. This allows tests to be viewed and modified, new tests to be uploaded, and provides security and authentication to grant access to functions and data internal to the GEANT4 collaboration.

The PrimeFaces JSF (java server faces) Framework [261] is used to create interactive, modern-looking web interfaces and charting tools based on JavaScript are used for plotting.

7. Outlook for the next decade

7.1. A brief summary of GEANT4 progress

Major changes and developments in the GEANT4 toolkit took place between the 8.1 and 10.1 releases. These include:

- the migration to multithreading,
- the addition of tessellated solids and a unified geometry description,
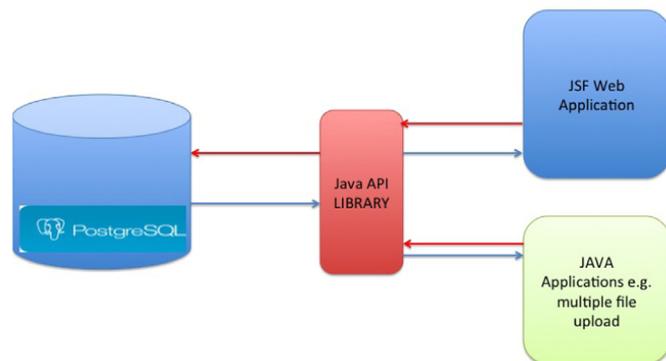

Fig. 23. Components of the GEANT4 validation repository.

- general biasing methods,
- improved and expanded physics models,
- expanded validation and testing,
- reference physics lists,
- the addition of rudimentry analysis methods, and
- improved and expanded visualization tools.

As a result the toolkit is more versatile, easier to use and makes more efficient use of available CPU and memory resources.

7.2. New directions

These changes were made in response to the demands of a diversifying user community and to the opportunities made available by advancing technology. It is expected that both these trends will continue and that GEANT4 will continue to evolve with them.

With this in mind the GEANT4 collaboration is studying new options for the future. GPUs and accelerated processors offer great potential for speeding up computationally intensive applications, and could possibly be adapted for use in physics simulations. Massively parallel computing and vectorization are also being examined as a way to exploit available supercomputer capacity [262,263].

The drive to make GEANT4 easier to use will continue. An increasing percentage of the GEANT4 user base requires more turn-key operation and better documentation. To enable this, improved user interfaces, simplified physics choices and still more powerful build tools will be required.

The expansion of GEANT4 into new physics domains will also continue. Users in nuclear physics require more detailed nuclear reactions and models, space and medical applications depend increasingly on precise, fast electromagnetic and radioactive decay modeling, biophysics and material science continue to expand the simulation of chemical kinetics and damage to micro-structures, and photon science is expected to be a user of radiation damage and accelerator dark current simulations. While new capabilities are currently being developed to meet the needs of experiments at the high energy, intensity and cosmic frontiers, it is clear that the increasing use of GEANT4 in other areas will also lead to new toolkit developments.

Acknowledgments

References

[1] J. Allison, et al., IEEE Trans. Nucl. Sci. 53 (2006) 270.
[2] X. Dong, G. Cooperman, J. Apostolakis, Multithreaded Geant4: semi-automatic transformation into scalable thread-parallel software, in: Euro-Par 2010 – Parallel Processing, Lecture Notes in Computer Science, vol. 6272, 2010, pp. 287–303.
[3] S. Ahn, et al., Geant4-MT: bringing multi-threading into Geant4 production, in: Joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2013 (SNA + MC 2013), Paris, 2014.
[4] Geant4 Collaboration, Toolkit Developers Guide, Available online at ⟨http://cern.ch/geant4⟩ (Chapter 2).
[5] A. Dotti, Description of hadron-induced showers in calorimeters using the Geant4 simulation toolkit, in: 2011 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2011, pp. 2128–2134.
[6] R. Chytracek, J. McCormick, W. Pokorski, G. Santin, IEEE Trans. Nucl. Sci. 53 (2006) 2892.
[7] CMS Collaboration, The CMS experiment at the CERN LHC, J. Instrum. 3 (S08004) (2008) 187.
[8] T. Leng, R. Ali, J. Hsieh, C. Stanton, A study of hyper-threading in high-performance computing clusters, Dell 6 (2002) 33.
[9] Intel Threading Building Blocks ⟨https://www.threadingbuildingblocks.org⟩.
[10] The Message Passing Interface Standard ⟨http://www.mcs.anl.gov/research/projects/mpi/⟩.
[11] G. Cosmo, The GEANT4 geometry modeler, IEEE Nucl. Sci. Symp. Conf. Rec. 4 (2004) 2196.
[12] J. Apostolakis, M. Asai, G. Cosmo, A. Howard, V. Ivanchenko, M. Verderi, Parallel geometries in GEANT4: foundation and recent enhancements, in: Proceedings of the IEEE NSS/MIC/RTSD Conference, 2008.
[13] P. Arce, J. Apostolakis, G. Cosmo, A technique for optimised navigation in regular geometries, IEEE Nucl. Sci. Symp. Conf. Rec. NSS08 (2008) 857.
[14] M. Gayer, J. Apostolakis, G. Cosmo, A. Gheata, J.M. Guyader, T. Nikitina, New Software library of geometrical primitives for modeling of solids used in Monte Carlo detector simulations, J. Phys.: Conf. Ser. 396(5) (2012) 052035.
[15] Geant4 Collaboration, Application Developers Guide, available online at ⟨http://geant4.web.cern.ch⟩.
[16] R.P. Brent, Algorithms for Minimisation without Derivatives, Prentice-Hall, Englewood Cliffs, New Jersey, 1973 (Chapter 4, ISBN 0-13-022335-2).
[17] Apache Xerces C Parser ⟨http://xerces.apache.org/xerces-c/⟩.
[18] STEP Tools, Inc., Open Source Code for 3-D Object Representation in Computer-Aided Design (CAD) ⟨http://www.steptools.com⟩.
[19] J. Allison, et al., Comput. Phys. Commun. 178 (5) (2008) 331.
[20] The Qt Project, An Open Source Cross-Platform Application and UI Framework ⟨http://www.qt.io⟩.
[21] OpenGL, the Industry's Foundation for High Performance Graphics ⟨https://www.opengl.org/⟩.
[22] OpenInventor, An Object-Oriented 3D Toolkit Offering a Comprehensive Solution to Interactive Graphics Programming Problems ⟨http://oss.sgi.com/projects/inventor/⟩.
[23] HepRApp Users Home Page ⟨http://www.slac.stanford.edu/~perl/HepRApp/⟩.
[24] S. Tanaka, M. Kawaguti, DAWN for GEANT4 Visualisation, in: Proceedings of the CHEP97 Conference, Berlin, April 1997.
[25] VRML Virtual Reality Modeling Language ⟨https://www.w3.org/MarkUp/VRML/⟩.
[26] A. Kimura, S. Tanaka, K. Hasegawa, Sasaki, Visualization for volume data scored by Geant4 simulation, IEEE Nucl. Sci. Symp. Conf. Rec. (2009) 2158–2161; see also ⟨http://geant4.kek.jp/gMocren/⟩.
[27] GL2PS ⟨http://guez.org/gl2ps/⟩.
[28] Coin3D, A High-Level, Retained-Mode Toolkit for Effective 3D Graphics Development ⟨https://bitbucket.org/Coin3D/coin/wiki/Home⟩.
[29] Anonymous, Distance from a Point to a Polyline ⟨http://programmizm.sourceforge.net/blog/2012/distance-from-a-point-to-a-polyline⟩.
[30] S. Agostinelli, et al., Geant4 a simulation toolkit, Nucl. Instrum. Methods Phys. Res. A 506 (2003) 250.
[31] V. Ivanchenko, et al., Recent improvements in Geant4 electromagnetic physics and interfaces, Prog. Nucl. Sci. Technol. 2 (2011) 898.
[32] J. Apostolakis, et al., Radiat. Phys. Chem. 78 (2009) 859.
[33] M. Karamitros, et al., J. Comput. Phys. 274 (2014) 841.
[34] V.N. Ivanchenko, et al., Geant4 electromagnetic physics for LHC upgrade, J. Phys.: Conf. Ser. 513 (2014) 022015.
[35] V.N. Ivanchenko, et al., Geant4 electromagnetic physics: improving simulation performance and accuracy, SNA+MC 2013 Joint International Conference on Supercomputing in Nuclear Applications + Monte Carlo, Paris, EDP Sciences, 2014, p. 03101.
[36] D. Cullen, J.H. Hubbell, L. Kissel, The Evaluated Photon Data Library, Report UCRL-50400, vol. 6, 1997.
[37] F. Salvat, J.M. Fernandez-Varea, J. Sempau, PENELOPE-2008: A Code System for Monte Carlo Simulation of Electron and Photon Transport, OECD-NEA Report 6416, 2009, Issy-les-Moulineaux, France.
[38] A.B. Migdal, Phys. Rev. 103 (1956) 1811.
[39] T. Stanev, C. Vankov, R.E. Streitmatter, R.W. Ellsworth, T. Bowen, Phys. Rev. D 25 (1982) 1291.
[40] G.O. Depaola, Nucl. Instrum. Methods Phys. Res. A 452 (2000) 298.
[41] G.O. Depaola, M.L. Iparraguirre, Nucl. Instrum. Methods Phys. Res. A 611 (2009) 84.
[42] V.F. Boldyshev, E.A. Vinokurov, N.P. Merenkov, Yu.P. Peresun'ko, Phys. Part. Nucl. 25 (1994) 3.
[43] M.L. Iparraguirre, G.O. Depaola, Eur. Phys. J. C 71 (2011) 1778.
[44] J.M.C. Brown, M.R. Dimmock, J.E. Gillam, D.M. Paganin, MULECS: The Monash University low energy Compton scattering package, in: Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), IEEE, Valencia, Spain, 2011, p. 1385.
[45] J.M.C. Brown, M.R. Dimmock, J.E. Gillam, D.M. Paganin, Nucl. Instrum. Methods Phys. Res. B 338 (2014) 77.
[46] J.W.M. Du Mond, Phys. Rev. 33 (1929) 643.
[47] R. Ribberfors, Phys. Rev. B 12 (1975) 2067.
[48] Y. Namito, S. Ban, H. Hirayama, Nucl. Instrum. Methods Phys. Res. A 349 (1994) 489.

[49] XCOM Evaluated Data, National Bureau of Standards ⟨http://www.nist.gov/pml/data/xcom/⟩.
[50] K. Amako, et al., IEEE Trans. Nucl. Sci. 52 (2005) 910.
[51] A.G. Bogdanov, H. Burkhardt, V.N. Ivanchenko, S.R. Kelner, R.P. Kokoulin, M. Maire, A.M. Rybin, L. Urban, IEEE Trans. Nucl. Sci. 53 (2006) 513.
[52] A.V. Bagulya, M.S. Vladimirov, V.N. Ivanchenko, N.I. Starkov, Bull. Lebedev Phys. Inst. 36 (2009) 127 (Original Russian Text in Kratk. Soobshch. Fiz. 36 (2009) 3).
[53] A. Lechner, V.N. Ivanchenko, J. Knobloch, Nucl. Instrum. Methods Phys. Res. B 268 (2010) 14.
[54] K. Lassila-Perini, L. Urban, Nucl. Instrum. Methods Phys. Res. A 362 (1995) 416.
[55] Q. Yang, D.J. O'Connor, Z. Wang, Nucl. Instrum. Methods Phys. Res. B 61 (1991) 149.
[56] J. Apostolakis, S. Giani, L. Urban, M. Maire, M. Bagulya, V.M. Grichine, Nucl. Instrum. Methods Phys. Res. A 453 (2000) 597.
[57] A. Schaelicke, et al., Geant4 electromagnetic physics for the LHC and other HEP applications, J. Phys.: Conf. Ser. 331 (2011) 032029.
[58] D. Antonczyk, et al., Nucl. Instrum. Methods Phys. Res. A 565 (2006) 551.
[59] P. Christiansen, Int. J. Mod. Phys. E 16 (2007) 2457.
[60] H. Bichsel, Rev. Mod. Phys. 60 (1988) 663.
[61] S. Chauvie, Z. Francis, S. Guatelli, S. Incerti, B. Mascialino, P. Moretto, P. Nieminen, M.G. Pia, IEEE Trans. Nucl. Sci. 54 (2007) 2619.
[62] S. Incerti, et al., Med. Phys. 37 (2010) 4692.
[63] A. Valentin, M. Raine, J.-E. Sauvestre, M. Gaillardin, P. Paillet, Nucl. Instrum. Methods Phys. Res. B 288 (2012) 66.
[64] A. Valentin, M. Raine, M. Gaillardin, P. Paillet, Nucl. Instrum. Methods Phys. Res. B 287 (2012) 124.
[65] G. Molière, Z. Naturforsch. 2a (1948) 133.
[66] H.A. Bethe, Phys. Rev. 89 (1953) 1256.
[67] S. Goudsmit, J.L. Saunderson, Phys. Rev. 57 (1940) 24.
[68] H.W. Lewis, Phys. Rev. 78 (1950) 526.
[69] O. Kadri, V. Ivanchenko, F. Gharbi, A. Trabelsi, Nucl. Instrum. Methods Phys. Res. B 267 (2009) 3624.
[70] V.N. Ivanchenko, O. Kadri, M. Maire, L. Urban, Geant4 models for simulation of multiple scattering, J. Phys.: Conf. Ser. 219 (2010) 032045.
[71] M.J. Boschini, et al., Nuclear and non-ionizing energy-loss for Coulomb scattered particles from low energy up to relativistic regime in space radiation environment, Proceedings of the 12th ICATPP, October 2010, p. 9.
[72] M.J. Boschini, et al., Radiat. Phys. Chem. 90 (2013) 39.
[73] J. Allison, et al., Geant4 electromagnetic physics for high statistics simulation of LHC experiments, J. Phys.: Conf. Ser. 396 (2012) 022013.
[74] J. Apostolakis, A. Bagulya, S. Elles, V.N. Ivanchenko, O. Kadri, M. Maire, L. Urban, The performance of the Geant4 standard EM package for LHC and other applications, J. Phys.: Conf. Ser. 119 (2008) 032004.
[75] C.K. Ross, M.R. McEwen, A.F. McDonald, C.D. Cojocaru, B.A. Faddegon, Med. Phys. 35 (2008).
[76] ATLAS Collaboration, Phys. Lett. B 716 (2012) 1.
[77] CMS Collaboration, Phys. Lett. B 716 (2012) 30.
[78] S.M. Seltzer, M.J. Berger, Nucl. Instrum. Methods Phys. Res. B 12 (1985) 95.
[79] S. Klein, Rev. Mod. Phys. 71 (1999) 1501.
[80] V.N. Ivanchenko, et al., Recent progress of Geant4 electromagnetic physics and readiness for the LHC start, in: PoS (ACAT2008), p. 108.
[81] A.B. Migdal, Phys. Rev. 103 (1956) 1811.
[82] S. Abdullin, et al., Calorimetry Task Force Report, CMS-NOTE-2010-007; CERN-CMS-NOTE-2010-007, Geneva, 2010.
[83] G.O. Depaola, F. Longo, Nucl. Instrum. Methods Phys. Res. A 566 (2006) 590.
[84] G.O. Depaola, Nucl. Instrum. Methods Phys. Res. A 512 (2003) 619.
[85] A. Schaelicke, G. Alexander, R. Dollan, K. Laihen, T. Lohse, S. Riemann, P. Starovoitov, A. Ushakov, Pramana 69 (2007) l171.
[86] A. Ushakov, A. Schaelicke, S. Riemann, Simulation of polarized positron sources for linear colliders, J. Phys.: Conf. Ser. 298 (2011) 012021.
[87] H. Burkhardt, S. Kelner, R. Kokoulin, Monte Carlo generator for muon pair production, CERN-SL-2002-016 (AP) and CLIC Note 511, May 2002.
[88] I. Agapov, H. Burkhardt, D. Schulte, A. Latina, G.A. Blair, S. Malton, J. Resta-Lopez, Tracking studies of the compact linear collider collimation system, Phys. Rev. ST Accel. Beams 12 (2009) 081001; I. Agapov, et al., Halo estimates and simulations for linear colliders, JACoW Server, CERN-AB-2007-045, CLIC-Note-714, August 2007.
[89] H. Burkhardt, Monte Carlo Generation of the Energy Spectrum of Synchrotron Radiation, CERN-OPEN-2007-018 ⟨http://cds.cern.ch/record/1038899⟩.
[90] S.T. Perkins, et al., Tables and Graphs of Atomic Sub-Shell and Relaxation Data Derived from the LLNL Evaluated Atomic Data Library (EADL), $Z=1$–100, Report UCRL-50400, vol. 30, 1997.
[91] S. Paltani, Unpublished User Contribution, Geneva University, 2013.
[92] A. Mantero, et al., X-Ray Spectrom. 40 (2011) 135.
[93] W. Brandt, G. Lapicki, Phys. Rev. A 23 (1981) 1717.
[94] A. Taborda, P.C. Chaves, M.A. Reis, X-Ray Spectrom. 40 (2011) 127.
[95] A. Taborda, P.C. Chaves, M.L. Carvalho, M.A. Reis, X-Ray Spectrom. 42 (2013) 177.
[96] M. Abramovitz, I. Stegun, Handbook of Mathematical Functions, 1st ed., Dover, New York, 1965.
[97] Z. Francis, et al., Nucl. Instrum. Methods Phys. Res. B 316 (2013) 1.
[98] S. Incerti, et al., Nucl. Instrum. Methods Phys. Res. B 358 (2015) 210.
[99] Z.S. Hartwig, P. Gumplinger, Nucl. Instrum. Methods Phys. Res. A 737C (2014) 155.
[100] S. Lee, J. Hauptman, Unpublished User Contribution, University of Iowa, 2007.
[101] X. Qian, V. Vasileiou, Unpublished User Contribution, California Institute of Technology and University of Maryland, 2010.
[102] M. Janecek, W.W. Moses, IEEE Trans. Nucl. Sci. 57 (2010) 964.
[103] M.A. Bernal, et al., Phys. Med. 31 (2015) 861.
[104] Geant4 DNA Project ⟨http://geant4-dna.org⟩.
[105] S. Chauvie, et al., in: Proceedings of the 7th International Workshop: Microbeam Probes of Cellular Radiation Response, 2006, p. 676.
[106] S. Incerti, et al., Int. J. Model. Simul. Sci. Comput. 1 (2010) 157.
[107] G. Garcia Gomez-Tejedor, M.C. Fuss (Eds.), Radiation Damage in Biomolecular Systems, Springer, Netherlands, Dordrecht, 2012.
[108] C. Villagrasa, Z. Francis, S. Incerti, Radiat. Prot. Dosim. 143 (2011) 214.
[109] C. Champion, S. Incerti, H. Tran, Z. El Bitar, Nucl. Instrum. Methods Phys. Res. B 273 (2012) 98.
[110] Z. Francis, S. Incerti, R. Capra, B. Mascialino, G. Montarou, V. Stepan, C. Villagrasa, Appl. Radiat. Isot. 69 (2011) 220.
[111] M. Karamitros, et al., Prog. Nucl. Sci. Technol. 2 (2011) 503.
[112] V.N. Ivanchenko, S. Incerti, Z. Francis, H.N. Tran, M. Karamitros, M.A. Bernal, C. Champion, P. Gueye, Nucl. Instrum. Methods Phys. Res. B 273 (2012) 95.
[113] Z. Francis, S. Incerti, M. Karamitros, H.N. Tran, C. Villagrasa, Nucl. Instrum. Methods Phys. Res. B 269 (2011) 2307.
[114] C. Champion, et al., Appl. Radiat. Isot. 83 (2014) 137.
[115] T. Andre, et al., Nucl. Instrum. Methods Phys. Res. B 319 (2014) 87.
[116] S. Incerti, et al., Nucl. Instrum. Methods Phys. Res. B 333 (2014) 92.
[117] Z. Francis, S. Incerti, V. Ivanchenko, C. Champion, M. Karamitros, Z.El. Bitar, Phys. Med. Biol. 57 (2011) 209.
[118] M.U. Bug, E. Gargioni, S. Guatelli, S. Incerti, H. Rabus, R. Schulte, A. B. Rosenfeld, Eur. Phys. J. D 60 (2010) 85.
[119] M. Dos Santos, C. Villagrasa, I. Clairand, S. Incerti, Nucl. Instrum. Methods Phys. Res. B 298 (2013) 47.
[120] S. Incerti, C. Champion, H.N. Tran, M. Karamitros, M. Bernal, Z. Francis, V. Ivanchenko, A. Mantero, Nucl. Instrum. Methods Phys. Res. B 306 (2013) 158.
[121] M.A. Bernal, C.E. deAlmeida, C. Sampaio, S. Incerti, C. Champion, P. Nieminen, Med. Phys. 38 (2011) 4147.
[122] M. Dos Santos, I. Clairand, G. Gruel, J.F. Barquinero, S. Incerti, C. Villagrasa, Radiat. Prot. Dosim. 161 (2014) 469.
[123] Z. Kuncic, H.L. Byrne, A.L. McNamara, S. Guatelli, W. Domanova, S. Incerti, Math. Methods Med. 2012 (2012) 1.
[124] S. Incerti, et al., IEEE Trans. Nucl. Sci. 51 (2004) 1395.
[125] S. Incerti, N. Gault, C. Habchi, J.L. Lefaix, P. Moretto, J.L. Poncy, Th. Pouthier, H. Seznec, Radiat. Prot. Dosim. 122 (2007) 327.
[126] S. Incerti, H. Seznec, M. Simon, P. Barberet, C. Habchi, P. Moretto, Radiat. Prot. Dosim. 133 (2009) 2.
[127] P. Barberet, F. Vianna, M. Karamitros, T. Brun, N. Gordillo, P. Moretto, S. Incerti, H. Seznec, Phys. Med. Biol. 57 (2012) 2189.
[128] V. Breton, et al., Trends Comput. Sci. Eng. 3 (2013) 21.
[129] M. Dos Santos, C. Villagrasa, I. Clairand, S. Incerti, Prog. Nucl. Sci. Technol. 4 (2014) 449.
[130] Z. Francis, et al., Phys. Med. Biol. 59 (2014) 7691.
[131] M.S. Kreipl, W. Friedland, H.G. Paretzke, Radiat. Environ. Biophys. 48 (2009) 11.
[132] M. Karamitros, Extension de l'outil Monte Carlo généraliste Geant4 pour la simulation de la radiolyse de l'eau dans le cadre du projet Geant4-DNA (thesis), Bordeaux University, 2012.
[133] J. Apostolakis, A. Bagulya, S. Elles, V.N. Ivanchenko, J. Jacquemier, M. Maire, T. Toshito, L. Urban, J. Phys.: Conf. Ser. 219 (2010) 032044.
[134] G.A.P. Cirrone, G. Cuttone, F. DiRosa, L. Pandola, F. Romano, Q. Zhang, Nucl. Instrum. Methods Phys. Res. A 618 (2010) 315.
[135] L. Pandola, C. Andenna, B. Caccia, Nucl. Instrum. Methods Phys. Res. B 350 (2015) 41.
[136] E. Bernardi, et al., Nucl. Instrum. Methods Phys. Res. A 262 (1987) 229.
[137] G. D'Agostini, et al., Nucl. Instrum. Methods Phys. Res. A 274 (1989) 134.
[138] S. Jan, et al., Phys. Med. Biol. 56 (2011) 881.
[139] M. Canadas, P. Arce, P. Rato Mendes, Phys. Med. Biol. 56 (2011) 273.
[140] G. Santin, V. Ivanchenko, H. Evans, P. Nieminen, E. Daly, IEEE Trans. Nucl. Sci. 52 (2005) 2294.
[141] J. Perl, J. Shin, J. Schumann, B. Faddegon, H. Paganetti, Med. Phys. 39 (2012) 6818.
[142] S. Ibarmia, J. Eck, V. Ivanchenko, D. Lavielle, A. Rivera, J. Cueto, G. Santin, IEEE Trans. Nucl. Sci. 60 (2013) 2486.
[143] I. Mishustin, I. Pshenichnov, W. Greiner, Eur. Phys. J. D 60 (2010) 109.
[144] V.S. Barashenkov, Pion-nucleus cross-sections, Preprint P2-90-158, Dubna, 1990.
[145] V.S. Barashenkov, Nucleon-nucleus cross-sections, Preprint P2-89-770, Dubna, 1989.
[146] V.M. Grichine, Eur. Phys. J. C 62 (2009) 399.
[147] V.M. Grichine, Nucl. Instrum. Methods Phys. Res. B 267 (2009) 2460.
[148] Particle Data Group ⟨http://pdg.lbl.gov/2006/reviews/hadronicrpp.pdf⟩.
[149] Institute for High Energy Physics Database, Protvino, Russia ⟨http://wwwppds.ihep.su⟩.
[150] Nuclear Energy Agency, France ⟨http://www.nea.fr/html/dbdata/bara.html⟩.
[151] M.V. Kossov, Eur. Phys. J. A 14 (2002) 265; P.V. Degtyarenko, M.V. Kossov, H.P. Wellisch, Eur. Phys. J. A 8 (2000) 217; P.V. Degtyarenko, M.V. Kossov, H.P. Wellisch, Eur. Phys. J. A 9 (2000) 411; P.V. Degtyarenko, M.V. Kossov, H.P. Wellisch, Eur. Phys. J. A 9 (2000) 421.

[152] V. Franco, R.J. Glauber, Phys. Rev. 142 (1966) 1195.
[153] V. Franco, Phys. Rev. 175 (1968) 1376.
[154] O.D. Dalkarov, V.A. Karmanov, Nucl. Phys. A 445 (1985) 579.
[155] A.M. Zadorozhnyi, V.V. Uzhinsky, S.Yu. xSxhmakov, Sov. J. Nucl. Phys. 39 (1984) 729;
A.M. Zadorozhnyi, V.V. Uzhinsky, S.Yu. Shmakov, Yad. Fiz. 39 (1984) 1155.
[156] S.Yu. Shmakov, V.V. Uzhinskii, A.M. Zadorozhny, Comput. Phys. Commun. 54 (1989) 125.
[157] W. Broniowski, M. Rybczynski, P. Bozek, Comput. Phys. Commun. 180 (2009) 69.
[158] V. Uzhinsky, J. Apostolakis, A. Galoyan, G. Folger, V.M. Grichine, V. N. Ivanchenko, D.H. Wright, Phys. Lett. B 705 (2011) 235.
[159] S.P. Denisov, et al., Nucl. Phys. B 31 (1971) 253.
[160] P. Shukla, Phys. Rev. C 67 (2003) 054607.
[161] H.L. Bradt, B. Peters, Phys. Rev. 77 (1950) 54.
[162] S. Barshay, C.B. Dover, J.P. Vary, Phys. Lett. 51B (1974) 5;
S. Barshay, C.B. Dover, J.P. Vary, Phys. Rev. C 11 (1975) 360.
[163] L. Sihver, C.H. Tsao, R. Silberberg, T. Kanai, A.F. Barghouty, Phys. Rev. C 47 (1993) 1225.
[164] S. Kox, et al., Phys. Rev. C 35 (1987) 1678.
[165] W.-Q. Shen, B. Wang, J. Feng, W.L. Zhan, Y.T. Zhu, E.P. Feng, Nucl. Phys. A 491 (1989) 130.
[166] A. Capella, U. Sukhatme, C.I. Tan, J. Tran Thanh Van, J. Phys. Rep. 236 (1994) 227.
[167] A.B. Kaidalov, Phys. Lett. 116 B (1982) 459;
A.B. Kaidalov, K.A. Ter-Martirosyan, Phys. Lett. 117 B (1982) 247.
[168] B. Andersson, G. Gustafson, B. Nilsson-Almqvist, Nucl. Phys. B 281 (1987) 289.
[169] B. Nilsson-Almqvist, E. Stenlund, Comput. Phys. Commun. 43 (1987) 387.
[170] Geant4 Collaboration, Geant4 Physics Reference Manual ⟨http://geant4.web. cern.ch/geant4/UserDocumentation/UsersGuides/PhysicsReferenceManual/ fo/PhysicsReferenceManual.pdf⟩.
[171] B. Andersson, G. Gustafson, G. Ingelman, T. Sjöstrand, Phys. Rep. 97 (1983) 31.
[172] M. Alvioli, M. Strikman, Phys. Lett. B722 (2013) 347.
[173] K.G. Boreskov, A.B. Kaidalov, S.M. Kiselev, N.Ya. Smorodinskaya, Yad. Fiz. 53 (1991) 569.
[174] Kh. Abdel-Waged, N. Felemban, V.V. Uzhinskii, Phys. Rev. C 84 (2011) 014905.
[175] Kh. Abdel-Waged, V.V. Uzhinsky, Phys. Atom. Nucl. 60 (1997) 828 (Yad. Fiz. 60 (1997) 925);
K. Abdel-Waged, V.V. Uzhinsky, J. Phys. G 24 (1997) 1723.
[176] M.I. Adamovich, et al., EMU-01 Collaboration, Z. Phys. A 358 (1997) 337.
[177] A.Y. Abul-Magd, W.A. Friedman, J. Hufner, Phys. Rev. C 34 (1986) 113.
[178] G. Folger, V.N. Ivanchenko, J.P. Wellisch, Eur. Phys. J. A 21 (2004) 407.
[179] V.A. Abramovsky, V.N. Gribov, O.V. Kancheli, Yad. Fiz. 18 (1973) 595 (Sov. J. Nucl. Phys. 18 (1974) 308).
[180] A. Bolshakova, et al., HARP-CDP Group, Eur. Phys. J. C 70 (2010) 543.
[181] D.H. Wright, M.H. Kelsey, Nucl. Instrum. Methods Phys. Res. A 804 (2015) 175.
[182] G. Folger, V. Ivanchenko, H.-P. Wellisch, Eur. Phys. J. A 21 (2004) 404.
[183] A. Boudard, J. Cugnon, J.-C. David, S. Leray, D. Mancusi, Phys. Rev. C 87 (2013) 014606.
[184] D. Mancusi, A. Boudard, J. Cugnon, J.-C. David, P. Kaitaniemi, S. Leray, Phys. Rev. C 90 (2014) 054602.
[185] J.-J. Gaimard, K.-H. Schmidt, Nucl. Phys. A 531 (1991) 709;
J. Benlliure, A. Grewe, M. de Jong, K.-H. Schmidt, S. Zhdanov, Nucl. Phys. A 628 (1998) 458.
[186] K. Gudima, S.G. Mashnik, V.D. Toneev, Nucl. Phys. A 401 (1983) 329.
[187] John Apostolakis, et al., J. Phys.: Conf. Ser. 160 (2009) 012073 (Proceedings of the XIII International Conference on Calorimetry in High Energy Physics (CALOR 2008), Pavía, Italy, May 26–30, 2008).
[188] A. Ivanchenko, V. Ivanchenko, J.M. Quesada, S. Incerti, Int. J. Radiat. Biol. 88 (2012) 171.
[189] J.M. Quesada, in: Proceedings of the International Topical Meeting on Nuclear Research Applications and Utilization of Accelerators, Satellite Meeting on Spallation Reactions, Viena, Austria, May 4–8, 2009.
[190] J.P. Bondorf, A.S. Botvina, A.S. Iljinov, I.N. Mishustin, K. Sneppen, Phys. Rep. 257 (1995) 133.
[191] V.S. Barashenkov, et al., Communications JINR, P4-10781, Dubna, 1977.
[192] G.D. Adeev, et al., Preprint INR 816/93 Moscow, 1993 (in Russian).
[193] V.E. Weisskopf, D.H. Ewing, Phys. Rev. 57 (1940) 472.
[194] S. Furihata, K. Niita, S. Meigo, Y. Ikeda, F. Maekawa, JAERI-Data/Code 2001-015, Japan Atomic Energy Research Institute, 2001.
[195] National Nuclear Data Center, Evaluated Nuclear Structure Data Files ⟨http:// www.nndc.bnl.gov/ensdf ⟩.
[196] T.T. Böhlen, F. Cerutti, M. Dosanjh, A. Ferrari, I. Gudowska, A. Mairani, J. M. Quesada, Phys. Med. Biol. 55 (2010) 5833.
[197] S. Leray, et al., J. Korean Phys. Soc. 59 (2011) 791.
[198] J.M. Quesada, V. Ivanchenko, A. Ivanchenko, M.A. Cortés-Giraldo, G. Folger, A. Howard, D.H. Wright, Prog. Nucl. Sci. Technol. 2 (2011) 936 (Proceedings of the Joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2010 (SNA+MC2010), Tokyo, Japan, October 2010).
[199] I. Pshenichnov, A. Botvina, I. Mishustin, W. Greiner, Nucl. Instrum. Methods Phys. Res. B 268 (2010) 604.
[200] G. Dedes, M. Pinto, D. Dauvergne, N. Freud, J. Krimmer, J.M. Letang, C. Ray, E. Testa, Phys. Med. Biol. 59 (2014) 1747.

[201] H.C. Fesefeldt, Simulation of Hadronic Showers, Physics and Application, Technical Report PITHA 85-02, 1985.
[202] R.J. Glauber, in: S. Devons (Ed.), High Energy Physics and Nuclear Structure, Plenum Press, New York, 1970.
[203] V.M. Grichine, Comput. Phys. Commun. 181 (2010) 921.
[204] G.D. Alkhazov, S.L. Belostotsky, A.A. Vorobyov, Phys. Rep. 42 (1978) 89.
[205] M.B. Chadwick, et al., Nucl. Data Sheets 112 (2011) 2887.
[206] H.D. Lemmel, P.K. McLaughlin, BROND-2.1 Russian Evaluated Neutron Data Library, IAEA-NDS-90, Rev. 7, 1993.
[207] L. Tingjin, L. Qichang, S. Zongdi, Nucl. Data Sci. Technol. Res. Rep. Phys. (1992) 804.
[208] H.D. Lemmel, EFF-2.4, The European Fusion File 1994, Including Revisions up to May 1995, Summary Documentation, IAEA-NDS-170, June 1995.
[209] A.B. Pashchenko, H. Wienke, S. Ganesan, J. Nucl. Mater. 233–237 (1996) 1601.
[210] C. Nordborg, M. Salvatores, Status of the JEF evaluated data library, in: Proceedings of the International Conference on Nuclear Data for Science and Technology, vol. 2, Gatlinburg, Tennessee, USA, May 1994, p. 680.
[211] S. Chiba, et al., J. Nucl. Sci. Technol. 39 (2002) 187.
[212] T. Nakagawa, et al., J. Nucl. Sci. Technol. 32 (1995) 1259.
[213] Yu.N. Shubin, et al., MENDL-2 Neutron Reaction Data Library for Nuclear Activation and Transmutation at Intermediate Energies, IAEA-NDS-136, 1995.
[214] E. Mendoza, D. Cano-Ott, T. Koi, C. Guerrero, IEEE Trans. Nucl. Sci. 61 (2014) 2357.
[215] International Atomic Energy Agency Nuclear Data Services ⟨https://www-nds.iaea.org/geant4/⟩.
[216] Lawrence Livermore National Laboratory General Nuclear Data ⟨ftp://gdo-nuclear.ucllnl.org/pub⟩.
[217] J.W. Wilson, et al., NUCFRG2: An Evaluation of the Semi-empirical Nuclear Fragmentation Database, NASA Technical Report 3533, 1995.
[218] A.V. Ivantchenko, V.N. Ivantchenko, J.-M.Q. Molina, S.L. Incerti, Int. J. Radiat. Biol. 88 (2012) 171.
[219] The TileCal Collaboration, Nucl. Instrum. Methods Phys. Res. A 606 (2009) 362.
[220] C. Adloff, et al., The CALICE Collaboration, J. Instrum. 5 (2010) 05007.
[221] A. Bodek, IEEE Trans. Nucl. Sci. 46 (1999) 407.
[222] A. Dotti, Nuclear Science Symposium and Medical Imaging Conference (NSS/ MIC), 2011, IEEE, Valencia, Spain, p. 2128.
[223] The ATLAS Collaboration, Phys. Lett. B 716 (2012) 1.
[224] The CMS Collaboration, Phys. Lett. B 716 (2012) 30.
[225] V. Ivanchenko, et al., The Joint International Conference of the 7th Super-computing in Nuclear Applications and the 3rd Monte Carlo (SNA+MC 2010), Tokyo, 2011.
[226] J. Allison, et al., J. Phys.: Conf. Ser. 396 (2012) 022013.
[227] E. Abat, et al., J. Instrum. 6 (2011) 04001.
[228] A.E. Kiryunin, H. Oberlack, D. Salihagic, P. Schacht, P. Strizenec, Nucl. Instrum. Methods Phys. Res. A 560 (2006) 278.
[229] A. Dotti, et al., J. Phys.: Conf. Ser. 293 (2011) 012022.
[230] F. Simon, C. Soldner, L. Weuste, J. Instrum. 8 (2013) 12001.
[231] V.B. Melas, On the efficiency of the splitting and roulette approach for sensitivity analysis, in: Proceedings of the 1997 Winter Simulation Conference, Atlanta, Georgia, 1997, pp. 269–274.
[232] H. Kahn, Applications of Monte Carlo, Report AECU-3259, Rand Corporation, Santa Monica, California, 1954.
[233] G. Rubino, B. Tuffin, Rare Event Simulation using Monte Carlo Methods, Wiley Publishing, Hoboken, New Jersey, 2009 (ISBN:0470772697 9780470772690, and references therein).
[234] L. Desorgher, F. Lei, G. Santin, Nucl. Instrum. Methods Phys. Res. A 621 (2010) 247.
[235] H. Kahn, Nucleonics 6 (5) (1950) 27;
H. Kahn, Nucleonics 6 (6) (1950) 60.
[236] X-5 Monte Carlo Team, Monte Carlo N-Particle Transport Code, Los Alamos Report LA-UR-03-1987, Los Alamos National Laboratory, 2003.
[237] M.H. Mendenhall, R.A. Weller, Nucl. Instrum. Methods Phys. Res. A 667 (2012) 38.
[238] V. Innocente, E. Nagy, Nucl. Instrum. Methods Phys. Res. A 324 (1993) 297.
[239] W. Wittek, EMC Internal Reports EMC/80/15, EMC/CSW/80/39, 81/13 and 81/ 18, Unpublished; A. Strandlie, W. Wittek, Nucl. Instrum. Methods Phys. Res. A 566 (2006) 687.
[240] Abstract Interfaces for Data Analysis ⟨http://aida.freehep.org⟩.
[241] G. Barrand, softinex, inlib, exlib, ioda, g4view, g4exa, wall, J. Phys.: Conf. Ser. 513 (2014) 022002; See also ⟨http://inexlib.lal.in2p3.fr⟩.
[242] ROOT Physics Analysis Library ⟨http://root.cern.ch⟩.
[243] R. Brun, P. Palazzi, in: C.E. Vandoni (Ed.), Proceedings from the International Conference and Exhibition on Eurographics, vol. 80, Geneva, 1980, p. 93.
[244] GEANT4 Collaboration Release Policy ⟨https://cern.ch/geant4/collaboration/ tag_release.shtml⟩.
[245] Apache, Subversion Software Version Control ⟨https://subversion.apache.org/⟩.
[246] CMake Building and Testing Suite ⟨https://cmake.org/Wiki⟩.
[247] Worldwide LHC Computing Grid ⟨http://home.cern/about/computing/worldwide-lhc-computing-grid⟩ and first link therein ⟨http://wlcg-public.web.cern.ch/⟩.
[248] CERN Virtual Machine File System ⟨https://cernvm.cern.ch/portal/filesystem⟩.
[249] Geant4 Profiling and Benchmarking ⟨https://g4cpt.fnal.gov⟩.
[250] The FAST Project ⟨https://cdcvs.fnal.gov/redmine/projects/fast⟩.

[251] The Ignominous Profiler ⟨http://igprof.org/index.html⟩.
[252] Open|Speedshop ⟨http://www.openspeedshop.org/wp/⟩.
[253] Valgrind Memory Debugging Tool ⟨http://valgrind.org/⟩.
[254] Valgrind Thread Error Detector ⟨http://valgrind.org/docs/manual/drd-manual.html⟩.
[255] Synopsys, Coverity Static Code Analysis Tool ⟨http://coverity.com/⟩.
[256] Geant4 Download Page ⟨https://cern.ch/geant4/support/download.shtml⟩.
[257] PostgresSQL Relational Database ⟨http://www.postgresql.org/⟩.
[258] Geant4 Validation Web Application ⟨http://g4validation.fnal.gov:8080/G4ValidationWebApp/⟩.

[259] Java Platform, Enterprise Edition ⟨http://www.oracle.com/technetwork/java/javaee/overview/index.html⟩.
[260] GlassFish Open Source Application Server ⟨https://glassfish.java.net/⟩.
[261] PrimeFaces Open Source User Interface Components ⟨http://primefaces.org/⟩.
[262] A. Dotti, et al., Extending Geant4 Parallelism with External Libraries (MPI, TBB) and its Use on HPC Resources, IEEE/NSS 2015 Conference Record, ArXiv: http://arxiv.org/abs/1605.01792 (to be published).
[263] P. Szostek, et al., Beyond core count: a look at new mainstream computing platforms for HEP workloads, J. Phys.: Conf. Ser. 513 (2014) 062036.