IPP-Report

Max-Planck-Institut
für Plasmaphysik

Wolfgang Eckstein, Renate Dohmen, Andreas Mutzke,
Ralf Schneider

SDTrimSP: Ein Monte-Carlo Code zur Berechnung von Stossereignissen in ungeordneten Targets

SDTrimSP: A Monte-Carlo Code for Calculating Collision Phenomena in Randomized Targets

IPP 12/3
Januar, 2007

SDTrimSP: A Monte-Carlo Code for Calculating Collision Phenomena in Randomized Targets

W. Eckstein, R. Dohmen A. Mutzke, R. Schneider

January 4, 2007

Abstract

The new program SDTrimSP is designed for atomic collisions in amorphous targets to calculate ranges, reflection coefficients and sputtering yields as well as more detailed information as depth distributions of implanted and energy distributions of backscattered and sputtered atoms. The program is based on the binary collision approximation and uses the same physics as its predecessors TRIM.SP and TRIDYN, but the structure of the new program has been completely changed. It runs on all sequential and parallel platforms with a F90 compiler. Table lookup is applied for all available atomic data needed for input, and different integration schemes for several interaction potentials are provided. Selected examples of typical results are given to show the manifold of possible applications.

Contents

1. Introduction **3**

2. Physical basis **3**

3. Structure of the code **4**

4. Implementation **7**

5. Performance **8**
5.1. Influence of NR and NH on the accuracy of the results 9
5.2. Parallel efficiency 11

6. Special applications **14**
6.1. Static mode 14
6.2. Dynamic mode 18

7. Conclusions **23**

8. Acknowledgment **23**

A. Global parameters **25**

B. Input variables in 'tri.inp' **25**
B.1. Necessary input variables in 'tri.inp' 25
B.2. Optional input variables in 'tri.inp' 27

C. Inputfile 'tri.inp' and 'layer.inp' of all examples **35**

1. Introduction

In the last 40 years many computer simulation programs have been developed to describe the interactions of ions bombarding solid, liquid and gaseous targets. Many of these programs were based on the binary-collision approximation dealing with crystalline and amorphous targets. Examples of programs dealing with amorphous targets are the static Monte-Carlo program TRIM and the corresponding dynamic version TRIDYN which were successful in describing collision effects in solids for many examples [1]. Many versions of these two programs evolved to handle specific physical problems. This triggered the idea to combine both programs in a new version SDTrimSP (where S stands for static and D for dynamic) with all possible output facilities used in the past like sputtering, backscattering and transmission. This offered the opportunity to introduce at the same time a modular structure, to have a more flexible output and to provide a higher portability. The program is suited equally well for all sequential architectures and for all parallel architectures, for which a Fortran 90 (F90) compiler and the MPI (Message Passing Interface) communication library are available. The new program also includes features which were used in the past, but were not incorporated in most versions of TRIM.SP and TRIDYN. The time dependence of the collision cascade can be chosen as described in [16, 17].

2. Physical basis

The new program SDTrimSP is based on TRIM.SP [2] and TRIDYN [3, 4]. Both programs, the static TRIM.SP and the dynamic TRIDYN, are described in [1]. The basic physics in the new program SDTrimSP is the same as in the former versions. SDTrimSP is a Monte Carlo program, which assumes an amorphous (randomized) target structure at zero temperature and infinite lateral size. The binary collision approximation is used to handle the atomic (nuclear) collisions. This means, that the change in flight direction due to the collision is given by the asymptotes of the real trajectory. For this evaluation an interaction potential has to be chosen (usually purely repulsive and only dependent on the distance between the colliding atoms) to determine the scattering angle of the moving atom and the recoil angle of the atom, which is set into motion. Then the energy loss (nuclear) of the moving atom and the energy gain of the recoil can be calculated. In addition, a moving atom looses energy to target electrons (electronic or inelastic energy loss). The program also provides the possibility to include simultaneous weak collisions, but strictly in the binary collision approximation. The program follows projectiles (incident atoms) and target recoil atoms three-dimensionally until their energy falls below some preset value or if they have left the target (backscattering, transmission, sputtering).
Besides a more modular structure many new features are included in the program. Most data needed for a calculation is taken from a database in form of tables: atomic numbers and masses of elements, densities of solid and liquid elements, surface binding energies (heat of sublimation), displacement energies are taken from Table 6.1 of [1]; one table

provides isotopic masses of elements; two other tables give the constants for the inelastic energy loss of hydrogen [5] and helium [6]. Different interaction potentials as KrC [7], ZBL [8], Molière [9], Nakagawa-Yamamura [21] , power potentials and a special Si-Si potential [10] can be chosen as well as different integration methods of the scattering integral as Magic [11], Gauß-Mehler [12], and Gauß-Legendre [13]. Magic is faster than the Gauß-Mehler and the Gauß-Legendre procedure, but is only available for KrC, ZBL and Molière. Evaluation of the accuracy [14] of the integration procedure Magic shows a maximum relative error of the scattering angle in the center-of-mass system of about 1 % nearly independent of the relative impact parameter (impact parameter/screening length). For the Gauß-Mehler procedure the corresponding error is increasing with an increasing relative impact parameter and depends on the number of pivots [14]. According to Robinson [15], the Gauss-Mehler method is generally more accurate than the Gauss-Legendre method in evaluating the scattering angle integral, but less accurate for the time integral.

3. Structure of the code

The code SDTrimSP treats the bombardment of incident ions on different target structures. Besides mono-atomic targets, layer structures, fixed and variable composition target structures are allowed. The kind of projectiles and/or target atoms is not limited. Both incident ions and recoil atoms are treated as series of subsequent collisions. There are two general cases in the code:

- static case: the target composition is fixed during the whole simulation

- dynamic case: modifications of the target caused by the ion bombardment are taken into account; in this case the target is updated at regular intervals, i.e. after a certain number NR$\geq$ 1 of projectiles and corresponding showers. NR has to be specified as a parameter in the input file.

The atoms are distinguished in projectiles (incident atoms) and recoils (target atoms). For each traced atom the important physical quantities, as energy, spatial coordinates, direction of motion, are recorded along its path using general data structures. Moreover, the path length and the number of collisions are stored for the projectiles, while for the recoils the collision number in which they are generated is stored (generation). Besides the information about the single projectile there are also quantities integrated over all projectiles to save memory. For projectiles, the inelastic (electronic) energy loss and the total elastic and the elastic loss larger than the displacement energy are stored. Other values derived from these basic quantities can be determined, if of interest.
The structure of the program is depicted in Fig. 1. In the projectile loop, groups of NR projectiles are followed from collision to collision. The recoils generated along the projectile trajectories of the NR incident ions are collected and treated in a separate loop, the recoil loop. After finishing the calculation of the NR projectiles and generated recoils the target is updated in the case of the dynamic mode. In the static mode no

target update is necessary, and it can be continued with the next group of projectiles until the total number NH of projectiles (number of histories) is reached. Finally, the output section is entered.
In the input file the target and incident particles are specified. A flag determines the static or dynamic mode. In the dynamic case the total fluence for a calculation has to be given in units of 10^{16}atoms/cm^2. The energy of the incident particles, the angles of incidence, the interaction potential and the inelastic energy loss model have to be chosen. The energy and angle of incidence of the projectile can be chosen fixed or by a given distribution. The input file is organized as a F90 namelist file and described in detail in the documentation delivered with the program package.
The output was designed in a very flexible manner allowing to store all important values of individual particles and offering at the same time the possibility to limit the output in order to save memory and computing time. By conditioning the different output sections in the code the user can switch on or off the different sections with corresponding flags and variables in the input file. Moreover, the output is structured in such a way that the user can insert own output sections in an obvious manner.
The general, obligatory output gives the reflection and sputtering coefficients, atomic fractions and densities as a function of depth, and the yield versus the generation. In the dynamic case the change of target thickness and atomic fractions and densities as a function of fluence is given. This minimal output has a size of some kBytes only. Optional output concerns trajectory information (evolution of spatial coordinates, directions of motion, energy, time), particle information (energy, number of collisions, path length, starting point and final coordinate), matrices (absolute frequency distributions of reflected, transmitted and sputtered particles in discrete levels of energy and exit angles). Note, that the amount of output can increase rapidly to hundreds of MBytes for the trajectory and particle output, especially when the incident energy is high. Especially for problems with a large number of reflected, transmitted and sputtered particles, the usage of matrices output is advantageous as it helps to save memory. There are several post processing programs concerning the matrix output and the visualization of calculated data by means of IDL.

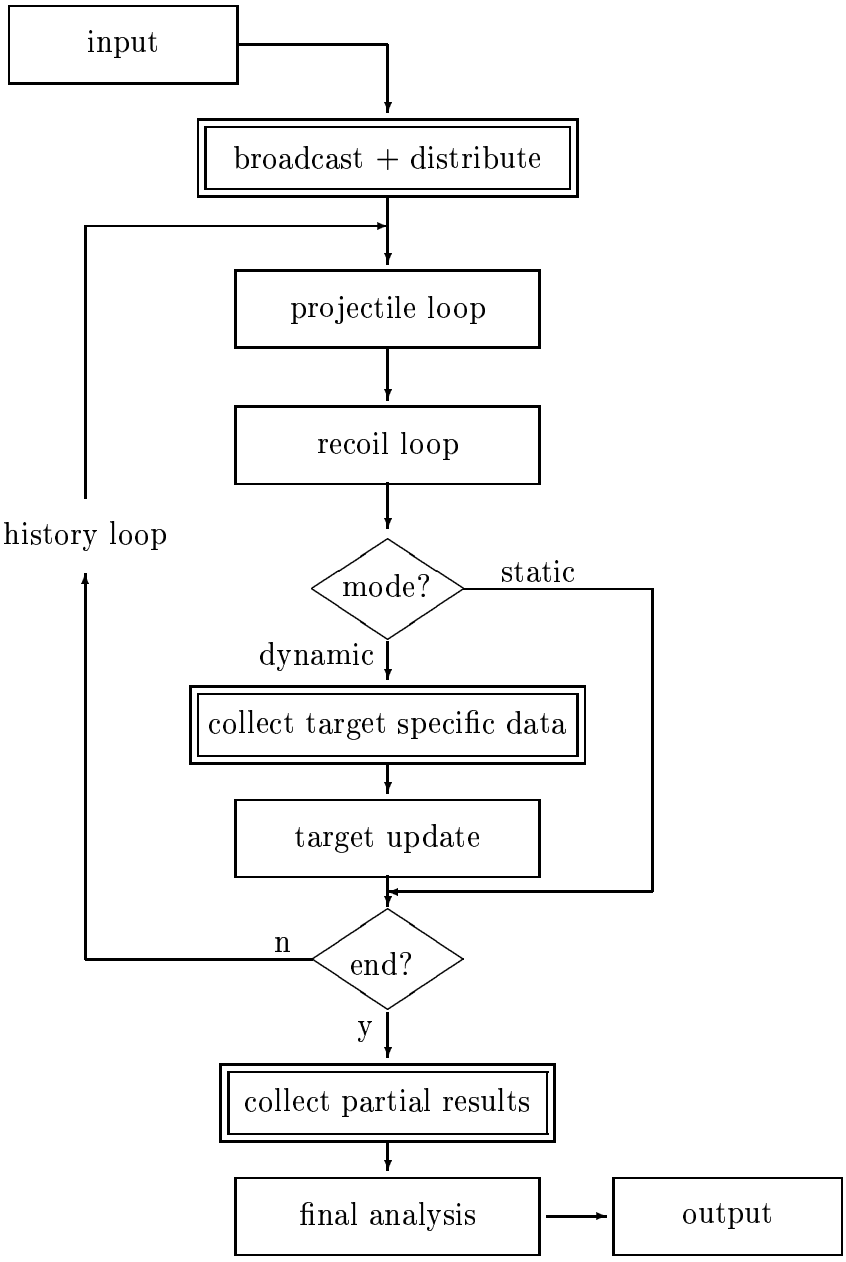

Figure 1: Main block flow chart of the program SDTrimSP. The double blocks indicate additional work necessary in the parallel mode. These parts are skipped in the sequential mode.

4. Implementation

The program SDTrimSP is implemented in Fortran 90. The work flow depicted in Fig. 1 is transferred into a modular structure of the code. The characteristic quantities belonging to larger data units like trajectories, particle states and other data blocks are combined in F90 user-defined structures to make data handling easier.
The code is drawn up to work in different modes and on different architectures. Basically, there are two modes:

1. the sequential mode for execution on any sequential architecture with a F90 and a C compiler,

2. the parallel mode for execution on any distributed-memory parallel architecture with a F90 and a C compiler and the MPI communication library available. In this version, the NR particle showers are distributed over the processors.

Both modes are included in the same source code, the different modes are distinguished by use of preprocessor directives. The user selects the mode at compile time by choosing the respective macro in the Makefile.
In the sequential mode the course of the program is as shown in blocks with simple boxes of Fig. 1. In the parallel mode some additional work as specified blocks with double boxes in Fig. 1 is necessary to distribute data structures and computational load and to summarize the results. The parallel algorithm works as follows: The target data is replicated on all processors (broadcast in Fig. 1), while the NR incident ions between two target updates are distributed over np processors (NR $\rightarrow$ NR/np). Furthermore, each processor has to be provided with an appropriate seed for the random number generator (see below). The ions and corresponding showers are simulated independently on the processors, each processor using a dedicated sequence of random numbers. The effects caused by the particles are recorded in processor private variables. This concerns target data, ion specific data and recoil specific data. In the dynamical case, the target data has to be summed up over the processors and made known to each processor in order to perform the target update. The target update is carried out quasi-sequentially on all processors and as a result each processor has a replicate of the new target data and can continue with the next group of ions and so on. In the static case the target update and the global sums are not necessary for the calculation and the summation of the target data is postponed to the end of the program where in any case all particle information gathered locally on the processors has to be collected and printed out. That means the static program is embarrassingly parallel with nearly no communication, while the amount of communication in the dynamic case can be considerable. Depending on the application the computing time may be rather long, therefore restart files can be written at regular time intervals.
The communication is based on the Message Passing Interface (MPI). By this the algorithm is portable between different parallel architectures. Special care has been taken for the generation of random numbers. By choosing the linear congruential random number generator from Cray's Scientific Library we have a true parallel random number generator having the advantage that the sequence of 2^{32} values can be divided into chunks

of equal size so that each processor has its own sequence which is not correlated to the sequences of all other processors. To facilitate debugging and providing reproducible results there is also the possibility to associate each particle with its own, determined seed so that the result of a computation with a certain number of incident particles is always the same irrespective of how many processors are involved. One must, however, be aware that this method of using reproducible random numbers does not yield reliable results in the sense of good statistics. Again, the mode of random number generation is controlled via preprocessor directives in the code and corresponding macros in the Makefile.

The calculation steps in the program SDTrimSP are determined by NR and NH. NR is number of projectiles between target updates, NH is the number of histories. In the dynamic case the target is relaxed after each history step.

Note that the structure of the whole package SDTrimSP is designed in such a way that the same source code, Makefile and run-time commands are used for all modes and architectures and distinctions are made via preprocessor and environment variables (e. g. OSTYPE). The object code is kept in different directories for the different architectures to facilitate the simultaneous usage of different architectures. A detailed description of the code with a list of all input and output variables and a description of all subroutines with references to the corresponding literature is provided. The code is available for free for non-commercial use. (contact mail-address: SDTrimSP@ipp.mpg.de or see the webside: www.ipp.mpg.de/~stel/SDTrimSP.html).

5. Performance

The program was tested on several sequential and parallel architectures, as e. g. IBM SP machines, IBM Power4 and Power5 systems, Cray T3E, NEC SX5, and Linux clusters with AMD or Intel processors, and is running in production mode for several years now with great success. For large, time-consuming applications it is advisable to use the parallel version of the code. In this case, the choice of the parameter NR, which is the number of incident particles and corresponding showers between two target updates, is decisive to have good performance, while in the sequential version, this parameter is of no relevance. The reason is that NR is a quantity closely related to the granularity of the parallel algorithm, as each processor has to treat NR/np incident particles together with their recoils, where np is the number of processors. That means, NR must not be less than np, and the larger NR, the better the efficiency of the parallel program, as the ratio between communication and computation decreases. On the other hand, NR has also a physical meaning, as it determines somehow the frequency of target updates. Therefore an investigation of the influence of NR and NH on the accuracy of the results has been carried out.

5.1. Influence of NR and NH on the accuracy of the results

A physical interpretation of NR and NH is that a larger number NR improves the statistical relevance of the target update, while a larger number of histories NH means a smaller fluence step (because the fluence step is the total fluence divided by NH) and by this inproves the overall statistics. In the static case, the statistics depends only on the product of NR and NH, and NR has no physical meaning.
With the following example of a dynamic case it is shown that the accuracy of the results depends merely on the product of NR and NH over a certain range of values for NR. The chosen example is a 1 keV bombardment of Fe on TaC with a fluence of 10^{17} atoms/cm^2. The results for different values of NR with NR $\cdot$ NH = constant are shown in Table 1 and Figs. 2 and 3 at the example of typical quantities. It shows that the plots and values differ only to the same extent as they would differ when using another seed for the random numbers. That means, the number NR may be increased to achieve a better parallel efficiency, while decreasing at the same time the number of histories. For statistical reasons it makes no sense to choose very small values of NH in the dynamic case. The fluence step (total fluence/NH) should of the order of 0.01 (10^{14} atoms/cm^2) to ensure that the target composition change is small in a fluence step.

NR	NH	change of thickness	qu(Fe)	qu(Ta)	qu(C)
1	1000000	4.47 nm	0.559	0.302	0.139
10	100000	4.51 nm	0.557	0.303	0.140
100	10000	4.50 nm	0.557	0.304	0.139
1000	1000	4.49 nm	0.558	0.303	0.139

Table 1: Change of thickness and atomic fraction (qu) of the surface composition with different numbers of NR and NH for the example of Fe $\rightarrow$ TaC (NR $\cdot$ NH = constant)

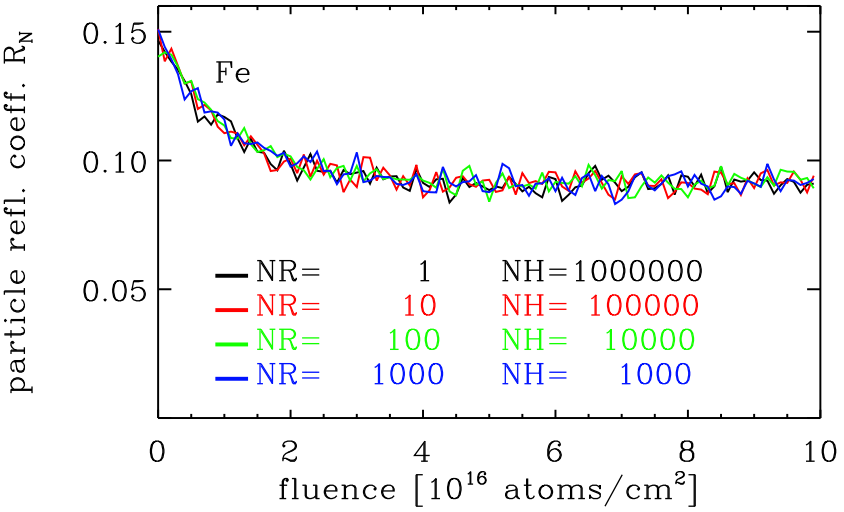

Figure 2: Particle reflection coefficient for different numbers of NR and NH ($NR \cdot NH = constant$) in the case of 1 keV Fe atoms impinging at normal incidence onto a TaC target

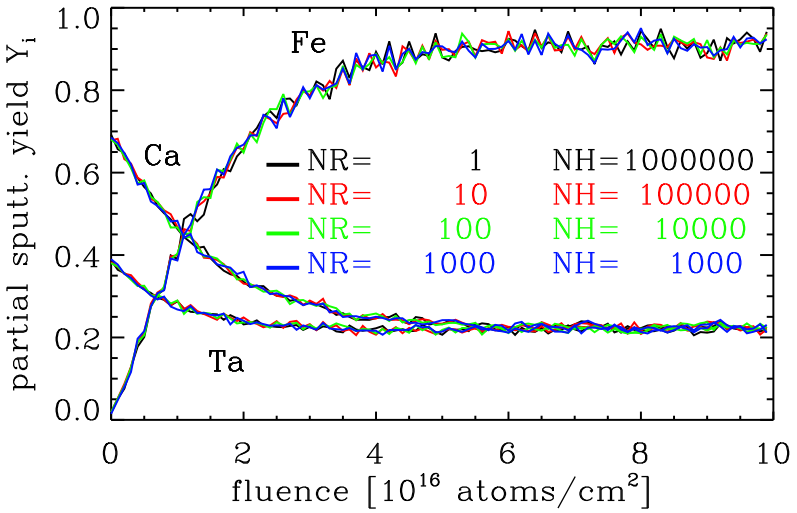

Figure 3: Partial sputtering yield for different numbers of NR and NH ($NR \cdot NH = constant$) in the case of 1 keV Fe atoms impinging at normal incidence onto a TaC target

5.2. Parallel efficiency

As already pointed out the performance of the program depends strongly on the mode and on the choice of NR on the one hand, but on the other hand on the characteristics of the used architecture. When working on a single-processor system, i. e. using the sequential version, it is mainly the clock rate of the processor which determines the computing time, irrespective of whether using the static or dynamic mode. The choice of NR is not of great importance. In the dynamic case, however, the code is rather communication-intensive, and the performance depends heavily on the choice of NR.
These dependencies are demonstrated at the example of 1 keV Fe bombardment on TaC at normal incidence with a fluence of 10^{16} atoms/cm^2. The benchmark has been carried out on two different parallel architectures, an IBM 1.3-GHz-Power4 (Regatta) system and a Linux cluster with Intel 2.8-GHz processors. The Regatta system is provided with a fast communication network with Federation switch, while the nodes of the Linux cluster are connected via Gigabit Ethernet. For these benchmarks the option of minimum output was used, the parameters NR and NH have been chosen as NR = 512 and NH = 20000 which allows to use up to 512 processors in the parallel mode.
The execution times and parallel efficiencies obtained for the static mode on the two architectures are shown in Table 2. As expected the parallel efficiency of the code is very good on both architectures, because the amount of communication is very low and consists mainly in broadcasting the data at the beginning of the calculation and summing up the partial results of the processors at the end of the calculation. This is also reflected in the corresponding speedup curves, cf. the solid lines in Fig. 4(b).
Table 3 and the dashed lines in Fig. 4(b) show the corresponding behavior for the dynamic case. There is a clear difference in the performance for the two architectures. The parallel efficiency obtained with the IBM Regatta is very good up to 64 processors. This is due to the fast communication achieved by the strong Federation switch of the Regatta system and the MPI implementation on top of the shared memory architecture of the Regatta. In contrast, the parallel efficiency on the Linux cluster is not that good. This is due to the fact that the communication network of the Linux cluster is rather slow compared to the processor performance and cannot cope with the amount of communication. The speedup curves demonstrate the somewhat poorer scaling and show that the Linux cluster is not specially suited for parallel calculations with more than 16 processors in the dynamic mode. Up to 16 processors, however, the performance of the Linux cluster is quite satisfactory, at least for the chosen value of NR = 512. It should be noted that the single-processor performance of the Linux cluster is much better than that of the IBM Regatta.
To conclude, the mode of the calculation, the choice of different parameters and the characteristics of the parallel architecture determine the efficiency of the calculation. To improve the performance in the dynamic mode it is advisable to reduce the communication by using a small number of NH and a large number of NR.

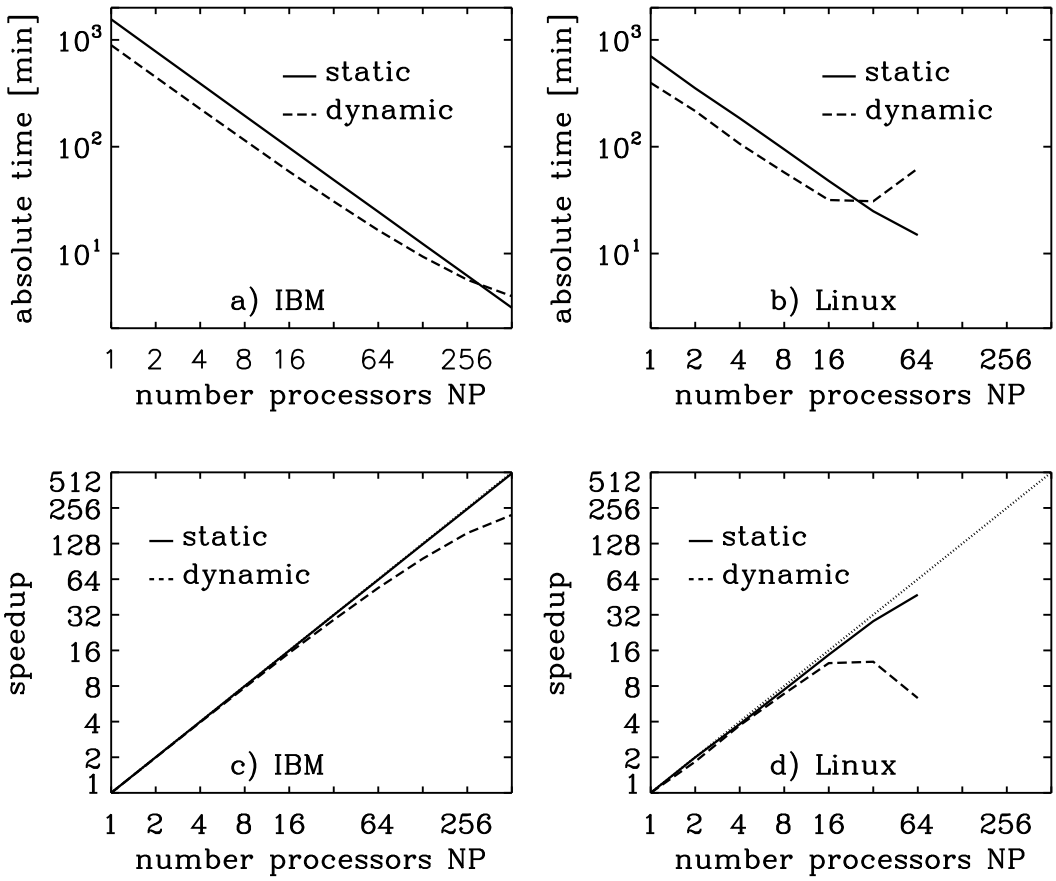

Figure 4: a, b) Absolute time and c, d) speedup of the static and dynamic cases in dependence of the number of processors for the IBM Regatta (IBM) and an Intel Linux cluster (Linux)

	IBM Regatta 1.3 GHz		Linux cluster 2.8 GHz (Intel)	
np	execution time [min]	parallel efficiency	execution time [min]	parallel efficiency
1	1557.38	1.000	703.84	1.000
2	778.61	1.000	352.47	0.998
4	389.96	0.998	184.78	0.952
8	194.81	0.999	94.40	0.931
16	97.65	0.996	47.86	0.919
32	48.87	0.995	24.90	0.883
64	24.86	0.989	14.89	0.738
128	12.31	0.988		
256	6.20	0.981		
512	3.11	0.978		

Table 2: Execution time and parallel efficiency of the static version of SDTrimSP on the IBM Regatta and on an Intel Linux cluster for the example of 1 keV Fe atoms impinging at normal incidence onto TaC with NR = 512, NH = 20000 (10240000 particles)

	IBM Regatta 1.3 GHz		Linux cluster 2.8 GHz (Intel)	
np	execution time [min]	parallel efficiency	execution time [min]	parallel efficiency
1	891.90	1.000	395.90	1.000
2	448.17	0.995	215.72	0.917
4	225.84	0.987	106.48	0.929
8	114.77	0.971	57.46	0.861
16	58.47	0.953	31.69	0.780
32	30.72	0.907	30.85	0.401
64	16.51	0.844	62.58	0.098
128	9.35	0.745		
256	5.68	0.613		
512	3.99	0.436		

Table 3: Execution time and parallel efficiency of the dynamic version of SDTrimSP on the IBM Regatta and on an Intel Linux cluster for the example of 1 keV Fe atoms impinging at normal incidence onto TaC with NR = 512, NH = 20000 particles

6. Special applications

The following examples illustrate the variety of possible applications and give an impression of the kind of dependencies which can be calculated. The different demonstrations proceed from simple examples to more elaborate cases.

6.1. Static mode

Trajectories As the first example a typical trajectory of a 2 keV He atom in a monoatomic Ni target is shown, see Fig. 5(a). The decreasing energy of the atom along its path through the solid is indicated by the color. The atom is stopped if its energy is smaller than the cutoff energy which is chosen to be 1.0 eV. In Fig. 5(b) the same trajectory as in the preceding figure is shown together with the generated Ni recoils. The recoils of the first generation are indicated in red, the recoils of the second generation in blue.

Energy and angular distribution In another example, Ni is bombarded with 1 keV Ar at 60° angle of incidence. The contour line plots in Fig. 6 show the angular distribution of the mean energy of the backscattered and sputtered atoms. The pictures illustrate the correlation between the mean energy and the emission angles. For backscattered atoms the highest mean energy is reached in the forward direction for nearly grazing exit angles, whereas for the backsputtered atoms the highest energy appears also in the forward direction, but for angles a somewhat larger than the specular direction. The lowest mean energy is reached in the backward direction (azimuthal angles larger than 90°) for both kinds of particles.
For the same example the contour line plots for the angular distributions of the intensity is shown in Fig. 7. The highest intensity for the backscattered particles is reached in the forward direction at an angle of about 75°, whereas for the backsputtered atoms a high-intensity ridge [19, 20] appears in the forward direction up to azimuthal angle of 60°. The highest intensity occurs in the forward direction at about 45°. The lowest intensity is in the backward direction for both kinds of particles.
For the same example the time-dependence of backscattered and sputtered atoms is presented in Fig. 8. Due to the lower energy of sputtered atoms (see Fig. 6) the maximum of the distribution appears later than that for backscattered atoms [16, 17].

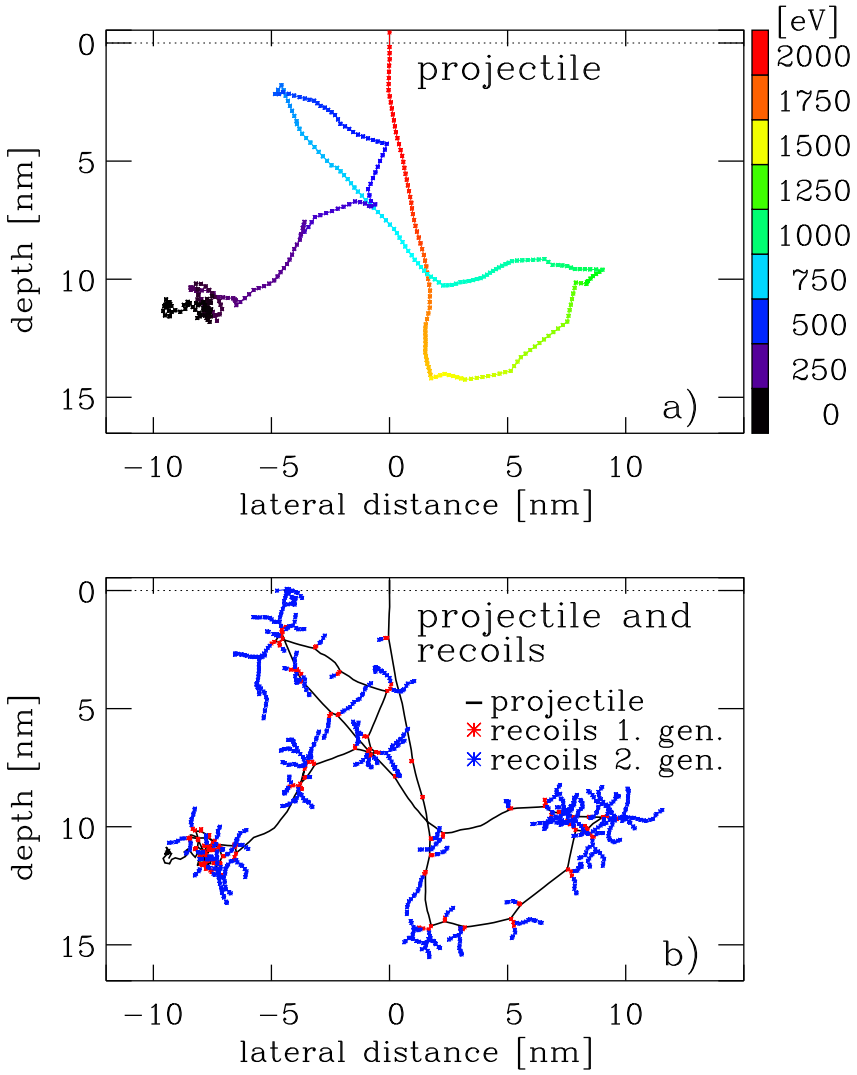

Figure 5: Trajectory of a 2 keV He atom penetrating a Ni target at normal incidence. a) The color indicates the decreasing energy of the He atom along its trajectory. b) In addition to the He trajectory also the generated recoils are shown. The color indicates the recoils of first generation (red) and the recoils of the second generation (blue).

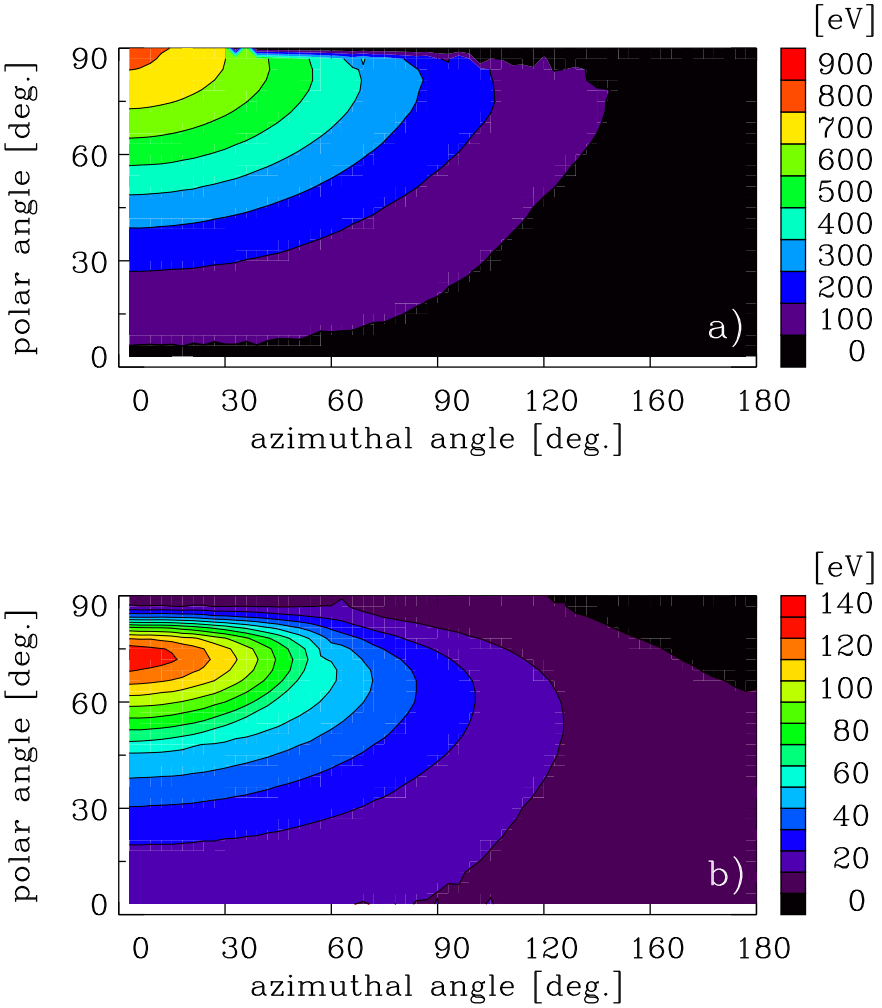

Figure 6: Contour line plot which represents the angular distribution of the mean energy of a) backscattered Ar atoms and b) sputtered Ni atoms. A Ni target is bombarded with 10^8 1 keV Ar atoms at 60° angle of incidence. The mean energy is indicated by the color.

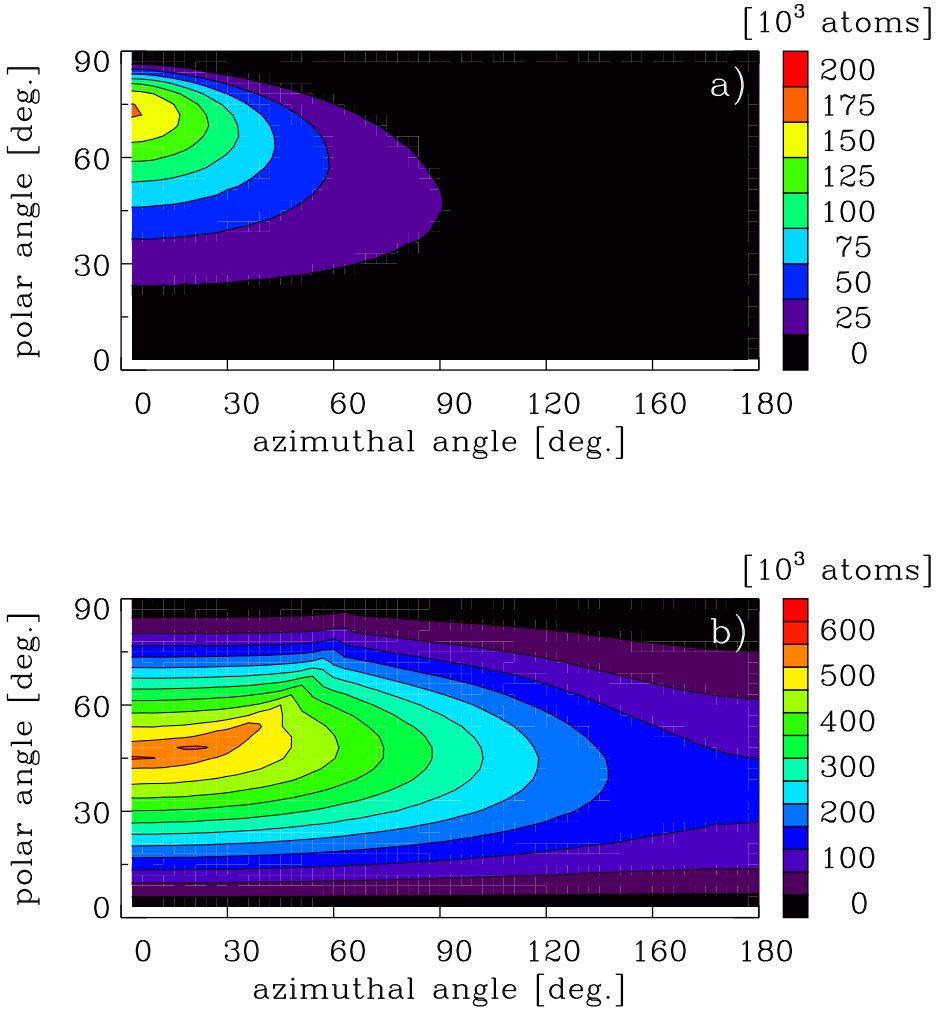

Figure 7: Contour line plot which represents the angular distribution of the intensity of a) backscattered Ar atoms and b) sputtered Ni atoms. A Ni target is bombarded with 10^8 1 keV Ar atoms at 60° angle of incidence. The intensity is indicated by the color.

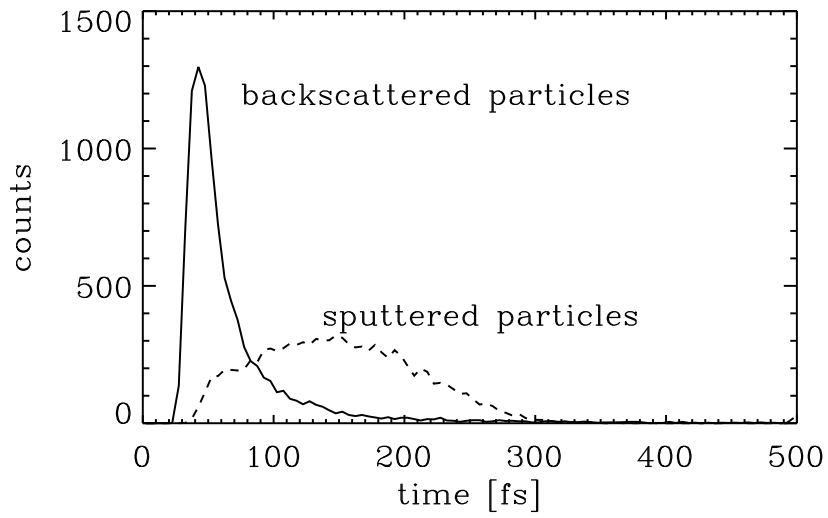

Figure 8: Counts of backscattered and sputtered particles dependent on time. A Ni target is bombarded with $3 \cdot 10^4$ 1 keV Ar atoms at 60° angle of incidence.

6.2. Dynamic mode

Backscattering coefficient, sputtering yield and atomic fraction Fig. 9 shows the fluence dependence of the particle backscattering coefficient, R_N, and the partial sputtering yields, Y_i, for the bombardment of the compound target WC with 10 keV Ni at normal incidence. At a fluence of about 10^{17} atoms/cm^2 R_N and Y_i become constant, which means that steady state or equilibrium is reached. The backscattering coefficient of Ni is decreasing with increasing fluence because some of the heavy W atoms are replaced by the lighter Ni atoms. This can be seen in Fig. 10, where the atomic fractions of the three species are shown versus depth at different fluences. The partial yield of Ni is increasing from zero (pure WC target) to a constant value. At steady state the amount of Ni in the target is not changing any more with fluence which means that $R_N + Y_{\mathrm{Ni}}$ must be unity. It can also be noted from Fig. 10, that the depletion of C in the target is larger than that of W. It is a well-known fact, that in many cases the lighter element in a multi-component target is preferentially sputtered. It should be remembered that diffusion and segregation effects are not included in the calculations.

Dynamic changes of the target composition Another interesting case is the bombardment of a target consisting of light atoms by heavy ions, in this case the bombardment of C by W at normal incidence. At the beginning of the bombardment, the target swells (positive value of surface position). This is a result of the deposition of W into the carbon target, which is larger than the sputtering of C. The composition of the target is changed particularly after a fluence of $5 \cdot 10^{16}$ atoms/cm^2. Therefore, the sputtering and reflection of W starts and the target shrinks (negative value of surface position),

see Fig. 11(a). The values of backscattering and sputtering change quasi-periodically according to the composition of the target, see Fig. 11(d). The peak of the partial yield of W (Y_{Si}) appears when the peak of the W implantation profile reaches the surface; the selfsputtering of W is much larger than the sputtering of C by W. The occurrence of further peaks is caused by the generation of further implantation profiles of W until they die out. After a fluence of $30 \cdot 10^{16}$ atoms/cm^2 a static state or equilibrium is reached and the coefficients R_N and Y_i get constant, see Fig. 11(b,c).

Target composition The program allows also layered target structures. As an example a target with several Si and Ta layers on Si is chosen, which is bombarded at normal incidence with 3 keV Ar.The oscillatory behavior of R_N and Y_i originates from the layered structure. For R_N the reason for the maxima is the higher reflection coefficient of Ar from Ta compared to that from Si due to the different mass ratio of target atom to incident ion. The peak of Y_{Si} at a fluence of about $5 \cdot 10^{17}$ atoms/cm^2 originates from the higher backscattering of Ar from the underlying Ta. Fig. 12 shows the broadening of the depth profile, the atomic mixing and the recoil implantation in the target. Again, in this example the lighter target element, Si, is preferentially sputtered. In this run, the implantation of Ar into the target is neglected.

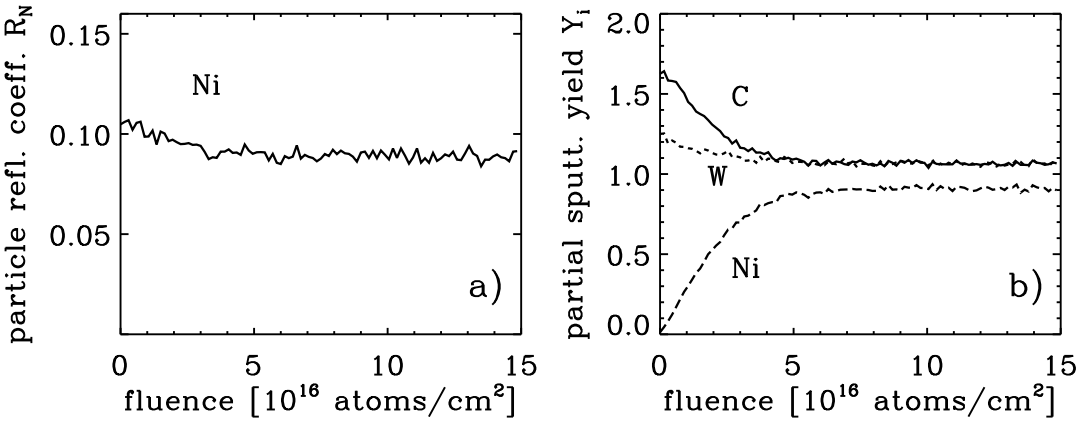

Figure 9: Fluence dependence of a) the particle reflection coefficient, R_N, and b) the partial sputtering yields, Y_i, by Ni on WC at normal incidence. A WC target is bombarded with 10 keV Ni.

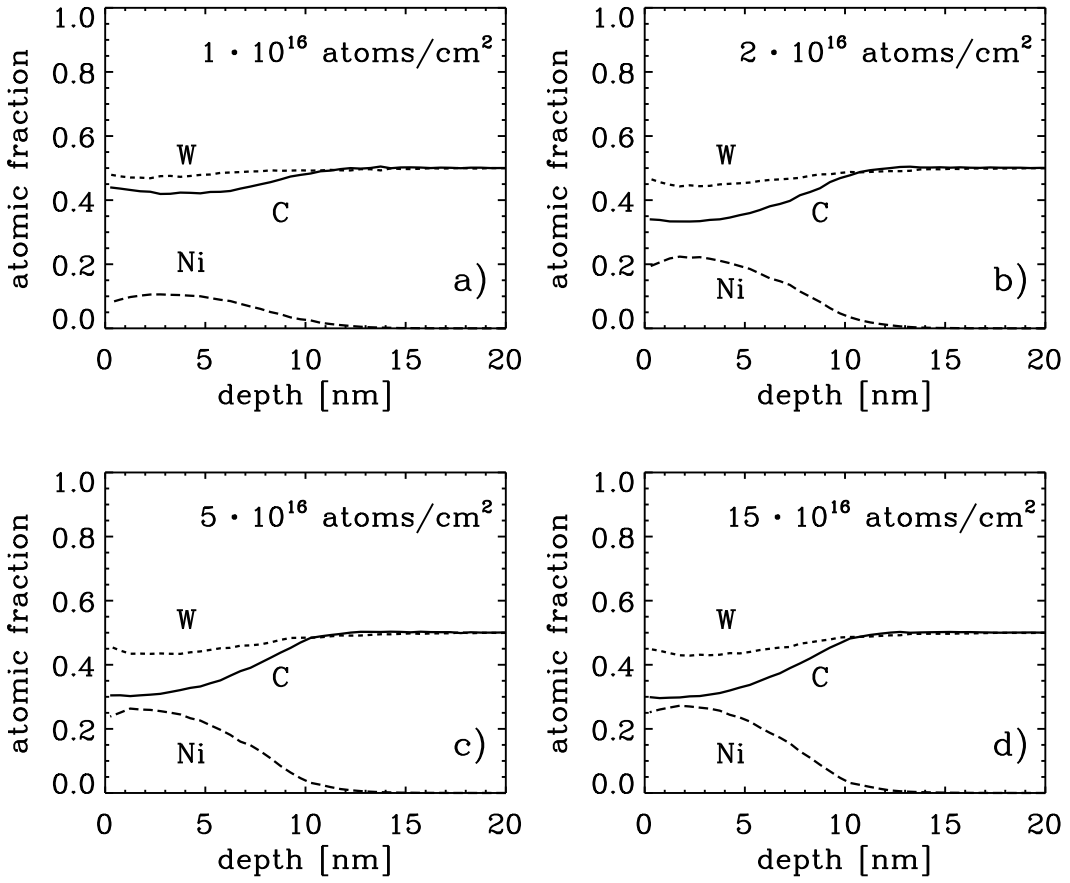

Figure 10: Atomic fraction of C, Ni and W versus depth dependent on fluence. A WC target is bombarded with 10 keV Ni at normal incidence.

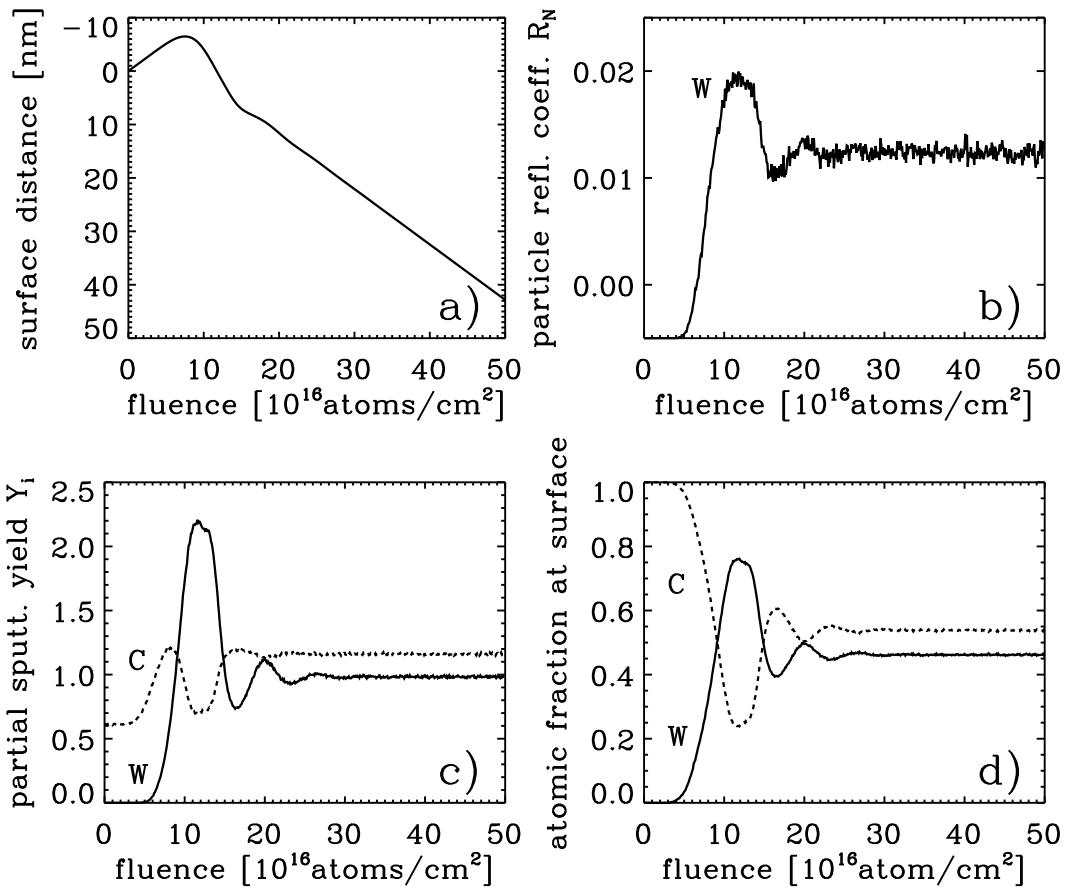

Figure 11: a) Surface distance, b) Particle reflection coefficient, R_N, of W, c) partial sputtering yields, Y_i, of C and W and d) atomic fractions of W and C for the bombardment of a C target at normal incidence with 5 keV W atoms.

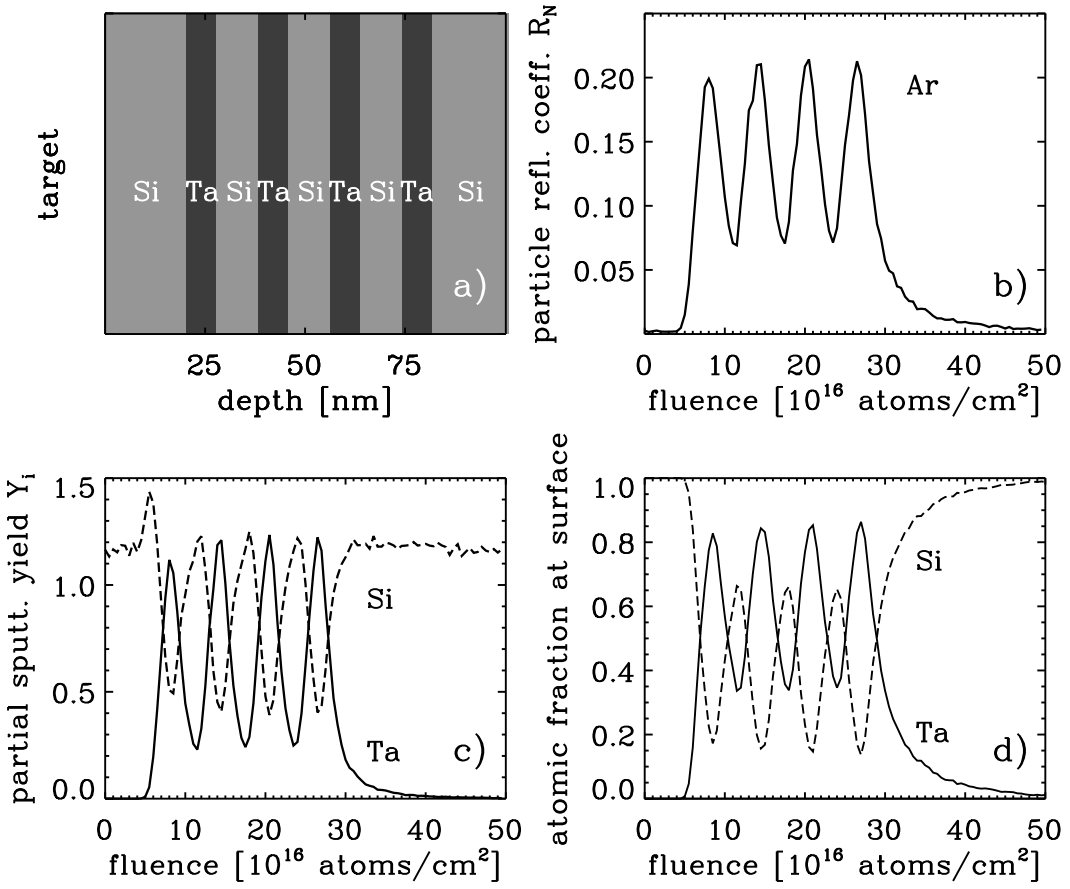

Figure 12: A Si(20 nm)[Ta(7.5 nm)Si(10.5 nm)]$_3$Ta(7.5 nm)Si target is bombarded by 3 keV Ar at normal incidence. a) Initial target composition, fluence dependence of b) the particle reflection coefficient, R_N, c) the partial sputtering yields, Y_i and d) the atomic fraction of Si and Ta at the surface (depth 0 - 0.5 nm).

7. Conclusions

The program SDTrimSP in its present form proves to be a very valuable tool for many ion-solid interactions as implantation, backscattering, transmission, sputtering and composition changes by ion bombardment. Further developments as an expansion to 2d and 3d target structures are in preparation. As these are much more memory consuming, an additional OpenMP parallelization for reduction of memory space seems to be the next step. This also fits the SMP architecture of modern supercomputers.

8. Acknowledgment

R. Schneider acknowledges funding of the work by the Initiative and Networking Fund of the Helmholtz Association.

References

[1] W. Eckstein, *Computer Simulation of Ion-Solid Interactions*, Springer Series in Material Science, Vol. 10, Springer Berlin, Heidelberg 1991

[2] J. P. Biersack, W. Eckstein, Appl. Phys. A 34 (1984) 73

[3] W. Möller, W. Eckstein, Nucl. Instrum. Meth. B 2 (1984) 814

[4] W. Möller, W. Eckstein, J. P. Biersack, Comput. Phys. Comm. 51 (1988) 355

[5] H. H. Andersen, J. F. Ziegler: In Hydrogen Stopping Powers and Ranges in All Elements, The Stopping and Range of Ions in Matter, Vol.3, ed. by J. F. Ziegler (Pergamon, NewYork, 1977)

[6] J. F. Ziegler: In Helium Stopping Powers and Ranges in All Elements, The Stopping and Range of Ions in Matter, Vol.4, ed. by J. F. Ziegler (Pergamon, NewYork, 1977)

[7] W. D. Wilson, L. G. Haggmark, J. P. Biersack, Phys. Rev. 15 (1977) 2458

[8] J. F. Ziegler, J. P. Biersack, U. Littmark: The Stopping and Range of Ions in Solids, The Stopping and Range of Ions in Matter, Vol.1, ed. by J. F. Ziegler (Pergamon, NewYork, 1985)

[9] G. Molière, Z. Naturforsch. A2 (1947) 133

[10] W. Eckstein, S. Hackel, D. Heinemann, B. Fricke, Z. Phys. D 24 (1992) 171

[11] J. P. Biersack, L. G. Haggmark, Nucl. Instrum. Meth. 174 (1980) 257

[12] K. Mehler, J. Reine Angew. Math. 63 (1864) 152

[13] Z. Kopal, *Numerical Analysis*, Chapman and Hall, London 1961, p. 367 ff.

[14] H. G. Schlager, W. Eckstein, IPP-Report 9/69, Garching 1991

[15] M. T. Robinson: Tables of Classical Scattering Integrals, Oak Ridge Natl. Lab., Oak Ridge, Tennessee (1970)

[16] R. Vichev, W. Eckstein, Nucl. Instrum. Meth. B 102 (1995) 272

[17] R. Vichev, W. Eckstein, Nucl. Instrum. Meth. B 122 (1997) 215

[18] W. Eckstein, R. Dohmen, Nucl. Instrum. Meth. B 129 (1997) 327

[19] R. Becerra-Acevedo, J. Bohdansky, W. Eckstein and J. Roth, Nucl. Instrum. Meth. B 2 (1984) 631

[20] W. Eckstein, Nucl. Instrum. Meth. B 27 (1987) 78

[21] S. T. Nakagawa, Y. Yamamura, Radiat.Eff.105, 239(1988)

[22] S. T. Nakagawa, Radiation Effects and Defects in Solids, Vol.116 (1991), page 21-28

A. Global parameters

parameter	value	description	program
ncpm	8	maximum number of elements	param.F90
nqxm	1000	maximum number of depth intervals	param.F90
time_between_r_files	18000	time in seconds between restart_files	param.F90
pemax	64	maximum number of PEs	work.F90
ntpmax	16384	size of global task queue	task_descr.F90
ntqmax	131072	size of local task queue	task_descr.F90
nhmax	8192	max num of trace records to be recorded	task_descr.F90

Table 4: Global parameters (set in programs)

B. Input variables in 'tri.inp'

B.1. Necessary input variables in 'tri.inp'

The sequence of the input values in the input file is arbitrary (namelist)

variable	description
alpha0(ncp)	angle of incidence (degree) of ncp species in case_alpha=0,5
e0(ncp)	energies (eV) of projectiles (qubeam $> 0.$) for case_e0=0 temperature (eV < 0) of projectiles for case_e0=2,3
flc	incident fluence (10^{16} atoms/cm^2 or atoms/A^2) in case of idrel=0
ipot	interaction potential: $= 1$: KrC $= 2$: Moliere $= 3$: ZBL $= 4$: Na-Ya $= 5$: Si-Si $= 4$: power

Table 5: Necessary input variables (no default values)

variable	description
isbv	surface binding model, determines the composition dependent surface binding energy sbv(ncp,ncp) from the elemental surface binding energies e_surfb(ncp) taken from table1 = 1 : sbv(ip,jp)=e_surfb(jp) for ip=jp, =0 else = 2 : sbv(ip,jp)=e_surfb(jp) for all ip, jp = 3 : sbv(ip,jp)=0., if e_surfb(ip)=0 or e_surfb(jp)=0 sbv(ip,jp)=0.5*(e_surfb(ip)+e_surfb(jp)) else = 4 : sbv(ip,jp)=f(e_surfb,qu,deltahf) for solid/solid compound = 5 : sbv(ip,jp)=f(e_surfb,qu,deltahf,deltahd) solid/gas compound
ncp	number of species (projectiles + target species) more than one projectile species is allowed
nh	number of histories (projectiles)
nqx	number of depth intervals of the target (discretization)
qubeam(ncp)	projectile atomic fractions (in incident beam) of ncp species, qubeam $> 0.$, Note: sum(qubeam(1:ncp))=1 qubeam $\leq 1.$ for projectiles qubeam $= 0.$ for target atoms
qu(ncp)	initial target atomic fractions of ncp species in case of homogenous initial composition (iq0 = 0)
symbol(ncp)	ncp chemical symbols of elements according to table1 (special symbol: 'H','D','T','He3','He','C_g', 'C_f','C_d')
two_comp	symbol of two-component target according to table.compound (e.g. two_comp ='Ta2O5') Note: only selected compounds in table.compound

Table 6: Necessary input variables (no default values) (continue)

B.2. Optional input variables in 'tri.inp'

These values have default values (see default_init.txt). If values different from the default values are needed, then these values have to be given explicitly in the input file. If values different from the default values are needed, then these values have to be given explicitly in the input file.

variable	default value	description
angleinp	'./'	directory of inputfile 'angle.inp' (see also: layerinp, tableinp, energyinp)
a_mass(ncp)	table	mass (in amu) of ncp elements; default from table1
a_num_z(ncp)	table	atomic number of ncp elements; default from table1
case_alpha	0	flag for the choice of the angle of incidence = 0 : angle of incidence (degree) counted from the surface normal (azimuthal angle phi = 0) alpha0 = 0... 90 (starting above surface) alpha0 = 90...180 (starting in solid) = 1 : random distribution of angles of incidence (only from above surface) (alpha and phi random) = 2 : cosine distribution of angles of incidence (only from above surface) = 3 : cosine distribution of angles of incidence alpha=0...$\pi/2$,max: by 0 phi=0...2π = 4 : input of a given incident angular distribution from file angle.inp = 5 : series of calculations with different angles of incidence (alpha= $(i-1)\cdot$alpha0; i = 1, number_calc) output : output.* dat default set :lmatrices = .false. ltraj_p = .false., ltraj_r = .false. lparticle_r = .false., lparticle_p = .false. case_e0 = 0 (note: all *.dat outputfile from last calculation)

Table 7: Optional input variables with default values

variable	default value	description
case_e0	0	flag for the choice of the incident energy = 0 : fixed incident energies(eV) of projectiles (qubeam>0) = 1 : input of a given energy distribution from file energy.inp = 2 : temperature (eV) of a Maxwellian velocity distribution of projectiles = 3 : temperature (eV) of a Maxwellian energy distribution of projectiles = 5 : series of calculations with different projectile energies e0(1)>0: linear energy= $i \cdot e0$; i = 1, number_calc e0(1)<0: logarithmic energy= $10^{(i-1)} \cdot e0$; i = 1, number_calc output: output.*dat default set: lmatrices = .false. ltraj_p = .false., ltraj_r = .false., lparticle_r = .false., lparticle_p = .false., case_alpha = 0 (note: all *.dat file from last calculation)
ca_scre(ncp,ncp)	1.	correction factor for the screening length in the interaction potential (not applicable for KrC and ZBL potentials)
charge(ncp)	0	charge of species if case_e0=2,3 and sheath>0 (plasma) $\geq$ 1. for qubeam>0 (projectiles) = 0. for qubeam=0 (target atoms)

Table 8: Optional input variables with default values (continue)

variable	default value	description
ck_elec(ncp,ncp)	1.	correction factor for the inelastic energy loss; correction factors for hydrogen (below 25 keV) are given in table3
deltahd(ncp)		heat of dissociation (eV) of a molecular target default from table1
deltahf		heat of formation (eV) of a molecular target default from table1
dist_nx	60	x-size of the matrix of energy distribution in target
dist_ny	60	y-size of the matrix of energy distribution in target
dist_nz	60	z-size of the matrix of energy distribution in target
dist_delta	2.0	distance between the matrix points of energy distribution in target
lenergy_distr	.false.	output of energy distribution in target (energy of stop, electric loss and elastic nuclear loss)
dns0(ncp)		atomic density ($atoms/A^3$) of ncp elements; default from table1
dsf	5.	average depth (A) for surface composition
e_bulkb(ncp)	0.	bulk binding energy; if e_bulkb>0., e_bulk has to be subtracted from the surface binding energy e_surfb
e_cutoff(ncp)		cutoff energy (eV) of ncp species; defaults from table1 (0.05 eV for noble gases; 1 eV for H, D, T; e_surf - 0.05 eV for selfbombardment)
e_displ(ncp)		displacement energy (eV); default from table1 (if in table1 e_displ=0 then e_displ=15)
e_surfb(ncp)		surface binding energy (eV) (heat of sublimation); default from table1
energyinp	'./'	directory of inputfile 'energy.inp' (see also: layerinp, tableinp, angleinp)
idrel	1	mode of simulation = 0 : full dynamic calculation (TRIDYN) > 0 : suppression of dynamic relaxation (TRIM), full static calculation < 0 : suppression of dynamic relaxation and cascades static calculation (TRIM) only projectiles (no recoils) are followed

Table 9: Optional input variables with default values (continue)

variable	default value	description
idout	-1	control output, determines the outputfiles: E0_31_target.dat, E0_34_moments.dat, partic*.dat, trajec*.dat and restart_file = -1 : output after each fluence step of nh/100, 100 fluence steps = 0 : output only after the last fluence step > 0 : output after each idout'th fluence step and last step
iintegral	0	integration method = 0 : MAGIC, only valid for KrC, ZBL, Moliere = 1 : Gauss-Mehler quadrature, ipivot ≥ 8 recommended = 2 : Gauss-Legendre quadrature, ipivot ≤ 16
imcp	0	flag indicating whether (flib)-moments of distributions are calculated = 0 : no moment calculation = 1 : moments of depth distributions for all projectiles (qubeam>0.)
inel0(ncp)	3	inelastic loss model = 1 : Lindhard-Scharff; nessary condition: $E < 25 \cdot Z^{4/3} \cdot M$ (in keV) where E, Z, M are the energy, the atomic number and the atomic mass of the moving particle = 2 : Oen-Robinson; nessary condition:$E < 25 \cdot Z^{4/3} \cdot M$ (in keV) = 3 : equipartition of 1 and 2 = 4 : high energy hydrogen (H,D,T) (energy > 25 keV) values from table3 = 5 : high energy helium (He3,He) (energy > 100 keV) values from table4

Table 10: Optional input variables with default values (continue)

variable	default value	description
ioutput_energ	0	energy in matrix = 0 : linear energy intervals = 1 : logarithmic energy intervals
ioutput_hist(6)	10	number of traced trajectories for: stopped, backscattered and transmitted projectiles, stopped, backsputtered, transmission sputtered recoils (see also: ltraj_p, ltraj_r)
ioutput_part(6)	10	number of traced particles for: stopped, backscattered and transmitted projectiles, stopped, backsputtered, transmission sputtered recoils (see also: lparticle_p, lparticle_r)
ioutput_polar	0	angle in matrix 0 : angle in degree intervals 1 : cosine intervals
ipivot	16	number of pivots in the Gauss-Mehler and Gauss-Legendre integration, the minimum number is 4 (larger numbers increase the computing time)
iq0	0	initial composition flag < 0 : initial depth dependent composition taken from file layer.inp = 0 : initial composition homogeneous, one layer with constant depth intervals
irand	1	random seed
irc0	-1	flag for subthreshold recoil atoms < 0 : subthreshold recoil atoms free ≥ 0 : subthreshold atoms bound
isot(ncp)	0	flag for isotope mass = 0 : natural isotope mixture (mass from table1) = 1 : isotope masses and natural abundances from table2 (valid for projectiles as well as for target species)

Table 11: Optional input variables with default values (continue)

variable	default value	description
i_two_comp	1	method to determine the densities dns0(:) from the compound density in a two-component target (table.compound) =1 : dns0 for the first target species is set equal to the elemental density; nessary if the second element is a gas (e.g. Ta2O5) =2 : dns0 for the second target species is set equal to the elemental density =3 : iterative determination of both dns0(:); recommended if the elemental densities are different
iwc=2	2	number of ring cylinders for weak simultaneous collisions for projectiles; for high energies (MeV H or He) iwc can be reduced to 1 or 0 to reduce computing time
iwcr=2	2	number of ring cylinders for weak simultaneous collisions for recoils
layerinp	'./'	directory of inputfile 'layer.inp' (see also: tableinp, angleinp, energyinp)
lmatrices	.false.	.true. : output of matrices, if idrel /= 0 .false. : no matrix output
lmoments	.true.	output of moments for energy distributions (linear and logarithmic) of projectiles and recoils and for range distributions (linear) of projectiles .true. : moments are written .false. : moments are not written
lparticle_p	.false.	.true. : output of projectile information .false. : no output of projectile information (see also: ioutput_part)
lparticle_r	.false.	.true. : output of recoil information .false. : no output of recoil information (see also: ioutput_part)

Table 12: Optional input variables with default values (continue)

variable	default value	description
ltableread	.true.	.true. : read from table1, table2, table3, table4 or table.compound .false. : no table read, a_num_z, a_mass, dns0, e_surfb e_displ have to be given *table*1 : chemical symbol (symbol), nuclear charge (a_num_z), atomic mass (a_mass), mass density, atomic density (dns0), surface binding energy (e_surfb), displacement energy (e_displ), cutoff energy (e_cutoff) *table*2 : chemical symbol, nuclear charge, isotope mass, atomic weight (in amu), natural abundance *table*3 : inelastic stopping coefficients for hydrogen: symbol, nuclear charge, inelastic stopping coefficients a1 to a12 (ch_h), ck *table*4 : inelastic stopping coefficients for helium: symbol, nuclear charge, inelastic stopping coefficients a1 to a9 (ch_he) *table.compound* : symbol of two-component target and physical values
ltraj_p	.false.	.true. : output of projectile trajectories .false. : no output of projectile trajectories (see also: numb_hist, ioutput_hist)
ltraj_r	.false.	.true. : output of recoil trajectories .false. : no output of recoil trajectories (see also: numb_hist, ioutput_hist)
lrestart	.false.	.true. : output of restartfiles after each idout .false. : no restart-files

Table 13: Optional input variables with default values (continue)

variable	default value	description
nm	-1	=-1 : not a molecular target > 1 : number of atoms in a two-component molecule
nr_pproj	10	number of projectiles between two target updates (idrel = 0)
matrix_e_min	0	minimum of lin. energy distribution in matrices
matrix_e_max	max(e0)	maximum of lin. energy distribution in matrices
numb_hist	20	number of traced trajectories of projectiles and recoils
number_calc	1	number of calculations if a series of calculations is carried out (case_e0 = 5 or case_alpha = 5)
quint	.false.	linear interpolation of atomic fractions between the depth intervals
qumax(ncp)	1.	maximum atomic fractions in the target for ncp species, if idrel=0
rhom		atomic density of a two-component target; default from table.compound $[g/cm^3]$
sfin	0.	$= 0$: no inelastic energy loss outside the target surface ($x = 0.$) $= 1$: inelastic energy loss outside the target surface ($-su > x > 0.$)
shth	0.	$= 0$: no sheath potential > 0 : sheath potential (eV), usually $= 3 \cdot \|e0\| \cdot$ charge, only if case_e0=2,3 (Maxwellian distribution, plasma)
tableinp	'../tables'	directory of inputfile for tables (see also: layerinp, angleinp, energyinp)
ttarget		total target thickness in Angstrom (A)
ttemp	0.	target temperature, only of interest at high temperatures, it reduces the surface binding energy according to a Maxwellian energy distribution
x0(ncp)	0.	starting position of projectile $\leq 0.$: outside the surface at $x = xc = -su$ $> 0.$: inside the solid

Table 14: Optional input variables with default values (continue)

C. Inputfile 'tri.inp' and 'layer.inp' of all examples

Inputfile 'tri.inp' of first static example He − > Ni

```
2 keV He − > Ni
& TRI_INP
        case_e0 = 0
        e0 = 2000, 0.00
        case_alpha = 0
        alpha0 = 0.000, 0.000
        ncp = 2
        symbol = "He", "Ta"
        idout = 10
        nh = 10
        nr_pproj = 1
        idrel = 1
        flc = 10.000E+0
        ipot = 1
        isbv = 1
        ttarget = 5000E+0
        nqx = 500
        qu = 0.0 , 1.0
        qubeam = 1.000, 0.000
        qumax = 0.000, 1.000
        lmatrices = .true.
        ltraj_p = .true.
        ltraj_r = .true.
        numb_hist = 1
        ioutput_hist = 1, 1, 0, 1, 1, 0
        lparticle_r = .true,
        lparticle_p = .true.
        ioutput_part = 100, 100, 0, 100, 100, 0
/
```

Inputfile 'tri.inp' of second static example Ar − > Ni

```
1 keV Ar − > Ni
& TRI_INP
        case_e0 = 0
        e0 = 1000, 0.00
        case_alpha = 0
        alpha0 = 60.000, 0.000
        ncp = 2
        symbol = "Ar", "Ni"
        nh = 10000000
        idout = 100000
        nr_pproj = 10
        idrel = 1
        flc = 10.000E+0
        ipot = 1
        isbv = 1
        ttarget = 5000E+0
        nqx = 500,
        qu = 0.0 , 1.0
        qubeam = 1.000, 0.000
        qumax = 0.000, 1.000
        lmatrices = .true.
/
```

Inputfile 'tri.inp' of first dynamic example Ar − > Ni

```
10 keV Ni − > WC
& TRI_INP
        case_e0 = 0
        e0 = 10000, 0.00
        case_alpha = 0
        alpha0 = 0.0 , 0.000, 0.000
        ncp = 3
        symbol = "Ni","W", "C_g"
        nm=2
        two_comp='WC'
        idout = 500
        nh = 50000
        nr_pproj = 32
        idrel = 0
        flc = 15.00
        ipot = 1
        isbv = 3
        inel0 = 3
        ttarget = 500
        nqx = 100
        qubeam = 1.000, 0.000, 0.000
        qumax = 1.000, 1.000, 1.000
        qu = 0.0, 0.5, 0.5

/
```

Inputfile 'tri.inp' of second dynamic example W − > C

```
5 keV W − > C
& TRI_INP
        case_e0 = 0
        e0 = 5000, 0.00
        case_alpha = 0
        alpha0 = 0.0 , 0.000
        ncp = 2,
        symbol = "W", "C_g"
        nh = 1000000
        nr_pproj = 32
        idout = 2000
        idrel = 0
        flc = 50.00
        ipot = 1
        isbv = 3
        inel0 = 3
        iwc = 0
        ttarget = 1000
        nqx = 100
        qubeam = 1.000, 0.000
        qumax = 1.000, 1.000
        qu = 0.0, 1.0
        lmatrices = .true.
/
```

Inputfile 'tri.inp' of third dynamic example Ar -> Si Ta

```
3 keV Ar -> Si Ta
& TRI_INP
        case_e0 = 0
        e0 = 3000, 0.00
        case_alpha = 0
        alpha0 = 0.000, 0.000,0.000
        ncp = 3,
        symbol = "Ar", "Si", "Ta"
        nm=3
        idout = 1000
        nh = 100000
        nr_pproj = 10
        idrel = 0
        flc = 50
        ipot = 1
        isbv = 3
        nqx = 263
        iq0 = -1
        qu = 0.0 , .5, 0.5
        qubeam = 1.000, 0.000,0.000
        qumax = 0.000, 1.000,1.000
/
```

Inputfile 'layer.inp' of third dynamic example Ar -> Si Ta

number of layer	thick-ness	target composition 2...ncp qu_2	qu_3	name of layer
40	5.00	1.0000	0.0000	Si 1
15	5.00	0.0000	1.0000	Ta 1
21	5.00	1.0000	0.0000	Si 2
15	5.00	0.0000	1.0000	Ta 2
21	5.00	1.0000	0.0000	Si 3
15	5.00	0.0000	1.0000	Ta 3
21	5.00	1.0000	0.0000	Si 4
15	5.00	0.0000	1.0000	Ta 4
100	5.00	1.0000	0.0000	Si 5
0	0	0	0	end