Available online at www.sciencedirect.com
ScienceDirect
Nuclear Instruments and Methods in Physics Research B 266 (2008) 1880–1885
NIM B
Beam Interactions with Materials & Atoms
www.elsevier.com/locate/nimb

Fast Monte Carlo for ion beam analysis simulations

François Schiettekatte *

Regroupement Québécois sur les Matériaux de Pointe, Département de Physique, Université de Montréal, Montréal, QC, Canada H3C 3J7

Received 24 September 2007; received in revised form 27 November 2007
Available online 23 December 2007

Abstract

A Monte Carlo program for the simulation of ion beam analysis data is presented. It combines mainly four features: (i) ion slowdown is computed separately from the main scattering/recoil event, which is directed towards the detector. (ii) A virtual detector, that is, a detector larger than the actual one can be used, followed by trajectory correction. (iii) For each collision during ion slowdown, scattering angle components are extracted form tables. (iv) Tables of scattering angle components, stopping power and energy straggling are indexed using the binary representation of floating point numbers, which allows logarithmic distribution of these tables without the computation of logarithms to access them. Tables are sufficiently fine-grained that interpolation is not necessary. Ion slowdown computation thus avoids trigonometric, inverse and transcendental function calls and, as much as possible, divisions. All these improvements make possible the computation of 10^7 collisions/s on current PCs. Results for transmitted ions of several masses in various substrates are well comparable to those obtained using SRIM-2006 in terms of both angular and energy distributions, as long as a sufficiently large number of collisions is considered for each ion. Examples of simulated spectrum show good agreement with experimental data, although a large detector rather than the virtual detector has to be used to properly simulate background signals that are due to plural collisions. The program, written in standard C, is open-source and distributed under the terms of the GNU General Public License.

PACS: 52.65.Pp; 34.50.−s; 82.80.Yc

Keywords: Monte Carlo simulation; Ion beam analysis

1. Introduction

Monte Carlo methods (MC) represent an attractive solution for the simulation of ion beam analysis (IBA) data as they encompass, in principle, all the important phenomena that affect ion transport by directly simulating their trajectory. These effects include deviations, small or large, form straight-line trajectory usually assumed in slab simulations [1], and a physical representation of the detection system. However, MC simulations imply the trajectory computation of one ion at the time. Since the fraction of incident ions that actually produce an event in the detector is of the order of 10^{-6}, the trajectory of billions of ions would have to be computed to get a spectrum with reasonable statistics. Direct simulation using TRIM [2] without other improvement than keeping the target thin was reported to take months on personal computers (PC) [3].

In order to improve this state of matters, different approaches have been suggested [4–6]. Here, we will focus on three such improvements: (i) As suggested by Arstila [4], the incoming and outgoing atom slowdown process can be separated form the main scattering event. At some depth, an atom is deliberately thrown within a cone along the detector axis. The detection efficiency is thus no longer affected by the angular dependence of the cross-section. (ii) The concept of a virtual detector several times larger than the actual detector can be used [4]. An atom trajectory intersecting the virtual detector is rotated to the actual detector and a correction to the kinematic factor and electronic energy losses is applied. (iii) Yuan et al. suggested to

* Tel.: +1 514 343 6049; fax: +1 514 343 7357.
E-mail address: francois.schiettekatte@umontreal.ca

0168-583X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.nimb.2007.11.075

interpolate the value of $\sin^2\theta_{CM}/2$, where θ_{CM} is the scattering angle in the center-of-mass, from a table [7] rather than computing it using Biersack's MAGIC algorithm [2]. The table was indexed using the binary representation of floating point numbers, allowing a logarithmic progression of the index related to ion energy without computing logarithms.

In this paper, an open-source program called CORTEO is presented. It regroups those three improvements and further extends the third one to avoid any trigonometric, inverse and transcendental function calls during ions slowdown computation. This makes MC simulation of IBA data possible within seconds or minutes on modern PCs. In the next section, the program parameters, features and computational strategies are summarized. Then, the effects of the different approximations are discussed through the simulations of Rutherford backscattering spectrometry (RBS) and elastic recoil detection (ERD) data.

2. Program description

CORTEO is a MC simulation code, written in standard C language that computes ion transport through a target consisting of layers. The thickness t and atom concentration N is specified for each layer, together with the atomic number, mass and fraction of each element in a layer. The user sets the beam ions atomic number Z_1, mass M_1, initial energy E_0, initial directional cosines and beam diameter. The beam is assumed to have a Gaussian section. The user also sets the atomic number Z_2 and mass M_2 of the partner for the main collision and the layer where to simulate it. This element is usually one of the elements of which this layer consists, but it could be another unrelated element if the user wants for example to simulate the spectrum of a trace element in this layer. Including this element in the layer description would not significantly change the slowdown process, but will increase the resources and time required for its computation.

Currently, in order to simplify computations, namely those related to the detection, the sample surface is considered to sit in the $x = 0$ plane and the detector axis lays in the $z = 0$ plane. The user sets the detector distance and its aperture radius and the x and y components of the detector axis. A foil can be included at the entrance of the detector (either as an absorber or as a timing foil) and a second detector at a certain distance from the first one and laying in the same axis can be considered.

2.1. Separation of slowdown and scattering

As suggested by Arstila [4], CORTEO separates the slowdown process from the main scattering event. The slowdown process of each ion is computed to a certain collision depth, taking into account multiple/plural collisions and the related energy transfer to target atoms and the electronic energy loss and straggling. Either Bohr [8], Chu [9] or Wang [10] energy straggling can be considered.

During a simulation, the collision depth is scanned from the top to the bottom of the simulated layer. At that depth, the ion or recoil is randomly thrown within a cone towards the detector. The cone angle can be established by a presimulation to estimate the angular distribution of the ions reaching the detector [4]. But as it will be discussed below, in order to properly account for the long tails due to plural scattering, the cone may have to cover $\sim 4\pi$ str. Since ions are emitted in equal number in all directions within the cone, the rate at which ions reach the detector is independent of the cross-section angular dependence and a significant fraction of the atoms are allowed to reach the detector.

After the main scattering or recoiling event, the slowdown process of the backscattered ion or recoil is computed until it reaches the surface. Calculation is interrupted if the particle reaches the back of the sample, if its energy reaches a certain minimum or if the outgoing atom is emitted in a direction not allowed by kinematics during the main collision.

For each scattered/recoiled atom reaching the sample surface, intersection with the detector is determined. If the ion is detected, the cross-section is computed considering the angle of the main collision. Andersen screening of the cross-section [11] can be taken into account. The program also includes provisions for computing the detected ion slowdown through a foil and its intersection with a second detector in the case of time-of-flight (TOF) measurements.

Finally, the energy spectrum is incremented by the value of the cross-section, which is computed only for detected atoms. Since the scattered atoms or recoils were thrown equally in all directions, incrementing the spectrum by the cross-section accounts for the probability of detection. A spectrum being a sum of cross-sections $d\sigma_i/d\Omega$ over i detected atoms in an energy interval, the number of detected atoms in this energy interval is thus simply

$$n = Nt\left(\frac{1}{i}\sum_i \frac{d\sigma_i}{d\Omega}\right)q\Delta\Omega, \tag{1}$$

where q is the incident number of atoms and $\Delta\Omega$ is the detector solid angle. However, normalization by the number of detected particles can be problematic when a significant fraction of the particles reach minimum energy. i can be determined more reliably as the number of ions that should have reached the detector, that is, $i = I\Delta\Omega/\Omega_{cone}$ where I is the number of simulated ions and Ω_{cone} is the solid angle withstood by the cone in which scattering/recoil events are thrown.

2.2. Virtual detector

To further improve the detection efficiency, the concept of virtual detector proposed by Arstilla [4] is implemented in CORTEO. For ions exiting at surface, the intersection with a detector 5–10 times larger in diameter than the

actual detector is considered, improving by a factor of 25–100 the detection efficiency. For atoms entering the virtual detector, a spot is selected randomly on the actual detector and the ion trajectory is rotated. The kinematic factor of the main collision is recalculated and electronic energy loss is corrected for the difference of distance traveled in the target. The number or angles of the collisions during the slowdown process, however, are not corrected. A drawback of this implementation is that this kind of correction applied to trajectories that are far from being approximately straight may result in unrealistic corrections. This is discussed below.

2.3. Improving computation speed

During the slowdown process, an atom undergoes a series of collision, usually at wide angles, keeping trajectory almost straight, but not always. Full details about how this process can be simulated are found in [2]. The computation usually involves calling numerous trigonometric and transcendental functions. In order to avoid such slow function calls, CORTEO relies on tables to directly get the needed values. In the following part, quantities that are extracted form a table are included in {braces}.

In CORTEO, the average flight length ℓ_0 is set by the user. A value of a few nanometers has usually to be used to properly reproduce all effects affecting the slowdown process. According to Poisson's statistic, the flight length distribution follows $\exp(-\ell/\ell_0)$. For each collision, the square root of the flight length is computed following

$$\sqrt{\ell} = \sqrt{\ell_0}\sqrt{-\ln r_1}, \quad \ell = (\sqrt{\ell})^2, \tag{2}$$

where r_1 is a random number between 0 and 1. $\sqrt{\ell}$ rather than ℓ is computed because it is needed for energy straggling calculations, below. CORTEO computes in advance $\sqrt{\ell_0}$ and values of $\{\sqrt{-\ln r_1}\}$ come from a long ($\sim 10^5$) pre-computed list, so no square root or logarithmic function calls are necessary. Similarly, the reduced impact parameter s of the collision is computed as

$$s = s_0\{\sqrt{r_2}\}\{1/\sqrt{-\ln r_1}\}, \tag{3}$$

where the last term is a table containing reciprocal values of the table used to compute Eq. (2). The value

$$s_0 = 1/a\sqrt{\pi N \ell_0}, \tag{4}$$

where a is the screening length, is computed in advance. In order to make sure that the tables are unbiased, uniformly distributed tables are generated and they are then randomly ordered.

Using s and the reduced energy $\varepsilon = fE$, where $f = a/(M_1/M_2 + 1)$ and E is the ion energy, the scattering angle can be computed. As already mentioned, TRIM uses the MAGIC algorithm to compute $\sin^2\theta_{CM}/2$ considering a screened Coulomb potential and this is already an improvement by several orders of magnitude over the direct computation of the scattering integral. Yuan et al. [7] showed that a significant execution speed gain is further obtained by interpolating the value $\sin^2\theta_{CM}/2$ in a table. As the value of θ_{CM} varies smoothly as a function of s and $\log\varepsilon$ (except at small values of s and large values of ε), one can interpolate in a table of $\sin^2\theta_{CM}/2$ mapped on these parameters. However, this would involve the computation of a logarithm at each collision. To avoid that, Yuan et al. suggested using the exponent bits of the binary representation of ε to index the table [12]. They interpolate between four values to get the value of $\sin^2\theta_{CM}/2$ corresponding to a given $\log\varepsilon$ and s. There is however some concern with the precision of this method at small values of s and large values of ε .

In CORTEO, $\sin^2\theta_{CM}/2$ is mapped over logarithmically progressing indexes of both s and ε, ensuring a smooth variation over all the parameter space. Furthermore, the four most significant mantissa bits are used in addition to the exponent bits, so the index increment corresponds 1/16 of the power of 2 of a value. The table is thus sufficiently fine-grained that not interpolation is necessary. The relative error for values of $\sin^2\theta_{CM}/2 > 10^{-6}$ is about 5% on average, but after several collisions, this error will smear out and becomes negligible. An index spanning from 0 to 500 covers more than nine orders of magnitude in terms of s and ε values. A 500 by 500 single-precision floating point matrix requires 1 MB of memory to store. The matrix $\sin^2\theta_{CM}/2$ is computed once, at installation time, by numerically integrating the scattering integral using the Gauss-Mehler quadrature [7].

What is needed then for the computation of the trajectory rotation are the scattering angle components in the laboratory reference frame, $\cos\theta$ and $\sin\theta$, which can be computed from $\sin^2\theta_{CM}/2$ through inverse trigonometric and square root function calls. To avoid these lengthy operations within an ion slowdown computation, CORTEO pre-computes from the $\sin^2\theta_{CM}/2$ matrix two tables, $\{\cos\theta\}$ and $\{\sin\theta\}$, for each pair M_1, M_2 considered in the simulation. Hence, scattering angle components are obtained at memory access speed from the indexes corresponding to s and ε. The memory burden, however, can be of several Mb, depending on the number of elements constituting the layers. For this reason, the layer description may not include trace elements and isotopes when they do not significantly impact ion slowdown. Their spectrum, as a detected element, can be computed anyway because scattered/recoiled ions are generated independently of the layers description.

Scattering in the random phase approximation involves a rotation by a random azimuthal angle ω. Tables of $\{\cos\omega\}$, $\{\sin\omega\}$ for $\omega = 2\pi r_3$ are generated at the beginning of the program execution, with a large number of r_3 values uniformly distributed between 0 and 1 and randomly ordered. Still, the computation of the directional cosines rotation involves the calculation of a square root that cannot be avoided. This single function call turns out represent 25% of the computing time of the whole

program if the standard C function is used, so CORTEO relies on bit-shifting and table look-up to compute this function.

Finally, the electronic energy loss and straggling are calculated. At installation time, CORTEO invokes SRIM's SRModule to generate stopping power tables of all ions in every element. When a simulation starts, the program computes stopping power tables for each layer considering Bragg's rule. (A correction factor or loading external tables could also be implemented easily.) These tables are mapped over a logarithmically progressing index extracted from the binary representation of the ion energy E, so the table values are varying smoothly. For the energy straggling, a factor excluding $\sqrt{\ell}$ is pre-computed for each layer and the energy straggling is multiplied by a value randomly picked from a list of uniformly distributed values of inverse error function, thus assuming a Gaussian distribution for this effect. Convolution by the energy resolution of the detector is carried out on the final spectrum at the end of the simulation.

3. Examples

In this section, a comparison to SRIM-2006 is shown for transmitted ions and examples of spectrum simulations are discussed. Computation times reported here are those obtained using a computer featuring a 1.7 GHz Intel Centrino processor and 512 MB of memory.

3.1. Comparison to SRIM-2006

Fig. 1 presents distributions of energy and directional cosines components for 500 keV He ions at the exit of a 100 nm gold layer. ℓ_0 was set to 1 nm so 100 collision occurred on average in the layer. The simulation of 10^5 ions, that is, 10^7 collisions, required 1.3 s. Very good agreement is found between these simulations, as for all other combinations tested, provided that enough collisions occurred. (CORTEO simulations with less than 10 collisions in the layer often started to differ significantly from SRIM results.) It thus appears that the approximations used for the computation of ion slowdown are valid. This particular example was chosen to underline the broad angular distributions that affect light ions traveling through heavy substrates (Fig. 1(b) and (c)). This makes plural collisions a significant effect that has to be taken into account in such samples.

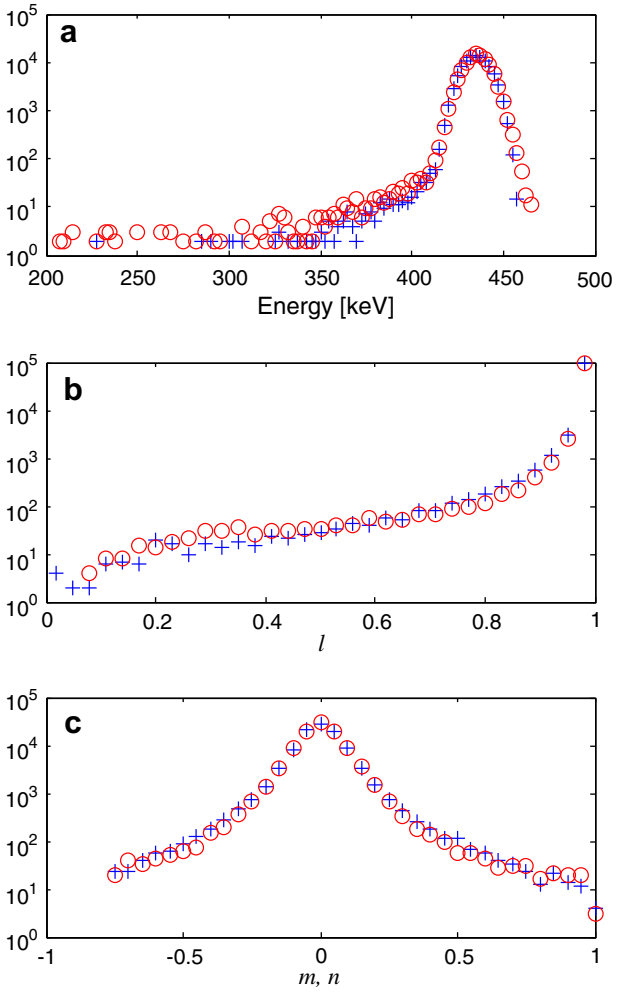

Fig. 1. Comparison between CORTEO (circles) and SRIM-2006 (plus) of (a) the energy distribution and directional cosines parallel (b) and perpendicular (c) to initial ion trajectory for 500 keV He ions at the exit of a 100 nm gold layer. CORTEO simulations carried out considering Bohr energy straggling.

3.2. RBS

In Fig. 2, the RBS spectrum of a sample consisting of 100 nm of gold on silicon measured using a 500 keV He beam is shown. The solid line in Fig. 2(a) is a spectrum obtained considering backscattering events within a relatively narrow cone (60°) about detector axis. A virtual detector five times larger than the actual detector was considered. 10^6 incident ions were simulated for each of the Au and Si layers and about 6×10^4 ions were detected in each case. The simulation for Au ran in 4 s and that for Si in 19 s. It is seen that the resulting spectrum (solid line), while well reproducing the gold peak, misses some amplitude at level of the Si signal and does not reproduce the background signal between the Au and Si. This background is the result of plural collision taking place when ions are initially scattered at more than 60° from the detector axis. One may thus consider a wider cone. The dashed line in Fig. 2(a) shows the resulting spectrum if the simulation is carried out with a wider cone (150°) using a virtual detector. Clearly, this simulation overestimates the background above 250 keV and underestimates it below that energy. This is a result of the trajectory correction carried out on virtually detected ions. The correction assumes small deviations from straight line trajectory, but this is far from being the case for ions forming the low-energy background, as it involves particles initially directed more than 60° away from the detector before being scattered towards it by plural collisions.

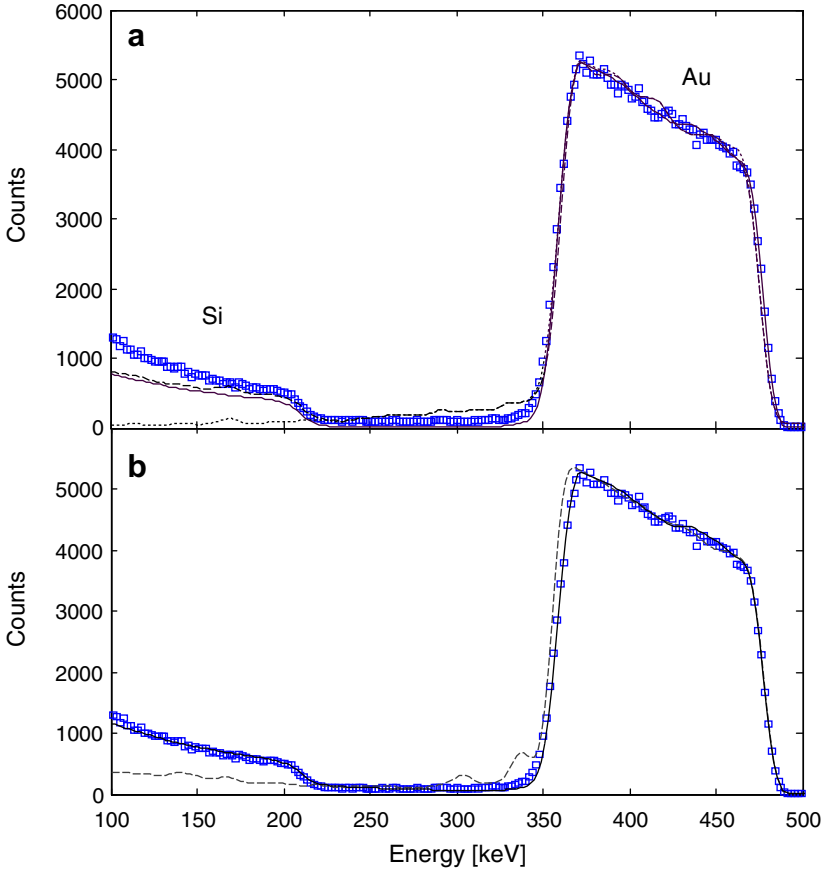

Fig. 2. RBS spectrum of 100 nm of gold on a silicon substrate obtained with a 500 keV He beam incident at 7° off surface normal and with detector at 10° from beam. Squares: experimental data. (a) Solid line: simulation with ions backscattered in a 60° cone about detector axis and using a virtual detector five times larger than the actual detector; dashed line: 150° cone (3.7π str); dotted line: Au contribution to the dashed line simulation. (b) Dashed line: Au layer simulation considering a detector five times larger than the actual detector (but not virtual) and a 150° cone about detector axis; solid line: same as solid line in (a) but with a background added considering a fit to the dashed line below 300 keV. ℓ_0 was set to 5 nm for these simulations, which are convoluted for detector resolution.

As a workaround, one can simulate the Au spectrum still considering a wide cone and a detector five times larger than the actual one, but not carrying out the trajectory corrections as with the virtual detector. Such a spectrum, represented by the dashed line in Fig. 2(b), is kinematically broadened, as seen from the back of the Au spectrum. But it produces the right background in the region below 300 keV. Kinematic broadening on this background has little effects. It is also seen that the background contains bumps, which are the result of the fact that some detected ions that contributed to this background where heading far away from the beam and detector after the main collision and are thus contributing an enormous cross-section to the spectrum compared to other particles scattered at a smaller angle. Still, these are rare events that, on average, would not significantly contribute to the spectrum amplitude. For this reason, the cone is restricted to 150° for the wide angle simulations. Finally, one can fit the background obtained with the large detector and add it to the narrow angle spectrum of Fig. 2(a) to obtain the spectrum represented by the solid line in Fig. 2(b). Simulating the background spectrum with enough detected atoms to avoid too many fluctuations required 5×10^6 incident ions, which ran in 79 s for a total of 102 s for the three simulations forming the solid line of Fig. 2(b). This, however, is a peculiar example of a spectrum featuring a strong background. Usual simulations can be achieved only with a virtual detector at narrow angle and would take a few seconds to compute.

Another remark is that one could have accelerated the simulation by choosing n ℓ_0 larger than 5 nm, as used in this example. But choosing an ℓ_0 of 10 nm already significantly broadens the edge of the silicon signal. This value must remain small to produce correct simulations.

3.3. ERD

Finally, an example of simulation applied to ERD-TOF mass separated spectra appears in Fig. 3. These spectra are accumulated using the energy provide by the final detector and simulations are convoluted for the energy resolution. It is seen that simulations, using a virtual detector 10 times larger in diameter than the actual detector (solid lines), reproduce relatively well the different spectra. An exception

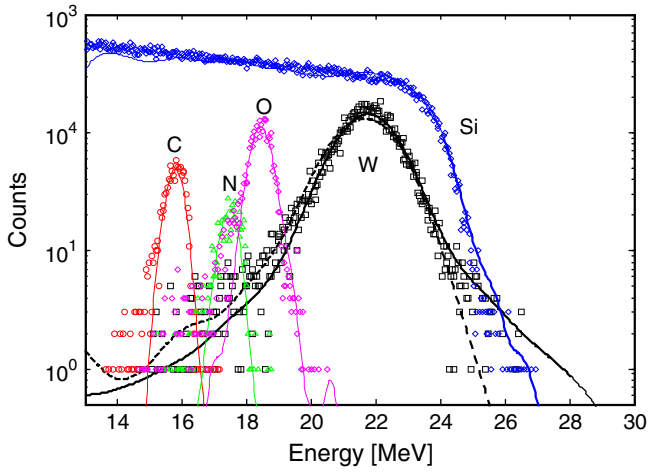

Fig. 3. ERD-TOF recoils energy spectra of a thin WCN layer on a thin SiO_2 layer on a Si substrate measured using a 40 MeV Cu beam incident at 15° off surface with detector at 150° of the beam. Symbols are the experimental spectra and solid lines the corresponding simulations using a virtual detector 10 times larger than the actual one. Recoils were thrown in a cone forming an angle of 90° (that is, a half sphere). The number of incident ions was 10^5 for C, N and O (~2 s each) and 10^6 for W and Si (16 and 20 s). The W spectrum was also simulated with only the real detector (dashed line) and required 10^8 incident ions, the simulation lasting almost 30 min.

to that might be the long high energy tail in the W simulation. Again, such a tail is due to trajectory far from being straight lines. A simulation considering only the real detector (100 times longer to run) shows no such tail, but rather a longer low-energy tail. Again, trajectory corrections must be improved for ions undergoing plural collisions. The virtual detector can be used to reproduce in a small amount of time the main features of a spectrum, but because of the way the trajectory is corrected, it cannot be used to reproduce backgrounds.

4. Conclusion

A Monte Carlo simulation program has been designed with the objective of simulating IBA spectra in a short amount of time on a modern PC. To do so, it relates on strategies that include the scattering/recoiling of ions in the direction of the detector, on the use of a virtual detector and on a fast access to tables that provide directly the values necessary for trajectory computation without computing trigonometric, inverse or transcendental functions. Simulations are possible within seconds if the spectrum does not contain significant contributions from plural straggling. If it does, the use of a virtual detector cannot reproduce such background considering the way the trajectory correction is currently carried out. Strategies relating on the estimation of the background using a large detector can be used, involving simulations of the order of a minute. A spectrum can also be simulated without these approximations in less than an hour.

Future developments will include provisions to take into account roughness. Since each ion can probe a different thickness, this should be possible without lengthening the simulations. It will also include, as suggested by Arstila [4], the emission of more than one scattered/recoiled ion per incident ion, cutting simulation time by 2. Arbitrary positioning and orientation of detectors will be implemented. More efficient use of memory will also be examined. The program is available online [13] for collaborative development under the terms of the GNU General Public License (GPL).

Acknowledgements

The author would like to thank Subash Gurathi for providing the ERD-TOF spectrum and Kai Arstila for fruitful discussions. This project was supported by the NSERC, FQRNT and NanoQuébec.

References

[1] L.R. Doolittle, Nucl. Instr. and Meth. 92 (1971) 481.
[2] J.F. Ziegler, J.P. Biersack, U. Littmark, The Stopping and Range of Ions in Solids, Pergamon, New York, 1985.
[3] P.N. Johnston, R.D. Franich, I.F. Bubb, M. El Bouanani, D.D. Cohen, N. Dytlewski, R. Siegele, Nucl. Instr. and Meth. B 161–163 (2000) 314.
[4] K. Arstila, T. Sajavaara, J. Keinonen, Nucl. Instr. and Meth. B 174 (2001) 163.
[5] R.D. Franich, P.N. Johnston, I.F. Bubb, Nucl. Instr. and Meth. B 219–220 (2004) 87.
[6] R.D. Franich, P.N. Johnston, I.F. Bubb, N. Dytlewski, D.D. Cohen, Nucl. Instr. and Meth. B 190 (2002) 252.
[7] B. Yuan, F.C. Yu, S.M. Tang, Nucl. Instr. and Meth. B 83 (1993) 413.
[8] N. Bohr, K. Dan, Vidensk. Selsk. Mat.-Fys. Medd. 18 (1948) 8.
[9] W.K. Chu, Phys. Rev. A 13 (1976) 2057.
[10] Q. Yang, D.J. O'Connor, Z. Wang, Nucl. Instr. and Meth. B 61 (1991) 149.
[11] H.H. Andersen, F. Besenbacher, P. Loftager, W. Möller, Phys. Rev. A 21 (1980) 1891.
[12] In computers based on the IEEE Standard 754, 32-bit floating point numbers are represented as 23 mantissa bits, followed by 8 exponent bits (base 2) and one sign bit.
[13] http://www.lps.umontreal.ca/~schiette.