User Guide: *Kalypso* 1.0 and *Simulation Kit* 2.4

Software packages for molecular dynamics simulation of atomic collisons in solids

Marcus Karolewski

Author e-mail:
mkarol@ualberta.ca
karolewski@alum.mit.edu

Version 1 of this document, released April 2001.

The *Simulation Kit* and *Kalypso* are distributed from:
`http://www.ualberta.ca/~mkarol`.

1. Table of Contents

2. The *Simulation Kit* and *Kalypso*

This manual introduces two closely related simulation packages: the *Simulation Kit* (SK) and *Kalypso*. The main difference between them lies in the interaction models which they use. *Kalypso*, which uses many-body potentials, gives a more realistic modelling of metal lattices, but as a result is slower than the SK (which uses a Morse potential). As a rough guide, the SK should be used for keV ion scattering spectroscopy simulations, while *Kalypso* should be used for simulations of low energy collision processes (scattering, thin film growth) and sputtering where processes in the target are important. In general, input files developed for SK projects cannot be read by *Kalypso*, and vice versa (current exceptions: PRJ, IMP and INL files, but this may change in future).

Previous versions of the SK came with extensive online Help systems, but much material has been moved into the manuals. This manual starts by describing the various input files used for the calculations, and gives small exercises for the user. Afterwards, the preparation and analysis of a realistic simulation project is described on a step-by-step basis. You may also need to refer to the tutorial (`tutorial` directory) or the various UI (user interface) documents (`docs` directory) while you read through this guide.

No simulation package can, or should, protect the user from the physics underlying the calculations. In order to use the packages effectively, users must eventually become familiar with a substantial portion of the simulation and ion-surface literature. For example, it is the user's responsibility to select an appropriate inelastic energy loss model (or none at all): no computer program (or programmer) can make this decision on his behalf. An understanding of surface crystallography is also very important: for example, the user must be able to work out by himself the dimensions of a statistically representative sampling zone based on symmetry and experimental geometry.

Realistically, a considerable investment of time, and careful attention to detail, will be required in order to coerce these packages to your requirements. Students who are in a hurry to complete their theses should probably look elsewhere. It would be especially naive to expect that one could run a serious (research-level) simulation without reading all of the documentation.

The most difficult part of a simulation is setting up the input files, which is achieved by the *Spider* utility (different versions exist for SK and *Kalypso*). Here are some tips, which new users are strongly recommended to follow which creating these files.

- First, use the same root for all 5 input filenames if possible (TEST.TRG, TEST.PRJ etc.), and create a new directory to store them in.

- Second, begin at the leftmost menu (Target) and work your way to the right (Inelastic). You can create new input files using the New menu options, and modify saved files using the Open menu options. [There is no Open option for Target files. Note also that the Open option for Impact files will not honour any changes that you might have made to the body of the file. You cannot Open an Impact file if you have previously deleted the header information on line 1.]

- Third, work slowly and make good use of the context-sensitive on-line Help. For example, if you position the cursor on an edit box, you can hit F1 to get more information about the input which is required (and, where relevant, a list of representative data).
- Finally, document your serious simulations properly in a notebook, save all input files, and make a note of the simulation options which you choose from the Options menu in *Snook* or *Kalypso* (or save the CFG file which is created when you save your simulation options to disk).

In the author's experience, most time is wasted through incorrect specifications of (a) simulation geometry (particularly the shape/dimensions of the impact zone) and (b) the timestep (if in doubt, set it too small, rather than too large).

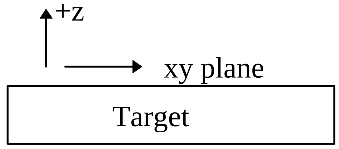
3: Target files and projectile files

3.1. Function of the target file

Target files, which have the extension .TRG, store information about the target lattice. To generate a target lattice within *Spider*, select Target|New[1] then (for example) Face-centred cubic|(100) surface. This selection will allow you to generate a fcc target whose *xy* surfaces have a (100) orientation. Once you understand the file format (section 3.4), you can also generate your own TRG files (via a spreadsheet, computer program etc.).

3.2. Coordinate system

Coordinates in a TRG file are expressed in Ångstroms. The *z*-direction corresponds to the surface normal. By default, the surface plane coincides with $z = 0$, although this is not mandatory. The positive z-direction is directed away from the target surface, while the negative *z*-direction is directed into the surface.



Coordinates in a TRG file correspond to the laboratory system of coordinates. For off-normal incidence, the projectile approaches the target in the *xz* plane from the +*x* direction with a negative *x* velocity ($v_x < 0$, $v_y = 0$, $v_z < 0$) by default. Thus the *x*-direction of the coordinate system and the azimuthal direction of projectile incidence which you require for your simulation should coincide.[2] This condition can be achieved (if necessary) by changing the azimuthal orientation of the lattice (i.e. rotating it), as discussed later in the chapter. Although it is possible to achieve the same effect by changing the default azimuthal direction of projectile incidence, the author does not recommend this.

3.3. Anchor atom

The first atom specified in the TRG file is known as the **anchor atom**. *Spider*'s lattice generation options will assign this atom to the position (0,0,0) by default. When you later define the starting coordinates of the projectile (in the impact file), these will be expressed relative to the position of the anchor atom, for example:

Anchor atom: (0.0, 0.0, 0.0)
Impact file: (0.0, 0.0, 3.0)
Projectile initial position (normal incidence): (0.0, 0.0, 3.0)

Anchor atom: (0.0, 2.5, 3.2)
Impact file: (0.0, 0.0, 3.0)
Projectile initial position (normal incidence): (0.0, 2.5, 6.2)

---

[1] The expression 'select Target|New' means 'select menu command Target, then sub-menu command New'.

[2] This point is only relevant for non-normal directions of projectile incidence.

3.4. Format

The following excerpt from a target file illustrates the format:

```
 0.00000   0.00000  0.00000  29  63.54600 0 Cu
10.22392  10.22392  0.00000  29  63.54600 0 Cu
```

Columns 1-3: (*x*, *y*, *z*) coordinates (real numbers)
Column 4: Atomic number (integer)
Column 5: Atomic weight (real number)
Column 6: Flag parameter (integer)
Column 7: Label (normally chemical symbol, any 2 characters)

All numeric data are space-delimited (i.e. the columns must be separated by 1 or more blank spaces, but do not have to be aligned). Real numbers can be entered in a variety of formats (32.0, 32, 3.2E1, 3.2e1 etc.) The symbol label ('Cu') should preferably be separated by only one space from the flag parameter (any extra spaces will be read as part of the label, but no error will result).

TRG files can be modified in any text editor. The only thing to be careful about is the end-of-file position: if you create TRG files yourself using a text editor make sure that the cursor is flush with the left-most margin of the last (empty) row when you save the file (i.e. no blank spaces after the last linefeed character). Except for the anchor atom, the order of the atoms in the file does not matter. Chemical symbols are included only for the user's information - they do not affect the calculations in any way.

3.4. Flags parameter

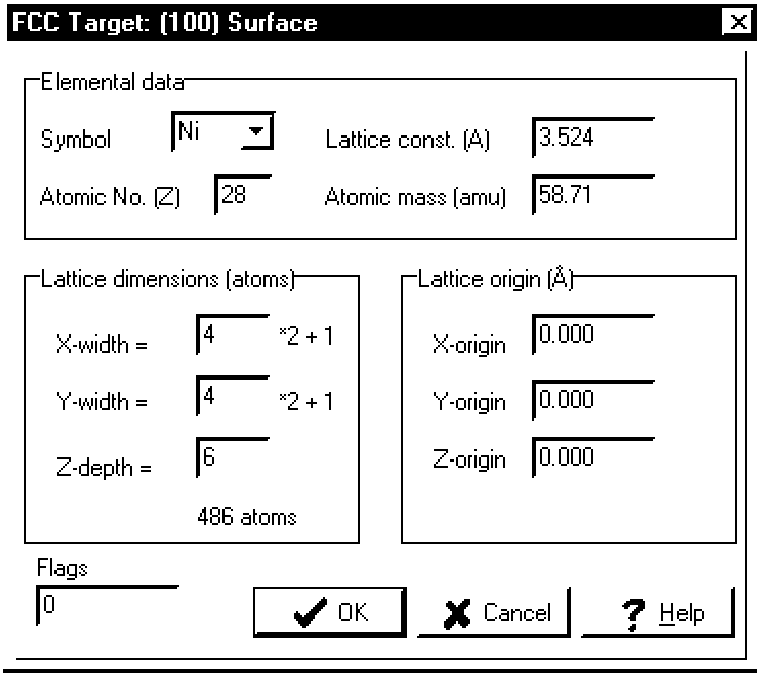
The flags parameter (FP) is used for various purposes which are described in a dedicated chapter of the Simulation Primer which ships with the *Kalypso* and SK packages. Generally it can be ignored by users of the SK, but is needed for *Kalypso* (see note below).

Note for Kalypso users only
For 99+% of *Kalypso* simulations, the only use of the FP is to distinguish the two kinds of atoms (type A, type B) which are present in a binary compound target. For a binary compound target you must always set the FP correctly. For an elemental target, leave the FP at its default value (0). For a binary compound target, the FP takes one of two values (0 or 1 for atom types A and B respectively). For example, in the Cu/Ni(100) system, the Cu atoms are labelled with a FP = 0 (type A), while FP = 1 (type B) for Ni atoms. Your type A/type B designations in the TRG file and in the model file should be mutually consistent so that potentials are applied correctly. The anchor atom should be of type A.

3.5. Exercise 1: Generate a small (486-atom) Ni(100) lattice

Select from the menu Target|New|Face-centred cubic|(100) surface. From the drop-down symbol box select 'Ni'. The data shown in the table below come up in the dialog box. All data input fields (including the symbol) can also be edited manually.

Symbol	Ni	Lattice const.	3.524
Atomic No.	28	Atomic mass	58.71
X-width	4	X-origin	0
Y-width	4	Y-origin	0
Z-depth	6	Z-origin	0
Flags	0		



Symbol: Ni
Lattice const. (A): 3.524
Atomic No. (Z): 28
Atomic mass (amu): 58.71
X-width = 4 *2 + 1
Y-width = 4 *2 + 1
Z-depth = 6
486 atoms
X-origin: 0.000
Y-origin: 0.000
Z-origin: 0.000
Flags: 0

Click OK to generate the lattice, and save it in a file which you specify (e.g. 'Ni100.trg'). Examine the file in a text editor (e.g. the Editor|Open option in *Spider*). There are 81 atoms per layer in this target file. The first line is the anchor atom, followed by other atoms in the outermost layer of the lattice (z = 0):

xy z *Z* mass FP symbol

```
0.00000 0.00000 0.00000 28 58.71000 0 Ni<-- 'anchor atom'
9.96738 9.96738 0.00000 28 58.71000 0 Ni
9.96738 7.47553 0.00000 28 58.71000 0 Ni
```

The second layer, identified by its *z* value (-1.762 Å), begins at line 82:

```
11.21330 11.21330  -1.76200 28 58.71000 0 Ni
11.21330  8.72145  -1.76200 28 58.71000 0 Ni
11.21330  6.22961  -1.76200 28 58.71000 0 Ni
```

If you examine the target file further, you will notice that the atomic positions in each layer (except for the anchor atom) are sorted systematically according to their *x* values.

3.6. Exercise 2: Modify target file

Suppose you wanted to apply an *inward* relaxation of 0.1 Å on the outermost layer of the Ni target (relative to the ideal positions). One way to proceed with the required modifications to the TRG file is to use a text editor, and carry out the following operation, or its equivalent, then save the result:

REPLACE: `'0.0000028 58.71000'` WITH: `'-0.100028 58.71000'`
This operation will only modify the *z* values of the first layer. If you did a search-and-replace on the string '0.00000' by itself, you would also modify any *x* or *y* coordinate with the same value.

3.7. Exercise 3: Target file for (1×1) metal monolayer system

The simplest type of metal monolayer is the pseudomorphic (1×1) type, e.g. Cu/Ni(100), where to a first approximation the Cu overlayer simply replaces Ni atoms in the surface layer. To create such a target, we begin by creating a target file 'Cu.trg' which consists of a single layer of Cu atoms based on the Ni lattice constant:

Symbol	Cu	Lattice const.	3.524
Atomic No.	29	Atomic mass	63.54
X-width	4	X-origin	0
Y-width	4	Y-origin	0
Z-depth	1	Z-origin	0
Flags	0		

A second target 'Ni.trg' constants of a Ni crystallite as before, except that the *z*-origin of the lattice (i.e. the lattice surface layer) is shifted by one interlayer distance (to -1.762 Å), and the Flags parameter is changed to 1 to indicate a different atomic type:

Symbol	Ni	Lattice const.	3.524
Atomic No.	28	Atomic mass	58.71
X-width	4	X-origin	0
Y-width	4	Y-origin	0
Z-depth	6	Z-origin	-1.762
Flags	1		

The two files are then spliced together as one file 'CuNi100.trg' using an editor (copy/paste) or the Dos command: `copy Cu.trg + Ni.trg CuNi100.trg`.

Other types of target can be quite tricky to create using *Spider*. A knowledge of basic surface crystallography is necessary. The example of a c(2x2) structure on a (100) fcc surface is dealt with in the next section.

3.8. Orienting targets

The Target|New command generates a lattice with a specified surface orientation in the *z*-direction. Often, however, you may have to modify the orientation of the lattice in the (*x*, *y*) plane. Table 1 shows the default orientations generated by *Spider*'s Target|New… option. The online Help file ('Default Azimuthal Orientations' topic) and the file `spider.pdf` have pictures of the default lattice orientations.

Target lattice orientations in (*x*, *y*) plane generated by *Spider*.

Lattice type	Orientation	y	x
fcc	(100)	[011]	[01-1] [b]
fcc	(110)	[-110]	[001]
fcc	(111)	[-1-12] [a]	[-110]
bcc	(100)	[-100]	[010]
bcc	(110)	[001]	[-110]
diamond & ZnS	(100)	[011]	[01-1]
diamond & ZnS	(110)	[-110]	[001]
diamond	(111)	[-1-12] [a]	[-110]
hcp	(0001)	[-1010]	[-12-10]

[a] The edges of these lattices actually lie at 60º to the *x*-axis, not parallel to the *y*-axis, and they are also <110> type edges.

[b] The notation [01-1] is used in this manual to mean $[01\bar{1}]$.

You can change the orientation of a target file using the Target|Display/Orient menu command. This brings up a dialog box with a display of your lattice. The detailed instructions for using this utility are found in the online help, which is context sensitive. Some of the main features are:

1. There are options for displaying TRG files, as well as re-orienting them.
2. The orientation options can be tested without committing changes to disk.
3. Orientation options include the ability to rotate, clip (cut edges off) and translate lattices.
4. Display options allow you to view the lattice from different directions, zoom in/out, translate and rotate.
5. There is an option to superimpose coodinates from another target file or an impact file on the current display. This allows you to check, for example, that the impact file correctly reflects surface structure and symmetry.

A very frequent, operation is to create a fcc(100) target with <001> edges: this example is discussed in the online Help, and involves applying a rotation of 45° in the (*x*, *y*) plane to the <011> terminated default lattice, followed by a trimming operation that reveals the <001> edges. You can also write a computer or spreadsheet program which achieves the same goal. More complex operations can be envisaged (e.g. creating a Cu lattice which exposes the (210) surface), but in practice these will rarely be used.

It is a good idea to start off with large target files, then trim them down to your requirements when all operations have been completed. Otherwise, you may find that the rotated overlayer does not cover the corners of your substrate lattice, and you will have to start all over again.

Exercise 3 showed how to create the target file for a (1×1) metal overlayer system. On occasion you may have to work with a more complex system, for example, a c(2×2) overlayer on a (100) fcc surface, more properly known as a $(\sqrt{2}\times\sqrt{2})R45°$ overlayer. The latter is equivalent to a fcc (100) layer which is rotated by 45° with respect to the substrate, and which has a lattice constant expanded by a factor $\sqrt{2}$ relative to that of the substrate (a). Thus, one way of creating such an overlayer is as follows:

1. Create a target file with 1 atomic layer, and a lattice constant $\sqrt{2}a$.
2. Use the Orient/Display utility to rotate this layer by 45° with respect to the substrate.
3. Translate the x and y coordinates by an amount $a/2\sqrt{2}$, to place the overlayer atoms in the substrate hollow sites.
4. Combine the rotated target file with that for the substrate, as in example 3.
5. Use the Orient/Display utility to trim off unwanted edges of the composite target.

This example illustrates the importance of understanding the surface crystallography of your simulation problem.

3.9. Choosing the target size

For a keV sputtering simulation you will need a target with at least 1500 atoms. A more typical number these days would be 3000-5000 atoms. In small targets, the failure to contain collision cascades laterally may introduce errors in the sputter yield data.

For an ion scattering spectroscopy (ISS) simulation, the 'lattice' will consist of 2-20 atoms typically, or as few as is required to illustrate the physics of the problem. One reason for this is that ISS simulations run for no more than 10 fs, which means that interactions between target atoms can be neglected. Furthermore, ISS data require only realistic modelling of the hard, short-range interactions experienced by the projectile in the topmost 1 or 2 surface layers. For serious ISS simulations you should always use the SK, since *Kalypso* is much slower than *Snook* for this type of problem.

The CPU time required for simulation is roughly a linear function of target size. For ISS simulations, a careful choice of the termination time can also reduce the CPU time. However, in a sputtering simulation there is little to gain from reducing the termination time from 1000 fs to 500 fs.

3.10. Projectile files

The projectile file (extension PRJ) contains the projectile characteristics: symbol, atomic number and mass, energy (in keV) and the projectile flags (see above and chapter 9 of the Simulation Primer). For most purposes you can ignore the projectile flags. They are only useful if you want to simulate a system in which the projectile is of the same atomic type (type A or B) as one of the atoms in the target. This is termed a 'self-bombardment' experiment. In this case, if you select the 'self-bombardment' option in *Snook*/*Kalypso*, you can use a many-body potential for the projectile-surface interaction (the default is screened Coulombic potential at all distances). The flags parameter should have the same value (0 or 1) as the corresponding atom type in the target file.

The projectile file data is thus very simple, and little further explanation is required. Just click Projectile|New in *Spider* and enter the projectile characteristics. However, note that several simulation parameters defined in other files depend on the projectile's atomic attributes. These dependencies are listed below for reference. As a result, changing the projectile atomic species for a new simulation (e.g. from Ar^+ to He^+) is not as trivial as it may seem, because parameters must be updated in several files, as the next table shows.

Simulation parameter(s)	*File*	*Remarks*
Mass, atomic number, energy	Projectile	
Timestep	Run	Depends on projectile velocity
Inelastic energy loss models	Inelastic file	Depend on projectile mass, atomic number

4: The Run file

4.1. Function

A simulation follows the evolution of the atomic system from its initial conditions. The simulation of events following each incident projectile trajectory is described as a 'run'. For example, a sputtering simulation might consist of 1000 runs, each lasting for 1000 fs. The initial conditions at the start of each run will be similar, except for the starting position (impact parameter) of the projectile, which will sample different impact points within the surface unit cell (see Chapter 6).

The Run file (extension RUN) defines (a) the kind of output data which the simulation will generate, (b) some miscellaneous parameters needed to define the initial and termination conditions of the simulations, (c) projectile incident angle characteristics. The items (c) should logically be found in the Projectile file, but for some historical reason that now eludes me they have found their way into the Run file.

4.2. Output data

There are two aspects to defining the output characteristics: *what* output to write, and *when* to write it. Typically, for sputtering or ISS simulations we need only write output data at the termination of each run. There may be situations, however, when you need to record output periodically during a run (if you want to examine the development of a collision cascade, for example). Either or both options is allowed. In the latter case, you must also specify the frequency (in timesteps) of the write operation. Your choice(s) should be entered into the 'Sampling Frequency' box of the run file dialog.

[Note: *Spider* for *Kalypso* offers an extra option labelled as 'Limit 1 record per atom', which is rarely used. This means output data will only be written once, if at all, for each atom. For example, you might wish to record the first occasion on which a sputtered atom leaves the surface region, and ignore that atom subsequently.]

4.3. Specifying output data

Output data consists of dynamical variables information for some or all particles in the system. The data are written to disk in the form of a record which contains the following information (one record per particle recorded):

- rx, ry, rz = particle coordinates
- px, py, pz = particle momentum coordinates
- ti = time elapsed
- rw = row number, or row occupied by particle in the target file (a value of 0 is assigned to the projectile)
- rn = run number, or row occupied by current run in the impact file

These records are stored are stored in a binary file ('*Snook* file') with a name *.snk (by default called `dynvars.snk`). All dynamical variables are stored in SI units (kg m s). The `source\winnow` directory contains small programs (with source code) that show how to read these files programatically.

The user is required to specify what kind of data should be recorded by the simulation. The purpose of these options is to avoid unnecessarily writing data to disk that won't be used later.

For example, if the purpose of the simulation is to calculate sputtering coefficients, you can simply record data for ejected particles. If your simulation is modelling ion scattering processes you can record data for the projectile only. However, it is also possible to filter data (using *Winnow*) after completion of the simulation if disk space is not a restriction for you.

There are several standard output choices, one of which must be selected by clicking the appropriate button. Each of the options records dynamical variables information in a standard record format ('SNK' record). The difference between the options is that they record the information for a subset of particles whose momentary dynamical characteristics satisfy certain conditions. Once again, the purpose of having these options is to avoid recording data irrelevant to the purpose of the user.

4.4. Output options

- Projectile only: always records data for the projectile, but no other particles.
- All particles with KE > 1 eV: records data for particles with kinetic energies above 1 eV.
- All particles with KE > 10 eV: records data for particles with kinetic energies above 10 eV.
- All particles with KE > 100 eV: records data for particles with kinetic energies above 100 eV.
- All emitted particles: records data for all particles with rz > 0.5 A, and vz > 0.0, including the projectile.
- Projectile emission: records data for the projectile only, provided z > 0.5 A, and vz > 0.0.
- All particles emitted within 5 degrees of surface normal: similar to the 'All emitted particles' option; however, this option only records data for those particles emitted within a 5 degree cone around the surface normal.
- User-programmed option: see below.

4.5. User-Programmed Option

The User-Programmed Option is an advanced option which allows the user to specify a condition which must be fulfilled by a particle's dynamical variables before its data are recorded. The conditional expression specified by the user will be interpreted by *Snook* at run-time.

The language in which conditional expressions are framed is the same as the query language used by *Winnow*. For a formal definition, see the file `winnow.pdf` found in the `\docs` directory.

As a simple example, the following expression will achieve the recording of data only for particles which leave the surface (arbitrarily bounded at 0.5 Å):

```
 [rz > 5.0e-11] & [vz > 0.0]
```

Here rz represents the particle's z-coordinate, while vz represents the z-component of its velocity vector. The '&' symbol is the logical AND operator. The output produced by this specification is identical to the 'All emitted particles' option described above. In pseudo-code this specification is programmed as follows:

```
for n = 0 to NATOMS do
  begin
    if (rz[n] > 5.0e-11) AND (vz[n] > 0.0) then
      WriteToFile(rx,ry,rz,px,py,pz...)
  end;
```

Likewise, the expression: `[ke/ep > 100.0]` is only satisfied if a particle's kinetic energy (`ke/ep`, where `ep` is the proton charge) is greater than 100 eV. Therefore the expression specifies that data are only to be recorded for those particles with more than 100 eV kinetic energy. In psuedo-code:

```
for n = 0 to NATOMS do
  begin
    KE = (sqr(px[n]+sqr(py[n])+sqr(pz[n]))/(2*mass[n]);
    if (KE/ep > 100.0) then WriteToFile(rx,ry,rz,px,py,pz...)
  end;
```

User-programmed expressions may be up to 255 characters in length. Various examples of expressions are provided in the Help topic associated with the expression memo box, which is accessed via the Samples button. Note that comments bracketed within curly braces `{like this}` will be ignored by the expression parser, and they can be used as required to improve readibility.

Note. The validity of the syntax of any user-programmed option is not checked by *Spider*. Instead, syntax checking is carried out by *Snook* at run-time, and you will be informed if an error is found.

4.6. Projectile incident angles

The initial direction of the projectile flight path is specified by means of its altitudinal (φ) and azimuthal (ϕ) angles respectively in the same (x, y, z) coordinate system which describes the .TRG file lattice sites. For normal incidence, φ = 90° and ϕ is arbitrary. The azimuthal angle is measured as an anti-clockwise rotation in the (x, y) plane, starting from the x-axis (ϕ = 0°). The velocity components for the projectile (v_{0x}, v_{0y}, v_{0z}) are initialised as follows:

$$\begin{aligned} v_{0x} &= -\sqrt{E/2m}\cos\varphi\,\cos\phi \\ v_{0y} &= -\sqrt{E/2m}\cos\varphi\,\sin\phi \quad (4.1) \\ v_{0z} &= -\sqrt{E/2m}\sin\varphi \end{aligned}$$

If your simulation requires φ < 90° and you select ϕ <> 0° (not recommened), you will need to have an understanding of how the specified value of ϕ relates to the major crystallographic axes (like <100> or <110>) of your lattice file. In other words, you need to know where these axes lie in the (x, y) plane in order to set the value of ϕ that corresponds with your experimental interest. Read Chapter 6 to see how the projectile position is initialised.

The author strongly recommends that you always set ϕ = 0.0°, and that you rotate your lattice to achieve the correct crystallographic alignment with the beam.

The reason for this strong statement is that it is all too easy to forget that the shape of the 'reduced impact zone' depends on the relative azimuthal beam-lattice alignment. (You may need to read about impact files in Chap. 6 before the meaning of this and following remarks becomes clear) Users should appreciate that when the projectile azimuthal alignment is changed, the impact file specifications should also be changed. For example, you must create one impact file for <011> azimuthal incidence, and another (different) impact file for <001> azimuthal incidence. You also need to align the projectile azimuthal incidence with the appropriate azimuth of the target. You can achieve this by changing ϕ and keeping the target fixed, or you can keep ϕ fixed (at 0°, for example) and rotate the target by 45°.

The recommended procedure for setting up the azimuthal alignment is:
(a) Set ϕ = 0.0° in the Run file;
(b) Rotate your lattice so that the axis of interest lies parallel to the *x*-direction;
(c) Identify the 'reduced impact zone';
(d) Prepare a .IMP file corresponding to the reduced impact zone (Chap.6).

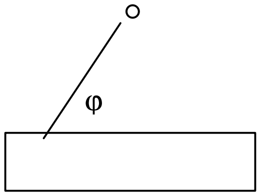
If you are not sure about the azimuthal orientation of your lattice with respect to the (*x*, *y*) plane, you can inspect it visually via the Target|Display/Orient menu option, and read the *Spider* online Help topic that summarises the default azimuthal orientations of lattices generated by *Spider* (indexed in the Help file under: azimuthal orientations).

For fcc lattices, the the targets are generated with the following default azimuthal orientation:

- 100 target: *y*-rows: [011], *x*-rows: [01-1]
- 110 target: *y*-rows: [-110], *x*-rows: [001]
- 111 target: *y*-rows: [-1-12], *x*-rows: [-110]

4.7. Example

The following illustration depicts the starting trajectory for a projectile (◦) with ϕ = 0° and φ ~ 70° (note that in this configuration both *x* and *z* components of the projectile velocity are initially negative):



The choice of a value for φ is not difficult, and is governed by the experimental set-up. Normally ϕ is specified in terms of a crystallographic direction: e.g. a projectile may be incident on Cu(001) along a <001> azimuth ('<001> incidence'). As emphasised above, the number you select for ϕ must be chosen with reference to the actual alignment of the .TRG file which you intend to use for the simulation. For example, by default, the Target|FCC|(100) menu option generates a (001) surface with <110> type rows parallel to the *x* and *y* axes. In this case, you should specify ϕ = 0.0° if you want <110> incidence, and ϕ = 45° if you want <100> incidence. Equivalently, as recommended above, you can rotate your Cu(001) target by 45° to achieve <100> incidence with the same value of ϕ = 0.0°. As emphasised previously, the author recommends the second procedure.

4.8. Periodic and miscellaneous parameters

Depending on the projectile energy and other characteristics, different simulation systems will evolve over different timescales. The 'Periodic Parameters' are parameters relating to the passage of time. Although their values are not critical to the success of the simulation, they have to be appropriately chosen for the user's specific system in order to ensure efficient computation. The individual parameters (Initial Timestep, Neighbour Update Time, Termination Time) will be considered in the following sections.

4.8.1. Initial Timestep

The Initial Timestep (expressed in s) represents the width of the integration interval, *dt*, used by *Snook*. The timestep is chosen in order to effect a compromise between computational speed and integration accuracy. (The larger the timestep, the faster the simulation.) The timestep scales roughly with velocity, i.e. as sqrt(*m/E*), where *m* and *E* are respectively the mass and energy of the projectile.

Snook/Kalypso gradually increases the timestep as the atoms in the collision cascade slow down, in such a way that the maximum displacement *dr* (for the fastest particle) stays approximately constant for every timestep. So by selecting an Initial Timestep, *dt*, you are also in effect specifying the spatial distance, *dr*, to be used for the entire simulation. (You can use the Utilities|Velocity Reckoner gadget to quickly calculate displacements per timestep corresponding to particular energies.)

Typically *dt* is set so that *dr* is no more than ca. 5% of the interatomic distance in the target atomic lattice (or ca. 0.1 Å, although shorter values are needed for very energetic collisions). For example, a timestep of 6.0E-17 s or less would be suitable for 5 keV Ar projectiles (v = 1.6 Å/fs), but setting it lower would do no harm.

If the timestep is too large, there is an unacceptably high error in the energy conservation reported by *Snook*. This effect is most significant for hard collision configurations, where the potential changes most rapidly with position. The optimum choice of timestep depends on the kind of information to be extracted from the simulation. For processes associated with hard collisions (e.g. impact collision ion-scattering spectroscopy) a value of perhaps 3.0E-17 is appropriate for the 5 keV Ar-Cu system, whereas for processes with larger cross-sections, a value of 8.0E-17 might be adequate.

It should be mentioned that small impact parameter projectile-target collisions are a rare event. The average energy conservation error reported by *Snook* is dominated by large impact parameter collisions (on the order of 1 Å).

If the timestep is increased, you may also find that *Snook* aborts after reporting the message: `"Too many partners: increase number of partners or reduce timestep"`. You should not be alarmed. This occurs either because longer timesteps require larger neighbour lists, or because of integration errors. The solution in the first case is to increase the "Maximum No. of Partners" parameter in the Run dialog, and in the second (more common) case to reduce the timestep.

4.8.2. Neighbour Update Time

The Update Time is expressed in 'timesteps' (integration cycles). This parameter determines how neighbour lists are managed. (A neighbour list is a list of current and possible interaction partners for each atom, which is compiled on the basis of system velocities, current interatomic distances and potential ranges.)

During a simulation run, neighbour lists are updated periodically, according to the Neighbour Update time parameter (1= every timestep, 10 = every 10th timestep etc). A typical value for this parameter is around 10. No error will result from setting a sub-optimal Update Time. However, if the lists are updated too frequently, the calculations run more slowly; if the lists are updated too infrequently, then *Snook* has to track far more potential collision partners which requires more memory, and may slow the calculations unacceptably. Note that if the neighbour update time is increased, you will normally have to increase the parameter specifying the number of partners (neighbours) that *Snook* can track (otherwise you will get an error message when you run *Snook*). To set an optimum Update Time, you generally should carry out a few preliminary runs to test the sensitivity of the calculation speed to this parameter.

4.8.3 Termination Times

The Minimum Termination Time specifies the minimum time for which the simulation will be run, regardless of what happens to the dynamical variables. There are situations in which this may be useful. It is a good idea to use a non-zero value of this parameter in impact collision ISS simulations, for reasons explained in the Spider Help file or `spider.pdf`.

The Maximum Termination Time specifies a time (in fs) at which *Snook* will automatically terminate the current run, regardless of the state of the system dynamical variables. The termination time should not be shorter than the timescale of the process which is being modelled.

One reason you need this parameter is to prevent infinite looping of the calculations if for some reason the normal termination condition (low energy) is not met.

The energy-termination condition doesn't always give satisfactory results because it only takes account of the atoms within a 'lattice' region defined by the initial target atom coordinates. If all target atoms are liable to move outside this region (e.g. if the target consists of a single atom) then you must control the simulation termination via a time condition.

A value of 1000-2000 fs would be typical for sputtering studies. After this time, any atoms that are going to be sputtered would already be on their journey out of the surface. There is no point in wasting computing resources by calculating beyond this point. (A much shorter time would be appropriate for ion scattering, because of the abrupt nature of the interaction.)

You should attempt to combine both time and energy termination conditions to control the exit point of your simulations. For instance, in a study of sputtering you could specify that the simulation should terminate if $t > 1000$ fs, or if the most energetic particle in the lattice has < 2 eV kinetic energy. A careful choice of these exit conditions may speed the calculations.

4.9. Cut-Off Parameters

[Note: for *Spider*-SK, the switching function selection/range described in 4.9.2 is specified in the Run file, but for *Spider-Kalypso* it is specified in the model file.]

The simulation algorithms used by *Snook*/*Kalypso* only take account of interactions over a finite range specified by the Potential Cut-Off parameter (R_c). A consequence of this is that each atom interacts at any time with only those neighbouring atoms that lie within a sphere of radius R_c. *Snook* maintains a list of these atoms, as well as any others that may potentially move into interaction range between list updates. The Maximum No. of Partners parameter represents the number of interaction partners that may be tracked for each atom in the lattice. Some guidelines for setting these parameters are given below.

4.9.1. Potential Cut-Off

A typical choice for the Potential Cut-Off would be some distance between the second and third nearest neighbour distance in your lattice. For example, the nearest and next-nearest neighbours in Cu are 2.56 and 3.62 A respectively. A Potential Cut-Off of around 3.8 A would be appropriate for this case. The maximum value of the Potential Cut-Off allowed by *Snook* is 8.0 Å, but you will probably never use such a large value. For attractive potentials (Morse, TB etc.) the Potential Cut-Off will normally be specified by whoever fitted the potential. You cannot arbitrarily modify the cut-off without changing the lattice properties (e.g. the cohesive energy of the solid).

For exploratory work, or for calculations involving huge lattices, you can consider reducing the Potential Cut-Off to a value between the first and second nearest neighbour distances (e.g. 3.0 Å for Cu). If you are only interested in hard-collisions (as in Ion Scattering Spectrometry) you should probably set a Potential Cut-Off which reflects this (say 1.0 A), because you will benefit from a great increase in computational speed.

Calculations run more slowly as the potential cut-off is increased, because this leads to a larger number of interactions. Memory usage by the computer is also higher, since these extra interaction partners have to be tracked via the neighbour lists.

4.9.2. Switching Function (Spider-SK only)

A polynomial switching function can be applied near the potential cut-off, to bring the potential and forces smoothly to zero at the cut-off. If the user selects this option (by checking the box), he or she must also specify the distance at which to turn on the switching function. This should normally be about 0.2 A (or more) below the cut-off distance. Take care that the switching function is not applied in a region which contains a shell of lattice atoms, otherwise it will alter the lattice properties. See Goldstein and Jonsson, Phil. Mag. B (1995) 71 1041-1056 (Eq. 14) for the implementation details, and section 5.2.4.

4.9.3. Maximum Number of Partners

This is admittedly an awkward parameter to set. It is required to keep *Snook*'s memory usage flexible for users with older PCs that don't have an abundance of memory (say 4 MB, which is common on 80486 machines). If you have 16 or 32 MB then set this parameter permanently to a large value (60-100 say) and don't worry about it. (You will be advised if the setting is found to be too small during a simulation.)

The significance of the parameter is that it is a measure of the number of neighbouring atoms that have to be included in the biggest neighbour list. (In other words, it determines the amount of memory which will be allocated for the lists.)

The neighbour list of a given atom is a list of all other atoms in the system which are currently within interaction range, or might possibly move within interaction range before the next update of the neighbour list.

The value might be between 70-100 for a bulk target depending on the simulation conditions, including: the potential cut-off distance, the projectile velocity, the structure/size of the target lattice and the Neighbour Update Time. This value is somewhat higher than the number of atoms in the primary and secondary coordination shells because the neignbour list also has to take account of moving atoms near the border of the shell.

For 5 keV Ar-Cu (1000 atoms), with a timestep of 0.08 fs, update time of 10 timesteps and 3.7 A potential cut-off, the 'number of partners' is found to be less than 45.

There is no harm in setting the Number of Partners too high, if your PC has the memory available (say > 16 MB). Otherwise, consult the example files that are shipped with *Snook*, or carry out some exploratory runs.

Snook/*Kalypso* will generate an error message at run-time ('Too many partners required') if it finds that the neighbour lists need more partners than you specified. The simulation will then abort gracefully. You can sometimes get the same error message if you set the Initial Timestep to a spuriously high value (for instance, 1 fs for 5 keV Ar projectiles). When *Snook*/*Kalypso* runs in verbose mode, it will inform you how many partners were actually required so that you can familiarise yourself with the size of the parameter (look for the message: Maximum no. of collision partners required was:...').

In general, if you increase the primary projectile energy, the number of partners required will also increase (because of higher speeds of atoms in the collision cascade). Simulation runs involving hard collisions will also require a higher number of partners than those involving large-impact parameter collisions (because of greater lattice disruptions).

5: The Model file

5.1. Function

The Model file (extension MDL) contains the parameters which define the projectile-target and target-target interaction model. Lattice vibrational effects and other miscellaneous parameters are also specified in the Model file.

The major difference between the SK and *Kalypso* lies in the way attractive interactions between target atoms are modelled. The SK uses a single Morse potential for all such interactions, which means among other things that the SK describes all attractive forces in the same manner, regardless of the type of atom involved. You can work around this to some extent in sputtering simulations by applying an extra surface binding energy to one type of atom only, but a better approach is to use *Kalypso* for simulations of binary compound targets where attractive forces are important. If attractive forces are not important (e.g. ISS simulations), you should use the SK, which models the short-range repulsive forces correctly and is considerably faster than *Kalypso*.

Kalypso models attractive forces realistically for a binary compound target using many-body Tight-Binding or Sutton-Chen potentials. In *Kalypso*, these potentials are generalised slightly, so that by suitable choice of coefficients you can cast them into the form of Morse or Lennard-Jones potentials. There are three sets of coefficents for the attractive potentials in *Kalypso*. If the target contains atoms of types A and B, then the coefficients describe A-A, A-B, and B-B interactions respectively. As discussed in section 3.4, *Kalypso* recognises an atom's type according to its flag parameter, so don't forget to set this correctly when you create the target file.

By default, the projectile interacts with target atoms via a screened Coulombic potential at all internuclear separations. However, both the SK and *Kalypso* allow a departure from this model in the case that the projectile of the same atomic type as one of the target atoms (the so-called self-bombardment experiment). In this case, the appropriate attractive potential for that atom may be used. A typical application is thin film growth by low energy ion bombardment. [Note: you specify a self-bombardment option in *Snook*/*Kalypso* at run-time, not in *Spider*'s model file, which should be set up in the usual way.]

At short internuclear distances, target atoms also interact with each other via a repulsive screened Coulomb potential. A cubic spline function is used to bridge the repulsive and attractive regions. The overall potential is termed a composite potential.

The next sections describe the potential parameter inputs for *Kalypso* and the SK respectively. Topics common to both (lattice vibrations, specific energies) follow afterwards. See section 8.5 for an example of the creation of a model file for *Kalypso* (section 8.6 for the SK).

5.2. Model file for *Kalypso*

5.2.1. Repulsive potentials

Select the appropriate types of projectile-target and target-target screened Coulombic potentials. Add a screening length correction if you wish. (If in doubt, select ZBL potentials and a correction factor of 1.0 for both items.)

5.2.2. Potential cut-off

Enter the range at which the potentials cut off. See the *Simulation Primer* document for a discussion. Note that potential parameters change when the cut-off changes, so you will normally obtain the cut-off distance as one of the parameters of your potential.

5.2.3.Spline function parameters

Enter the atomic numbers ('Z(A), Z(B)') of type A and type B (if any) target atoms here. These values are not used directly for simulation purposes, but only to generate the spline function. (You can look at the spline coefficients by opening up the Model file in a text editor.) Enter the range over which the spline function is to be fitted. [These will normally be guessed values, in a region somewhat below the first neighbour separation. After you have entered the attractive potential parameters, you can use the 'Graphs' tab to view the composite potential, and adjust the spline range iteratively to your satisfaction.]

5.2.4. Attractive potentials

Select the potential type (Sutton-Chen or TB-SMA, Tight-Binding Second Moment Approximation) via the radio button, bottom right. A switching function can be applied to the attractive potentials, in order to terminate them smoothly at the cut-off distance. If you want to use a switching function (recommended) select the appropriate box, and pick a switching distance (R_{SW}). This should be less than the potential cut-off distance, but should be outside the last atomic shell for which interactions are counted. For example, suppose the cut-off is 4 Å for Cu, whose second neighbour shell lies at 3.62 Å. Then an appropriate value for R_{SW} would be around 3.8 Å. The reason for applying the switching functionis outside the last shell of atoms is to make sure that the inclusion of the switching function doesn't modify the properties of the undisturbed lattice. (See also section 4.9.2.)

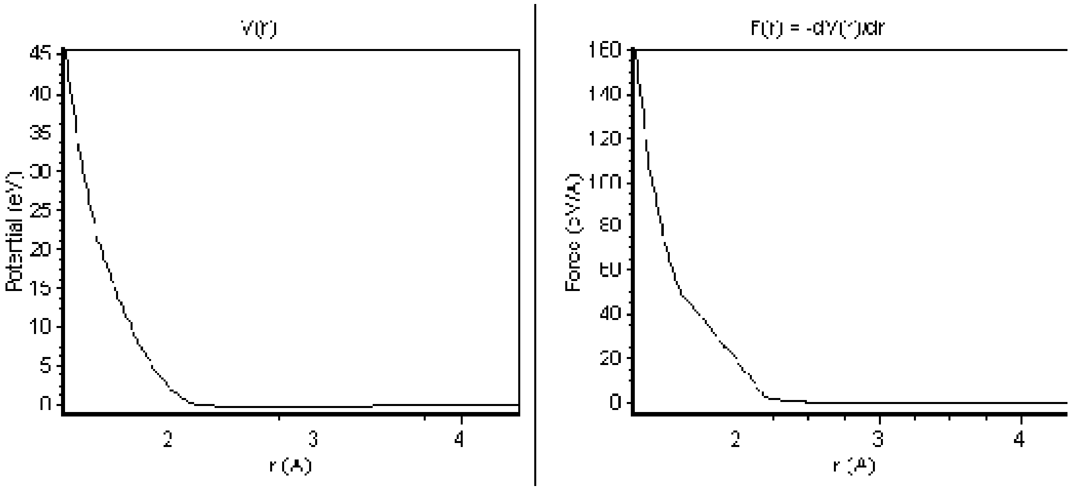
For each potential enter the appropriate parameters (see the *Simulation Primer* for a list of elemental materials). The last parameter, S_n may be tricky. This is a lattice sum which has to be calculated on the basis of the other parameters (including the potential cut-off) and is to some extent arbitrary for compound targets. It is used for specifying the spline function. Again you are referred to the Simulation Primer for details of its calculation. Since the target is an element, the V(A-B) and V(B-B) parameters are unimportant.

5.2.5. Effective potentials

Now you can proceed to view the composite target-target potentials. Select the Graphs tab. Select a graph to display (e.g. the A-A potential). The left-side graph displays the 'effective potential', while the right-side graph displays its derivative, the force-function. There is also an option to display the coefficients of the spline function (or you can view them by opening up the model file in a text editor).

What is the effective potential? This is an approximate representation of the many-body potential inside the lattice environment (Simulation Primer, chapter 10). It has a pairwise character, and is needed for fitting the spline function. Note that the effective potential depends on the lattice sum S_n which was discussed in the previous section.

Typically, you will see discontinuities (maxima and minima) in the $F(r)$ plot on your first try. Normally you will have to go back and adjust the spline range iteratively until you get a satisfactory result (no minima or maxima in $F(r)$). Some cases (e.g. Ca) are quite difficult to fit. Normally you can get a very smooth $V(r)$ composite, but $F(r)$ will have significant gradient discontinues at one or both spline nodes. Maxima and minima in *F*(*r*) should not be tolerated, however. The model files in the example projects offer an illustration of the type of fit that you should aim for.



Every time you generate a graph display (as above), the corresponding potential and force functions are written to disk in text files named POTENTL.SPL and FORCE.SPL respectively. These are not needed for the simulation, but you may wish to plot or examine them for your own information.

5.3. Model file for SK

5.2.1. Repulsive screened Coulombic potentials

Select the appropriate types of projectile-target and target-target screened Coulombic potentials. Add a screening length correction if you wish. (If in doubt, select ZBL potentials and a correction factor of 1.0 for both items.)

5.2.2. Attractive Morse potential

If you want to use an attractive Morse potential, enter the Morse parameters. Otherwise disable this option via the check box ('use Morse potential'). For ISS simulations an attractive potential is usually unnecessary.

Set a range for the splining function which will be applied between the Morse and screened (target-target) Coulombic potentials (this step is not necessary if you disabled the Morse option). You will probably have to make a guess at first (the spline range should be somewhat below the nearest neighbour lattice distance). To test your guess, click the 'View Composite Potential' button. You will be prompted for the atomic number of your lattice atom (this is needed to compute the spline), then a dialog will come up with the spline parameters. Click OK, and you will then get graphs of the potential, V(r), and its derivative, F(r). You will probably have to modify the spline range in order to get a well-behaved spline function. See the remarks in 5.2.5 for hints on how to proceed from here.

5.4. Lattice vibrations

You can optionally apply Debeye-Waller (DW) vibrational displacements corresponding to your lattice temperature. You will need to enter the appropriate Debeye temperatures (a list is provided in the online Help), as well as the lattice atom mass. The Debeye temperatures for surface atoms differ from those of bulk atoms. You can optionally enter values appropriate for perpendicular (z) and parallel (*xy* plane) vibrations of surface atoms.

***Kalypso* users take note**: this option generates two identical lines of output in the model file, for type A and type B atoms respectively. These consist of 3 numbers:

```
0.00605 0.01885 0.00827 // rms vib A
0.00605 0.01885 0.00827 // rms vib B
```

These are DW mean square vibrational amplitudes (in $Å^2$) based on the parameters you entered. For a compound target, these parameters will be different for the two different types of atoms. You may, of course, ignore this difference where the atoms are similar (e.g. for a Cu-Ni system), but you will probably want to edit the model file in a text editor and insert the appropriate values for each atom type. However, beware - if you later modify the model file again in *Spider*, your edited values will be overwritten (so make the file read-only to protect it).

5.5. Specific energies

These are miscellaneous parameters that you might wish to set, but will often ignore.

1. Termination energy - the simulation will terminate when all atoms in the lattice region fall below this energy. This condition will override the maximum termination time defined in the run file, but not the minimum termination time. If you set the termination energy to zero the simulation will only terminate after the maximum termination time has elapsed.
2. Surface binding energy - you can apply an additional binding energy to surface atoms when they attempt to leave the surface. The surface plane lies above the anchor atom at a distance equivalent to the potential cut-off distance. If the atom's energy is too low to pass through the surface barrier it will be reflected, otherwise it will be refracted. You might want to apply a surface binding energy if you think that the surface binding energy predicted by your potential was too small. Otherwise leave this parameter at zero. You can control which atoms experience and ignore this surface binding energy by setting flags in the Target file. See the chapter on Flags in the *Simulation primer*.
3. Bulk binding energy - this is an energetic barrier applied to atoms which attempt to leave the edges of the simulation crystallite. The implementation is similar to the surface binding energy. The idea is that you may wish to have a lattice with stiff or rigid atoms at the edges. Avoid using this option unless you have good reason. If your lattice is breaking up, it is better to make it larger than to constrain it in this manner.

6: The Impact file

6.1. Function

The impact file consists of rows of Cartesian coordinates (b_x, b_y, b_z, expressed in Å).[1] The z coordinates, (b_z) in the impact file refer to the vertical starting position of the projectile relative to the z coordinate of the anchor atom (row #1) of the target file. The x and y coordinates (b_x, b_y) are expressed relative to the coordinates of the anchor atom, and represent the impact parameters of the projectile in the surface plane, i.e. the points towards which the projectile is directed at the start of the simulation.

For a projectile directed towards the surface at an azimuthal angle ϕ, and an altitudinal angle j , the projectile starting position (x_0, y_0, z_0) is expressed relative to the coordinates of the anchor atom (x_1, y_1, z_1) as follows:

$$
\begin{aligned}
x_0 &= x_1 + b_z \cos f / \tan j + b_x \\
y_0 &= y_1 + b_z \sin f / \tan j + b_y \quad (5.1) \\
z_0 &= z_1 + b_z
\end{aligned}
$$

For the special case of normal projectile incidence $(j = 90^\circ)$ the relationship (5.1) becomes:

$$
\begin{aligned}
x_0 &= x_1 + b_x \\
y_0 &= y_1 + b_y \quad (5.2) \\
z_0 &= z_1 + b_z
\end{aligned}
$$

It is important to grasp that the coordinates listed in the impact file refer to a projectile position relative to the location of the anchor atom. This is not usually obvious from inspection of the impact file, since the anchor atom is often placed at (0, 0, 0) by default.

For completeness, the initial values of the projectile velocity components (Chapter 4, Eq. 4.1) are repeated here:

$$
\begin{aligned}
v_{0x} &= -\sqrt{E/2m}\cos j \cos f \\
v_{0y} &= -\sqrt{E/2m}\cos j \sin f \quad (5.3) \\
v_{0z} &= -\sqrt{E/2m}\sin j
\end{aligned}
$$

In the author's view. the creation of impact files is the most difficult aspect of a simulation project. The statistics of sampling is something which an experimenter never has to think about, but it is important to get this right in a simulation model. The determination of the shape and size of the sampling area based on surface symmetry requires a familiarity with surface structure. This knowledge is assumed in the following sections. There is usually more than one way to

---

[1] The first row of the impact file also contains data stored from *Spider*'s impact file dialog box. This data is not used by the simulation program, however.

generate a target file + impact file combination, so the procedures outlined here should be regarded as descriptive rather than prescriptive.

6.2. Impact file format

The first line of a .IMP file is formatted as follows:

```
 1.0000E+0000  1.0000E+0000 3.0 1.000000 1.000000 1.000000 1.000000 90 0 1 1
```

The first two numbers represent the impact parameter of the projectile in the *xy* plane (1.0 Å). The next number (3.0) represents the height above the surface (3.0 Å) from which the projectile flight begins (z_0), relative to the anchor atom in the target lattice (usually located by default at (0,0,0)).

The remaining numbers (columns 4-11) represent data used by *Spider*'s Impact|Read dialog box; however, these numbers are not used in simulations. All information that *Spider* needs and uses to generate an Impact file is stored in columns 4-11 of line 1 of the Impact file. (Tip: If you want to archive a very large Impact file created by *Spider*, just store the first line in a text file to save on disk storage.)

Subsequent lines of the impact file (if any) contain only the (b_x, b_y, b_z) coordinate information, like this:

```
 1.0934E+0000  1.0000E+0000 3.0
```

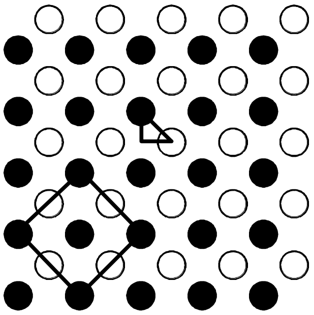
The numbers in the impact file must be delimited by one or more space characters, and may be written in a variety of conventional floating point formats (1.04, 1.04e0, 1.04E0 and so on). The format is very simple, and if you prefer, you could create your own impact files without the help of *Spider*. (It is not necessary to include the extra inputs, cols. 4-11, on the first line in this case.)

6.3. Creating impact files

The impact file must define a set of statistically representative points on the target surface. These represent a sample of points within a zone of irreducible symmetry, which will be called the reduced impact zone (RIZ) here. The RIZ depends on both surface symmetry, and collision geometry (see Harrison *et al*.[1] for a discussion).

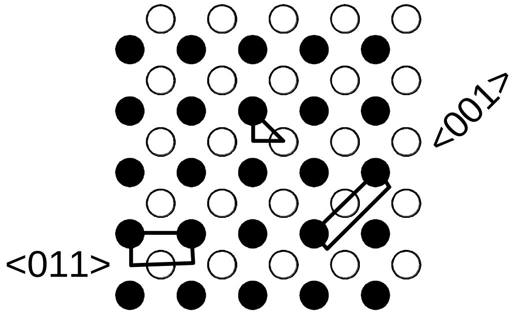
---

[1] D.E. Harrison, C.E. Carlston, G.D. Magnuson, Phys. Rev. 139 (1965) A737.



The figure to the right illustrates a fcc (100) surface (black circles are surface atoms, white circles are 2nd layer atoms). The RIZ for a normally incident projectile ($J = 90^{\circ}$) is represented by the small triangle in the centre of the figure, while the square region (bottom left) represents one face of the bulk crystallographic unit cell.

For non-normal projectile incidence, the RIZ shape depends on the azimuthal angle, ϕ. Examples are shown in the lower figure for <011> and <001> azimuthal directions of incidence. The area of the RIZ is the same in both cases, but the shapes are different.



For an arbitrary azimuthal angle with no special symmetry properties, the RIZ is equivalent to the area occupied by a face of the bulk unit cell. The unit cell could, in fact, be used for the other cases, but this is normally avoided because it is regarded as being statistically inefficient.[1] Likewise, any integral multiple of the true RIZ is acceptable, if inefficient.

There are two stages to the creation of an impact file. First, you must identify the RIZ for your experimental system and configuration. Second, you must decide how to represent this RIZ using *Spider*.

Tables 6.1-6.6 summarise the procedures for creating RIZs for cubic (100) and (110) surfaces for non-normal (Tables 6.1-6.3) and normal (Tables 6.4-6.6) projectile incidence. These are the easiest cases: (111) surfaces require a more elaborate procedure.

Before you can benefit from these tables, you will have to read the following sections, which explain the meaning of the notation. The tables employ the following abbreviations:

- N/A: Not applicable (parameter is irrelevant)
- *a*: bulk unit cell lattice parameter
- d_0, d_1: See definitions in footnotes to tables.
- ϕ: Projectile azimuthal direction of incidence.
- *R*45, *R*00 etc.: Azimuthal rotation which must be applied to the target file *as generated by Spider* in order to align ϕ with the specified axial direction. (R00 means no rotation is required.) Use the Target|Display/Orient command to apply these rotations.
- *R*00/*R*45/*R*90: Target file must be rotated azimuthally by 0°, 45° or 90°.
- b_x(min) etc.: refer to values entered for fields in the impact file dialog box.
- (x, y) angle: another field in the impact file dialog box.

---

[1] This argument is weak as far as the SK and *Kalypso* are concerned, due to the way in which the impact file coordinates are generated. Also, the addition of random vibrational displacements (if selected) removes the possibility of redundant sampling. However, other benefits of using the RIZ are: (1) reduced containment problems, and (2) simplified analysis of impact parameter dependent processes.

Table 6.1. Fcc and diamond lattices: **Non-normal** projectile incidence.[a]

Surface	ϕ	R_{xy}	b_x(min)	b_x(max)	b_y(min)	b_y(max)	(x, y) angle
(100)	<110>	*R*00	0.0	d_0	0.0	$d_0/2.0$	90
(100)	<100>	*R*45	0.0	$a/2.0$	0.0	$a/2.0$	90
(110)	<100>	*R*00	0.0	a	0.0	$d_0/2.0$	90
(110)	<110>	*R*90	0.0	d_0	0.0	$a/2.0$	90

[a] a is the fcc or diamond unit cell parameter, while $d_0 = a\sqrt{2}$.

Table 6.2. Bcc lattice: **Non-normal** projectile incidence.[a]

Surface	ϕ	R_{xy}	b_x(min)	b_x(max)	b_y(min)	b_y(max)	(x, y) angle
(100)	<100>	*R*00	0.0	a	0.0	$a/2.0$	90
(100)	<110>	*R*45	0.0	d_0	0.0	d_0	90
(110)	<110>	*R*00	0.0	$2\ d_0$	0.0	d_1	90
(110)	<110>	*R*90	0.0	d_1	0.0	$2\ d_0$	90

[a] a is the bcc unit cell parameter, while $d_0 = a\sqrt{2}$ and $d_1 = (a\sqrt{3})/4$.

Table 6.3 Rocksalt lattice: **Non-normal** projectile incidence[a]

Surface	ϕ	R_{xy}	b_x(min)	b_x(max)	b_y(min)	b_y(max)	(x, y) angle
(100)	<100>	*R*00	0.0	a	0.0	$a/4.0$	90
(100)	<110>	*R*45	0.0	d_0	0.0	$d_0/2.0$	90

[a] a is the fcc or diamond unit cell parameter, as applicable, while $d_0 = a\sqrt{2}$.

Table 6.4. Fcc and diamond lattices: **Normal** projectile incidence.[a]

Surface	ϕ	R_{xy}	b_x(min)	b_x(max)	b_y(min)	b_y(max)	(x, y) angle
(100)	N/A	*R*00	0.0	d_0/2.0	0.0	d_0/2.0	45
(110)	N/A	*R*00	0.0	*a*/2.0	0.0	d_0	35.2644

[a] *a* is the fcc or diamond unit cell parameter, while $d_0 = a\sqrt{2}$.

Table 6.5. Bcc lattice: **Normal** projectile incidence.[a]

Surface	ϕ	R_{xy}	b_x(min)	b_x(max)	b_y(min)	b_y(max)	(x, y) angle
(100)	N/A	*R*00	0.0	d_0/2.0	0.0	*a*/2.0	45
(110)	N/A	*R*00	0.0	d_0/2.0	0.0	d_1	35.2644

[a] *a* is the bcc unit cell parameter, while $d_0 = a\sqrt{2}$ and $d_1 = (a\sqrt{3})/4$.

Table 6.6 Rocksalt lattice: **Normal** projectile incidence[a]

Surface	ϕ	R_{xy}	b_x(min)	b_x(max)	b_y(min)	b_y(max)	(x, y) angle
(100)	N/A	*R*45	0.0	d_0/2.0	0.0	d_0/2.0	45

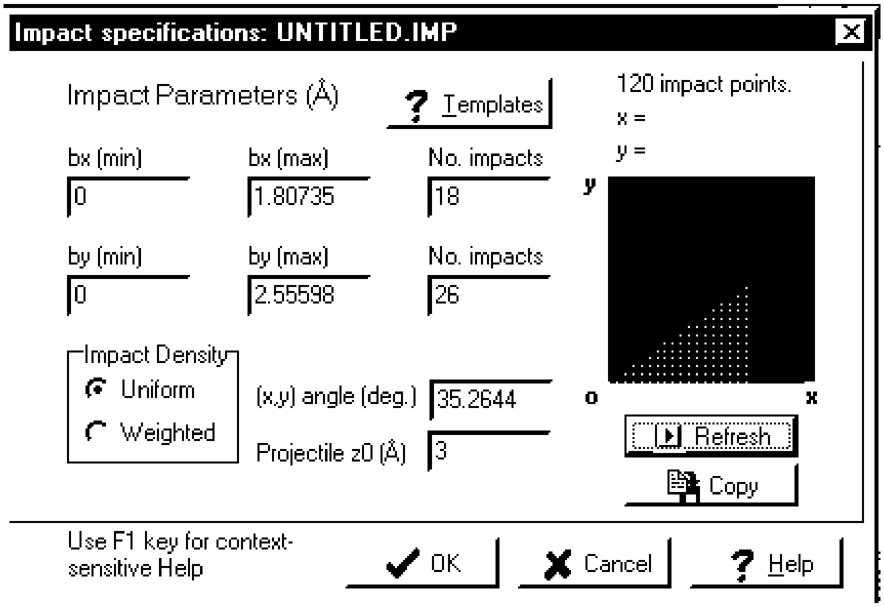
[a] *a* is the fcc or diamond unit cell parameter, while $d_0 = a\sqrt{2}$.

6.4. Examples

6.4.1. Normal projectile incidence on Cu(110) surface

A RIZ for normal incidence on a cubic lattice (100) or (110) surface can be generated relatively easily. For fcc, bcc and diamond lattices, the RIZ can be used immediately with the corresponding target file generated by *Spider*; in other words, no azimuthal rotation of the target is required ($R_{xy} = 0.0$, represented as *R*00 in Tables 6.4 and 6.5).

Before you begin, you will need to know the lattice parameter (*a*) for Cu (= 3.6147 Å) and compute the parameter d_0 defined in Table 6.4 as $\sqrt{a}$ (i.e. $d_0 = 2.55598$ Å). Select the Impact|New menu item in *Spider*. A dialog box comes up. With reference to Table 6.4, enter the values shown in the figure below, then click the Refresh button to see the RIZ points represented in the impact file (you should see a triangular region, as in the figure).



Impact specifications: UNTITLED.IMP

Impact Parameters (Å)

120 impact points.
x =
y =

bx (min): 0
bx (max): 1.80735
No. impacts: 18

by (min): 0
by (max): 2.55598
No. impacts: 26

Impact Density
● Uniform
○ Weighted

(x,y) angle (deg.): 35.2644
Projectile z0 (Å): 3

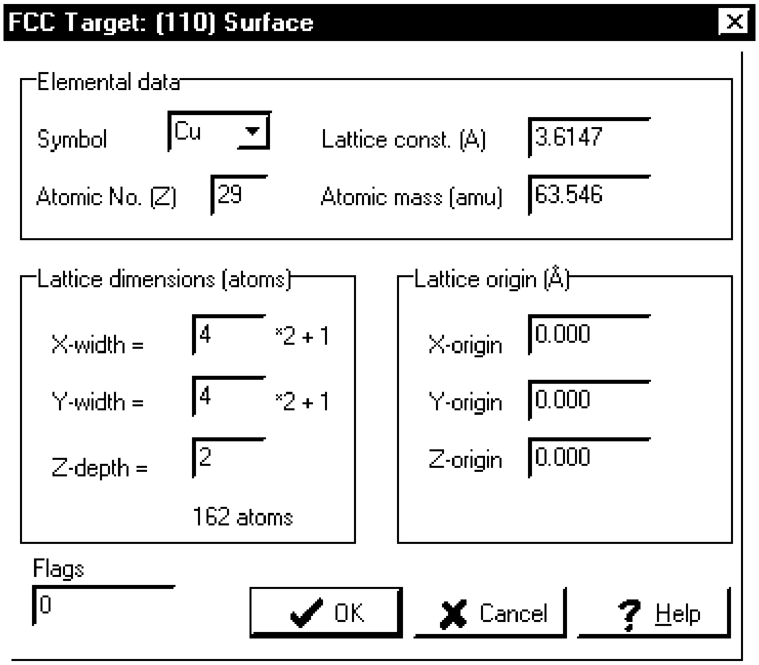
Use F1 key for context-sensitive Help

Note the following:

- The Impact Density option is not used for the public release of these programs, so you can ignore it.
- The Number of Impacts options determine the mesh widths of the impact points in the *x*- and *y*-directions respectively. In this case, the values reflect the ratio b_x(max): b_y(max). The number of impact points required in the impact file depends on the goal of the simulation. For a sputtering simulation, 500-2000 impact points would be typical, whereas for an angle-resolved ion scattering spectrum simulation, you might need 10^5 impact points if backscattering is weak.
- The Projectile z_0 parameter indicates the starting projectile z-coordinate relative to the anchor atom. in this case, the projectile z-coordinate will be initialised as $z_{\text{anchor atom}}$ + 3.0 Å.
- Since b_x(min) and b_y(min) are both zero, the first projectile trajectory will start off directly above the anchor atom (see Eq. 5.2).
- The (x, y) angle parameter refers to the bottom, left angle of the triangle (dotted region).

- The Copy button copies the image to the Windows clipboard, while the Templates button brings up a help screen with information from Tables 5.1-5.6.

Click the OK button and save the file as 'cu110.imp'. It is always wise to check the dimensions of the RIZ. To do this, go to Target|New|Face-centred cubic, and generate a 2-layer Cu(110) lattice using the inputs shown in the next figure. Save the file as 'cu110.trg'.



FCC Target: (110) Surface

Elemental data

Symbol: Cu

Lattice const. (A): 3.6147

Atomic No. (Z): 29

Atomic mass (amu): 63.546

Lattice dimensions (atoms)

X-width = 4 *2 + 1

Y-width = 4 *2 + 1

Z-depth = 2

162 atoms

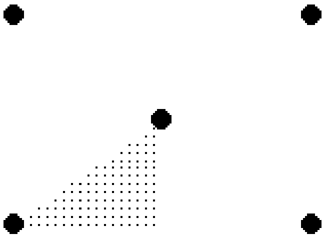
Lattice origin (Å)

X-origin: 0.000

Y-origin: 0.000

Z-origin: 0.000

Flags: 0



Next, use the Impact|Display/Orient command to load the file cu110.imp for display. To superimpose the target file on this display, use the Display|Link file option on the dialog box menu to load cu110.trg. You should now see the relationship between the surface symmetry and the RIZ which you generated, as in the diagram above.

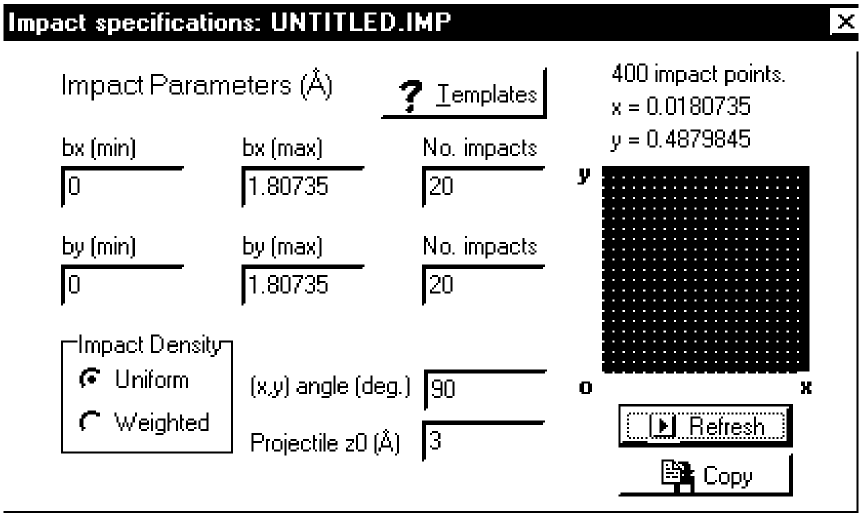
6.4.1. Non-normal projectile incidence on Cu(100) surface: f =<001>

If you refer to Table 6.1, you will see that this configuration requires a target file which is rotated azimuthally by 45° ($R_{xy} = R45$). The reason for this is that the target file is created, by default, with <011> edges parallel to the x- and y-directions. However, we need to match the target file with a RIZ which has edges parallel to <001>.[1] We will thus first create the impact file, then briefly show how to rotate the target file.

---

[1] Of course, it is also possible to rotate the RIZ instead of the target, but we choose not to for this example.

The impact file is created using the inputs shown in the figure below (with reference to Table 6.1, using values of a and d_0 appropriate to Cu from the previous example).


Impact specifications: UNTITLED.IMP

Impact Parameters (Å)

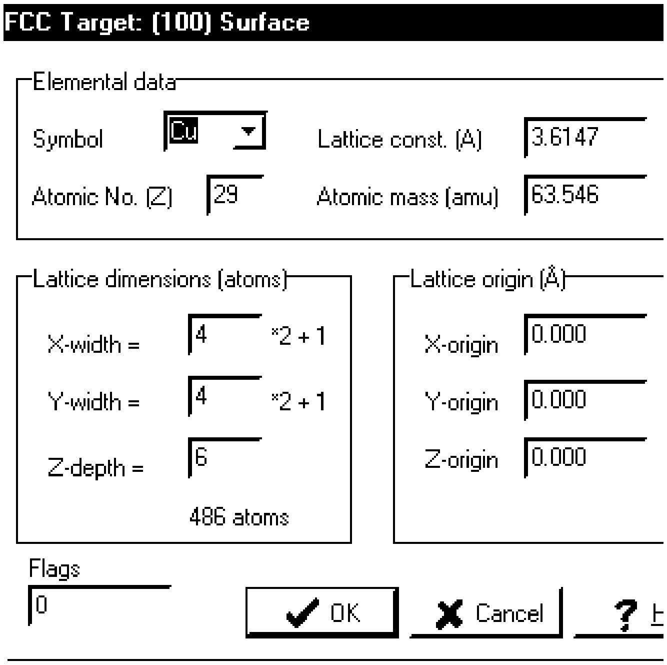
bx (min): 0
bx (max): 1.80735
No. impacts: 20
by (min): 0
by (max): 1.80735
No. impacts: 20

Impact Density: ● Uniform ○ Weighted

(x,y) angle (deg.): 90
Projectile z0 (Å): 3

400 impact points.
x = 0.0180735
y = 0.4879845

The RIZ is a square region. Next use the Target|New|Fcc|(100) command to generate a target file using the following inputs, and save as cu100a.trg.


FCC Target: (100) Surface

Elemental data
Symbol: Cu
Lattice const. (A): 3.6147
Atomic No. (Z): 29
Atomic mass (amu): 63.546

Lattice dimensions (atoms)
X-width = 4 *2 + 1
Y-width = 4 *2 + 1
Z-depth = 6
486 atoms

Lattice origin (Å)
X-origin: 0.000
Y-origin: 0.000
Z-origin: 0.000

Flags: 0

The coordinates in this target file must be rotated azimuthally by 45° before the file can be used in conjunction with the impact file. The first step is to display cu100a.trg using the Target|Display/Orient command. This brings up a dialog box, to which the next remarks refer. Enter 45 into the z-axis rotation field in this dialog box (top left) and click the following buttons in this order: Test, Reload, Apply. The last command will allow you to save the rotated file to disk (e.g. as cu100b.trg). Before doing so, you may wish to superimpose the RIZ defined in your impact file. If so, click Display|Link file, and select cu100.imp. Click the Impose button whenever you need to refresh the display of the RIZ. You will see that the RIZ reflects the symmetry of the rotated file (i.e. after you click Test), but not that of the original file (cu100a.trg). To be sure that everything is specified correctly after you save cu100b.trg, you can use the latter in conjunction with the Impact|Display option applied to cu100.imp, as described in the preceding section.

6.4.2. Impact file for fcc (111) surface, normal incidence

This example is covered in the *Spider* online help file (under the topic: Display/Orient Overview: see the help index). In this case, it is necessary to trim the default RIZ generated by the Impact|New command to the desired shape (equilateral triangle) using the Impact|Display/Orient tool. The difficulty in generating impact files for the more unfamilar cases is not in using the tools, but in knowing the crystallography of your surface. It is easy to make mistakes, so you should always check that the final results are reasonable using *Spider*'s graphical tools, and by any other means available to you.

7: The Inelastic file

7.1. Function

The Inelastic file is an *optional* component of a simulation project (you can choose not to use it). New users of *Kalypso* and the *Simulation Kit* should probably ignore inelastic effects until they are familiar with the implementation of a simulation based on elastic processes only (like the great majority reported in the literature). Inelastic effects grow in importance as the particle velocity increases.

The Inelastic file specifies the parameters which are used for modelling inelastic energy losses (responsible for so-called "electronic stopping") in a collision process. Three different inelastic loss models are available, which may be used singly or in any arbitrary combination. These models are:

(a) Lindhard-Scharff-Schiott (LSS) model
(b) Oen-Robinson (OR) model
(c) Shapiro-Tombrello (ST) model

In addition, the inelastic file allows you to specify parameters relating to thermal cooling effects. The physical background to models (a) - (c) is discussed in some detail in Chapter 8 of the Simulation Primer file which ships with the SK and *Kalypso*. The Compute K(LSS), K(OR) gadget found on the inelastic file input data dialog box helps you compute the parameters used for the first two models.

7.2. Implementation of the inelastic models

The implementation of each of the inelastic loss models needs to take into account the fact that individual atoms or pairs of atoms are associated with different inelastic loss parameters. This is achieved by "tagging" each set of parameters with a number, which is used like an index or hashing function at runtime to look up the appropriate set of parameters.

For the LSS model, which represents electronic stopping of an atom by a continuum target material, the tag is simply the atomic number (Z_1) of the atom which is being stopped.

For the OR and ST models, which predict the inelastic energy loss in a binary collision event, the tag is the product of the atomic numbers of the participating atoms, i.e. Z_1*Z_2. The tag for an Ar-Ar collision event is thus 324, while for an Ar-Cu close-encounter it is 522.Up to 10 different tags may be used. (This scheme permits the tracking of inelastic events between the projectile and the different components of a target with 1-3 different kinds of atoms in it.)

For all of these models, tags of zero ('0') and their associated parameters will be ignored by the simulation at run-time. You can use this feature to disable parameter sets without deleting them. Duplicate tags are allowed but will also be ignored (only the first match is used). Invalid tags (i.e. tags that never occur in the simulation) are allowed, but have no effect.

The "scale" parameters for the LSS and OR models, and the "dE" parameter for the ST model, can be modified to partition the inelastic energy loss over 2 or more models. The scale parameter modifies (scales) the theoretical value of the energy loss function (thus, the default value is 1.0). For example, an equipartition of the energy loss between the LSS and OR models can be achieved by setting both LSS and OR scale parameters to 0.5. A scale factor of 0.0 ("no energy loss") has the same effect as a null tag parameter (see preceding).

The electronic stopping effects are applied to the system in the following order: ST, OR, LSS. For example, the OR energy loss is calculated after the ST energy loss (if any) has been removed from the system, but before the LSS loss (if any) has been removed. (There is no particular significance in this order.)

Important
In order to incorporate inelastic effects into your simulation model, you must also select any model(s) (LSS, OR, ST or cooling effects) you wish to use in the *Snook*/*Kalypso* Simulation Options dialog box at runtime. All inelastic models are disabled by default. Only those you select will be used. You don't need to supply parameters, realistic or otherwise, for inelastic models which you don't want to use in your simulations.

7.3. Cooling effects

Cooling effects (e.g. by phonon-electron coupling) are implemented by defining a 'time constant' (cooling period, t) and a target temperature (T_0, usually room temperature). Times for starting (t_1, switching on) and stopping (t_2, switching off) the cooling algorithm must also be specified. The implementation follows a simple model (see, for example, Eqs. (8.23-8.24) in Smith[1]) in which the acceleration (**a**) of each particle is modified by the addition of a friction-like term:

$$m\mathbf{a} = \mathbf{F} - \frac{\mathbf{v}}{2t}\left(1 - \frac{T_0}{T}\right) \quad (7.1)$$

where T is the current temperature, m is the particle mass, **v** is its velocity, **F** is the classical force due to other atoms.

Cooling effects are a relatively unexplored area of simulation. It is probably a good idea not to turn them on too soon (i.e. t_1 should be > 0) to prevent excessive energy loss, but no optimal value can be specified (~50 fs?).

---

[1] Atomic and Ion Collisions in Solids and at Surfaces, R. Smith (Ed.), Cambridge University Press (1996).

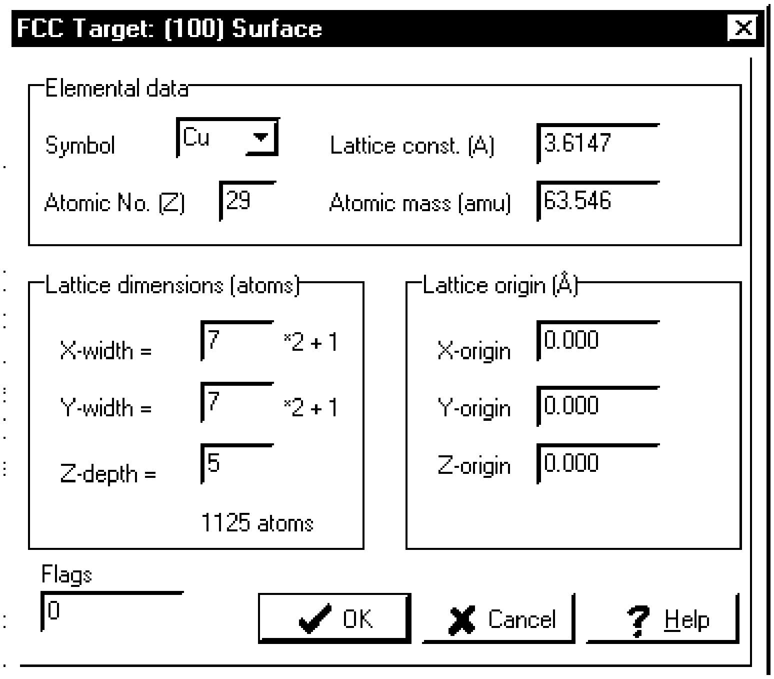
8: Sputtering Example

8.1. Description

This example will examine the sputtering of a Cu(100) surface by 1 keV Ar projectiles incident from the normal direction. The goal is to determine the sputter coefficient and energy distribution for sputtered particles. In order to keep the example tractable, we will compromise on both the target size, and on the number of projectile trajectories (runs) which we choose to study. In other respects the simulation will be similar to one that you might find in the research literature. In section 8.11 the simulation will be developed to study the effect of changing the direction of the incident projectile. Input files generated by the author can be found in the `examples\User Guide Chapter8` directory.

8.2. Target file (both SK and *Kalypso*)

- Start *Spider*, and select Target|New|Face-centred cubic|(100) surface
- Enter the parameters shown below and click OK. Save as 'cu100.trg' in a new folder.



FCC Target: (100) Surface

Elemental data

Symbol: Cu

Lattice const. (A): 3.6147

Atomic No. (Z): 29

Atomic mass (amu): 63.546

Lattice dimensions (atoms)

X-width = 7 *2 + 1

Y-width = 7 *2 + 1

Z-depth = 5

1125 atoms

Lattice origin (Å)

X-origin: 0.000

Y-origin: 0.000

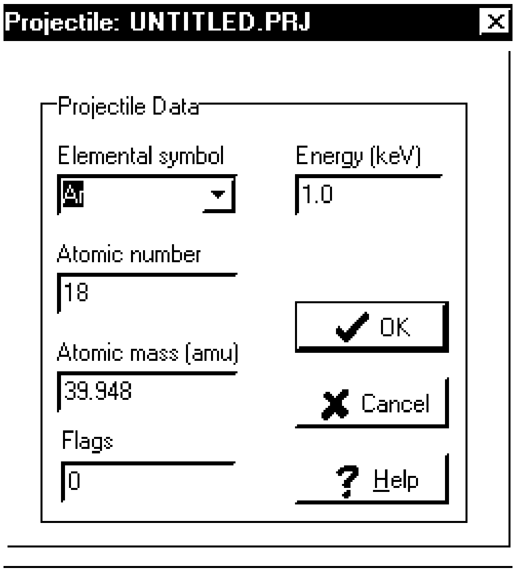
Z-origin: 0.000

Flags: 0

- Examine the target file using Target|Display/orient. Click the zoom button if you wish, and examine the target from different perspectives. You will see a 5 layer target with 225 (=15×15) atoms per layer. The *x* and *y* axes of the coordinate system are aligned with <011> rows of the target surface. The lattice anchor atom is at (0,0,0). Close the display when you are satisfied.
- This target is not very large, but it serves as a good example. In a real project you would use about 3000 atoms or more to ensure lateral containment of collision cascades.

8.3. Projectile file (SK and *Kalypso*)

- Use the Projectile|New menu command to set up the parameters for the projectile file as shown below, then save the file as 'cu100.prj' in the same directory as 'cu100.trg'.

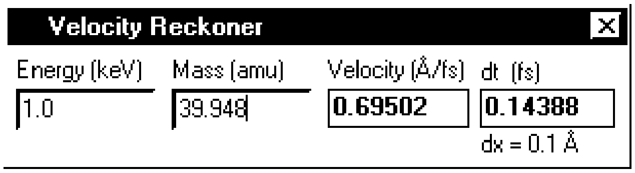


Elemental symbol: Ar
Energy (keV): 1.0
Atomic number: 18
Atomic mass (amu): 39.948
Flags: 0

8.4. Run file

8.4.1. Kalypso run file

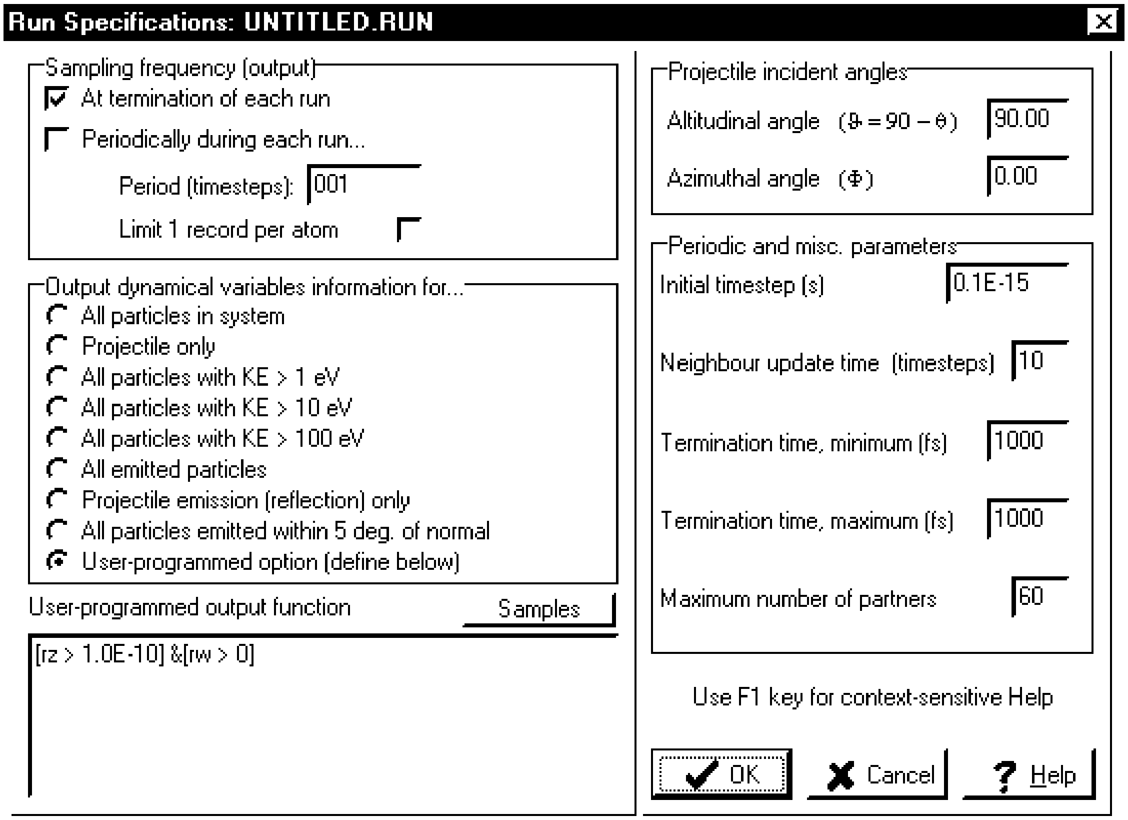
- Begin with the Utilities|Velocity reckoner command.
- Enter the energy and mass of the projectile as shown.
- Note the result for dt (= 0.14×10^{-15} s). This is the time required for a 1 keV Ar projectile to travel 0.1 Å. We will use a value slightly smaller than this for our timestep in the run file.



Energy (keV)	Mass (amu)	Velocity (Å/fs)	dt (fs)
1.0	39.948	0.69502	0.14388

dx = 0.1 Å

- Select the Run|New command, and enter the following parameters:



Run Specifications: UNTITLED.RUN

Sampling frequency (output)
☑ At termination of each run
☐ Periodically during each run...
Period (timesteps): 001
Limit 1 record per atom ☐

Output dynamical variables information for...
○ All particles in system
○ Projectile only
○ All particles with KE > 1 eV
○ All particles with KE > 10 eV
○ All particles with KE > 100 eV
○ All emitted particles
○ Projectile emission (reflection) only
○ All particles emitted within 5 deg. of normal
● User-programmed option (define below)

User-programmed output function
[rz > 1.0E-10] &[rw > 0]

Projectile incident angles
Altitudinal angle (ϑ = 90 − θ) 90.00
Azimuthal angle (Φ) 0.00

Periodic and misc. parameters
Initial timestep (s) 0.1E-15
Neighbour update time (timesteps) 10
Termination time, minimum (fs) 1000
Termination time, maximum (fs) 1000
Maximum number of partners 60

Use F1 key for context-sensitive Help

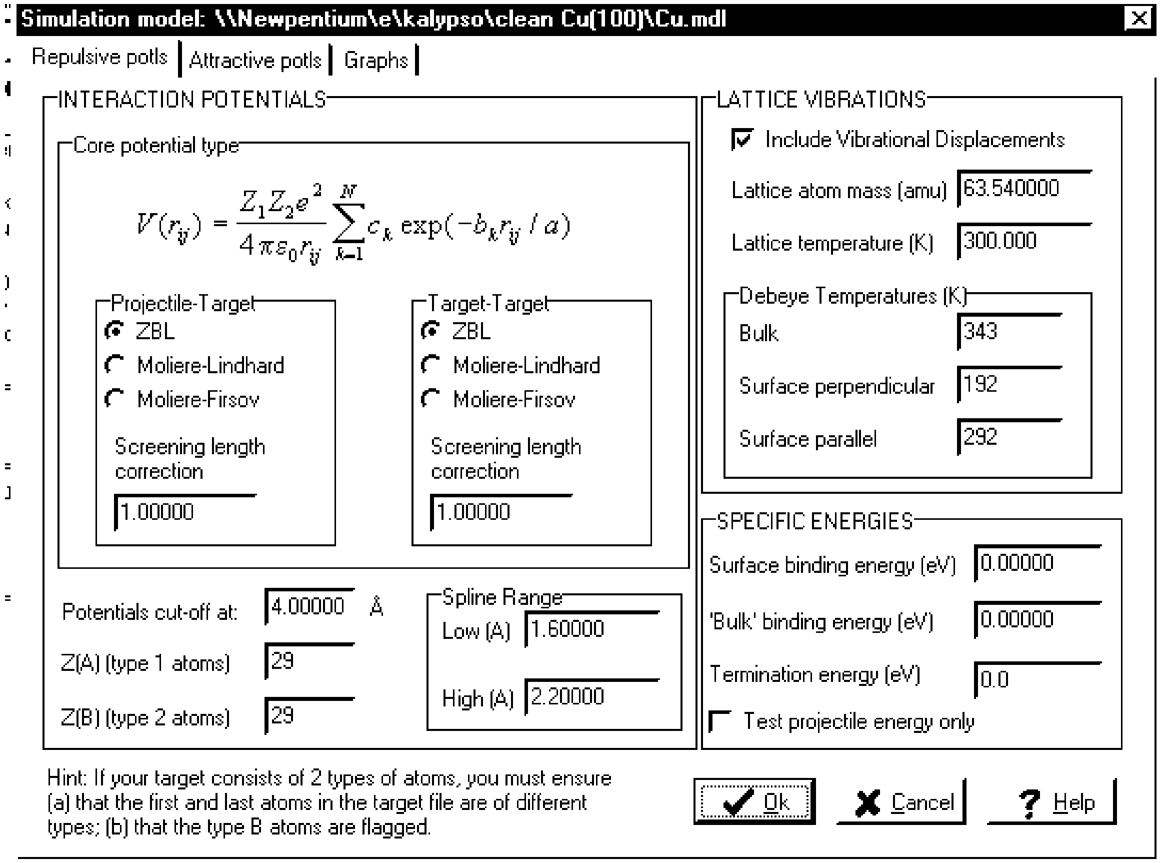
- This simulation will write data to disk at the end of each run, which will terminate after 1000 fs. The data to be written must satisfy the following condition: [rz > 1.0E-10] &[rw > 0]. This means that the z-position of the particle must be > 1 Å above the surface, and the particle must not be the projectile (which has row number, rw, = 0). The projectile is incident from the normal direction. In this case, the value specified for the azimuthal angle is obviously irrelevant. The timestep is set small enough to ensure that the projectile moves less than 0.1 Å in any time step. The neighbour update time indicates the number of timesteps between updates of the neighbour list (a value of 10 is suggested). The maximum number of partners should be set above 50, but not higher than 100 or so: this parameter allocates memory for the neighbour lists. You will be informed at runtime if it has been set too small.
- Save the file as 'cu100.run'.

8.4.2. SK run file

- The SK run file requires two extra parameters not found in the *Kalypso* run file.
- First, prepare the run file dialog box using the values given in section 8.4.1, then add the following values: Potential cut-off (in Å) = 4.0; Switching function (R_{sw}) : Select the checkbox option, and enter a value of 3.8 Å. These choices will cut the potential and forces off at a distance of 4 Å. Above 3.8 Å, a switching function will be applied to bring the Morse potential smoothly to zero at the cut-off point.
- Save the file as 'cu100.run'.

8.5. Model file (*Kalypso*)

- Because the target is elemental, only one set of target-target potential parameters is required. The parameters needed for specifying the A-B and B-B can be specified with arbitrary values.
- Use the tab control to enter the following values into the 'Repulsive potentials' and 'Attractive potentials' screens, as shown below.
- The Core potential type is chosen to be a standard (uncorrected) ZBL potential for both Cu-Cu and Ar-Cu interactions.
- All potentials will be cut off at 4 Å. The Cu-Cu ZBL potential will be joined, via a cubic spline in the region 1.6 to 2.2 Å, to the Cu-Cu attractive potential (as yet undefined).
- Vibrational effects will be included in the simulation (the parameters come from the *Spider* online Help).
- The specific energies can be ignored. Since we elected (in the run file) to terminate the simulations after 1000 fs and not before, any value set for the termination energy will be ignored.
- The atomic number for the type A atoms, Z(A), is set for Cu (29). the value set for Z(B) is irrelevant for an elemental lattice.



Simulation model: \\Newpentium\e\kalypso\clean Cu(100)\Cu.mdl

INTERACTION POTENTIALS

Core potential type

$$V(r_{ij}) = \frac{Z_1 Z_2 e^2}{4\pi\varepsilon_0 r_{ij}} \sum_{k=1}^{N} c_k \exp(-b_k r_{ij} / a)$$

Projectile-Target
● ZBL
○ Moliere-Lindhard
○ Moliere-Firsov
Screening length correction: 1.00000

Target-Target
● ZBL
○ Moliere-Lindhard
○ Moliere-Firsov
Screening length correction: 1.00000

Potentials cut-off at: 4.00000 Å
Z(A) (type 1 atoms): 29
Z(B) (type 2 atoms): 29

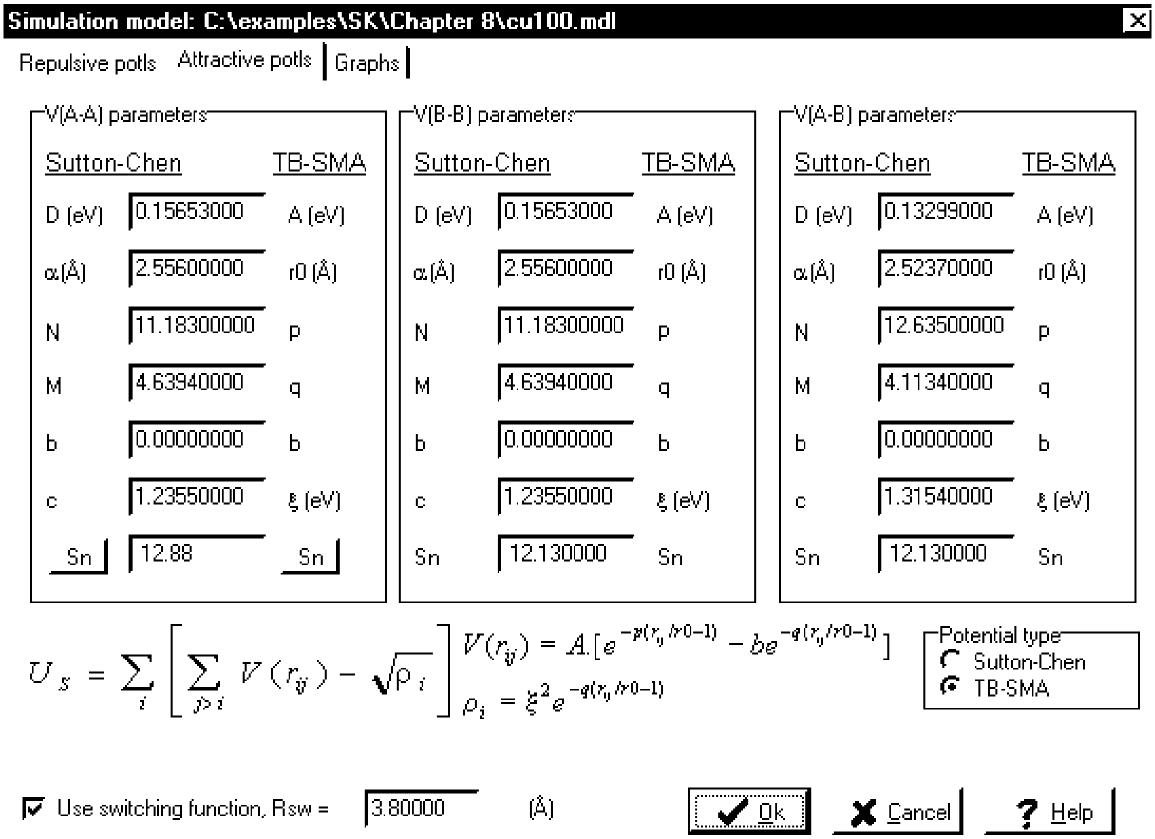
Spline Range
Low (A): 1.60000
High (A): 2.20000

LATTICE VIBRATIONS
☑ Include Vibrational Displacements
Lattice atom mass (amu): 63.540000
Lattice temperature (K): 300.000

Debeye Temperatures (K)
Bulk: 343
Surface perpendicular: 192
Surface parallel: 292

SPECIFIC ENERGIES
Surface binding energy (eV): 0.00000
'Bulk' binding energy (eV): 0.00000
Termination energy (eV): 0.0
☐ Test projectile energy only

Hint: If your target consists of 2 types of atoms, you must ensure (a) that the first and last atoms in the target file are of different types; (b) that the type B atoms are flagged.



Simulation model: C:\examples\SK\Chapter 8\cu100.mdl

V(A-A) parameters

Sutton-Chen		TB-SMA
D (eV)	0.15653000	A (eV)
α(Å)	2.55600000	r0 (Å)
N	11.18300000	p
M	4.63940000	q
b	0.00000000	b
c	1.23550000	ξ (eV)
Sn	12.88	Sn

V(B-B) parameters

Sutton-Chen		TB-SMA
D (eV)	0.15653000	A (eV)
α(Å)	2.55600000	r0 (Å)
N	11.18300000	p
M	4.63940000	q
b	0.00000000	b
c	1.23550000	ξ (eV)
Sn	12.130000	Sn

V(A-B) parameters

Sutton-Chen		TB-SMA
D (eV)	0.13299000	A (eV)
α(Å)	2.52370000	r0 (Å)
N	12.63500000	p
M	4.11340000	q
b	0.00000000	b
c	1.31540000	ξ (eV)
Sn	12.130000	Sn

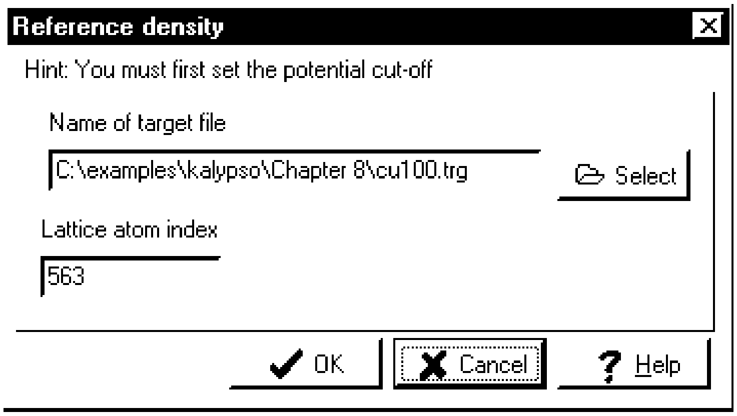
$$U_S = \sum_i \left[\sum_{j>i} V(r_{ij}) - \sqrt{\rho_i} \right] \quad \begin{matrix} V(r_{ij}) = A[e^{-p(r_{ij}/r0-1)} - be^{-q(r_{ij}/r0-1)}] \\ \rho_i = \xi^2 e^{-q(r_{ij}/r0-1)} \end{matrix}$$

Potential type
○ Sutton-Chen
● TB-SMA

☑ Use switching function, Rsw = 3.80000 (Å)

- On the attractive potentials screen, only the column of V(A-A) parameters will be used by the simulation program since the target is an element (the A-B and B-B parameters can be set to arbitrary values).

- The simulation will use a tight-binding (second moment approximation) potential. See the Simulation Primer (chapter 10) for an explanation of where the parameters A, r0, p, q, b, ξ come from. The last parameter, S_n, is a lattice sum which is calculated for the Cu(100) target: click the Sn button, and enter the name of the target file and the index (e.g. 563) of any bulk-like atom in that file:



Reference density

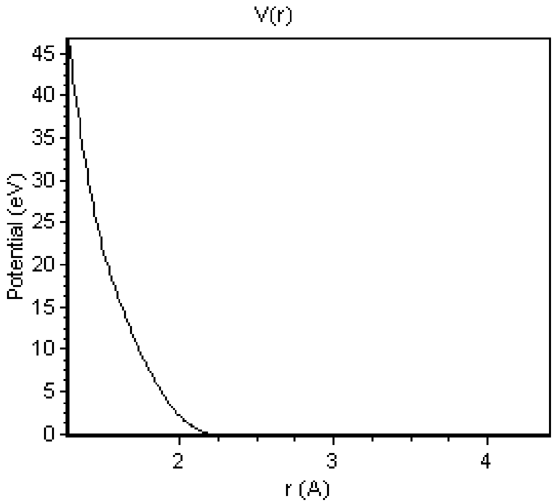
Hint: You must first set the potential cut-off

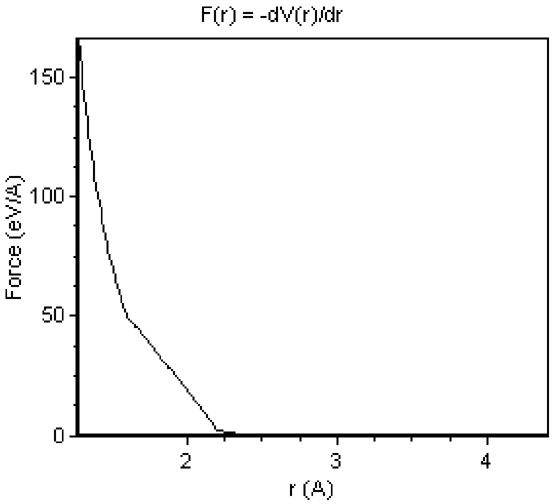
Name of target file: C:\examples\kalypso\Chapter 8\cu100.trg

Lattice atom index: 563

- After you click OK, you will get a message informing you what the reference density is (12.878...). You can also get this parameter from the compilation in the Simulation Primer document. It is not critical to specify it to high precision.

- Lastly, you can view a plot of the composite potential/force function which you have created by going to the Graph screen. The plot uses an effective pair potential approximation to the TB potential in the region r > 2.2 Å. The plotted data are stored on disk as the files force.spl and potential.spl (in case you want to save them).

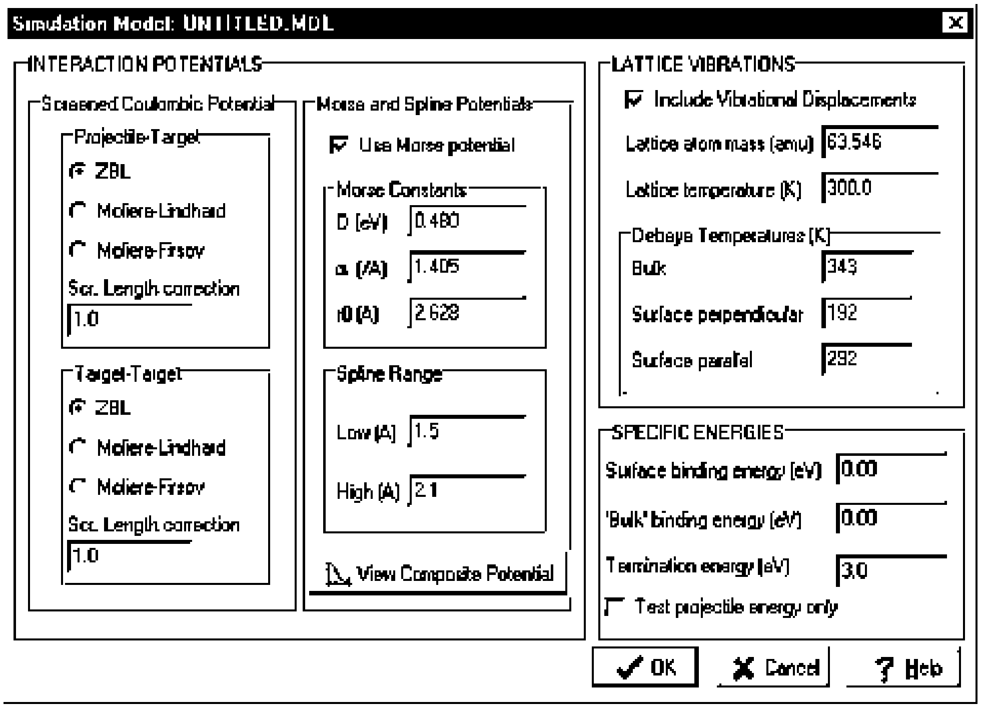




- Save the file as cu100.mdl.

8.6. Model file (*Simulation Kit*)

- There are fewer choices for a SK model file compared with *Kalypso*.
- Fill out the fields in the model file dialog as shown in the image below. Read the description of the model file for Kalyspo in the preceding section in order to find out their meaning.
- The Morse potential is taken from the literature. (The parameters differ slightly from what the author has fitted in the Simulation Primer, probably due to different values of physical data).
- Use the 'View...' button to view a graphical plot of the composite potential.
- Save as cu(100).mdl. When you do so, the root mean square vibrational amplitudes will also be reported.



Simulation Model: UNTITLED.MDL

INTERACTION POTENTIALS

Screened Coulombic Potential

Projectile-Target

● ZBL

○ Moliere-Lindhard

○ Moliere-Firsov

Scr. Length correction: 1.0

Target-Target

● ZBL

○ Moliere-Lindhard

○ Moliere-Firsov

Scr. Length correction: 1.0

Morse and Spline Potentials

☑ Use Morse potential

Morse Constants

D (eV): 0.480

α (/A): 1.405

r0 (A): 2.628

Spline Range

Low (A): 1.5

High (A): 2.1

LATTICE VIBRATIONS

☑ Include Vibrational Displacements

Lattice atom mass (amu): 63.546

Lattice temperature (K): 300.0

Debaye Temperatures (K)

Bulk: 343

Surface perpendicular: 192

Surface parallel: 292

SPECIFIC ENERGIES

Surface binding energy (eV): 0.00

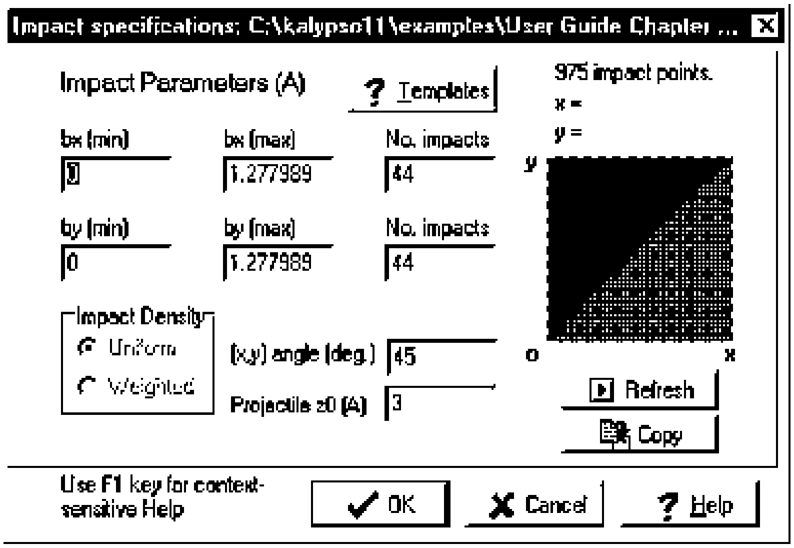
'Bulk' binding energy (eV): 0.00

Termination energy (eV): 3.0

☐ Test projectile energy only

8.7. Impact file

- The reduced impact zone is triangular in shape. See Table 6.1 and the figure below for the inputs required to create it.
- Save the data as cu100.imp.



Impact specifications; C:\kalypso11\examples\User Guide Chapter ...

Impact Parameters (A)

975 impact points.

x =

y =

bx (min): 0

bx (max): 1.277989

No. impacts: 44

by (min): 0

by (max): 1.277989

No. impacts: 44

Impact Density

● Uniform

○ Weighted

(x,y) angle (deg.): 45

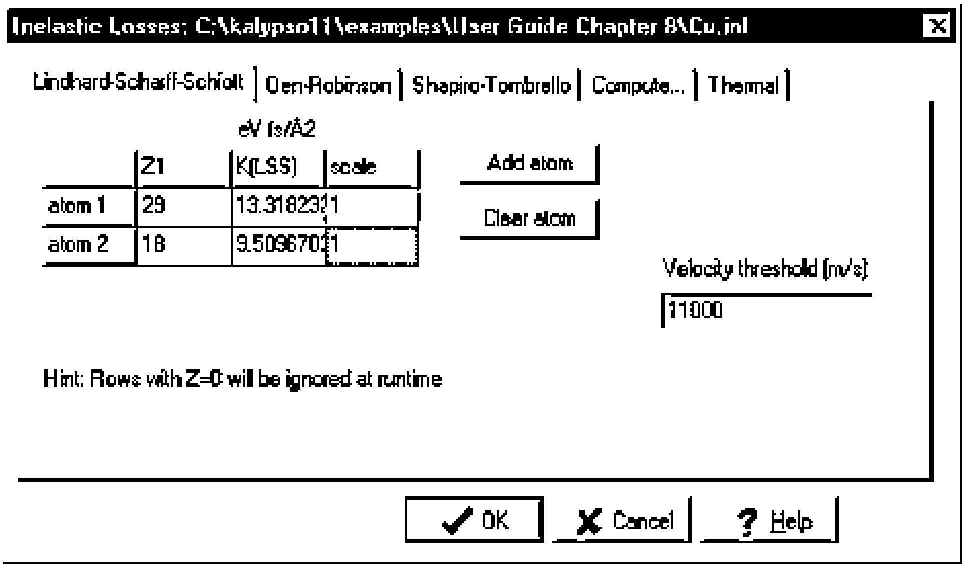
Projectile z0 (A): 3

Use F1 key for context-sensitive Help

- To examine the properties of the Impact file visually, select Impact|Display/Orient and load cu100.imp (this displays the impact points). While the latter is displayed, select cu100.trg from the Link File menu item (this superimposes the lattice sites of the target).

8.8. Inelastic file

- The procedure for creating this file, which should be saved as `cu100.inl`, was described in Chapter 7. For this example, we use the LSS model to describe the inelastic losses (Ar-Cu and Cu-Cu). This entials calculating the *K*(LSS) constant, using the "Compute..." gadget on a tab of the dialog box (see the figure below). The dialog box relevant to the LSS model is shown below with appropriate data. The OR and ST models will not be used.



Inelastic Losses: C:\kalypso11\examples\User Guide Chapter 8\Cu.inl

Lindhard-Scharff-Schiott | Oen-Robinson | Shapiro-Tombrello | Compute... | Thermal

eV fs/Å2

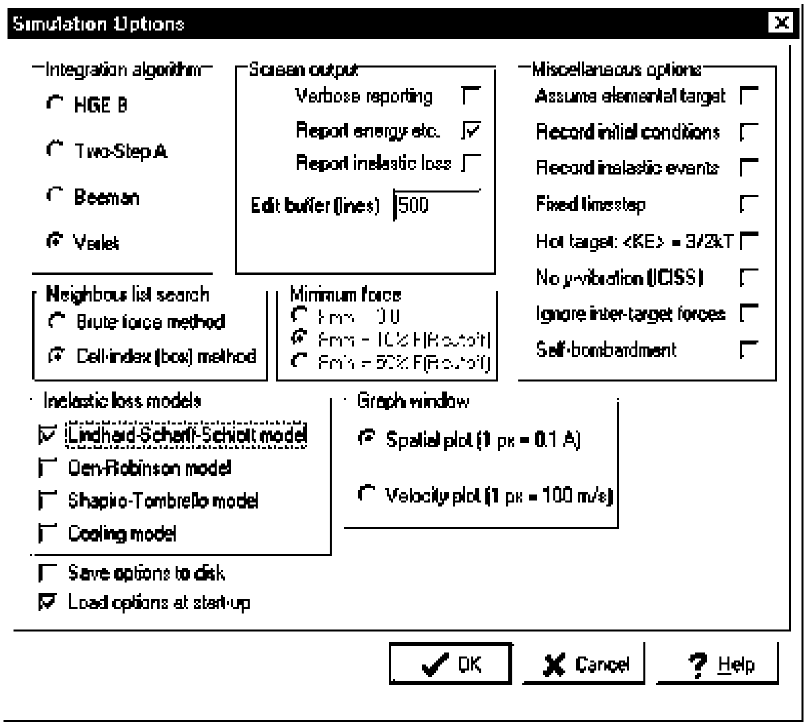
	Z1	K(LSS)	scale
atom 1	29	13.31823	1
atom 2	18	9.509670	1

Velocity threshold (m/s): 11000

Hint: Rows with Z=0 will be ignored at runtime

8.9. Running the Simulation

- Start *Kalypso* or *Snook*, as relevant, and set the simulation options as shown below.



Simulation Options

Integration algorithm
○ HGE B
○ Two-Step A
○ Beeman
● Verlet

Screen output
Verbose reporting ☐
Report energy etc. ☑
Report inelastic loss ☐
Edit buffer (lines) 500

Miscellaneous options
Assume elemental target ☐
Record initial conditions ☐
Record inelastic events ☐
Fixed timestep ☐
Hot target: <KE> = 3/2kT ☐
No y-vibration (ICISS) ☐
Ignore inter-target forces ☐
Self-bombardment ☐

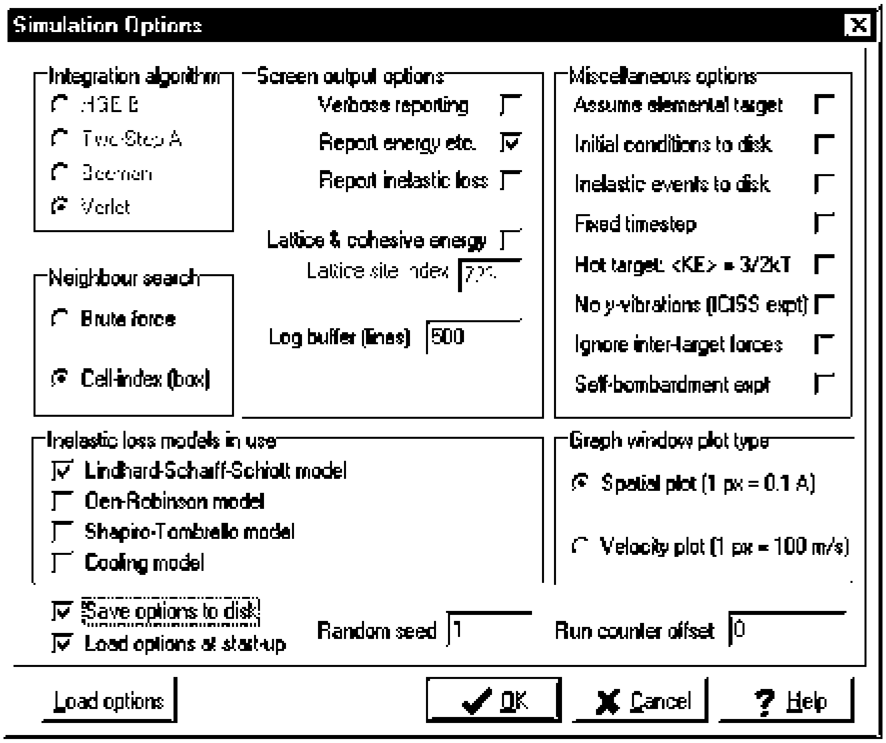
Neighbour list search
○ Brute force method
● Cell-index (box) method

Minimum force
○ Fmin = 0.0
● Fmin = 10% F(Rcutoff)
○ Fmin = 50% F(Rcutoff)

Inelastic loss models
☑ Lindhard-Scharff-Schiott model
☐ Oen-Robinson model
☐ Shapiro-Tombrello model
☐ Cooling model

Graph window
● Spatial plot (1 px = 0.1 A)
○ Velocity plot (1 px = 100 m/s)

☐ Save options to disk
☑ Load options at start-up



Simulation Options

Integration algorithm
○ HGE B
○ Two-Step A
○ Beeman
● Verlet

Neighbour search
○ Brute force
● Cell-index (box)

Screen output options
Verbose reporting ☐
Report energy etc. ☑
Report inelastic loss ☐
Lattice & cohesive energy ☐
Lattice site index 72
Log buffer (lines) 500

Miscellaneous options
Assume elemental target ☐
Initial conditions to disk ☐
Inelastic events to disk ☐
Fixed timestep ☐
Hot target: <KE> = 3/2kT ☐
No y-vibrations (ICISS expt) ☐
Ignore inter-target forces ☐
Self-bombardment expt ☐

Inelastic loss models in use
☑ Lindhard-Scharff-Schiott model
☐ Oen-Robinson model
☐ Shapiro-Tombrello model
☐ Cooling model

Graph window plot type
● Spatial plot (1 px = 0.1 A)
○ Velocity plot (1 px = 100 m/s)

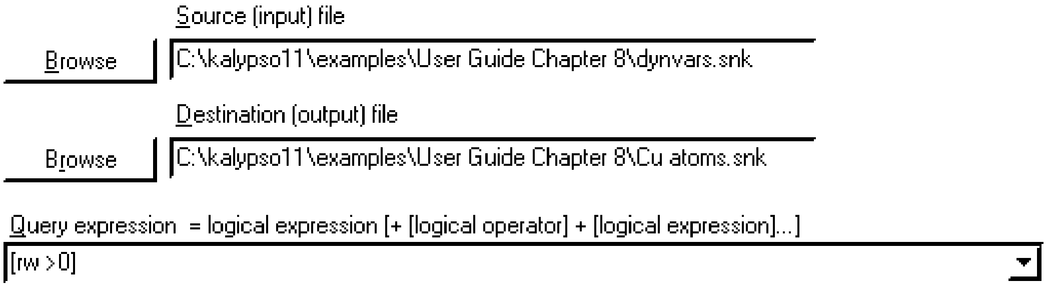
☑ Save options to disk
☑ Load options at start-up
Random seed 1
Run counter offset 0

[Simulation options for *Snook* (top) and *Kalypso* (bottom)]

- Specify the input files for *Kalypso* (or *Snook*), then run the simulation. [File|Specify Input Files, followed by File|Run]. optionally you can also Specify a location for the dynamical variables output file (`dynvars.snk`).[1]
- The simulation will take a few hours to run to completion. The output is written to `dynvars.snk` which you will need to locate for the data analysis procedure.

8.10. Data analysis with *Winnow*

- When the simulation is completed, start *Winnow*, and execute the Process|Filter menu command. Enter [rw < 0] for the filter condition, as shown below, and locate the file `dynvars.snk` on your system:



Source (input) file
C:\kalypso11\examples\User Guide Chapter 8\dynvars.snk

Destination (output) file
C:\kalypso11\examples\User Guide Chapter 8\Cu atoms.snk

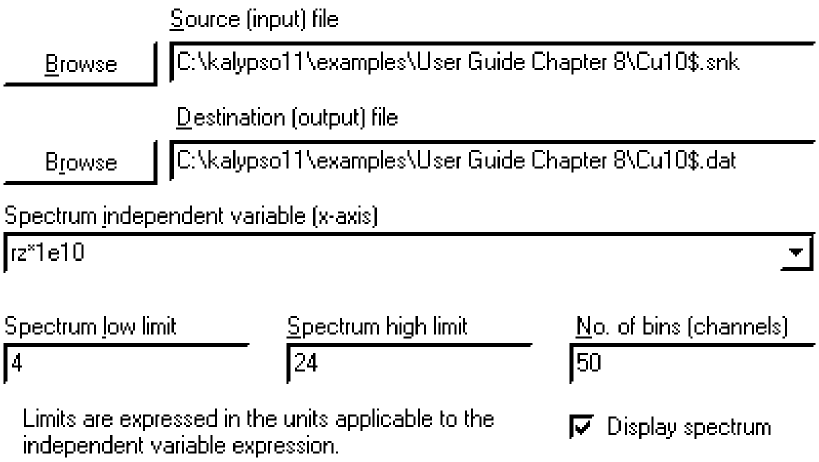
Query expression = logical expression [+ [logical operator] + [logical expression]...]
[rw >0]

- This filter will remove all projectile atoms from the input file.

---

[1] I usually put it in the same directory initially as the input files for the project.

- Next, plot the distribution of atoms along the z-axis in the output file (`Cu atoms.snk`) in the range 4-24 Å. This will help us to locate the boundary between the solid surface (including adatoms produced during sputtering) and the gas phase. Execute the menu command Process|Spectrum and complete the fields as shown:



Source (input) file: C:\kalypso11\examples\User Guide Chapter 8\Cu10$.snk

Destination (output) file: C:\kalypso11\examples\User Guide Chapter 8\Cu10$.dat

Spectrum independent variable (x-axis): rz*1e10

Spectrum low limit: 4

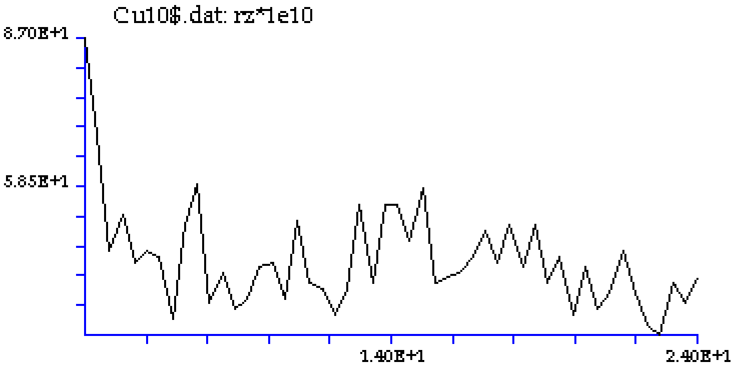
Spectrum high limit: 24

No. of bins (channels): 50

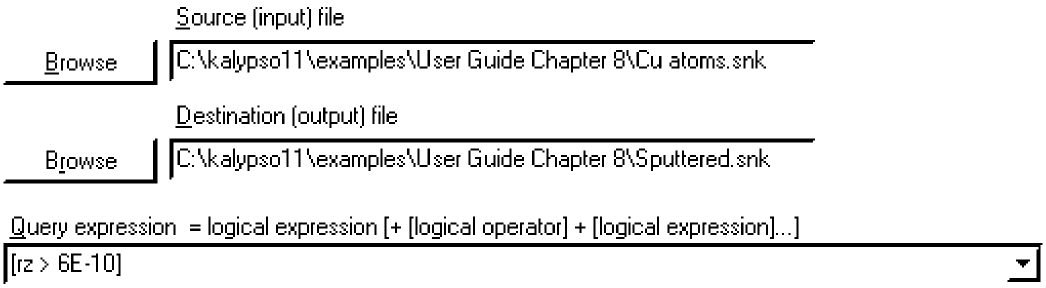
Limits are expressed in the units applicable to the independent variable expression.

☑ Display spectrum

- When you click OK you will obtain a distribution ('spectrum') like this (these data were produced by Kalypso):



- The solid/gas phase boundary is seen to be located at about $z = 6$ Å. At lower values of z, the number of atoms starts to rise sharply (you can examine this by changing the spectrum limits).
- To estimate the sputter yield, we filter out the atoms with z < 6 Å:



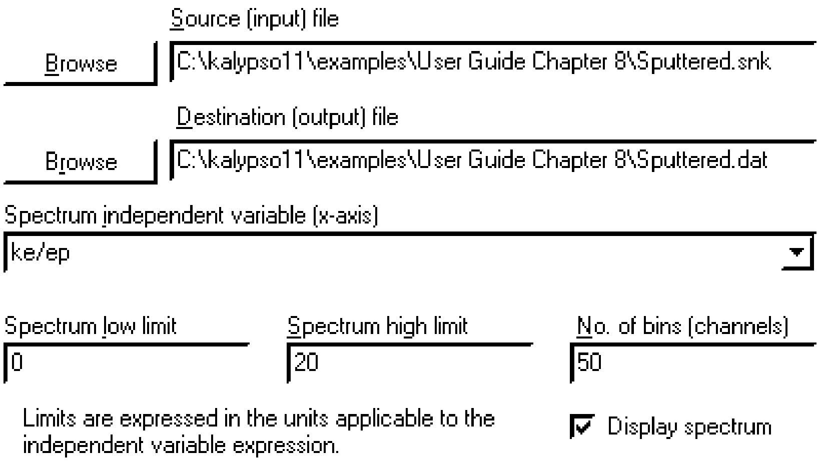
Source (input) file: C:\kalypso11\examples\User Guide Chapter 8\Cu atoms.snk

Destination (output) file: C:\kalypso11\examples\User Guide Chapter 8\Sputtered.snk

Query expression = logical expression [+ [logical operator] + [logical expression]...]

[rz > 6E-10]

- The filter routine reports: "Sputtered.snk (4567 recs. = 22.043%)". Since there were 990 incident projectile impacts in the simulation, the predicted sputter yield is 4567/990 = 4.61.[1] The experimental value is about 3. This level of discrepancy is typical for Cu.[2]
- To compute the energy distribution in the range 0-20 eV, we again use the Spectrum feature:



Source (input) file

Browse C:\kalypso11\examples\User Guide Chapter 8\Sputtered.snk

Destination (output) file

Browse C:\kalypso11\examples\User Guide Chapter 8\Sputtered.dat

Spectrum independent variable (x-axis): ke/ep

Spectrum low limit: 0

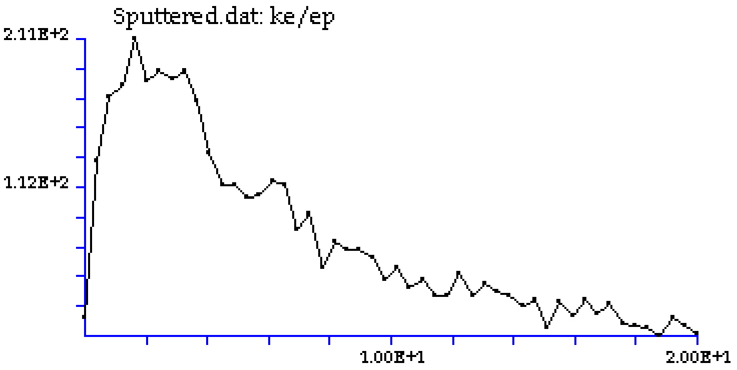
Spectrum high limit: 20

No. of bins (channels): 50

Limits are expressed in the units applicable to the independent variable expression.

☑ Display spectrum

- Here, 'ke' is the kinetic energy of a particle, while ep is the proton charge (remember, *Winnow* uses SI units in general, except for the Convert command).
- The result is shown below (the data are written to a file `Sputtered.dat` in ascii format, so you can plot them with a dedicated graphics package):



8.11. Modifying the projectile incident direction

In this final section, the project is modified to use a projectile incident along a <001> azimuthal direction at an altitudinal angle φ of 60° (i.e. 30° of the normal direction). This will entail three modifications:

- Rotation of the target lattice by 45° in the *xy* plane in order to align the <001> rows of the target with the x-direction. (The lattice was created with <011> alignment by default.)
- Setting φ to 60° in the Run file.

---

[1] When I ran the simulation, I forgot to include the lattice vibrational effects in the MDL file. The sputter yield which you obtain may be slightly higher than mine as a result.

[2] See M.A. Karolewski, Surf. Sci. 440 (1999) 87 and refs. therein.

- Modification of the impact file to conform to the symmetry of the new experimental arrangement.

To rotate the lattice, execute *Spider*'s Display/Orient command, and select the `cu100.trg` file. Enter '45' for the Z-axis rotation (top left of the dialog box), and hit the 'Test' button. Notice how this rotates the lattice shown on the graphical display. To create a rotated Target file, hit the Reload button, followed by the Apply button. Save the file under a new name (e.g. `cu100 rotated.trg`.

To change the altitudinal angle, open up the Run file, and enter '60.0' for the altitudinal angle, then save under a new name.

To change the Impact file, first consult Table 6.1. Note that the impact zone has dimensions $a/2 \times a/2$, where the lattice parameter a = 3.6147 Å ($a/2$ = 1.80735 Å) for Cu. So, create a new impact file in which *bx* and *by* both range from 0 to 1.80735 Å. Set the number of impacts at 31 along both axes (961 trajectories), and make sure that the *xy* angle is 90°. Save the file under a new name.

These files can now be substituted in the `cu100` project developed in this chapter to give a simulation of the specified geometry. The project will not be further discussed, since the procedures for running and analysing the simulation are identical to those given above. The input files modified relative to the original project can found in the `modified` sub-directory. To use them with *Kalypso*, you should proceed as described in section 8.9. However, make sure that you load the TRG, RUN and IMP files from the modified subdirectory (they have names of the form `cu100 rotated.xxx`). This can be achieved from the File|Specify Input Files command in *Kalypso*, or by clicking the icon of the file type you wish to load in the project window.