

NASCAM (NanoSCALE Modeling)

**Kinetic Monte Carlo code for the simulation of deposition,
diffusion, nucleation and growth of a film on a surface**

Plugins Manual

(release October, 2021)

S. Lucas, P. Moskovkin, J. Müller.

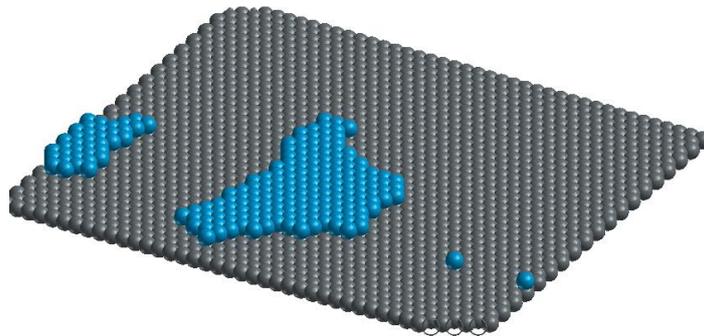


Table of contents

Table of contents	2
1 Introduction	5
2 Make substrate.....	5
2.1 Input parameters overview	5
2.2 Make substrate methods overview	6
3 Scan to Nascam	13
4 Simtra to Nascam	16
5 Thermal Evaporation.....	20
5.1 Code requests.....	20
5.2 Geometry	20
5.3 Inputs:	21
5.3.1 Rectangular source	21
5.3.2 Punctual source	22
5.4 Outputs.....	23
6 Roughness	25
7 Porosity.....	27
7.1 Algorithm.....	27
7.1.1 Data loading	27
7.1.2 Structure reconstruction	28
7.1.3 Boundaries detection	31
7.1.4 Pores detection	32
7.1.5 Pores characterization	33
7.2 NASCAM GUI example	34
7.2.1 Input parameters	34
7.2.2 JMOL results	36
7.2.3 Statistical results.....	38
8 Optics	41
8.1 Algorithm.....	41

8.1.1	Data loading	41
8.1.2	Structure reconstruction	42
8.1.3	Z discretization	44
8.1.4	Optical characterization.....	44
8.2	NASCAM GUI example	47
8.2.1	Input parameters	47
8.2.2	Optical characterization.....	50
9	Colors	54
9.1	Algorithm.....	54
9.1.1	Data loading	54
9.1.2	Chromaticity coordinates calculation.....	55
9.2	NASCAM GUI example	60
9.2.1	Input parameters	61
9.2.2	Results	62
10	Electrics.....	64
10.1	Algorithm.....	64
10.1.1	Data loading	64
10.1.2	Structure reconstruction	65
10.1.3	Finite-element calculation (incomplete description!)	65
10.2	NASCAM GUI example.....	69
10.2.1	Input parameters.....	69
10.2.2	Results	71
11	Slices	73
12	Thermal Conductivity	74
12.1	Algorithm.....	74
12.1.1	Input and Coating data loading	74
12.1.2	Calculation of a thermal conductivity of a single deposited layer.	75
12.1.3	Calculation of thermal conductivity of a film consisted of a multiple layers	76
12.2	NASCAM GUI example.....	76
12.2.1	Input parameters.....	76
12.2.2	Results	78



Innovative Coating Solutions,
Place Saint-Pierre, 11, B-5380 Forville., Belgium
slu@incosol4u.com www.incosol4u.com

12.2.3	Temperature profile across the coating.....	80
13	References	83

1 Introduction

NASCAM is developed to simulate the time evolution of atoms deposited on a substrate.

In this manual, the reader will find a list of plugins that either analyze the NASCAM film growth simulation results, or help the user to build input (e.g. substrates).

For each plugin ordered, there is a help file with the last changes.

2 Make substrate

This code is used for creating an initial structure made of either substrate atoms or deposited atoms. It generates different types of structures: rows, columns, and hemispheres. The user determines the type, size and number of objects. The code generates an *xyz file* which stores information about atom positions in Cartesian coordinates (name_of_atom, x, y, z) in JMOL format, which can be used as an input to NASCAM and for visualization.

2.1 Input parameters overview

By loading the *Make Substrate* plugin, you will be able to access four groups of parameters:

- **the method:** enables to choose if you want to create a new substrate, load a file with atoms coordinates (provided for example by AFM scans), or modify an already existing substrate (by adding a new pattern, changing the size or duplicating it). More details are given below.
- **the files parameters:** enables to choose the name of the output files, and (for some method options) the name and the format of the input file (either atoms coordinates text file, or already existing xyz files provided by NASCAM or by *Make Substrate* itself). After computation, the output files are stored in the output and save folders, but the output folder content is erased if you relaunch a new simulation.
- **the substrate mesh parameters:** enables to define the meshing option and, for some methods, the size of the substrate.
- **the species:** for some method options, enables to define the material of the film (first monolayer) and of the pattern.
- **the pattern parameters:** for the "create" and "update" methods, enables to define the type of pattern (spheres, cones, ...), the method (extrude or subtract), and the parameters of the pattern. An option called "active random parameters" allows creating pseudo-periodic to random patterns.

2.2 Make substrate methods overview

Five methods are available now:

- **Create substrate**

This option enables to create simple periodic (Figure 1) or pseudo-periodic (Figure 2) geometrical patterns.

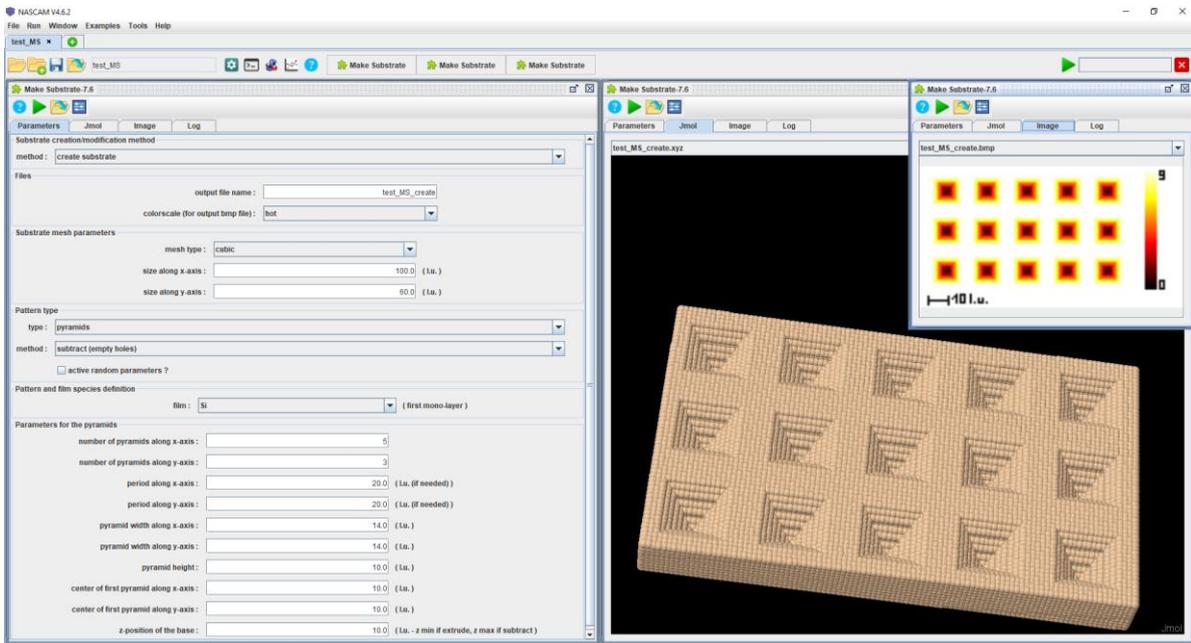


Figure 1: Create substrate option – subtraction of periodic pyramids

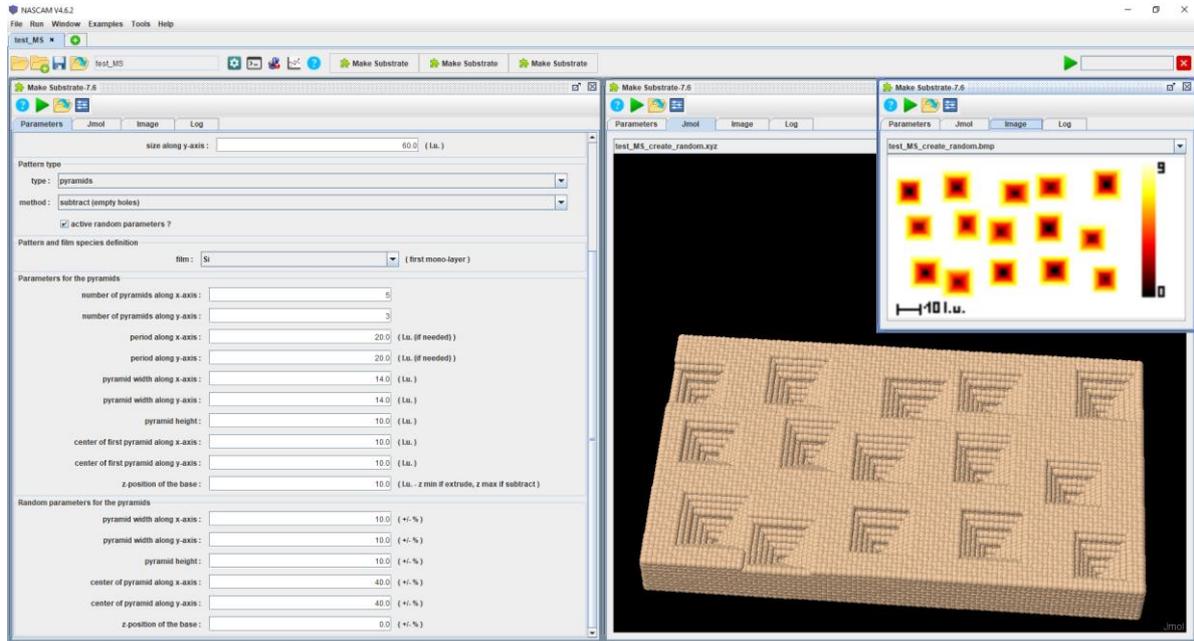


Figure 2: Create substrate option – subtraction of random pyramids

- **Convert coordinates to substrate**

This option allows to load files with atoms coordinates (provided for example by digitizing an AFM picture). The coordinate file can be 2D or 3D. In the current version of *Make Substrate*, the X and Y coordinates have to be in lattice unit already (1, 2, 3...) with a regular space step (future version will solve this by activating the “interpolation” option). However, it is still possible to rescale it if the asked number of atoms in the X and Y directions is smaller than the number of pixels (the Z coordinates can be rescaled freely). Figure 4 shows the case of an “artistic” structure: the loaded *.csv file* (where the coordinates are stored) is generated from a greyscale picture thanks to the freeware [imageJ](#) (after loading the picture in [imageJ](#), select the area you want to digitize and click on Analyze/Tools/Save XY Coordinates), as shown in Figure 3.

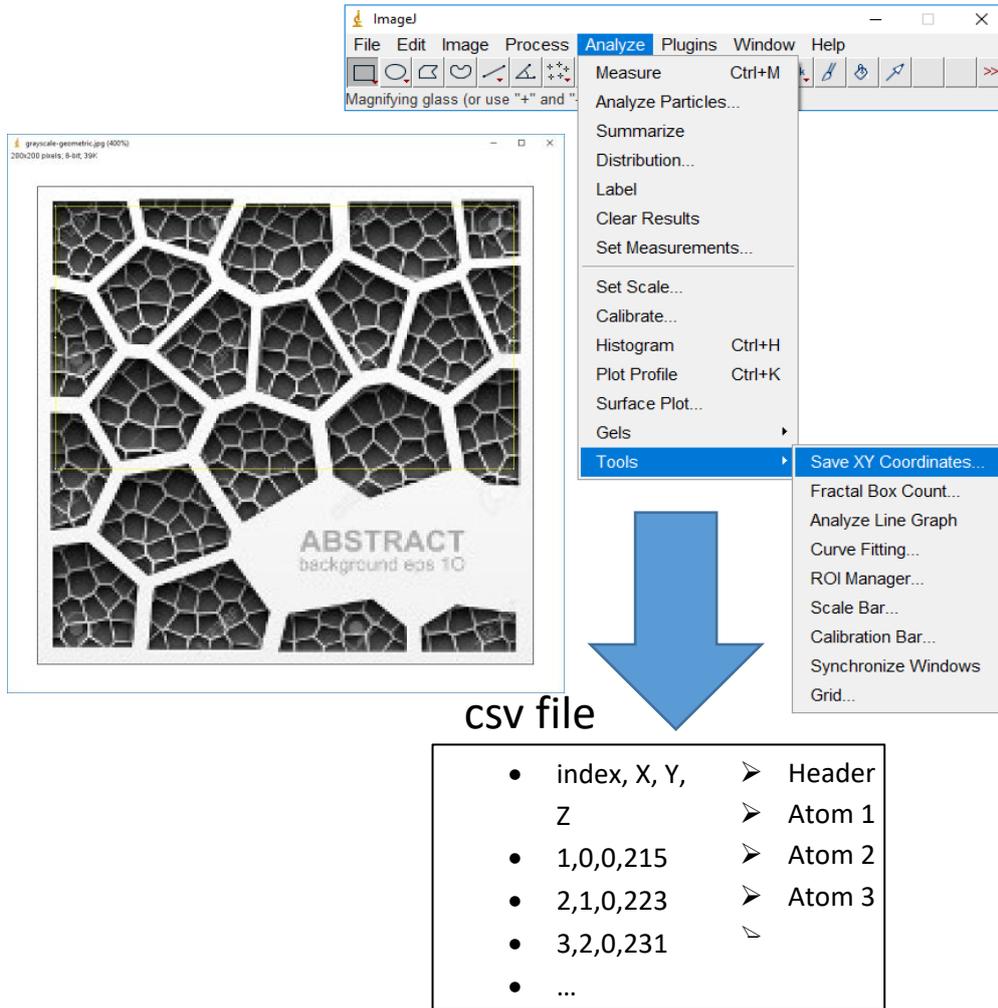


Figure 3: example of txt/csv file with atom coordinates (1 line header, “?,x,y,z” format) generated by imageJ from a random “artistic” picture

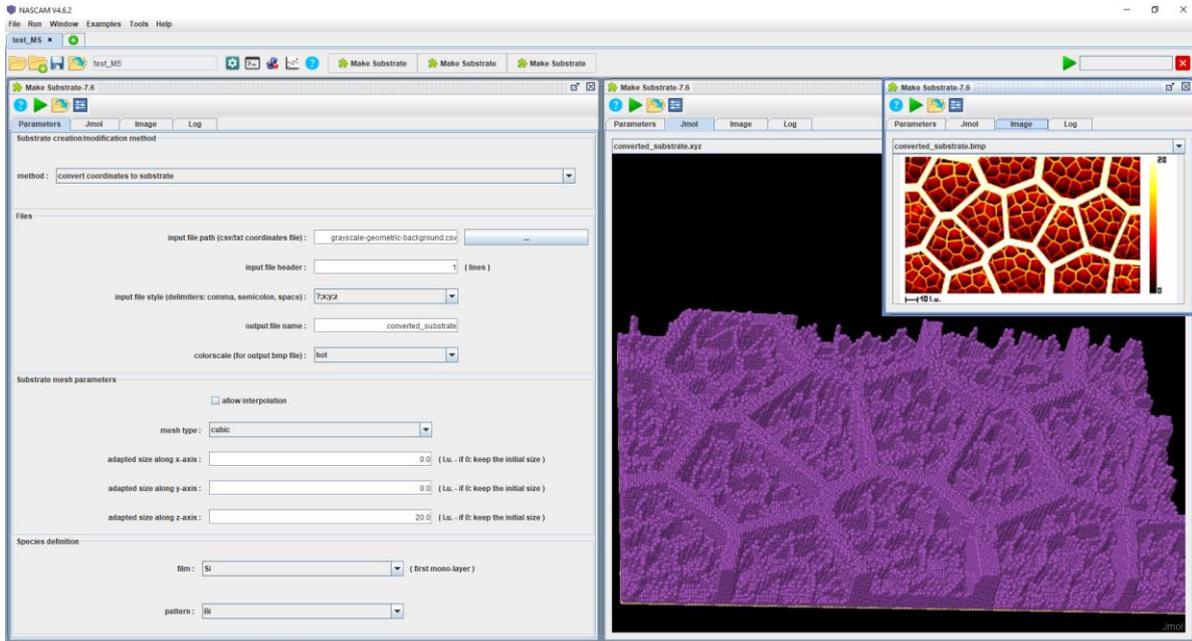


Figure 4: Convert coordinates to substrate option – from an “artistic” image (after digitalization by imageJ)

- **Update pattern**

This option enables to add a new pattern to an already existing substrate. Figure 5 shows how to update the substrate displayed in Figure 1 by adding spheres.

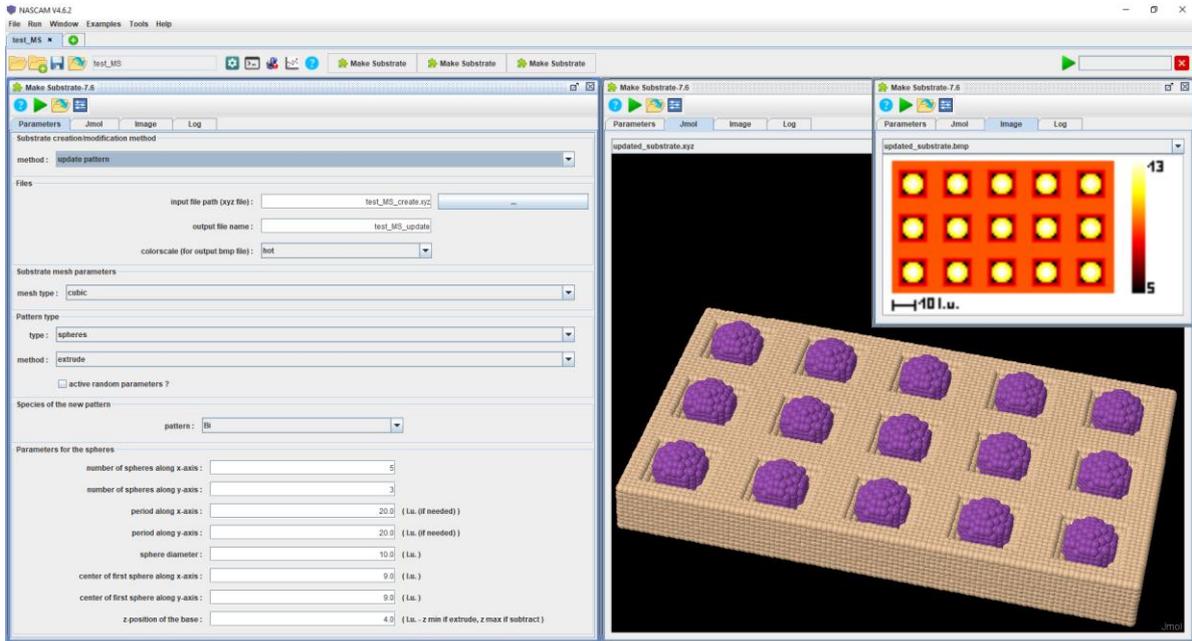


Figure 5: Update pattern option – extrude spheres

- **Extend/truncate substrate**

This option allows changing the size of an already existing substrate. Figure 6 and Figure 7 show how to extend and truncate (respectively) the substrate displayed in Figure 1.

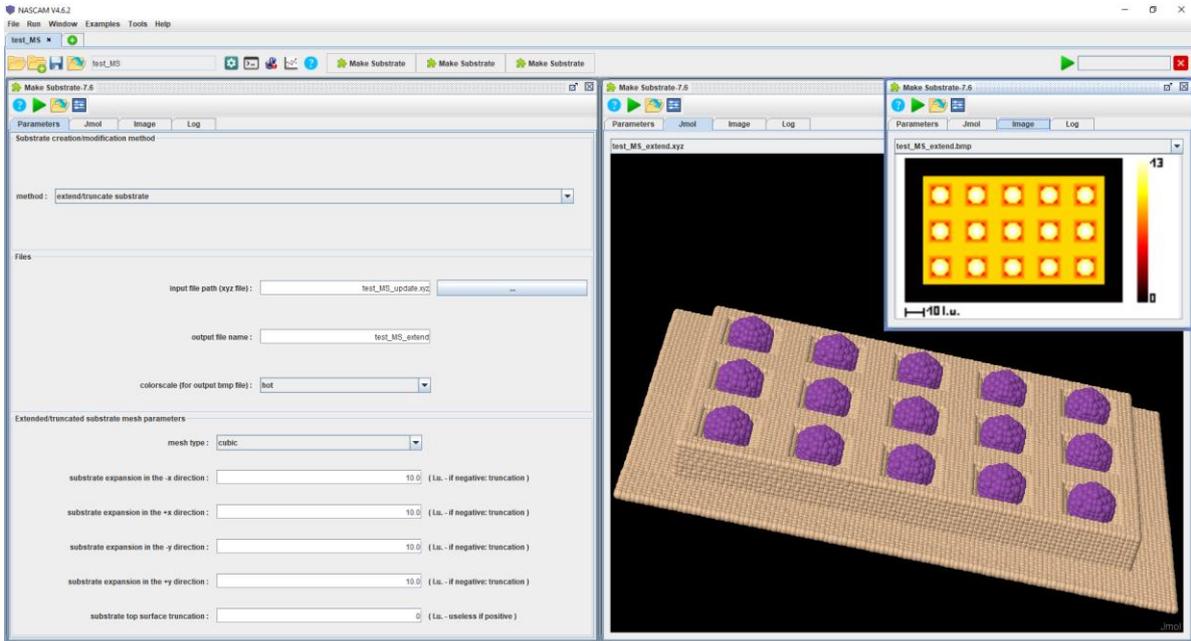


Figure 6: Extend substrate option (positive expansion parameters)

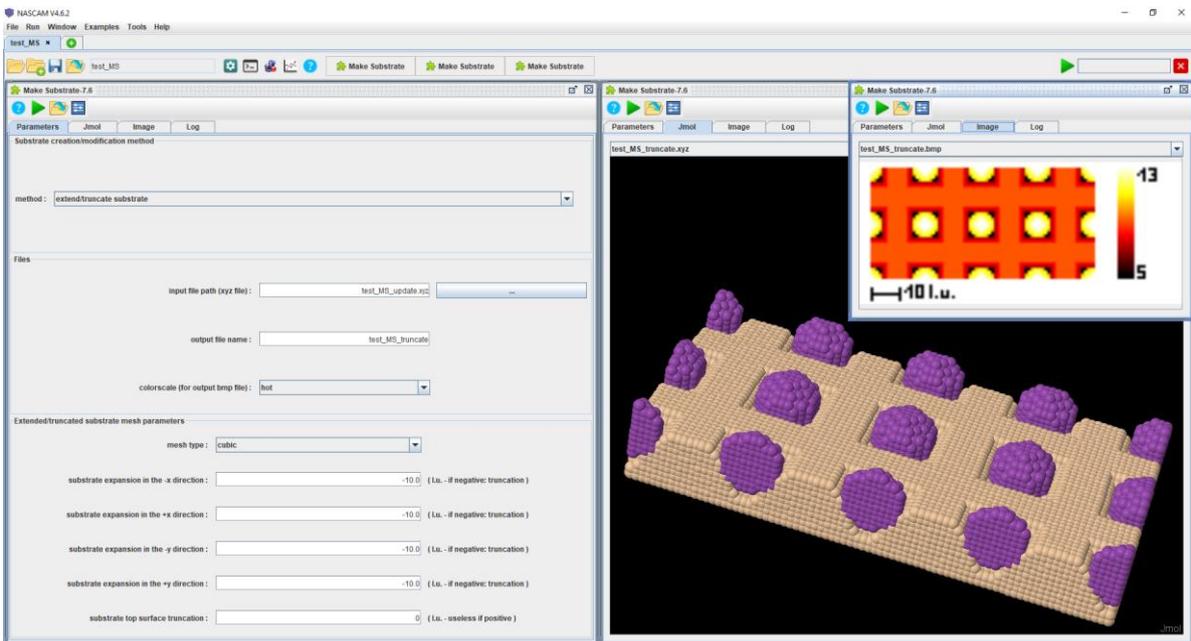


Figure 7: Truncate substrate option (negative expansion parameters)

- Duplicate substrate

This option enables to duplicate an already existing substrate periodically. Figure 8 shows how to duplicate the substrate displayed in Figure 1 (two periods in each direction).

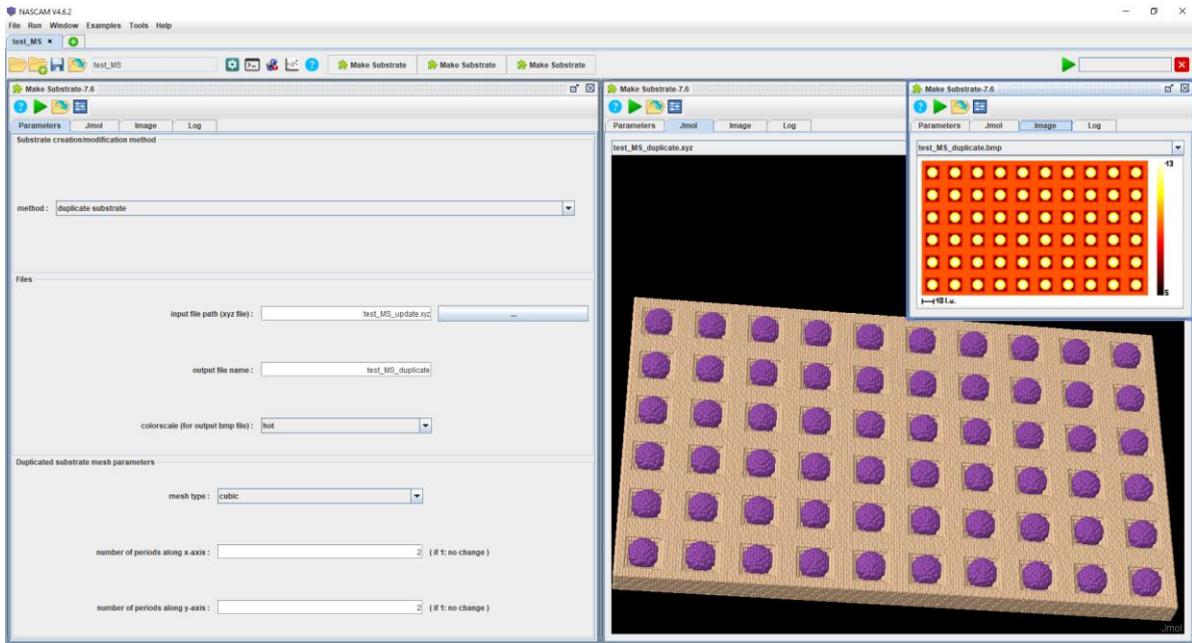


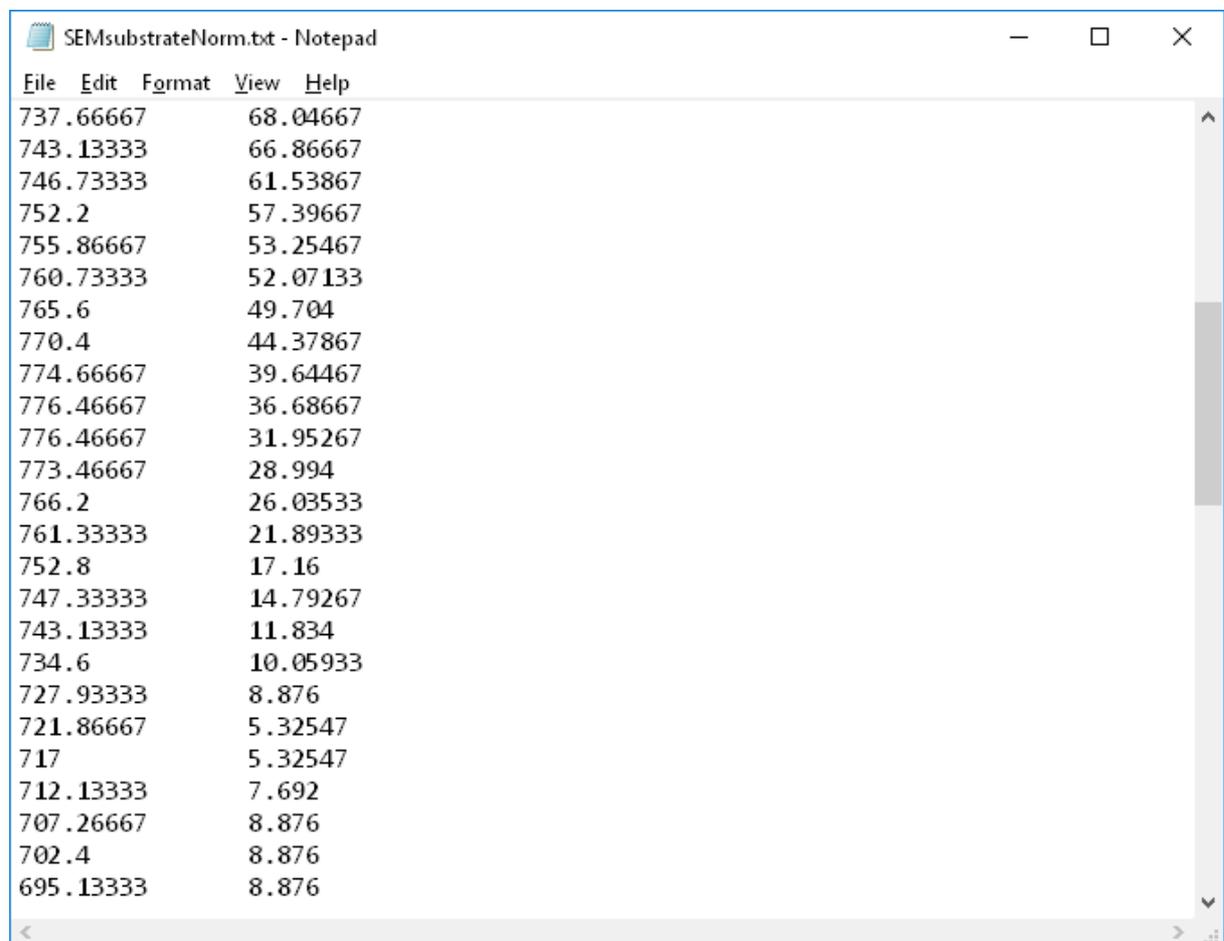
Figure 8: Duplicate substrate option

3 Scan to Nascam

This plugin can be used to create an initial structure file from a set of (x,z) data points. The plugin is useful to convert the data obtained from, for example, a sample cross-section into an input *substrate* file to NASCAM.

There are two steps in the conversion.

1. Scan the image of the cross-section and store the data in a file. Here is an example of how the file looks like (see also an example included in the "*SEMsubstrateNorm.txt*" package):



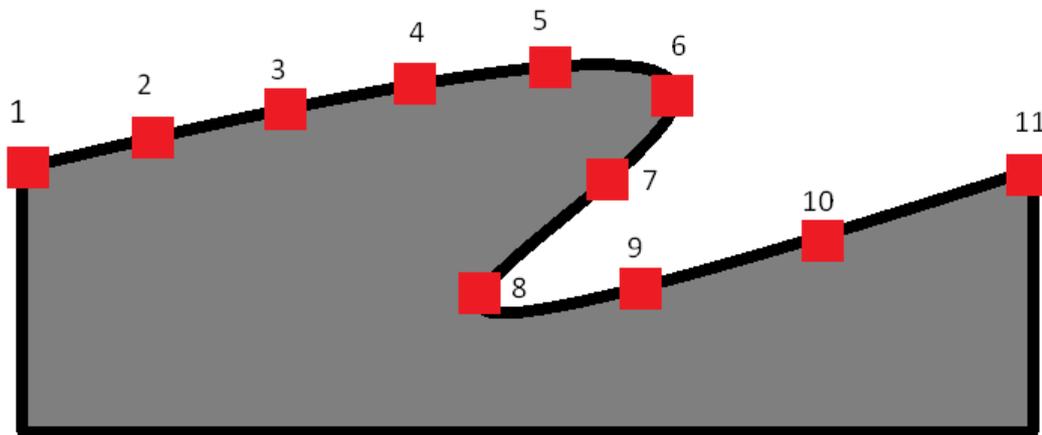
```
SEMsubstrateNorm.txt - Notepad
File Edit Format View Help
737.66667 68.04667
743.13333 66.86667
746.73333 61.53867
752.2 57.39667
755.86667 53.25467
760.73333 52.07133
765.6 49.704
770.4 44.37867
774.66667 39.64467
776.46667 36.68667
776.46667 31.95267
773.46667 28.994
766.2 26.03533
761.33333 21.89333
752.8 17.16
747.33333 14.79267
743.13333 11.834
734.6 10.05933
727.93333 8.876
721.86667 5.32547
717 5.32547
712.13333 7.692
707.26667 8.876
702.4 8.876
695.13333 8.876
```

2. The substrate file generated by the plugin has the following dimensions:

(X): the size in X-direction is defined by the input *scan* file. The X-size is equal to the difference between maximum and minimum values of x coordinate in the *scan* data file. Therefore, when digitizing the image, it is necessary to set the x coordinates of the left and right sides of the scan properly. In the example included in the package, $X_{min} = +3.2680e-001$ and $X_{max} = +9.9673e+001$ thus the dimension of the substrate is 100.

(Y): the size in Y-direction is always equal to 4.

IMPORTANT! The order of the lines in the *scan* file is very important. Data points MUST BE ordered in the way that, when connected in succession, they produce a sketch of the surface! For example, for the surface shown below, the points in the *scan* file must be ordered as shown in the figure below. To order the data points with an increasing of x-coordinates would be an error.



If you use the *UnScan* software to digitize the scan, you will get a wrong scan file, as *UnScan* always orders the data points with an increasing of x-coordinate.

3. Use the plugin to convert the file into a NASCAM *substrate* file.



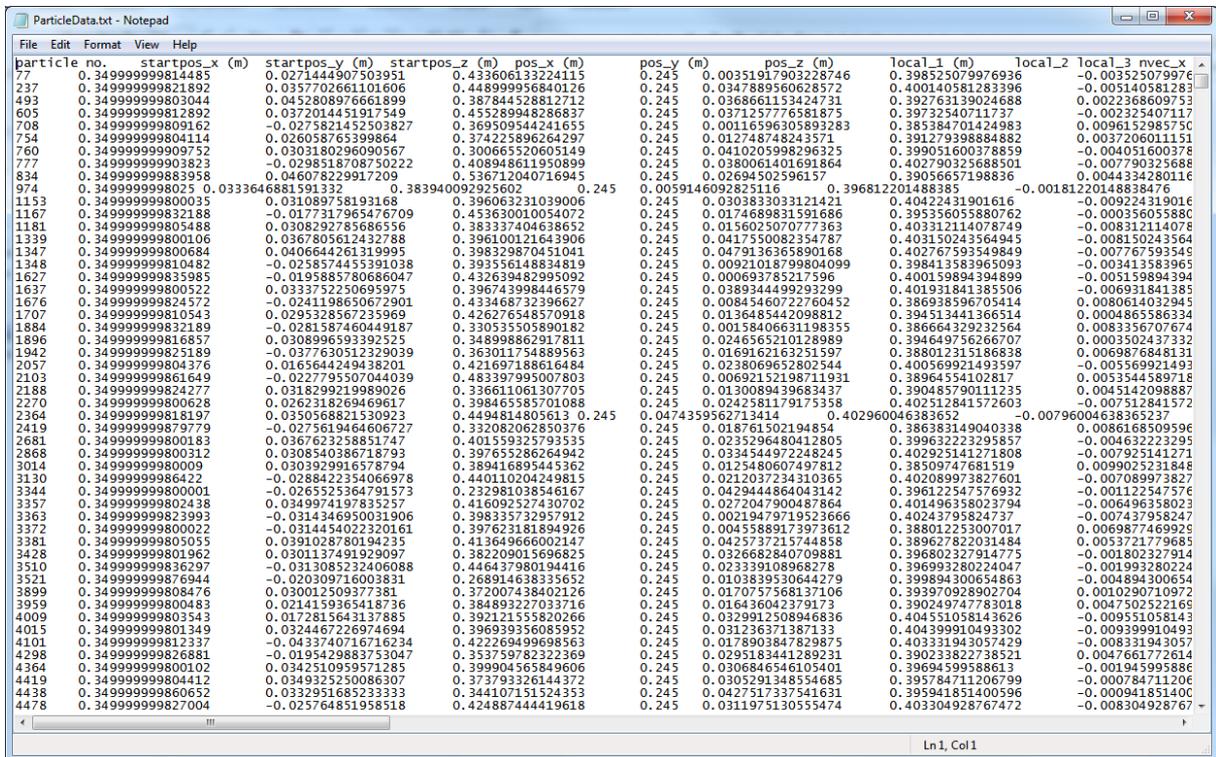
Innovative Coating Solutions,
Place Saint-Pierre, 11, B-5380 Forville., Belgium
slu@incosol4u.com www.incosol4u.com

PS. The *substrate.xyz* file stores information about atom positions in Cartesian coordinates (name_of_atom, x, y, z) in JMOL format.

4 Simtra to Nascam

This utility is used to create energy and angular distribution input files for NASCAM from SIMTRA v.2.1¹ [1] with all the steps evenly distributed. This is a requirement for NASCAM.

As an input *Simtra2Nascam* uses the *Simtra ParticleData.txt* output file. This file is located in the **depositionDummyObjects/Object/Surface** folder, where Object and Surface are the object and surface of interest. The *ParticleData.txt* file contains information about the vectors of velocity at which atoms hit the detector-sample (see the following figure for an example).



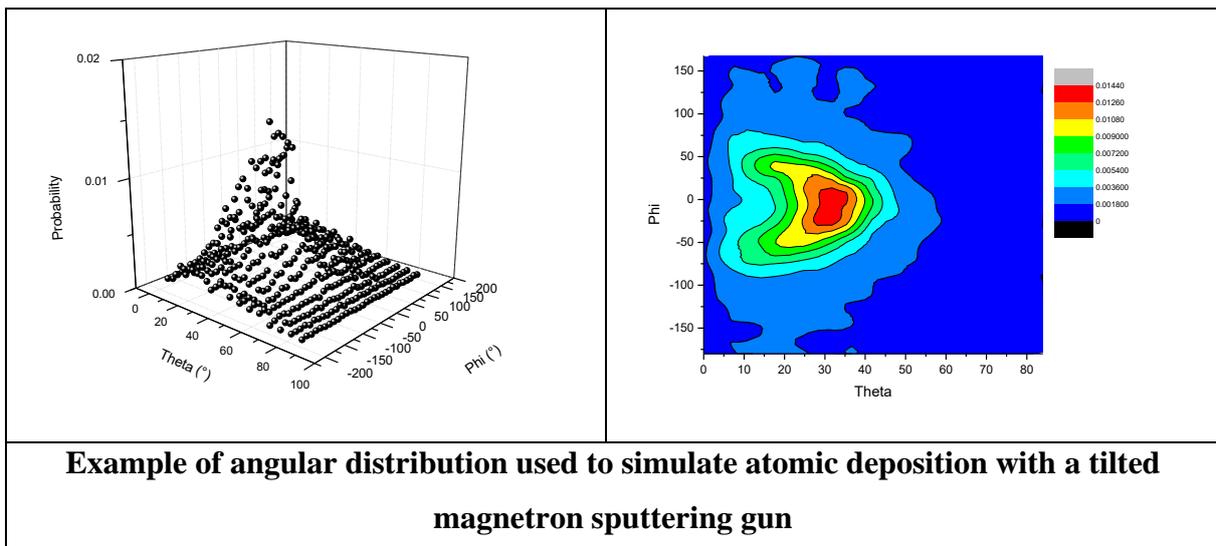
Simtra2Nascam.exe then extracts the information from the *ParticleData.txt* file in two NASCAM compatible files:

1. The *A_DST.TXT* file, containing the angular distribution of the deposited atoms. The column names are line number, theta, phi, probability (see next figure).

¹ **Simulation of Metal Transport**, available at <http://www.draft.ugent.be/>.

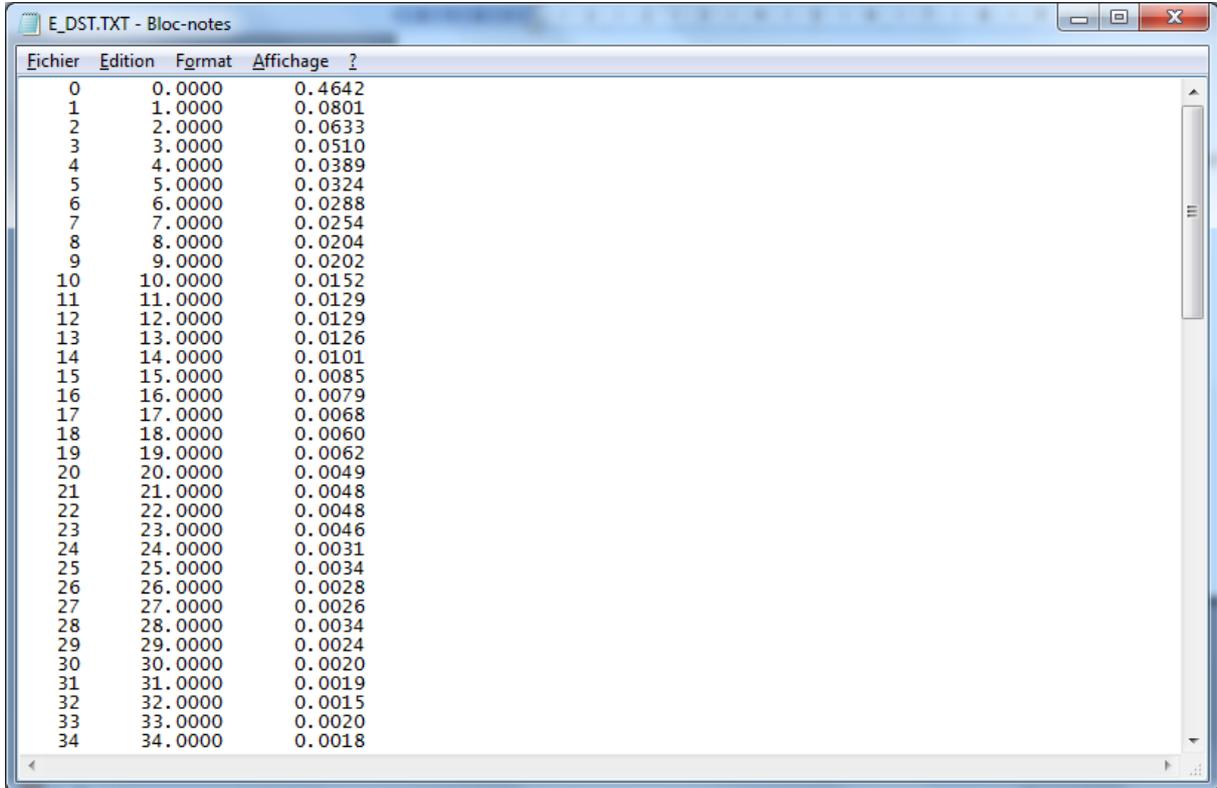
Fichier	Edition	Format	Affichage	?
0	0.0000	-180.0000	0.0002	
1	0.0000	-168.0000	0.0003	
2	0.0000	-156.0000	0.0001	
3	0.0000	-144.0000	0.0004	
4	0.0000	-132.0000	0.0003	
5	0.0000	-120.0000	0.0001	
6	0.0000	-108.0000	0.0003	
7	0.0000	-96.0000	0.0003	
8	0.0000	-84.0000	0.0004	
9	0.0000	-72.0000	0.0003	
10	0.0000	-60.0000	0.0002	
11	0.0000	-48.0000	0.0001	
12	0.0000	-36.0000	0.0003	
13	0.0000	-24.0000	0.0005	
14	0.0000	-12.0000	0.0002	
15	0.0000	-0.0000	0.0005	
16	0.0000	12.0000	0.0002	
17	0.0000	24.0000	0.0004	
18	0.0000	36.0000	0.0002	
19	0.0000	48.0000	0.0004	
20	0.0000	60.0000	0.0001	
21	0.0000	72.0000	0.0001	
22	0.0000	84.0000	0.0003	
23	0.0000	96.0000	0.0001	
24	0.0000	108.0000	0.0002	
25	0.0000	120.0000	0.0002	
26	0.0000	132.0000	0.0005	
27	0.0000	144.0000	0.0004	
28	0.0000	156.0000	0.0002	
29	0.0000	168.0000	0.0005	
30	6.0000	-180.0000	0.0009	
31	6.0000	-168.0000	0.0004	
32	6.0000	-156.0000	0.0009	
33	6.0000	-144.0000	0.0008	
34	6.0000	-132.0000	0.0006	

The following figures show a 3D representation of an *A_DST.TXT* file



2. The *E_DST.TXT* file, containing the energy of the deposited atoms (see next figure).

The columns names are line number, energy, probability.



Fichier	Edition	Format	Affichage	?
0	0.0000			0.4642
1	1.0000			0.0801
2	2.0000			0.0633
3	3.0000			0.0510
4	4.0000			0.0389
5	5.0000			0.0324
6	6.0000			0.0288
7	7.0000			0.0254
8	8.0000			0.0204
9	9.0000			0.0202
10	10.0000			0.0152
11	11.0000			0.0129
12	12.0000			0.0129
13	13.0000			0.0126
14	14.0000			0.0101
15	15.0000			0.0085
16	16.0000			0.0079
17	17.0000			0.0068
18	18.0000			0.0060
19	19.0000			0.0062
20	20.0000			0.0049
21	21.0000			0.0048
22	22.0000			0.0048
23	23.0000			0.0046
24	24.0000			0.0031
25	25.0000			0.0034
26	26.0000			0.0028
27	27.0000			0.0026
28	28.0000			0.0034
29	29.0000			0.0024
30	30.0000			0.0020
31	31.0000			0.0019
32	32.0000			0.0015
33	33.0000			0.0020
34	34.0000			0.0018

Then, these two files can be used as an input for NASCAM simulations. Do not forget to set the appropriate filenames in the *input.txt* file so that NASCAM can use them (see the NASCAM manual).

The code can be used as a part of NASCAM_GUI or a standalone. If the code is used as a standalone, it is necessary to create:

an “input” subfolder and copy the “*ParticleData.txt*” file in this subfolder;

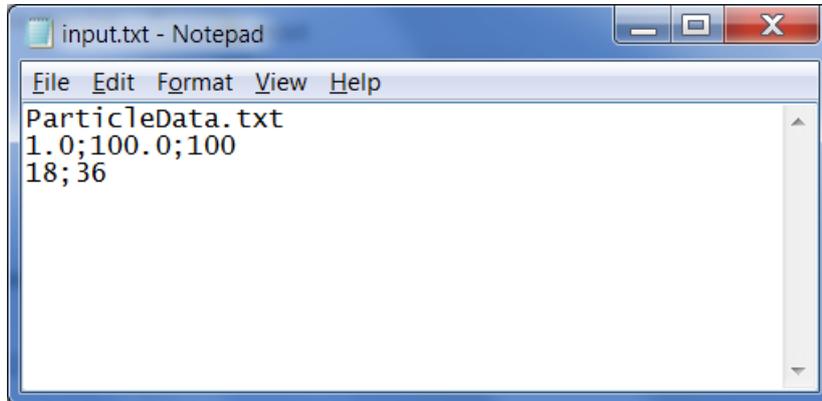
an “output” subfolder for output files.

In addition, an *input.txt* file must be created, with the following parameters (see figure below):

The first line is the name of the Simtra output file;

The second line specifies the energy range, minimum and maximum values in eV and number of intervals in which the whole range [min; max] is divided. Thus the step is (max - min)/number of intervals;

The third line specifies the numbers of intervals in which the whole ranges are divided for THETA $[0; \pi/2]$ and PHI $[-\pi; \pi]$.



```
input.txt - Notepad
File Edit Format View Help
ParticleData.txt
1.0;100.0;100
18;36
```

Content of an *input.txt* file for the *Simtra2Nascam* tool

The use of *Simtra2Nascam* is simple:

- Place *Simtra2Nascam* and the *input.txt* in the same folder and *ParticleData.txt* “input” subfolder.
- Double-click to run *Simtra2Nascam.exe*.

5 Thermal Evaporation

The purpose of the *Thermal Evaporation* plugin is to define a thermal evaporation source (punctual or rectangular) and calculate the emission probability of evaporated material that will condense on the substrate.

5.1 Code requests

- The source position with respect to the substrate position.
- The dimensions of the source in the rectangular case (see "Geometry" below).
- The shape of the vapour cloud emitted by the source: The punctual source emits according to a cosine law: $P(\theta) = \cos^n \theta$. For the rectangular case, the surface is divided in rectangular meshes emitting like a punctual source.

5.2 Geometry

- The substrate is located at (0 , 0 , 0) coordinates and the visible surface points in the $z < 0$ direction.
- The source has to be placed at the x,y and z ($z > 0!!$) coordinates.

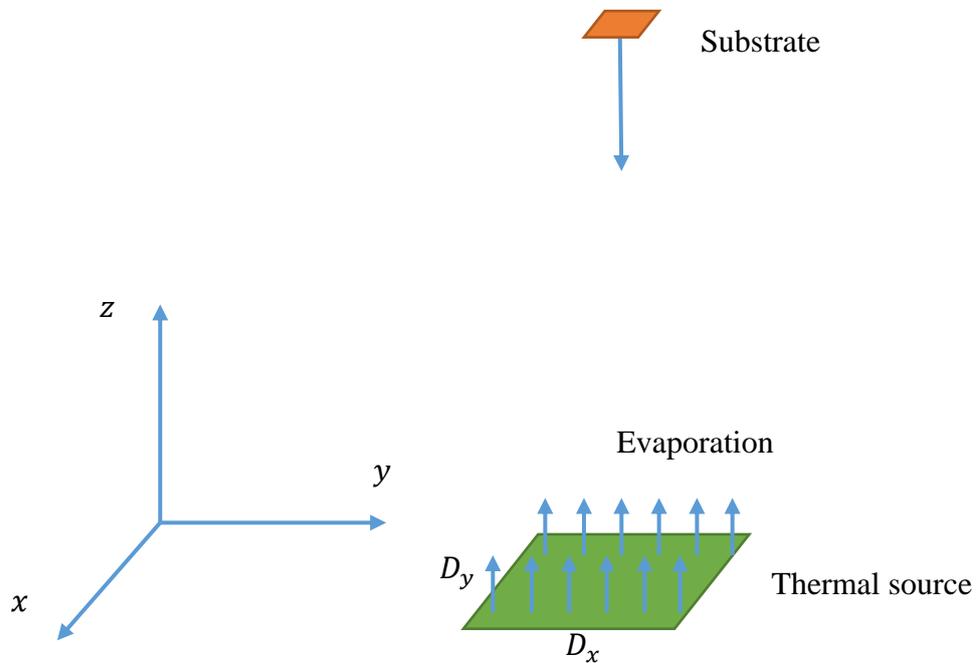
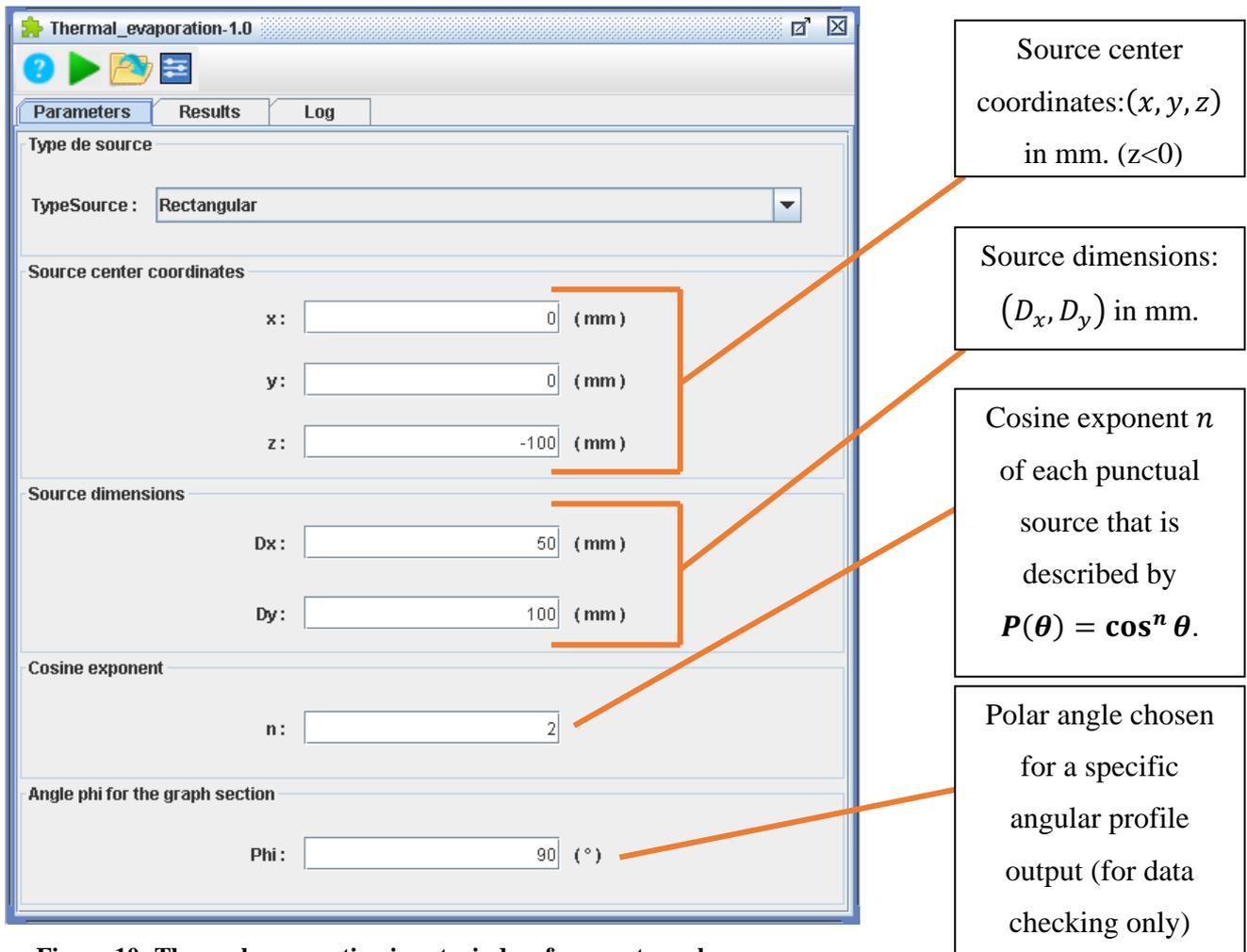


Figure 9: Geometry of the thermal evaporation plugin.

- By default, the source is not tilted (see Figure).

5.3 Inputs:

5.3.1 Rectangular source



The screenshot shows the 'Parameters' tab of the 'Thermal_evaporation-1.0' software. The 'Type de source' is set to 'Rectangular'. The 'Source center coordinates' are x: 0 (mm), y: 0 (mm), and z: -100 (mm). The 'Source dimensions' are Dx: 50 (mm) and Dy: 100 (mm). The 'Cosine exponent' n is 2. The 'Angle phi for the graph section' is 90 (°).

Source center coordinates: (x, y, z) in mm. ($z < 0$)

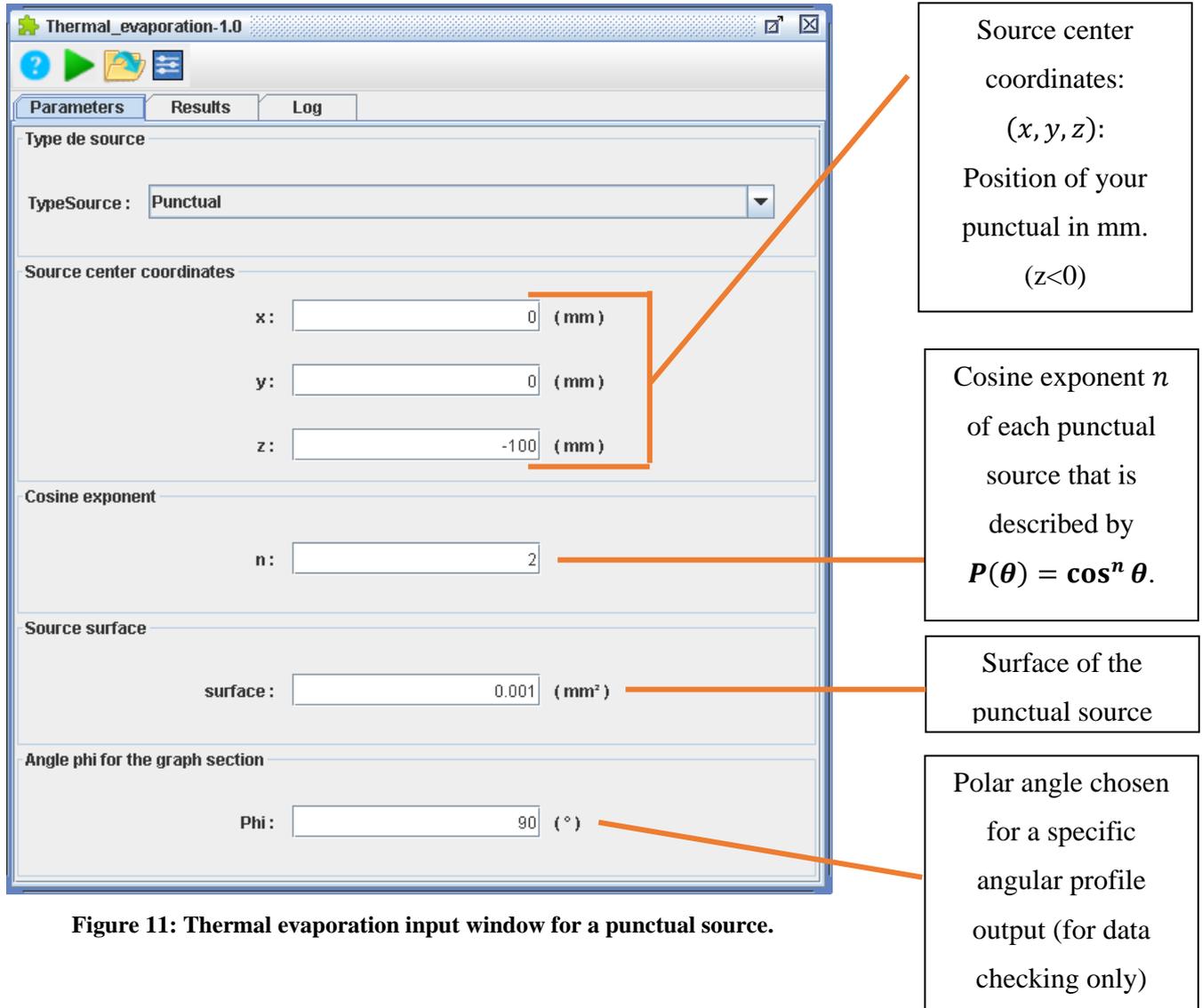
Source dimensions: (D_x, D_y) in mm.

Cosine exponent n of each punctual source that is described by $P(\theta) = \cos^n \theta$.

Polar angle chosen for a specific angular profile output (for data checking only)

Figure 10: Thermal evaporation input window for a rectangular source.

5.3.2 Punctual source



Source center coordinates:
 (x, y, z) :
 Position of your punctual in mm.
 $(z < 0)$

Cosine exponent n
 of each punctual source that is described by
 $P(\theta) = \cos^n \theta$.

Surface of the punctual source

Polar angle chosen for a specific angular profile output (for data checking only)

Figure 11: Thermal evaporation input window for a punctual source.

Note: The shape of the vapor cloud (as described by the n coefficient) strongly depends on the source geometry. The surface tension of the evaporated material increases the cloud convexity and increases the n value. If the evaporation is electron-beam generated, it can form a concave "vapor-emitting surface" due to a local increase of the vapor pressure. A high crucible wall can also affect the vapor shape by obstruction ($n > 1$). In any case, one can observe the formation of a vapor cloud ($n \ll 1$). In general, the coefficient is between 1 and 7.

5.4 Outputs

- The file to be used for NASCAM simulation ("*a_dst.txt*") in the appropriate format

N	θ	ϕ	Probability
Angular distribution			
0	0	-180	1
1	0	-178	1
2	0	-176	1
3	0	-174	1
4	0	-172	0.9
5	0	-170	0.9
6	0	-168	0.8
7	0	-166	0.76
8	0	-164	0.70
9	0	-162	0.4
10	0	-160	0.4
11	0	-158	0.39
12	0	-156	0.35
13	0	-154	0.1
14	0	-152	0.1
15	0	-150	0.1
16	0	-148	0.1
17	0	-146	0.1
18	0	-144	0.1
19	0	-142	0.1
20	0	-140	0.1
21	0	-138	0.1
22	0	-136	0.1

Figure 12: Output file of the thermal evaporation plug-in.

N is the line index, θ is the incident azimuthal angle, ϕ is the incident polar angle at the surface of your substrate and *Probability* is the incident probability at the angle (θ, ϕ) .

- Files to allow the user to plot the results of the plugin simulation automatically:
- "*a_dst.csv*" contains the data for the angular distribution at $\Phi =$ (the angle specified by the user).
- The files "*a_dst0deg.txt*", "*a_dst90deg.txt*", "*a_dst180deg.txt*" and "*a_dst_90deg.txt*" are also automatically generated for $\Phi = 0^\circ$, $\Phi = 90^\circ$, $\Phi = 180^\circ$ and $\Phi = -90^\circ$ for data checking.
- NASCAM being insensitive to angular distribution normalization, all the sub-graphs are normalized with the maximum set to 1.
- To use the angular distribution produced, get the *a_dst.txt* file in the project/plugins/Thermal_evaporation/output folder.

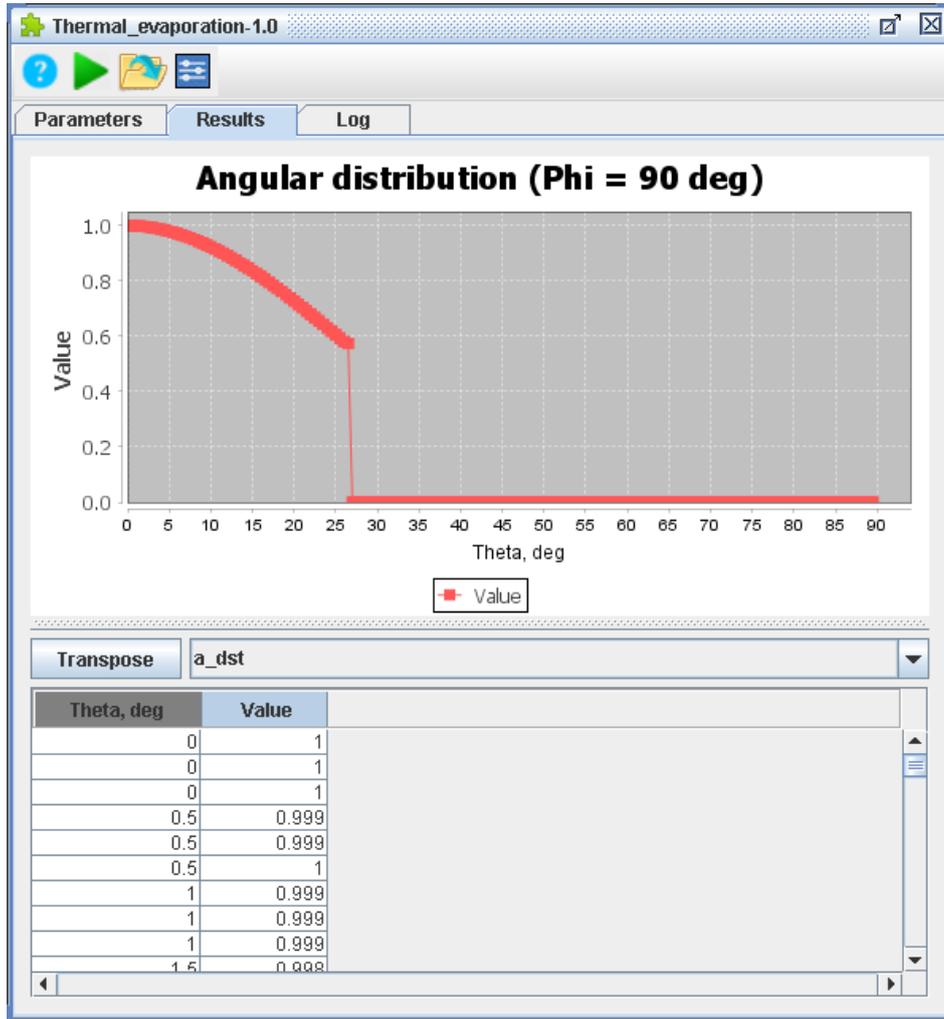


Figure 13: Angular plot for a rectangular source.

6 Roughness

This plugin calculates the film roughness, RMS :

$$RMS = \sqrt{\langle (h - \langle h \rangle)^2 \rangle} = \sqrt{\langle h^2 \rangle - \langle h \rangle^2},$$

where $h = h(x, y)$ is the film local thickness, $\langle h \rangle$ is an averaged film thickness, and $\langle \rangle$ means an average over the substrate surface.

Also two correlation functions, a structure function $G(R)$ and an autocovariance function $C(R)$ are calculated:

$$G(r) = \langle (h(x + R, y) - h(x, y))^2 \rangle,$$

$$C(r) = \langle 2(h(x + R, y)h(x, y)) \rangle.$$

These two functions and the roughness obey the equation:

$$G(R) + C(R) = 2 * RMS^2.$$

While the roughness gives information about film thickness variations, these functions are used to estimate the characteristic size of surface structures. Suppose the film surface is given by the equation:

$$h = H_0 + h_0 * \sin(k * x),$$

where H_0 is the average film thickness, h_0 is the variation, and the characteristic size of the surface structure is $R_0 = \pi/k$. Then the surface roughness is equal to

$$RMS = 0.5 * h_0^2,$$

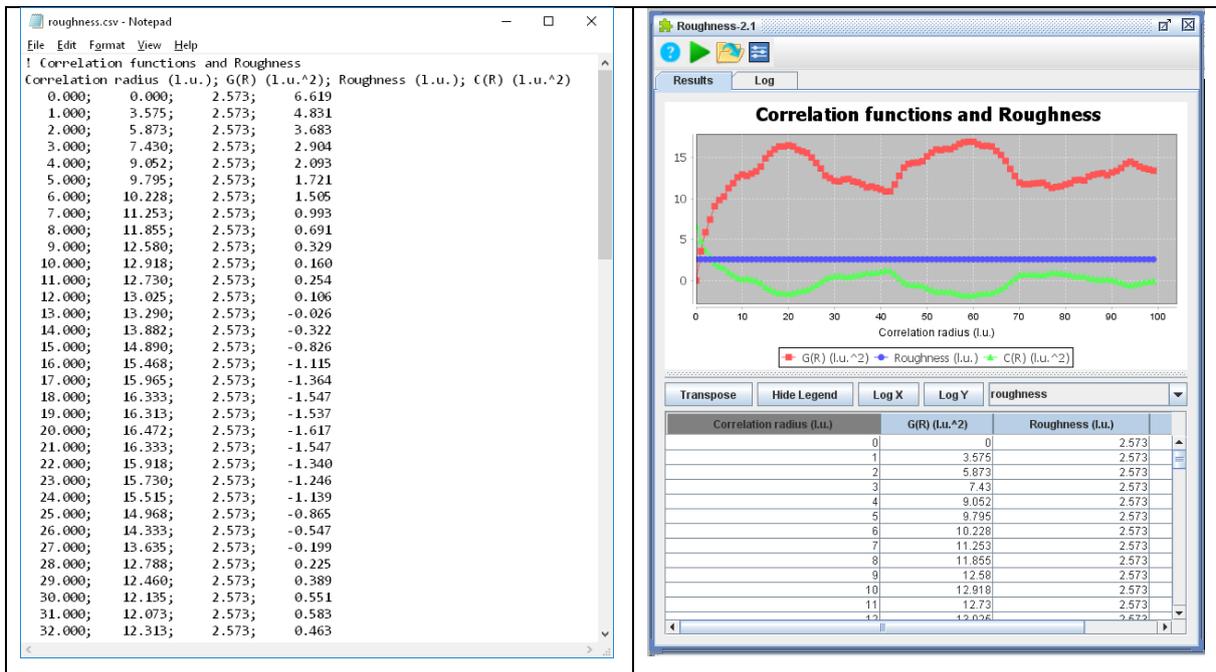
and gives no information about the spatial structure of the surface. At the same time:

$$G(R) = h_0^2 * \{1 - \cos(\pi R / R_0)\},$$

and it is possible to extract the information about the size of surface structures from this equation. For the surfaces where the objects are positions irregularly, one can use the following approximation for $G(R)$:

$$G(R) = 2 * RMS^2 * \{1 - \exp(-R^2/R_0^2)\}.$$

The plugin uses *coating.xyz* as an input file and its output file is *roughness.csv*. There is no input parameter for the plugin. To run it, just click on the green triangle. An example of a *roughness.csv* file and its graphical representation by the plugin is shown below. Roughness is given in lattice units and correlation functions are given in square lattice units.



7 Porosity

The *Porosity* plugin is designed to compute the porosity of a multilayer structure provided by NASCAM. The code is written in C.

7.1 Algorithm

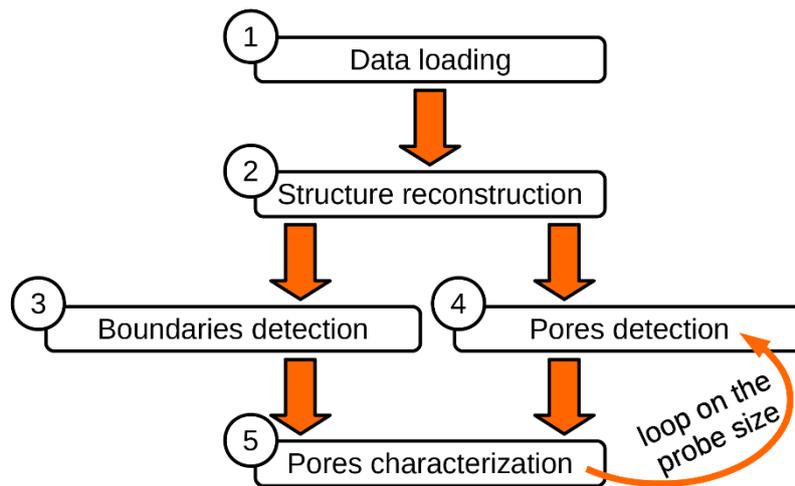
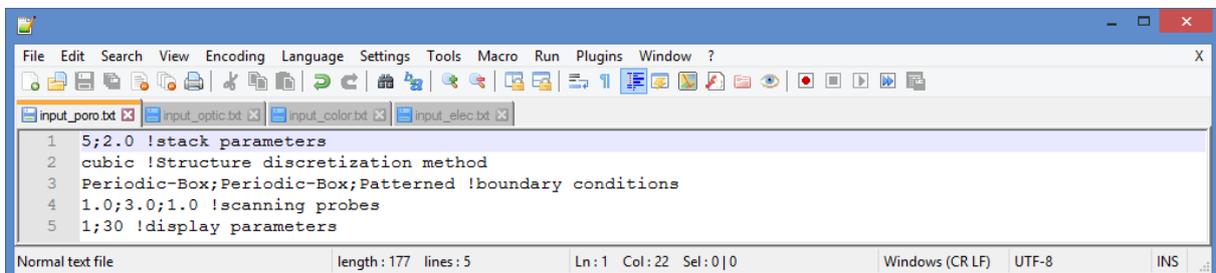


Figure 14: flowchart of the Porosity plugin.

Figure 14 summarizes the different steps of the *Porosity* algorithm. More details are given below.

7.1.1 Data loading

During this step, two (three) kinds of files are loaded. The first one is the “*input.txt*” file where all parameters set by the user are stored (see Figure 15). For more details, see subsection 7.2.1.



```

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
input_poro.txt input_optic.txt input_color.txt input_elec.txt
1 5;2.0 !stack parameters
2 cubic !Structure discretization method
3 Periodic-Box;Periodic-Box;Patterned !boundary conditions
4 1.0;3.0;1.0 !scanning probes
5 1;30 !display parameters
Normal text file length: 177 lines: 5 Ln: 1 Col: 22 Sel: 0|0 Windows (CR LF) UTF-8 INS
  
```

Figure 15: Porosity input.txt file example.

The second kind of data files are the “*deposited_layer.xyz*” provided by NASCAM (one file per layer). The coordinates of all atoms of a given layer are stored in each file.

The third one (if asked) is the “*substrate.xyz*” file where the coordinates of all atoms of a patterned (i.e. structured) substrate are stored.

7.1.2 Structure reconstruction

The structure is defined by a 3D structured mesh defined by space steps Δx , Δy and Δz . Each atom of the structure, loaded during step 1, is placed at the center of the most suitable mesh cell. This mesh is stored thanks to a 3D array of integers (named **Matrix 0**, see Figure 16a). The matrix size is given by the number of cells in the 3 directions ($N_x \cdot N_y \cdot N_z$).

Each medium is represented by an integer index as follows:

- 0: empty cell (vacuum)
- 1: cell filled by the medium of layer 1
- ...
- N_{layer} : cell filled by the medium of layer N_{layer}
- Additional indexes can be used in the case of structured substrates.

Figure 16a shows the vertical cross-section of the Matrix 0 for a 10-layer stack described by a cubic mesh.

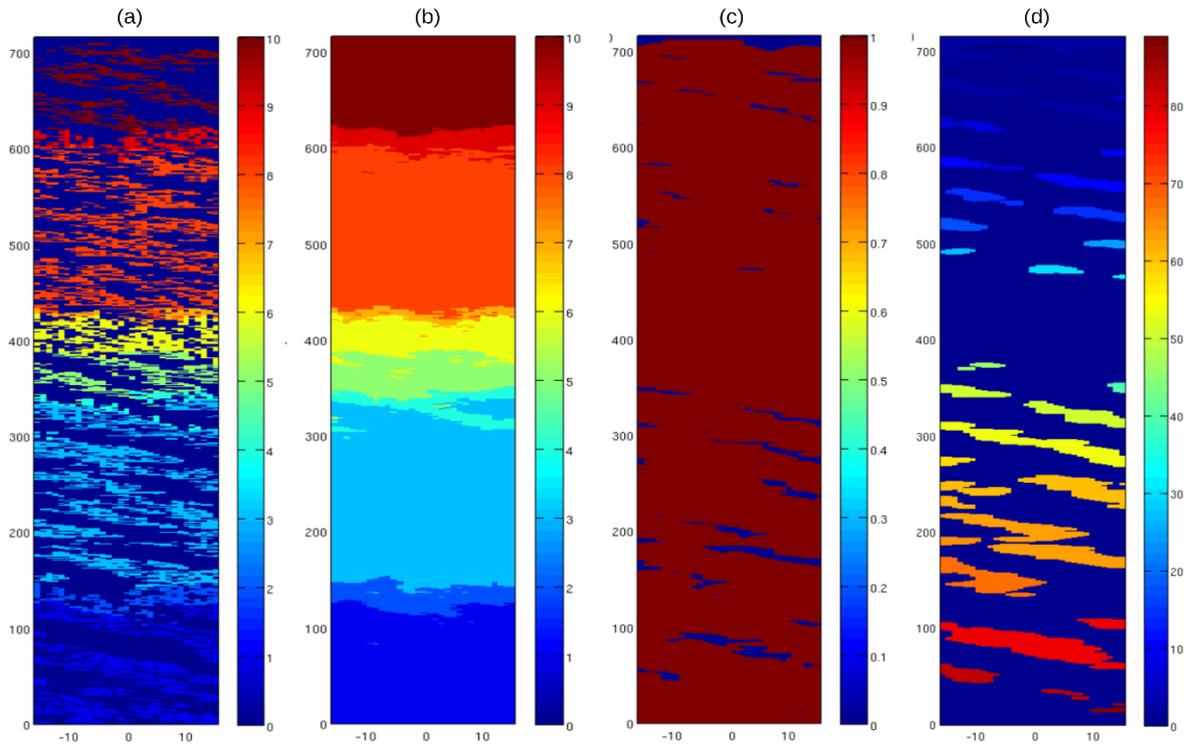


Figure 16: Matrix discretization of a structure. Example of a 10-layer stack, discretized by a cubic mesh ($\Delta x = \Delta y = \Delta z = 1$, $50 \times 50 \times 710$ cells) and scanned by a 5-lattice diameter probe. Each picture is a vertical cross-section (xz) of the structure. (a) Matrix 0: structure with one color (i.e. one integer index) per layer.

0 is for the vacuum. (b) Matrix A: filled structure with one color per layer. (c) Matrix B1: inflated binarized structure. 0 is for all areas accessible by the center of the probe. (d) Matrix B2: for the pores storage. Each color (i.e. each integer index $i > 1$) represents one pore. 1 is for the structure material and all the pores which are too small.

The choice of the mesh (i.e. Δx , Δy and Δz) has to be done wisely. Some examples are shown in Figure 17:

- The first example (a) can be used for a cubic crystal. Then, $\Delta x = \Delta y = \Delta z = 1^2$ is a good choice.
- The second case (b) is for a hexagonal crystal. The best choice is $\Delta x = 1/2$, $\Delta y = \sqrt{3}/2$ and $\Delta z = 1$.
- The last case (c) is for an amorphous structure. There is no particular rule but (obviously) a finer mesh will provide more accurate results (at the cost of a more expensive computation).

² Here, the unit length represents the lattice length of the crystal. It is set by the *deposited_layer.xyz* files.

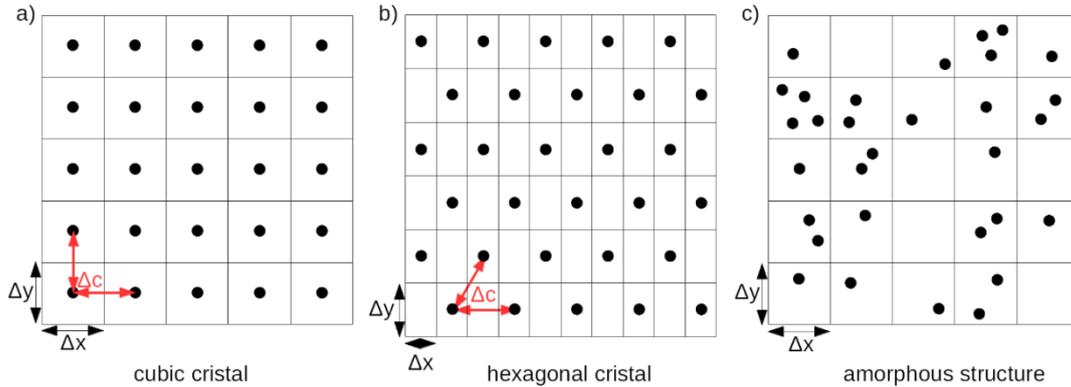


Figure 17: choice of the meshing method. 2D representation of the mesh in the case of a cubic crystal (a), a hexagonal crystal (b) and an amorphous structure. Δc represents the lattice of the crystal (equal to 1 in NASCAM).

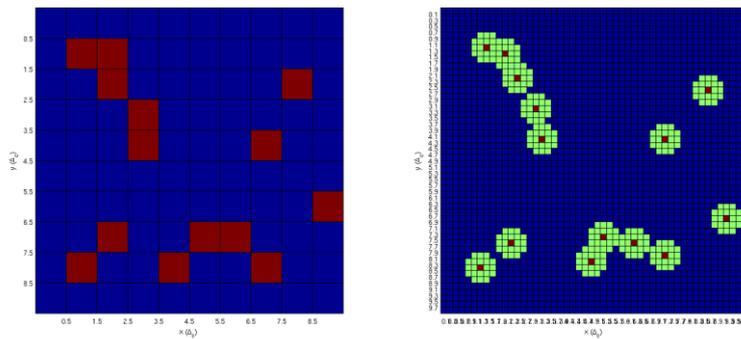


Figure 18: Spatial discretization of an atom (2D representation). Left: $\Delta x = \Delta y = 1$. Right: $\Delta x = \Delta y = 0.2$.

It is now important to talk about meshes with space steps smaller than one lattice. In this case, an atom is supposed to fill all mesh cells in a radius of 0.5 lattice. This shows the importance of a good choice of the space steps. For example, Figure 18 shows the case of an amorphous structure discretized with different space steps. In the first case (left), $\Delta x = \Delta y = \Delta z = 1$. Then, each atom fills only one cell (in red). On the right, $\Delta x = \Delta y = \Delta z = 0.2$, and then an atom fills the cell where its center is supposed to be (in red), plus all cells around the red one in a radius of 0.5 lattice (green cells).

7.1.3 Boundaries detection

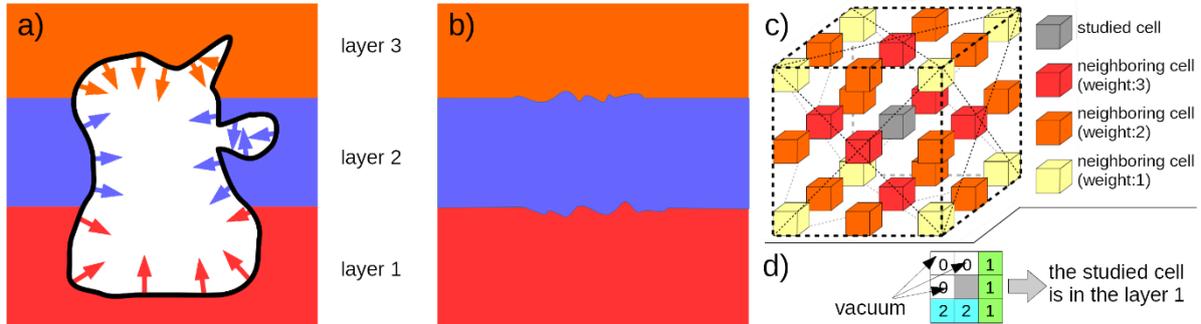


Figure 19: Detection of boundaries between layers. (a) Matrix 0 where pores are not filled yet. (b) Matrix A where all pores are filled by indexes of layers. (c) Characterization of a mesh cell in function of all the neighbouring cells, by using a weighted method. (d) 2D example of characterization of a cell surrounded by 2 layer indexes.

Boundaries between layers must be determined to distinguish the porosity of each layer. As shown in Figure 19, the method used here is based on the filling of all empty areas of the Matrix 0 by the material of each layer itself. The starting points to fill the structure are the material/vacuum interfaces. For each empty cell of the interface (marked by the index 0), we look at the $3^3-1=26$ neighbouring cells. Each neighbouring cell is linked to a weight which depends on the distance between this cell and the studied one (Figure 19c). Then, by summing up all the weights for each layer all around the empty cell, we can determine which layer has the highest weight³, i.e. the layer where the empty cell is supposed to be. When the interface material/vacuum is filled, the operation is repeated by starting from the new interface. The fully filled structure is called **Matrix A**.

Figure 16b shows the vertical cross-section of Matrix A for a 10-layer stack described by a cubic mesh.

³ If two (or more) layers have the same weight, then the choice between all these layers is done randomly.

7.1.4 Pores detection

The goal of this step is to detect all pores having a throat larger than the diameter of a spherical probe set by the user. Because of the structure pixelization, the probe that will scan this structure cannot be perfectly spherical. Indeed, it will depend on the diameter/space step ratio. Figure 20 shows the case of a sphere discretized with different space steps.

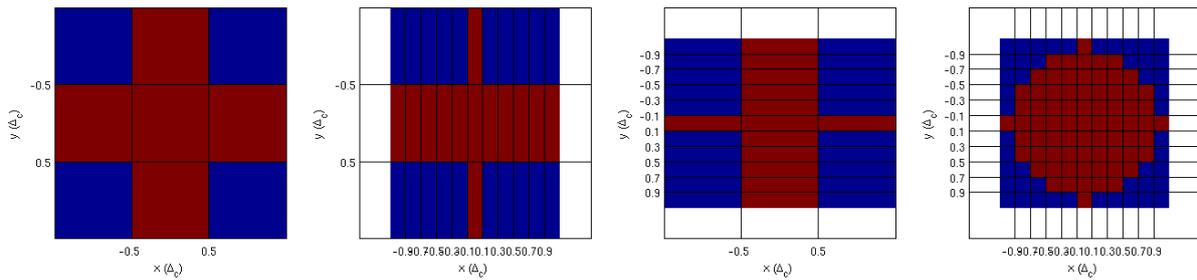


Figure 20: Discretization of a spherical probe with a diameter of 2 (2 lattices) in a structured mesh.
 From left to right: $(\Delta x = \Delta y = 1)$, $(\Delta x = 0.2, \Delta y = 1)$, $(\Delta x = 1, \Delta y = 0.2)$, $(\Delta x = \Delta y = 0.2)$.

The detection of pores has to follow 3 steps:

- The first step simply consists in binarizing Matrix 0 (Figure 21a) by giving each mesh cell the index 1, if one (or more) atoms are here, and 0 if not. This matrix is called **Matrix 0bin**.
- The second step consists in inflating Matrix 0bin by a thickness of (approximately) 1 probe radius (see Figure 21b). The approximation is due to the spherical probe discretization (see Figure 20). Then we can conclude that all the unfilled areas (i.e. which are still represented by the index 0) are reachable by the center of the probe. After inflation, the matrix is called **Matrix B1**.
 For example, Figure 16c shows the vertical cross-section of Matrix B1 for a 10-layer stack described by a cubic mesh. All blue cells are accessible by the center of a 5-lattice diameter probe.
- The last step is to reverse the previous one. Indeed, Matrix B1 has to be deflated (see Figure 21c). After deflation, we obtain the **Matrix B2**, where each area described by index 0 is supposed to be large enough to contain the whole probe. Then, Matrix B2 has to be updated by giving each detected pore one unique integer index $i > 1$ (one index per pore). For example, Figure 16d shows the vertical cross-section of Matrix B2 for a 10-layer stack described by a cubic mesh. All not-blue cells represent all pores accessible by a 5-lattice diameter probe.

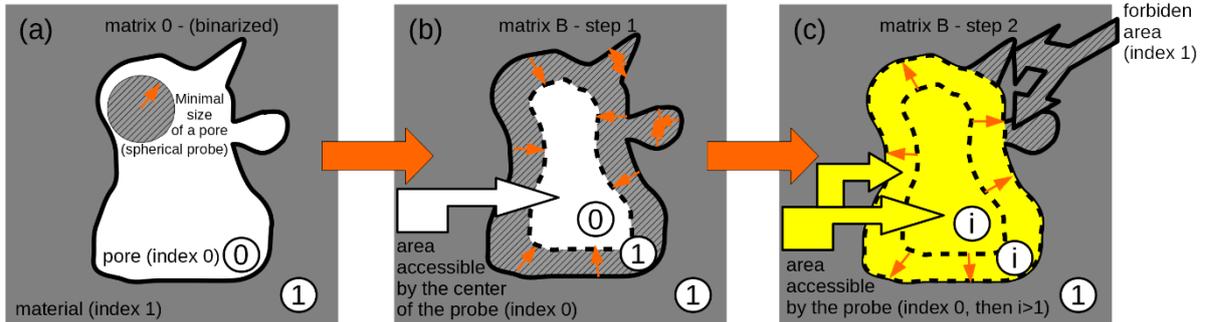


Figure 21: Method to detect pores reachable by a spherical probe. (a) Matrix 0bin given by the binarization of Matrix 0. (b) Matrix B1 after inflating matrix 0bin. (c) Matrix B2 after deflating Matrix B1.

7.1.5 Pores characterization

More information is found during this step. At first, pore kind determination. Three kinds of pores are considered here:

- Occluded pore: the pore is linked to only one layer
- Connected pore: the pore is linked to two layers at least
- Air-connected pore: the pore is connected to the top surface of the structure

Figure 22 shows the method used to do such thing, by comparing Matrices A and B2.

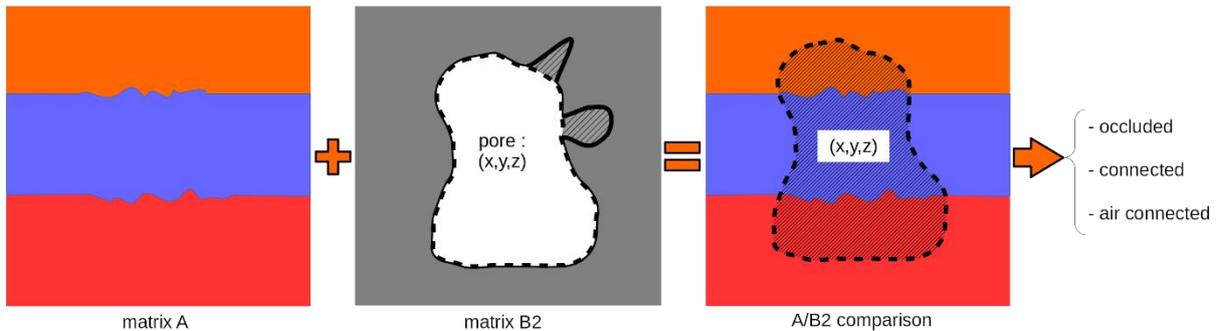
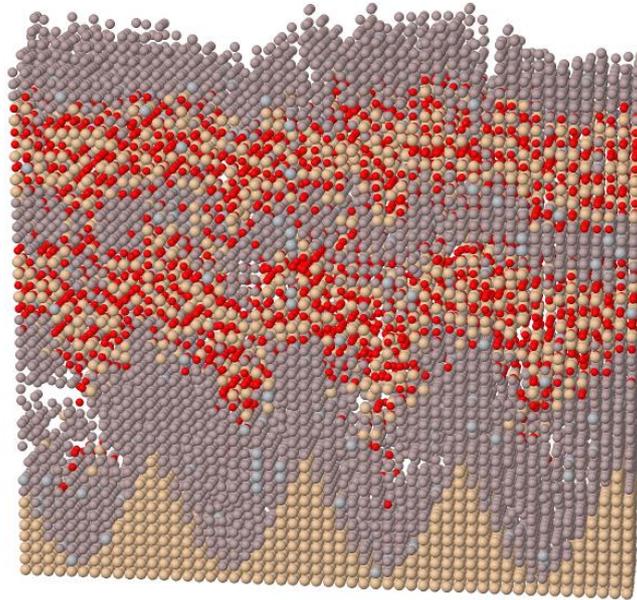


Figure 22: Pores classification by comparing matrices A and B2. In this example, the studied pore is a connected one.

Pore volume or volume ratio is also calculated here. More details can be found in subsection 7.2.

7.2 NASCAM GUI example



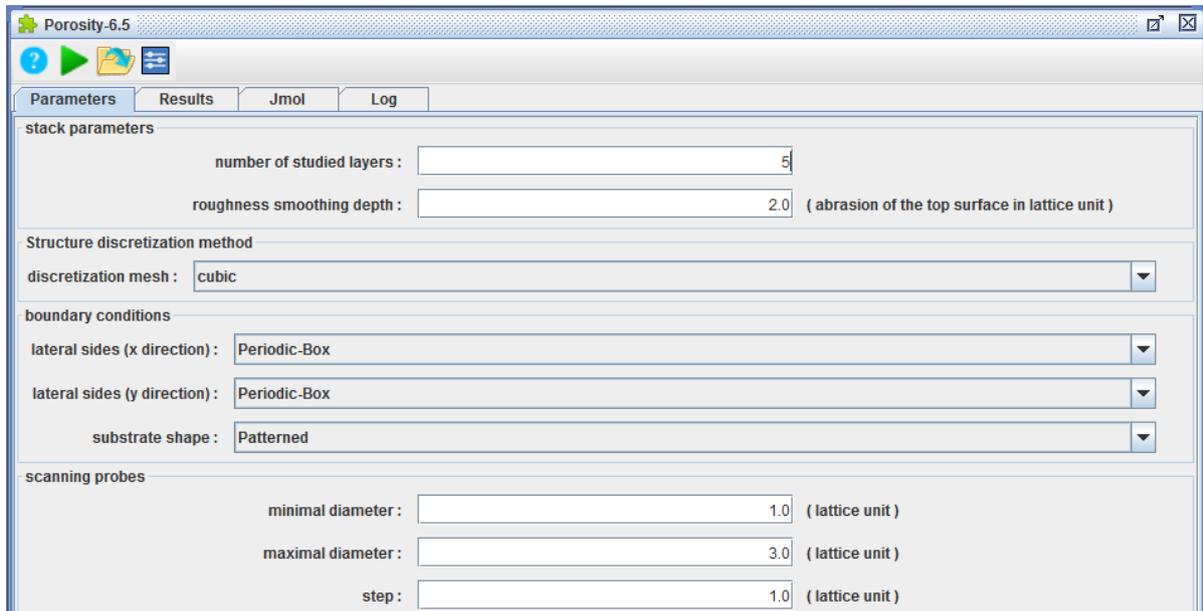
Jmol

Figure 23: A *coating.xyz* file displayed by Structure Viewer.

In this section, we study a 3D 5-layer stack (Al/SiO₂) on a Si substrate structured by triangular rows, as shown by the JMOL representation in Figure 23.

7.2.1 Input parameters

The *Porosity* parameter window pops up by clicking on Tools→Porosity.



The screenshot shows the 'Parameters' tab of the Porosity-6.5 software. The interface is divided into several sections:

- stack parameters:** 'number of studied layers' is set to 5, and 'roughness smoothing depth' is set to 2.0 (abrasion of the top surface in lattice unit).
- Structure discretization method:** 'discretization mesh' is set to 'cubic'.
- boundary conditions:** 'lateral sides (x direction)' and 'lateral sides (y direction)' are both set to 'Periodic-Box', and 'substrate shape' is set to 'Patterned'.
- scanning probes:** 'minimal diameter' is 1.0 (lattice unit), 'maximal diameter' is 3.0 (lattice unit), and 'step' is 1.0 (lattice unit).

Figure 24: Porosity input parameters.

The *Porosity* parameters defined in Figure 24 are:

- the number of studied layers asked by the user (it can be adapted automatically to its maximum possible value, i.e. the number of layers of the structure).
- the “roughness smoothing depth” : by setting a value n , the algorithm will simply remove all the atoms at the surface of the stack in a depth of n lattices.
- the meshing method (cubic, hexagonal or manual).
- the mesh cell size (Δx , Δy and Δz in lattice unit) if the manual meshing method is chosen. It is important to note that it is a 3D computation and thus using too small mesh cells can make your computation time explode exponentially! So just take what you need!
- the boundary conditions
 - for the left and right sides of the structure (recommended: periodic)
 - for the back and front sides of the structure (recommended: periodic)
 - for the substrate, which can be structured or not. It is important to note that if you study a structure with a patterned substrate, while choosing the “Flat” option for the *Porosity* calculation, all the volume occupied by the substrate will be considered as vacuum (see Figure 26-right).
- the scanning probe diameter (min, max and step values in lattice unit): these values are directly linked to the minimal pore throat.

Then, to launch the calculation, just click on the green arrow. All calculation details can be checked in the log windows (click on the Log tab).

7.2.2 JMOL results

The first results available are the 3D JMOL maps of pores detected during the calculation. Four files (stored in the output folder) can be loaded thanks to the Jmol window (click on the Jmol tab). Each file (xyz extension) represents one kind of pore (air-connected, connected, occluded and “all kind”). In Figure 25, all kinds of pores, detected by a 1 lu diameter probe, are displayed for two values of the “roughness smoothing depth” (left: 0 lu, right: 5 lu). The colour depends on the pore size.



Figure 25: JMOL 3D display of all the pores detected by a 1 lu diameter probe. Left: full structure. Right: reduced structure by setting the “roughness smoothing depth” parameter to 5 lu.

Note that results for all probe diameters are stored in the same Jmol file. As shown in Figure 26 (left), to access the different data, right-click on the Jmol window, go to “model i/N” (i: the current picture, N: the number of available pictures) and then choose the picture you want. It is possible to observe a cross-section of the structure by opening the console (right-click on the Jmol window and click on “Console”). Figure 27 shows the xz cross section of Figure 25 (right) by typing the command “display y==0” (the value of y depends on the coordinates of the structure).

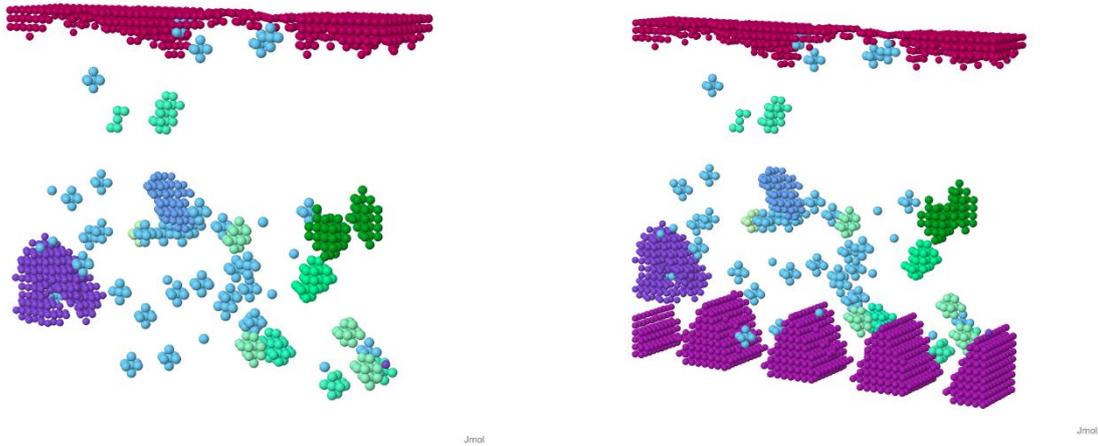


Figure 26: JMOL 3D display of all the pores detected by a 2 lu diameter probe. Left: the substrate is taken into account (patterned option). Right: the substrate is removed (flat option).

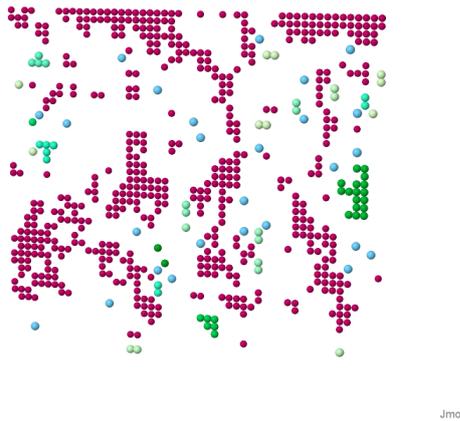


Figure 27: JMOL 2D display of all the pores detected by a 1 lu diameter probe (xz cross-section).

Finally, it could be interesting to visualize the boundaries between the different layers, as shown in Figure 28. To do that, you just have to choose the “Jmol_-_Layers_and_Boundaries.xyz” file.

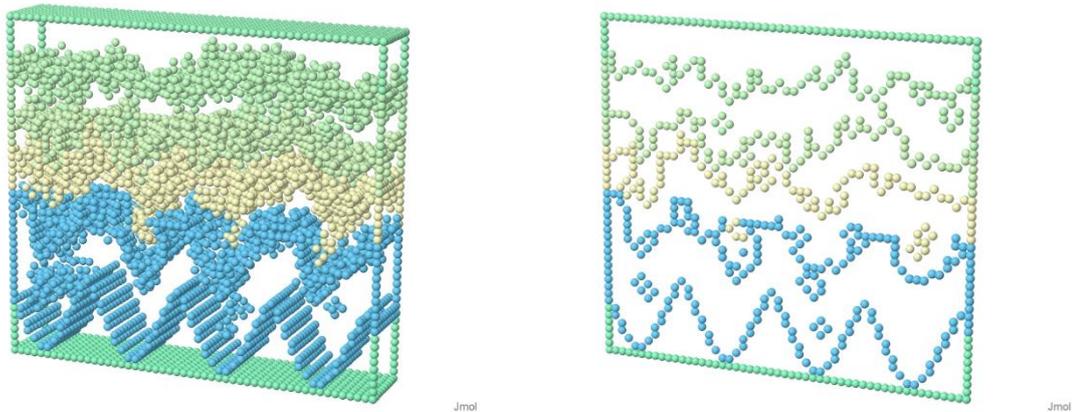


Figure 28: Jmol display of the boundaries between layers. Left: 3D display. Right: xz cross-section.

WARNING: if no pores are detected for a given situation (pore kind, probe size...), the corresponding file is not created/updated.

7.2.3 Statistical results

There are four kinds of statistical results (stored in the output folder and available by clicking on the “Results” tab):

- The size distribution (Figure 29 - left): it represents the number of pores for each possible size in each layer. There is one file for each kind of pore (connected or occluded pores) and each scanning probe diameter (in lattice unit). Note that a logarithmic hole size distribution mode is set by default, but you can still access to the linear one (both abscissa are stored in the first and the second columns of the table below the graph). Then, right-click on the one you want, and left-click on the other to remove it.
- The number of pores (Figure 29 - right): it represents the number of pores (of any size) for each layer and each studied scanning probe diameter. There is one file per pore kind (air-connected, connected, occluded and “all pores”).
- The volume of pores (Figure 30 - left): it represents the total volume of pores (in cubic lattice) for each layer and each studied scanning probe diameter. There is one file per pore kind (air-connected, connected, occluded and “all pores”).
- The volume ratio of pores (Figure 30 - right): it represents the volume ratio of pores (in %) for each layer and each studied scanning probe diameter. There is one file per pore kind (air-connected, connected, occluded and “all pores”).

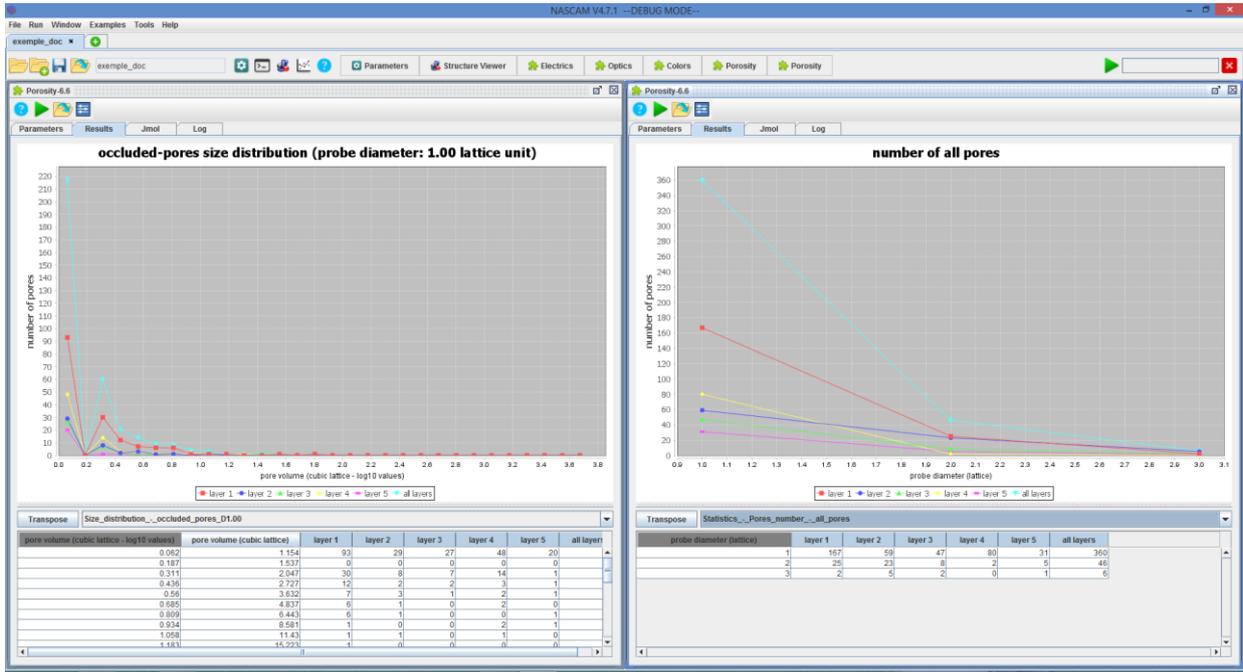


Figure 29: Statistical results. Left: size distribution of occluded pores detected by a 1-lattice diameter probe. Right: number of detected pores (any kind) in function of the probe size.

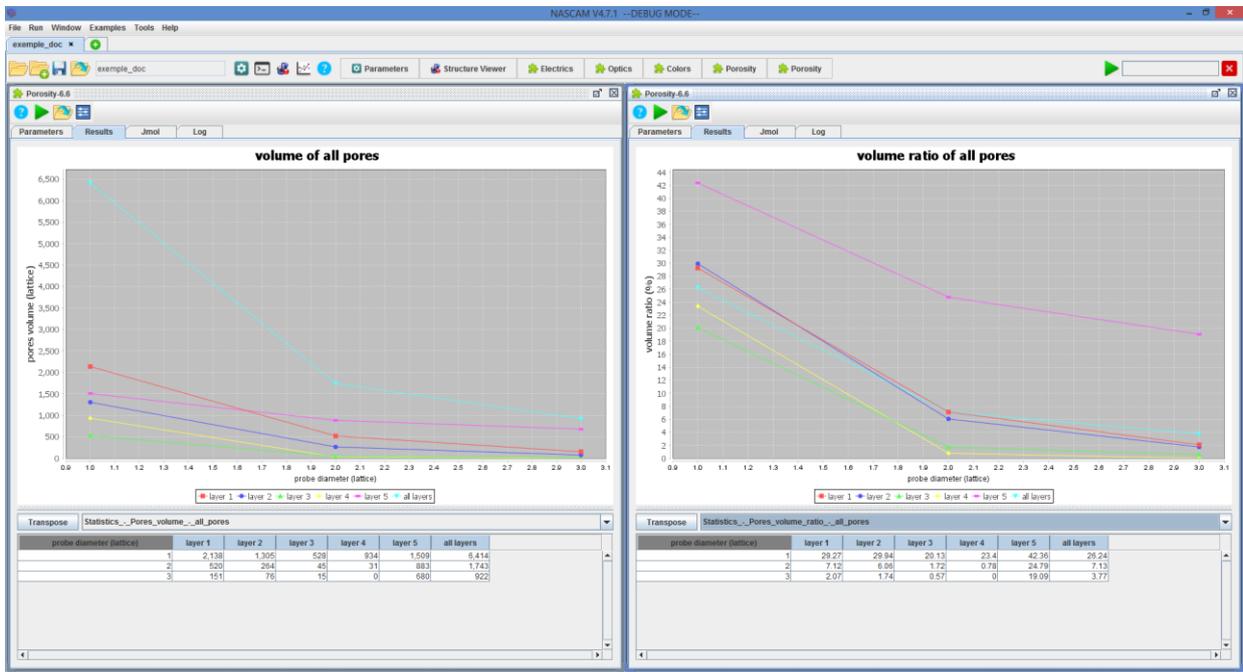


Figure 30: Statistical results. Left: volume (in cubic lattice) of detected pores (any kind) according to the probe size. Right: volume ratio of detected pores (any kind) according to the probe size.



Innovative Coating Solutions,
Place Saint-Pierre, 11, B-5380 Forville., Belgium
slu@incosol4u.com www.incosol4u.com

WARNING: if no pores are detected for a given situation (pore kind, probe size...), the corresponding file is not created/updated.

8 Optics

Optics is designed to compute the absorptance, the reflectance and the transmittance of a multilayer structure provided by NASCAM. The code is written in C.

8.1 Algorithm

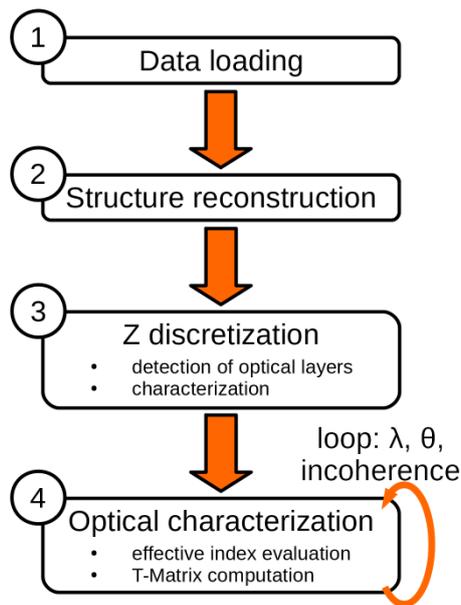


Figure 31: flowchart of the Optics plugin.

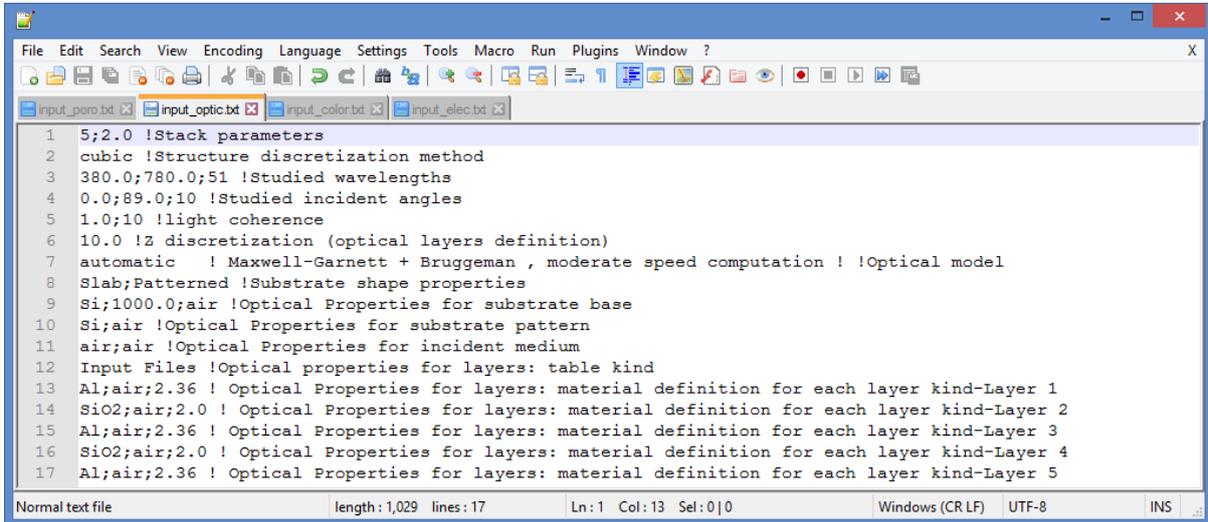
Figure 31 summarizes the different steps of the Optics algorithm. More details are given below.

8.1.1 Data loading

During this step, three kinds of files are loaded. The first one is the *input.txt* file where all parameters set by the user are stored (see Figure 32). For more details about parameters, see subsection 8.2.

The second kind of data file is the *deposited_layer.xyz* provided by NASCAM (one file per layer). In each file are stored the coordinates of all atoms of a given layer. If needed, the *substrate.xyz* file can be loaded too.

The last is the *Optical_database.nk* file (provided by the NASCAM SQL database) where the refractive indexes of several materials⁴ are stored.



```

1 5;2.0 !Stack parameters
2 cubic !Structure discretization method
3 380.0;780.0;51 !Studied wavelengths
4 0.0;89.0;10 !Studied incident angles
5 1.0;10 !light coherence
6 10.0 !Z discretization (optical layers definition)
7 automatic ! Maxwell-Garnett + Bruggeman , moderate speed computation ! !Optical model
8 Slab;Patterned !Substrate shape properties
9 Si;1000.0;air !Optical Properties for substrate base
10 Si;air !Optical Properties for substrate pattern
11 air;air !Optical Properties for incident medium
12 Input Files !Optical properties for layers: table kind
13 Al;air;2.36 ! Optical Properties for layers: material definition for each layer kind-Layer 1
14 SiO2;air;2.0 ! Optical Properties for layers: material definition for each layer kind-Layer 2
15 Al;air;2.36 ! Optical Properties for layers: material definition for each layer kind-Layer 3
16 SiO2;air;2.0 ! Optical Properties for layers: material definition for each layer kind-Layer 4
17 Al;air;2.36 ! Optical Properties for layers: material definition for each layer kind-Layer 5
  
```

Figure 32: Optics *input.txt* file.

8.1.2 Structure reconstruction

The method is similar to the one used by the *Porosity* plugin (see Section 7.1.2), with some differences however. The algorithm is summarized in Figure 33.

⁴ The can update the SQL database.

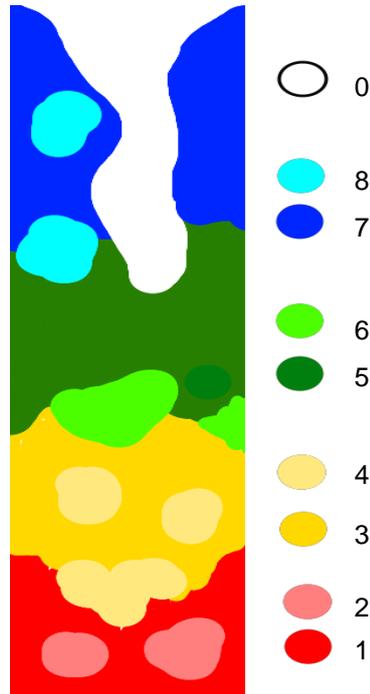


Figure 33: Structure reconstruction algorithm used by the *Optics* plugin

Each medium is represented by an integer index as follows:

- 0: air-connected pores
- 1: cell filled by the medium of layer 1
- 2: cell filled by the pores content of layer 1
- ...
- $2N_{layer}-1$: cell filled by the medium of layer N_{layer}
- $2N_{layer}$: cell filled by the pores content of layer N_{layer}
- Additional indexes can be used in the case of structured substrates.

Then, each index is directly linked to an optical index spectrum stored in the database.

8.1.3 Z discretization

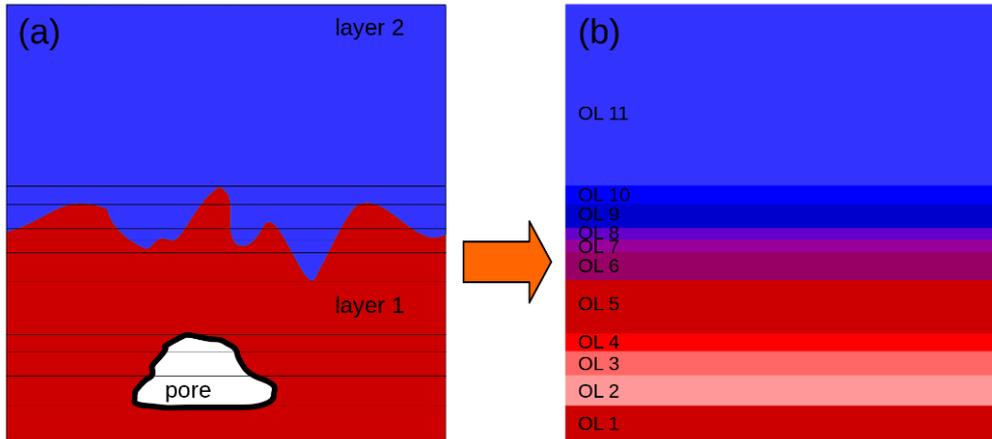


Figure 34: Optical layers detection. (a) 2-layer porous structure. (b) Equivalent 1D structure.

The Z discretization method is based on the calculation of the volume fraction gradient in the z direction for each material of the structure. If the sum of these gradients is bigger than a given value (set by the user – see Subsection 8.2.1), the algorithm considers that we are in a new optical layer (see Figure 34).

8.1.4 Optical characterization

Two steps are needed for the optical characterization of the structure. The first one is the calculation of the effective dielectric function ϵ_{eff} (in each optical layer and for each studied wavelength) based on the effective medium theory (Vedam, 1994). Five optical models (for an n-material optical layer) are available:

- the **Volume Average method**:

$$\epsilon_{eff} = \sum_{i=1}^n f_i \epsilon_i$$

With f_i and ϵ_i the volume fraction and the dielectric function of the material i (respectively). This method is quite fast, but not quite accurate. It can be used only for stacks with relatively smooth interfaces and a low porosity.

- the **Lorentz-Lorentz method**:

$$\frac{\epsilon_{eff} - 1}{\epsilon_{eff} + 2} = \sum_{i=1}^n f_i \frac{\epsilon_i - 1}{\epsilon_i + 2}$$

Same remark as for the Volume Average method.

- the **Maxwell-Garnett method**:

$$\frac{\epsilon_{eff} - \epsilon_h}{\epsilon_{eff} + y\epsilon_h} = \sum_{i=1}^{n-1} f_i \frac{\epsilon_i - \epsilon_h}{\epsilon_i + y\epsilon_h}$$

With ϵ_h the dielectric function of the host, $y = 1/l - 1$ and $0 \leq l \leq 1$ where l is the polarization factor (set by default at 1/3, for spherical pores). This method is particularly suitable to structures with a "dominant" material (called host) with a relatively uniform porosity.

- the **Bruggeman method**:

$$\sum_{i=1}^n f_i \frac{\epsilon_i - \epsilon_{eff}}{\epsilon_i + 2\epsilon_{eff}} = 0$$

This method has to be used when a host cannot be defined in the structure (especially for highly porous structures). This is the only model that cannot be solved analytically (except for 2 materials or less). The optimization method used to solve the Bruggeman equation is the bounded Nelder–Mead method (or downhill simplex method) [(Bosch, 2000), (Luersen, 2004)].

Figure 35 shows the resolution of the Bruggeman equation in a 3-materials domain case (air: 20%, Mo: 40%, Si: 40%) at a $\lambda=100\text{nm}$ wavelength.

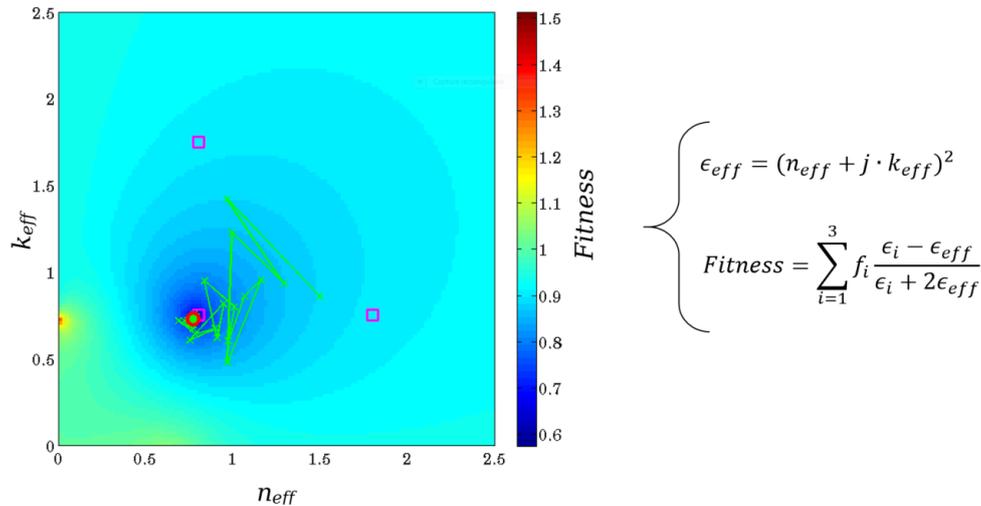


Figure 35: Convergence of the Bruggeman solution in function of optical indexes (n_{eff} and k_{eff}).

A 3-material domain case (air: 20%, Mo: 40%, Si: 40% at the wavelength $\lambda=100\text{nm}$). Optimization by the bounded Nelder-Mead method. Pink squares: starting points. Green crosses: path of the solution. Red circle: final solution (fitness $< 1e-5$ after 63 iterations).

- the **hybrid method**: this method uses the Maxwell-Garnett and the Bruggeman models. The choice of the model is function of the volume ratio of each material in an optical layer (the model can then change in each layer). If a "dominant" material (or host) is found, the Maxwell-Garnett model is used. If not, the Bruggeman model is chosen. The hybrid method needs all the

parameters of the Maxwell-Garnett and Bruggeman models, plus the minimal volume fraction needed to consider a material as a host.

The second and last step of the structure optical characterization consists in calculating its reflectance, transmittance and absorptance in function of the wavelength and incident angle. It is done by using a Transfer-matrix code based on the computation of the Fresnel coefficients in each optical layer of the structure (Heavens, 1955), as shown in Figure 36.

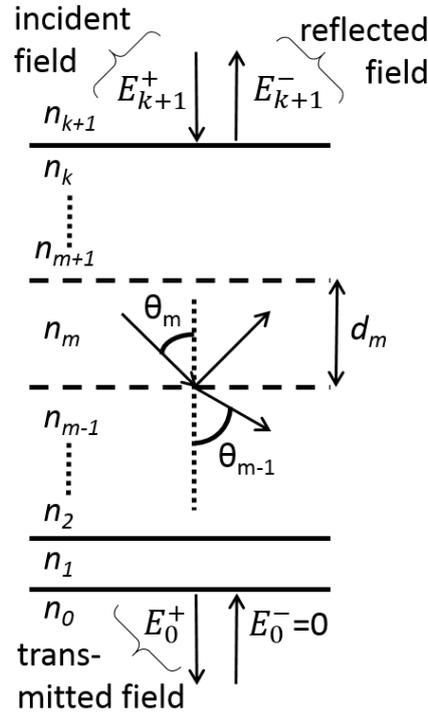


Figure 36: T-Matrix principle: simplified 1D isotropic stack.

At each interface between two optical layers, we need to compute the Fresnel coefficients:

$$r_m^p = \frac{n_m \cos \theta_{m-1} - n_{m-1} \cos \theta_m}{n_m \cos \theta_{m-1} + n_{m-1} \cos \theta_m} \quad r_m^s = \frac{n_m \cos \theta_m - n_{m-1} \cos \theta_{m-1}}{n_m \cos \theta_m + n_{m-1} \cos \theta_{m-1}}$$

$$t_m^p = \frac{2n_m \cos \theta_m}{n_m \cos \theta_{m-1} + n_{m-1} \cos \theta_m} \quad t_m^s = \frac{2n_m \cos \theta_m}{n_m \cos \theta_m + n_{m-1} \cos \theta_{m-1}}$$

Then, the complex Snell law gives the incidence angle of the light in each optical layer:

$$n_m \sin \theta_m = n_{m-1} \sin \theta_{m-1} \rightarrow \theta_{m-1} = \text{real} \left[\sin^{-1} \left(\frac{n_m}{n_{m-1}} \sin \theta_m \right) \right] \\ - i \cdot \text{abs} \left\{ \text{im} \left[\sin^{-1} \left(\frac{n_m}{n_{m-1}} \sin \theta_m \right) \right] \right\}$$

The phase of light is described by its phase shift factor:

$$\delta_m = \frac{2\pi}{\lambda} n_m d_m \cos \theta_m$$

Then, we can compute the transfer matrices in all the structure making the link between the incident, the reflected and the transmitted electric fields:

$$M_{m-1} = \frac{1}{t_m} \begin{bmatrix} \exp(i\delta_{m-1}) & r_m \exp(-i\delta_{m-1}) \\ r_m \exp(i\delta_{m-1}) & \exp(-i\delta_{m-1}) \end{bmatrix} \rightarrow \begin{bmatrix} E_{k+1}^+ \\ E_{k+1}^- \end{bmatrix} = M_k M_{k-1} \dots M_2 M_1 \begin{bmatrix} E_0^+ \\ E_0^- \end{bmatrix}$$

The reflectance, transmittance and absorbance can then be computed by:

$$R = \frac{|E_{k+1}^-|^2}{|E_{k+1}^+|^2} \quad T = \frac{n_0 \cos \theta_0}{n_{k+1} \cos \theta_{k+1}} \frac{|E_0^+|^2}{|E_{k+1}^+|^2} \quad A = 1 - R - T$$

The T-Matrix computation can be done several times in an incoherent incident light case. Indeed, a “trick” method is used to mimic such incoherent light by introducing a random phase shift of the light inside the different layers of the structure, as explained in (Troparevsky, 2010).

8.2 NASCAM GUI example

In this section, we study a 3D 5-layer stack (Al/SiO₂) on a Si slab (1μm) substrate structured by triangular rows (see Figure 23).

8.2.1 Input parameters

The Optics parameter window pops up by clicking on Tools→Optics. The Optics parameters defined in Figure 37 are:

- the number of studied layers asked by the user (it can be adapted automatically to its maximum possible value, i.e. the number of layers of the structure).
- the “roughness smoothing depth” : by setting a value n , the algorithm will simply remove all the atoms at the surface of the stack in a depth of n lattices.
- the meshing method (cubic, hexagonal or manual).
- (if manual meshing) the mesh cell size (Δx , Δy and Δz in lattice unit). It is important to note that it is a 3D computation and thus using too small mesh cells can make your computation time explode exponentially! So just take what you need!
- the studied wavelengths:
 - minimal wavelength in nm
 - maximal wavelength in nm
 - number of studied wavelengths
- the studied incident angles:
 - minimal incident angle in degrees (≥ 0 deg.)

- maximal incident angle in degrees (<90 deg.)
- number of studied incident angles
- the light coherence parameters:
 - the incoherence “level” of the incidence light: it can go from fully coherent (0.0) to fully incoherent (1.0)
 - the number of iterations: i.e. the number of simulations you have to do to mimic an incoherent light (this parameter is useless for coherent light).

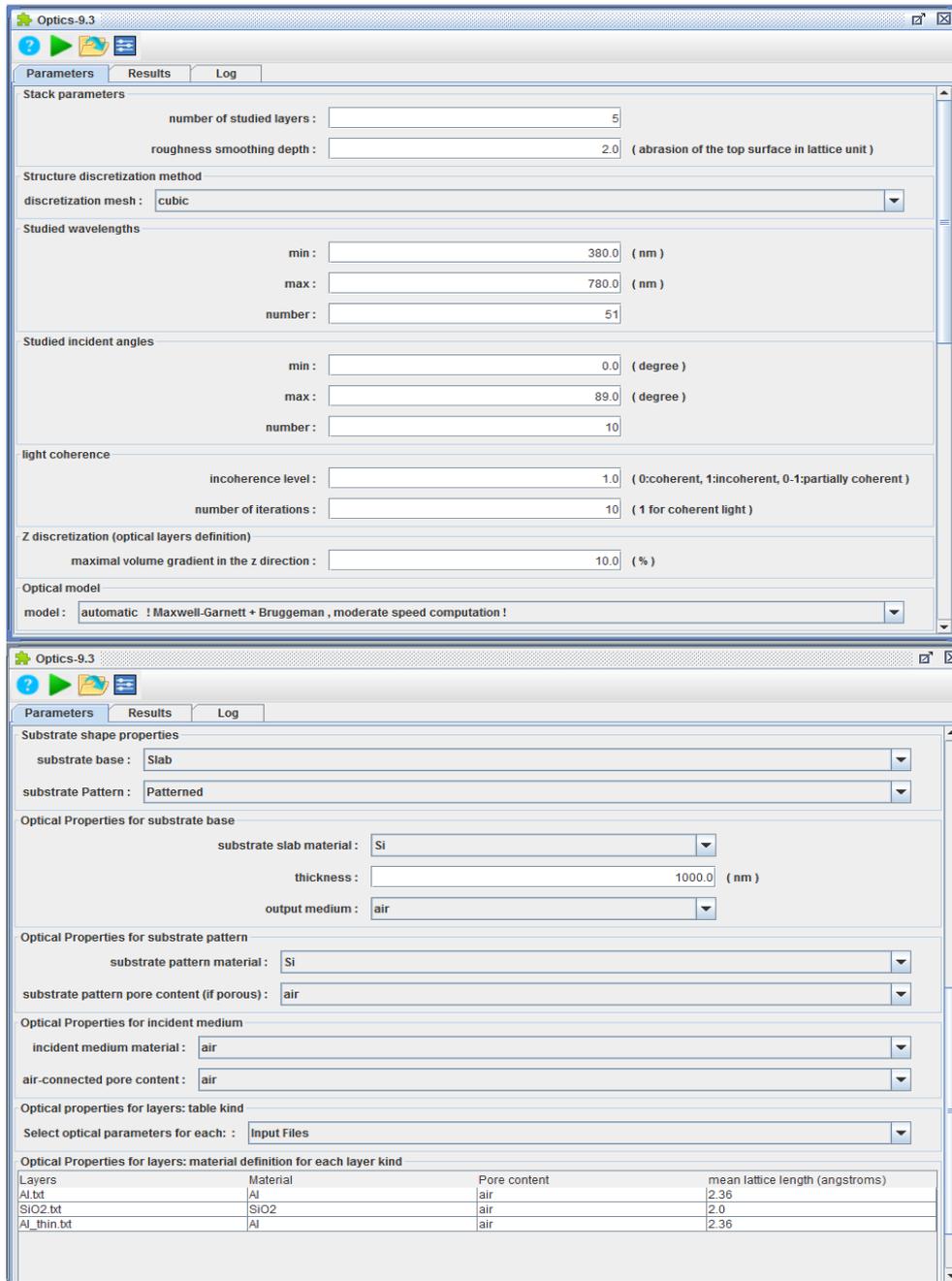


Figure 37: Optics input parameters.

- the maximal material volume gradient in the z direction: this parameter is used to define the boundary between the optical layers (in the *Optics* plugin, an optical layer is not necessary a structured layer!). By decreasing the parameter value, the optical layers thicknesses decrease too (for a more accurate computation, at the cost of an increased computation time).
- the optical model. Six models are available:
 - the Volume Average method: this method is really fast, but not really accurate. It can be used only for stacks with relatively smooth interfaces and low porosity.
 - the Lorentz-Lorentz method: same remark as for the Volume Average method.
 - the Maxwell-Garnett method: this method is particularly suitable to structures with a "dominant" material (called host) with a relatively uniform porosity. This model is defined by the polarization factor (set by default at 1/3, for spherical pores).
 - the Bruggeman method: this method has to be used when a host cannot be defined in the structure (especially for highly porous structures). This is the only model that cannot be solved analytically. The optimization method used to solve the Bruggemann equation is the bounded Nelder–Mead method (or downhill simplex method (Luersen, 2004)). The Nelder–Mead parameters are:
 - the reflection factor
 - the expansion factor
 - the contraction factor
 - the shrink factor
 - the maximal number of iterations (stopping criterion)
 - the targeted precision (stopping criterion)
 - the hybrid method: this method uses the Maxwell-Garnett or the Bruggeman models in function of the volume ratio of each material in an optical layer (the model can then change in each layer). If a "dominant" material (or host) is found, the Maxwell-Garnett model is used. If not, the Bruggeman model is chosen. The hybrid method needs all the parameters of the Maxwell-Garnett and Bruggeman models, plus the minimal volume fraction needed to consider a material as a host.
 - the automatic method: this is the hybrid method, but all parameters are already set.
- Substrate shape properties
 - the substrate base which can be finite (slab) or semi-infinite
 - the substrate structuration: it can be flat (no structuration) or patterned
- Optical properties for the substrate base:
 - material of the substrate base
 - (*if the substrate base is a slab*) the slab thickness
 - (*if the substrate base is a slab*) the material of the output medium (usually: air)
- (*if the substrate is structured*) Optical properties of the substrate pattern:
 - material of the pattern
 - pore content of the pattern (if porous)
- Optical properties for the incident medium:
 - material of the incident medium
 - pore content of the air-connected pores (which can be something else than air)

- Optical properties for layers - table kind: method to define optical properties for each layer (per layer or per input file).
- Optical properties for layers: choice of the material name of layers and pores contents and set of the mean lattice length for each layer or input file (depending on the choice of the table kind).

Then, to launch the calculation, just click on the green arrow. All calculation details can be checked in the log window (click on the Log tab).

8.2.2 Optical characterization

The first results that you can observe during the computation are in the log window, especially during steps 4 and 5, as shown in Figure 38 (left). In the “step 4” log section, you can access an estimation of the average thicknesses of the different layers of the stack. It is important to note that such values are highly dependent on the structuration. Therefore, it is dangerous to use them in “sensitive” cases like patterned substrates or inter-penetrating layers. In the “step 5” log section, you can access the reflectance, transmittance and absorbance for each wavelength and incident angle in the case of an unpolarized incident light.

Additional information, such as the different refractive indexes of the materials in the structure, is displayed in the Results section (click on the “Results” tab), as show in Figure 38 (right). Then, you choose “*Optical_Index_by_Areas_-_nk*” if you want to display the refractive indexes for each layers and pore layers, or “*Optical_Index_by_Materials_-_nk*” if you want to display the refractive indexes for each material.

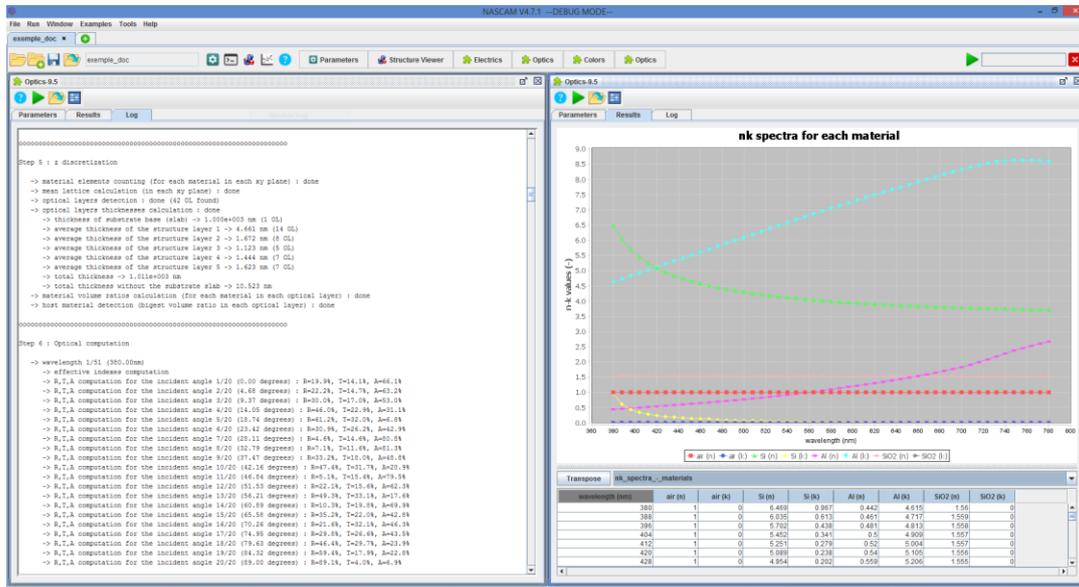


Figure 38: Left: log results. Right: refractive index spectra of all the materials in the structure.

The next results are the effective indexes (n and k) in function of z, as shown in Figure 39 (here, each curve represents one wavelength). They are stored in the “Effective_Optical_Index_VS_z_-_n/k” files.

The last and more important available results are the absorptance, the reflectance and the transmittance spectra in function of the wavelengths and the incident angles. The incident plane wave can be s-polarized (s), p-polarized (p) or unpolarized (sp). All results files are stored in the output folder and can be loaded by clicking on the “Results” tab.

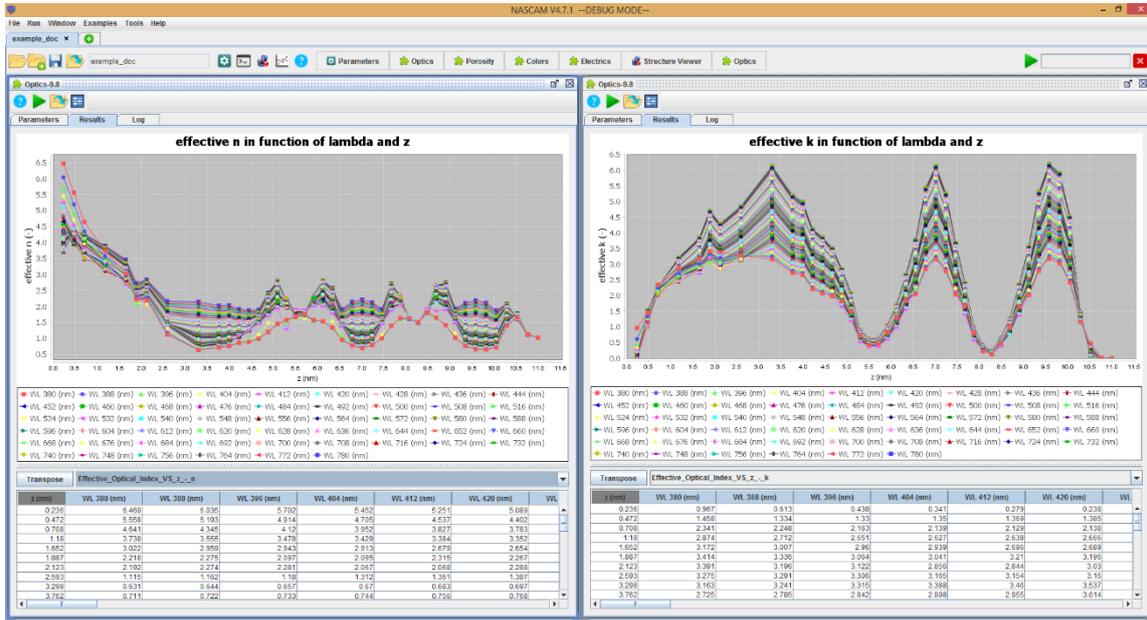


Figure 39: Effective indexes (n and k) in function of z and the wavelength.

Figure 40 shows the reflectance of the structure described in Figure 23 set on a slab-substrate (1 μm thickness). Each curve represents one incident angle. On the left, the incident light is unpolarized and coherent, and on the right, the incident light is unpolarized and incoherent. As you can see, there are some important differences between coherent and incoherent results. This is mainly due to interferences, which appear inside the slab-substrate (thicker than the wavelengths) in the case of coherent light.

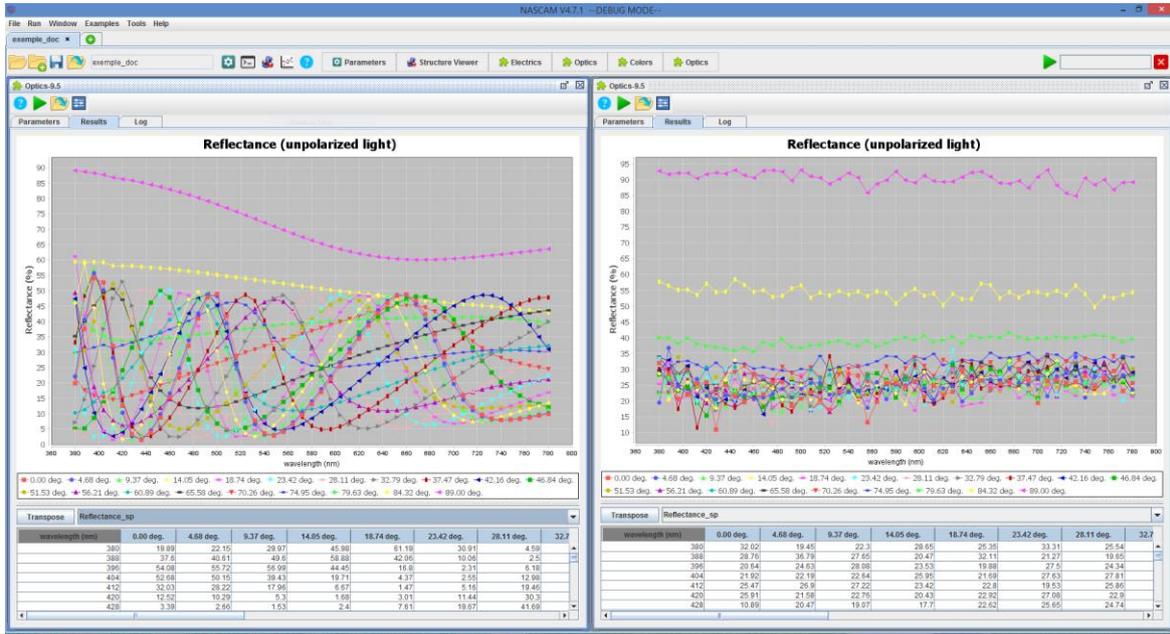


Figure 40: Reflectance in function of wavelengths and incident angles. Left: the incident light is unpolarized and coherent. Right: the incident light is unpolarized and incoherent (obtained by averaging 20 simulations).

9 Colors

Colors is designed to convert visible spectra to colour (i.e. to colour space coordinates as XYZ, Lab or RGB). By default, these visible spectra are the reflectance and the transmittance computed by the *Optics* plugin (see Section 8). The code is written in C.

9.1 Algorithm

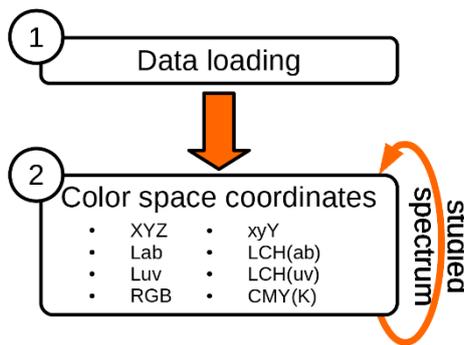
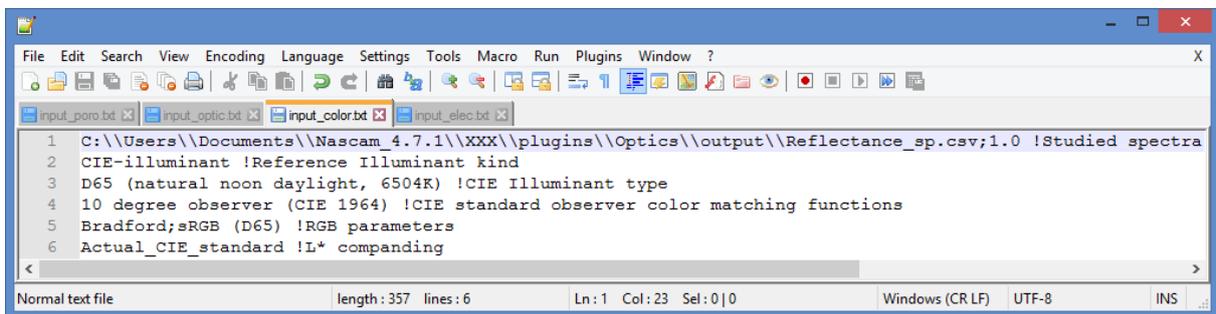


Figure 41: Flowchart of the Colors plugin.

Figure 41 summarizes the different steps of the *Colors* algorithm. More details are given below.

9.1.1 Data loading

During this step, three kinds of files are loaded. The first one is the *input.txt* file where all the parameters set by the user are stored (see Figure 42). For more details about parameters, see subsection 9.2.1.



```

1 C:\Users\Documents\Nascam_4.7.1\XXX\plugins\Optics\output\Reflectance_sp.csv;1.0 !Studied spectra
2 CIE-illuminant !Reference Illuminant kind
3 D65 (natural noon daylight, 6504K) !CIE Illuminant type
4 10 degree observer (CIE 1964) !CIE standard observer color matching functions
5 Bradford;sRGB (D65) !RGB parameters
6 Actual_CIE_standard !L* companding
  
```

Figure 42: Colors input.txt file.

The second kind of data are the spectra (reflectance and/or transmittance) you want to convert in colour. Usually, these spectra are stored in different *.csv* files provided by the *Optics* plugin,

but you can create your own spectra file if you follow the same “layout”. They have to be large enough (from 380nm to 780nm) to describe all the visible spectra. If not, the computation will be cancelled.

The last kind of data is the colour database (provided by the NASCAM SQL database⁵) where are stored:

- the reference illuminant spectra
- CIE standard observer colour matching functions (human eye sensibility)
- the reference white tristimulus values of the reference illuminant spectra
- the chromatic adaptation matrices
- the RGB working space parameters
- the CIE L* function parameters

9.1.2 Chromaticity coordinates calculation

This subsection resumes the steps to compute the chromaticity coordinates of a colour. More details are accessible on the [brucelindbloom](http://brucelindbloom.com) website.

❖ Flowchart

Figure 43 shows us all the useful steps to compute the chromaticity coordinates from a visible spectrum in different colour spaces.

⁵ The SQL database can be updated by the user.

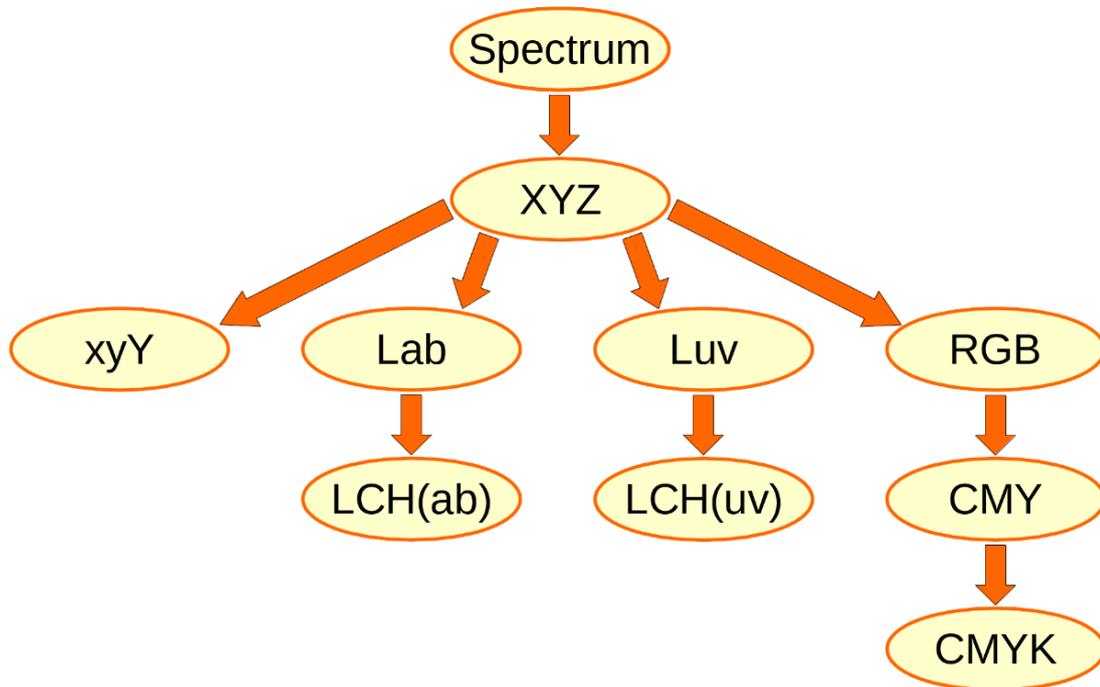


Figure 43: Colour space coordinates calculation - flowchart

❖ **Spectral data to XYZ tristimulus values and xyY chromaticity coordinates**

The XYZ tristimulus values are obtained by solving the following system:

$$\begin{cases} X = \frac{1}{N} \int_{\lambda} \bar{x}(\lambda) S(\lambda) I(\lambda) d\lambda \approx \sum_{\lambda} \bar{x}(\lambda_i) S(\lambda_i) I(\lambda_i) \Delta\lambda \\ Y = \frac{1}{N} \int_{\lambda} \bar{y}(\lambda) S(\lambda) I(\lambda) d\lambda \approx \sum_{\lambda} \bar{y}(\lambda_i) S(\lambda_i) I(\lambda_i) \Delta\lambda \\ Z = \frac{1}{N} \int_{\lambda} \bar{z}(\lambda) S(\lambda) I(\lambda) d\lambda \approx \sum_{\lambda} \bar{z}(\lambda_i) S(\lambda_i) I(\lambda_i) \Delta\lambda \\ N = \int_{\lambda} \bar{y}(\lambda) I(\lambda) d\lambda \approx \sum_{\lambda} \bar{y}(\lambda_i) I(\lambda_i) \Delta\lambda \end{cases}$$

where $S(\lambda)$ is the studied spectrum (e.g. a reflectance spectrum provided by the *Optics* plugin) and $I(\lambda)$ is the reference illuminant spectrum. The integration is done in the visible spectrum, from 380nm to 780nm. $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ and $\bar{z}(\lambda)$ are the CIE standard observer functions (2 or 10 degree). Then, the XYZ values can be converted to xyY chromaticity coordinates by using:

$$\begin{cases} x = \frac{X}{X + Y + Z} \\ y = \frac{Y}{X + Y + Z} \\ Y = Y \end{cases}$$

❖ XYZ tristimulus values to Lab and LCH(ab) chromaticity coordinates

The Lab coordinates are obtained by solving the following equations:

$$\begin{cases} L = 116f_y - 16 \\ a = 500(f_x - f_y) \\ b = 200(f_y - f_z) \end{cases} \quad \text{with} \quad \begin{cases} f_x = \begin{cases} \sqrt[3]{X/X_r} & (\text{if } X/X_r > \epsilon) \\ \frac{\kappa X/X_r + 16}{116} & (\text{if } X/X_r \leq \epsilon) \end{cases} \\ f_y = \begin{cases} \sqrt[3]{Y/Y_r} & (\text{if } Y/Y_r > \epsilon) \\ \frac{\kappa Y/Y_r + 16}{116} & (\text{if } Y/Y_r \leq \epsilon) \end{cases} \\ f_z = \begin{cases} \sqrt[3]{Z/Z_r} & (\text{if } Z/Z_r > \epsilon) \\ \frac{\kappa Z/Z_r + 16}{116} & (\text{if } Z/Z_r \leq \epsilon) \end{cases} \end{cases}$$

where X_r , Y_r and Z_r are the reference white tristimulus values of the illuminant and ϵ and κ are CIE standard L^* companding parameters defined by:

$$\begin{cases} \epsilon = \begin{cases} 0.008856 & (\text{actual CIE standard}) \\ 216/24389 & (\text{intent of the CIE standard}) \end{cases} \\ \kappa = \begin{cases} 903.3 & (\text{actual CIE standard}) \\ 24389/27 & (\text{intent of the CIE standard}) \end{cases} \end{cases}$$

Then, it is possible to compute the LCH(ab) coordinates thanks to:

$$\begin{cases} L = L \\ C = \sqrt{a^2 + b^2} \\ H = \begin{cases} \tan^{-1}(b/a) & (\text{if } \tan^{-1}(b/a) \geq 0) \\ \tan^{-1}(b/a) + 2\pi & (\text{if } \tan^{-1}(b/a) < 0) \end{cases} \end{cases}$$

Notes: in the results provided by the *Colors* plugin, H is expressed in degrees.

❖ XYZ tristimulus values to Luv and LCH(uv)

The Luv coordinates are obtained by solving the following equations:

$$\left\{ \begin{array}{l} L = \begin{cases} 116\sqrt[3]{Y/Y_r} - 16 & (\text{if } Y/Y_r > \epsilon) \\ \kappa Y/Y_r & (\text{if } Y/Y_r \leq \epsilon) \end{cases} \\ u = 13L(u' - u'_r) \\ v = 13L(v' - v'_r) \end{array} \right. \quad \text{with} \quad \left\{ \begin{array}{l} u' = \frac{4X}{X + 15Y + 3Z} \\ v' = \frac{9Y}{X + 15Y + 3Z} \\ u'_r = \frac{4X'_r}{X'_r + 15Y'_r + 3Z'_r} \\ v'_r = \frac{9Y'_r}{X'_r + 15Y'_r + 3Z'_r} \end{array} \right.$$

where X_r , Y_r and Z_r are the reference white tristimulus values of the illuminant and ϵ and κ are CIE standard L^* companding parameters defined by:

$$\left\{ \begin{array}{l} \epsilon = \begin{cases} 0.008856 & (\text{actual CIE standard}) \\ 216/24389 & (\text{intent of the CIE standard}) \end{cases} \\ \kappa = \begin{cases} 903.3 & (\text{actual CIE standard}) \\ 24389/27 & (\text{intent of the CIE standard}) \end{cases} \end{array} \right.$$

Then, it is possible to compute the LCH(uv) coordinates thanks to:

$$\left\{ \begin{array}{l} L = L \\ C = \sqrt{u^2 + v^2} \\ H = \begin{cases} \tan^{-1}(v/u) & (\text{if } \tan^{-1}(v/u) \geq 0) \\ \tan^{-1}(v/u) + 2\pi & (\text{if } \tan^{-1}(v/u) < 0) \end{cases} \end{array} \right.$$

Notes: in the results provided by the *Colors* plugin, H is expressed in degrees.

❖ XYZ tristimulus values to RGB, CMY and CMYK

The conversion of XYZ values to linear rgb values is done by:

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = M^{-1} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

where M is the rgb/XYZ transformation matrix given by:

$$M = \begin{bmatrix} S_r X_r & S_g X_g & S_b X_b \\ S_r Y_r & S_g Y_g & S_b Y_b \\ S_r Z_r & S_g Z_g & S_b Z_b \end{bmatrix}$$

$$\text{with: } \left\{ \begin{array}{l} X_r = x_r/y_r \\ Y_r = 1 \\ Z_r = (1 - x_r - y_r)/y_r \end{array} \right. ; \left\{ \begin{array}{l} X_g = x_g/y_g \\ Y_g = 1 \\ Z_g = (1 - x_g - y_g)/y_g \end{array} \right. ; \left\{ \begin{array}{l} X_b = x_b/y_b \\ Y_b = 1 \\ Z_b = (1 - x_b - y_b)/y_b \end{array} \right.$$

where $(x_r, y_r, x_g, y_g, x_b, y_b)$ are the chromaticity coordinates of the RGB working space, and

$$\begin{bmatrix} S_r \\ S_g \\ S_b \end{bmatrix} = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix}^{-1} \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$$

where (X_W, Y_W, Z_W) are the reference white tristimulus values of the RGB working space.

Furthermore, if the reference illuminant of the RGB working space is different from the one asked by the user (defined by its white reference tristimulus values (X_W^u, Y_W^u, Z_W^u)), a chromatic adaptation is needed and is done by using a modified rgb/XYZ transformation:

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = M^{-1} M_A^{-1} \begin{bmatrix} \rho/\rho_u & 0 & 0 \\ 0 & \gamma/\gamma_u & 0 \\ 0 & 0 & \beta/\beta_u \end{bmatrix} M_A \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

where $\begin{bmatrix} \rho_u \\ \gamma_u \\ \beta_u \end{bmatrix} = M_A \begin{bmatrix} X_W^u \\ Y_W^u \\ Z_W^u \end{bmatrix}$ is the cone response for the reference illuminant chosen by the user,

and $\begin{bmatrix} \rho \\ \gamma \\ \beta \end{bmatrix} = M_A \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$ is the one for the reference illuminant of the RGB working space.

The matrix M_A depends on the chromatic adaptation method chosen by the user:

$$M_A = \begin{matrix} \text{XYZ scaling} & \text{Bradford} & \text{Von Kries} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} ; & \begin{bmatrix} +0.895 & +0.266 & -0.161 \\ -0.750 & +1.714 & +0.037 \\ +0.039 & -0.069 & +1.030 \end{bmatrix} ; & \begin{bmatrix} +0.400 & +0.708 & -0.081 \\ -0.226 & +1.165 & +0.046 \\ 0 & 0 & +0.918 \end{bmatrix} \end{matrix}$$

Now, the next step consists in converting the linear rgb to non-linear RGB⁶ coordinates. Then, the method depends on the chosen RGB working space. The first method is the Gamma companding given by:

⁶ the RGB range is [0;1] here

$$\begin{cases} R = r^{1/\gamma} \\ G = g^{1/\gamma} \\ B = b^{1/\gamma} \end{cases}$$

The second method is the sRGB companding:

$$\begin{cases} R = \begin{cases} 1.055r^{1/2.4} - 0.055 & (\text{if } r > 0.0031308) \\ 12.92r & (\text{if } r \leq 0.0031308) \end{cases} \\ G = \begin{cases} 1.055g^{1/2.4} - 0.055 & (\text{if } g > 0.0031308) \\ 12.92g & (\text{if } g \leq 0.0031308) \end{cases} \\ B = \begin{cases} 1.055b^{1/2.4} - 0.055 & (\text{if } b > 0.0031308) \\ 12.92b & (\text{if } b \leq 0.0031308) \end{cases} \end{cases}$$

and the last one is the L* companding:

$$\begin{cases} R = \begin{cases} 1.16\sqrt[3]{r} - 0.16 & (\text{if } r > \epsilon) \\ \kappa r/100 & (\text{if } r \leq \epsilon) \end{cases} \\ G = \begin{cases} 1.16\sqrt[3]{g} - 0.16 & (\text{if } g > \epsilon) \\ \kappa g/100 & (\text{if } g \leq \epsilon) \end{cases} \\ B = \begin{cases} 1.16\sqrt[3]{b} - 0.16 & (\text{if } b > \epsilon) \\ \kappa b/100 & (\text{if } b \leq \epsilon) \end{cases} \end{cases}$$

where ϵ and κ are CIE standard L* companding parameters defined by:

$$\begin{cases} \epsilon = \begin{cases} 0.008856 & (\text{actual CIE standard}) \\ 216/24389 & (\text{intent of the CIE standard}) \end{cases} \\ \kappa = \begin{cases} 903.3 & (\text{actual CIE standard}) \\ 24389/27 & (\text{intent of the CIE standard}) \end{cases} \end{cases}$$

Finally, it is possible to compute the CMY and CMYK coordinates:

$$\begin{cases} C = 1 - R \\ M = 1 - G \\ Y = 1 - B \end{cases} \quad \text{and} \quad \begin{cases} K = 1 - \max(R, G, B) \\ C = \frac{(1 - R - K)}{(1 - K)} \\ M = \frac{(1 - G - K)}{(1 - K)} \\ Y = \frac{(1 - B - K)}{(1 - K)} \end{cases}$$

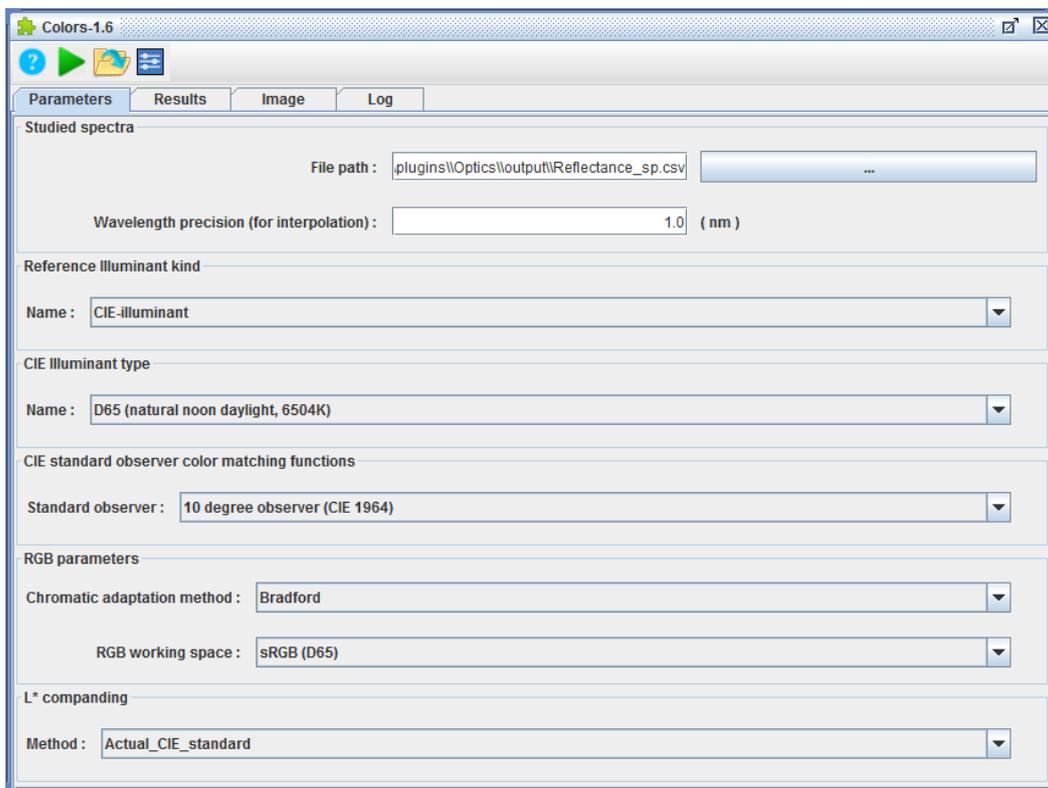
9.2 NASCAM GUI example

In this section, we study a 3D 5-layer stack (Al/SiO₂) on a Si slab (1 μ m) substrate structured by triangular rows (see Figure 23). All studied spectra (reflectance, transmittance) are obtained thanks to the *Optics* plugin (see subsection 8.2).

9.2.1 Input parameters

The *Colors* parameter window pops up by clicking on Tools→Colors. The *Colors* parameters defined in Figure 44 are:

- the file where are stored the studied spectra (usually the reflectance or transmittance spectra stored in the Optics\output folder)
- the wavelength step used to interpolate these spectra (useful if the spectra have just a few points)
- the reference illuminant used to light the related structure
- the CIE standard observer colour matching functions (relative to human eye sensibility)
- the RGB parameters:
 - the chromatic adaptation method (useful if the reference illuminant of the RGB working space is different from the one asked by the user)
 - the RGB working space. This is a device-dependent colour space, which has to be adapted to your display device such as your computer monitor or your printer.
- the L* companding parameters needed to compute the Lab and Luv coordinates



The screenshot shows the 'Colors-1.6' software window with the following settings:

- Studied spectra:** File path: `plugins\Optics\output\Reflectance_sp.csv`; Wavelength precision (for interpolation): `1.0` (nm)
- Reference Illuminant kind:** Name: `CIE-illuminant`
- CIE Illuminant type:** Name: `D65 (natural noon daylight, 6504K)`
- CIE standard observer color matching functions:** Standard observer: `10 degree observer (CIE 1964)`
- RGB parameters:** Chromatic adaptation method: `Bradford`; RGB working space: `sRGB (D65)`
- L* companding:** Method: `Actual_CIE_standard`

Figure 44: *Colors* input parameters.

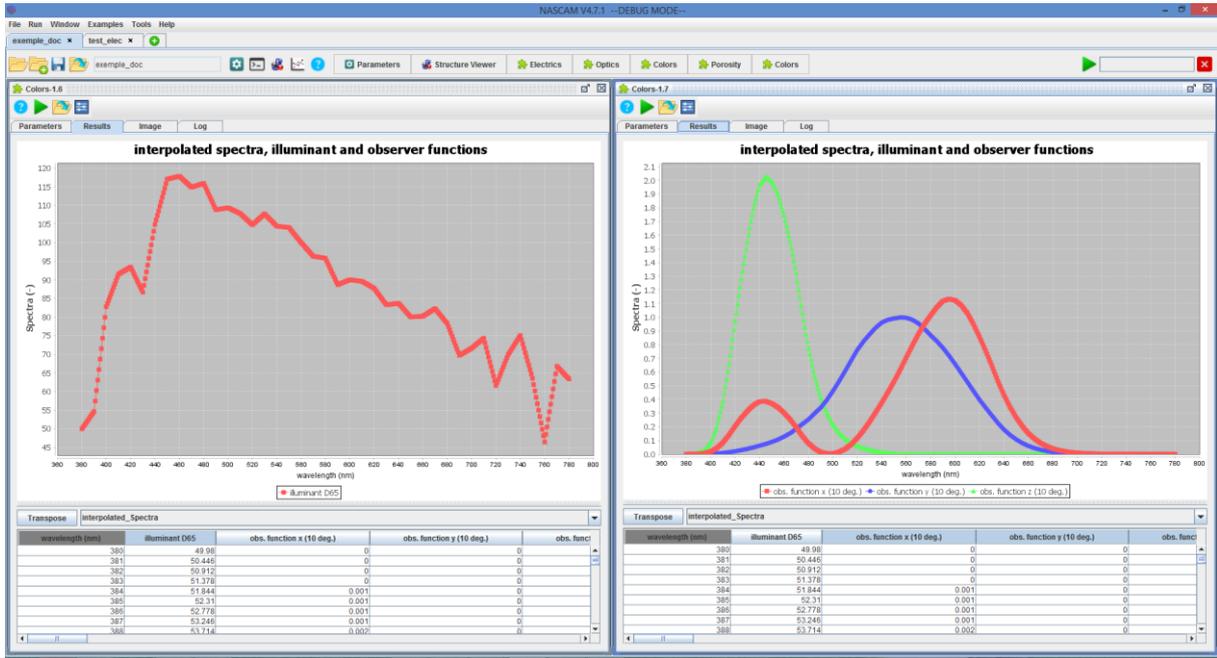


Figure 46: (left) CIE D65 reference illuminant spectrum.
 (right) CIE 10° standard observer colour matching functions.

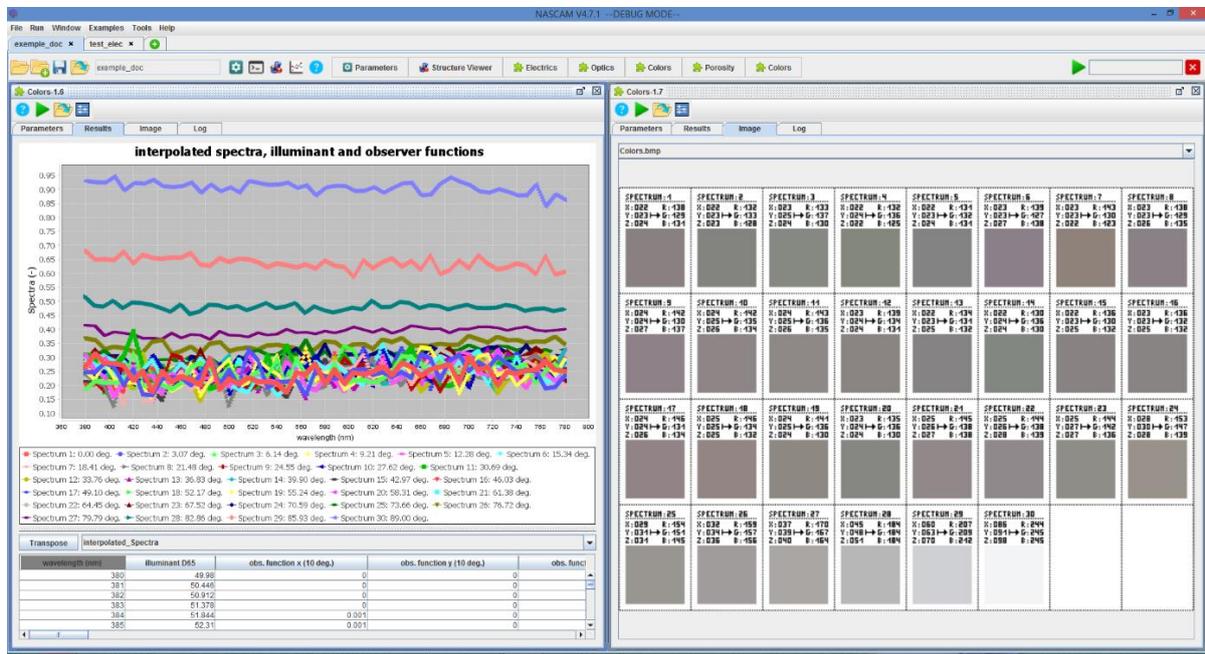


Figure 47: (left) interpolated studied spectra (for an unpolarized incoherent D65 incident light).
 (right) bitmap RGB colour display for each studied spectrum.

10 Electrics

Electrics is designed to compute the effective conductivity of a multilayer structure provided by NASCAM. The code is written in C.

10.1 Algorithm

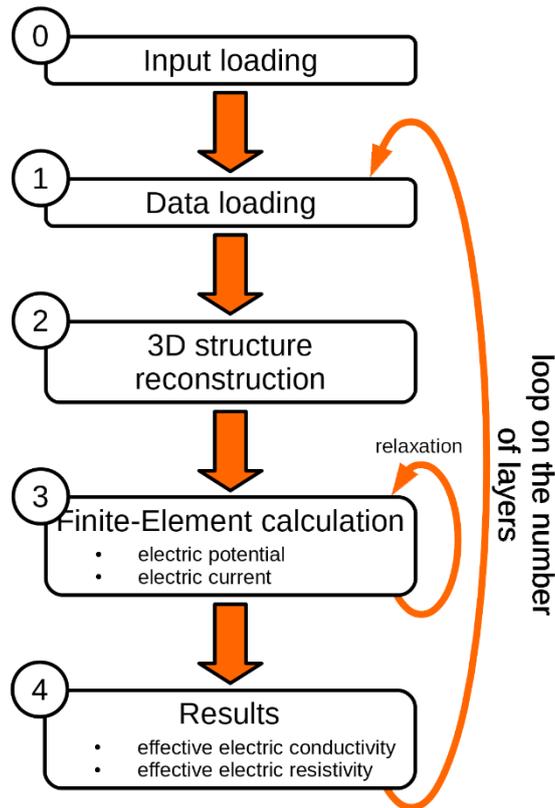


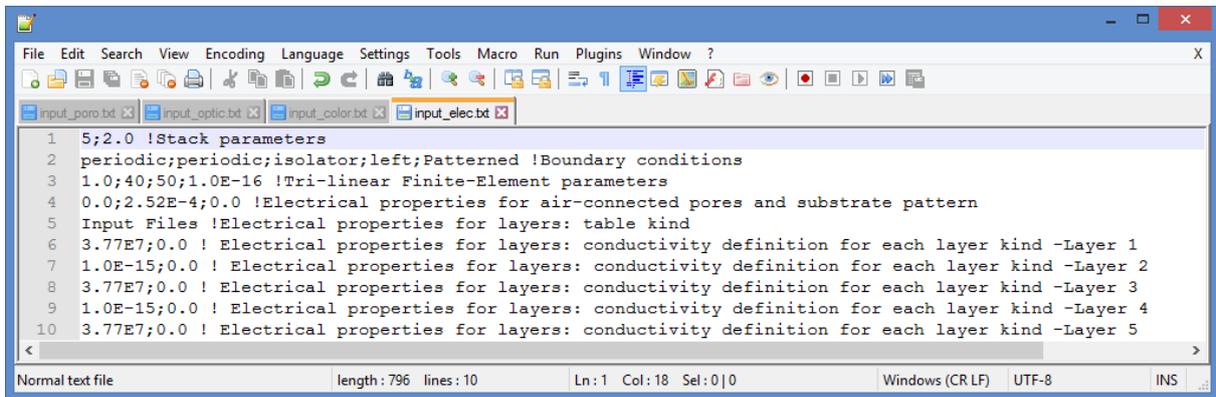
Figure 48: Flowchart of the *Electrics* plugin.

Figure 48 summarizes the different steps of the *Electrics* algorithm. More details are given below.

10.1.1 Data loading

During this step, two kinds of files are loaded. The first one is the *input.txt* file where all the parameters set by the user are stored (see Figure 49). For more details about the parameters, see subsection 10.2.1.

The second kind of data file is the *deposited_layer.xyz* provided by NASCAM (one file per layer). In each file are stored the coordinates of all atoms of a given layer. If needed, the *substrate.xyz file* can be loaded too.



```

1 5;2.0 !Stack parameters
2 periodic;periodic;isolator;left;Patterned !Boundary conditions
3 1.0;40;50;1.0E-16 !Tri-linear Finite-Element parameters
4 0.0;2.52E-4;0.0 !Electrical properties for air-connected pores and substrate pattern
5 Input Files !Electrical properties for layers: table kind
6 3.77E7;0.0 ! Electrical properties for layers: conductivity definition for each layer kind -Layer 1
7 1.0E-15;0.0 ! Electrical properties for layers: conductivity definition for each layer kind -Layer 2
8 3.77E7;0.0 ! Electrical properties for layers: conductivity definition for each layer kind -Layer 3
9 1.0E-15;0.0 ! Electrical properties for layers: conductivity definition for each layer kind -Layer 4
10 3.77E7;0.0 ! Electrical properties for layers: conductivity definition for each layer kind -Layer 5

```

Figure 49: Electrics *input.txt* file.

10.1.2 Structure reconstruction

The method to recreate a 3D structure is similar to the one used by the *Optics* plugin (see subsection 8.1.2). The only difference is that each integer representing one material inside the 3D matrix is directly linked to an electric conductivity (instead of an optical index spectrum).

10.1.3 Finite-element calculation **(incomplete description!)**

The algorithm is mainly based on a finite-element program developed by NIST⁷ (Garboczi, 1998; Scocchi, 2013). We will resume the main steps, summarized in Figure 50.

⁷ [National Institute of Standards and Technology](http://www.nist.gov)

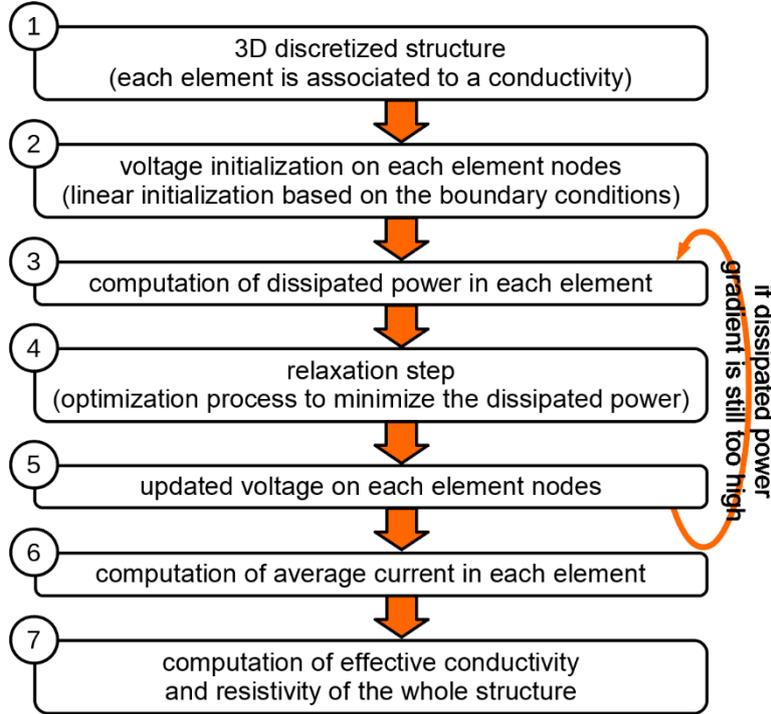


Figure 50: Flowchart of finite-element algorithm.

❖ Dissipated power calculation

The main goal of the algorithm is to compute the voltage field in the whole structure allowing minimal dissipated power. The general expression of the power is given by:

$$P = \iiint \vec{\mathbf{E}}^t \bar{\bar{\boldsymbol{\sigma}}} \vec{\mathbf{E}} dx dy dz$$

where $\vec{\mathbf{E}}$ is the electric field (3×1 vector) and $\bar{\bar{\boldsymbol{\sigma}}}$ is the 3×3 conductivity tensor.

To compute such power, we need to solve the Maxwell-Faraday equation:

$$\vec{\nabla} \times \vec{\mathbf{E}} = -\frac{\partial \vec{\mathbf{B}}}{\partial t}$$

with $\vec{\mathbf{B}}$ the magnetic field. In electrostatics (by considering a near-absence of varying magnetic field), this equation could be simplified in:

$$\vec{\mathbf{E}} = -\vec{\nabla}V$$

with V the electric potential or voltage. The local electric field components at the position (x, y, z) are then given by:

$$e_i(x, y, z) = -\frac{\partial u(x, y, z)}{\partial i} \quad \text{with } i = x, y, z$$

To solve such an equation in our system, the basic idea consists in discretizing the studied structure in elements (or pixels). In our case, these elements correspond to the cubic cells of the 3D meshed structure described above (see 10.1.2). As shown in Figure 51, these elements have 8 nodes and 26 neighbours. In the current plugin version, $\Delta x = \Delta y = \Delta z = \Delta$, where Δ is the cubic element edge size (in lattice unit) set by the user.

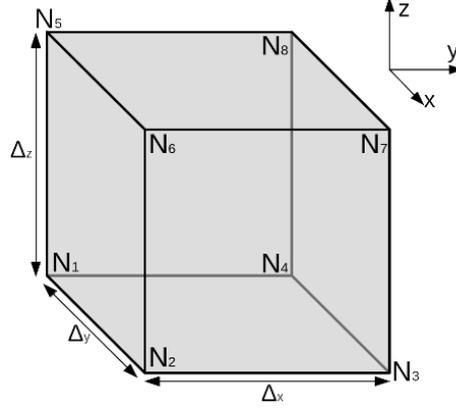


Figure 51: Tri-linear finite element (pixel) and its 8 nodes.

It is now possible to compute the electric potential at each node of the mesh, and then express $u(x, y, z)$ as a tri-linear interpolation of the 8 nodal voltages of an element n stored in the 8×1 vector \vec{u}_n .

Thereby, we can compute the power dissipated in this element n :

$$P_n = \int_0^\Delta \int_0^\Delta \int_0^\Delta \vec{E}^t \bar{\sigma}_n \vec{E} dx dy dz = \int_0^\Delta \int_0^\Delta \int_0^\Delta (-\vec{\nabla}u(x, y, z))^t \bar{\sigma}_n (-\vec{\nabla}u(x, y, z)) dx dy dz$$

This expression can be simplified in:

$$P_n = \vec{u}_n^t \bar{\mathbf{A}}_n \vec{u}_n$$

where $\bar{\mathbf{A}}_n$ is the 8×8 stiffness matrix of the element n given by :

$$\bar{\mathbf{A}}_n = \int_0^\Delta \int_0^\Delta \int_0^\Delta \vec{l}_n^t \bar{\sigma}_n \vec{l}_n dx dy dz$$

where $\vec{l}_n = \vec{\nabla} \vec{L}^t$ is a 3×8 matrix making the link between \vec{u}_n and \vec{E} . Such integration is done by using the Simpson method. \vec{L} is a 8×1 vector used for the linear interpolation of u inside the element, given by:

$$u(x, y, z) = \vec{L}_n^t \vec{u}_n$$

It is important to note that boundary conditions are not yet taken into account here. The exact expression of P_n is then:

$$P_n = \vec{u}_n^t \bar{\mathbf{A}}_n \vec{u}_n + \vec{b}_n^t \vec{u}_n + c_n$$

For more details about the boundary condition dependent variables \vec{b}_n and c_n , refer to the [NIST report](#).

To obtain the total energy P dissipated by the whole structure, a simple summation is needed:

$$P = \sum_n P_n$$

❖ Dissipated power gradient calculation

As explained above, the base of the algorithm is to minimize the dissipated power. It is done by using the conjugate gradient method, which needs the computation of the dissipated power gradient:

$$G = \partial P / \partial V$$

In our discretized structure, such a gradient has to be computed by taking into account all neighbours of the element:

$$\vec{G}_n = \begin{pmatrix} \partial / \partial u_1 \\ \vdots \\ \partial / \partial u_8 \end{pmatrix} P_n = \bar{\mathbf{A}}'_n \vec{u}_n + \vec{b}_n$$

where $\bar{\mathbf{A}}'_n$ is a global 8×8 stiffness matrix built from all the stiffness matrices of the eight neighbouring elements in contact with each node of the element n .

❖ Electrical current calculation

The local electrical current in each element is computed by averaging the diagonal currents in the 4 diagonal directions of the cubic element (see Figure 51):

$$\vec{i}_n = \frac{\vec{i}_{1 \rightarrow 7} + \vec{i}_{2 \rightarrow 8} + \vec{i}_{3 \rightarrow 5} + \vec{i}_{4 \rightarrow 6}}{4}$$

Each diagonal current is given by:

$$\vec{i}_{i \rightarrow j} = (u_i - u_j) \bar{\sigma}_n \hat{u}_{i \rightarrow j} \quad \text{with } \hat{u}_{i \rightarrow j} \text{ the unit vector giving the diagonal direction.}$$

❖ Effective conductivity calculation

It is possible to obtain the effective electrical conductivity thanks to the volume average current per element:

$$\vec{I}_{\text{mean}} = \frac{1}{N} \sum_n \vec{I}_n$$

The effective conductivity in the i direction is then given by ([see MIT lecture - chapter 3](#)):

$$\sigma_{\text{eff}}|_i = \frac{I_{\text{mean}}|_i N_i \Delta_i}{\Delta V_i \Delta_j \Delta_k} \quad \text{with } i, j, k = x, y, z$$

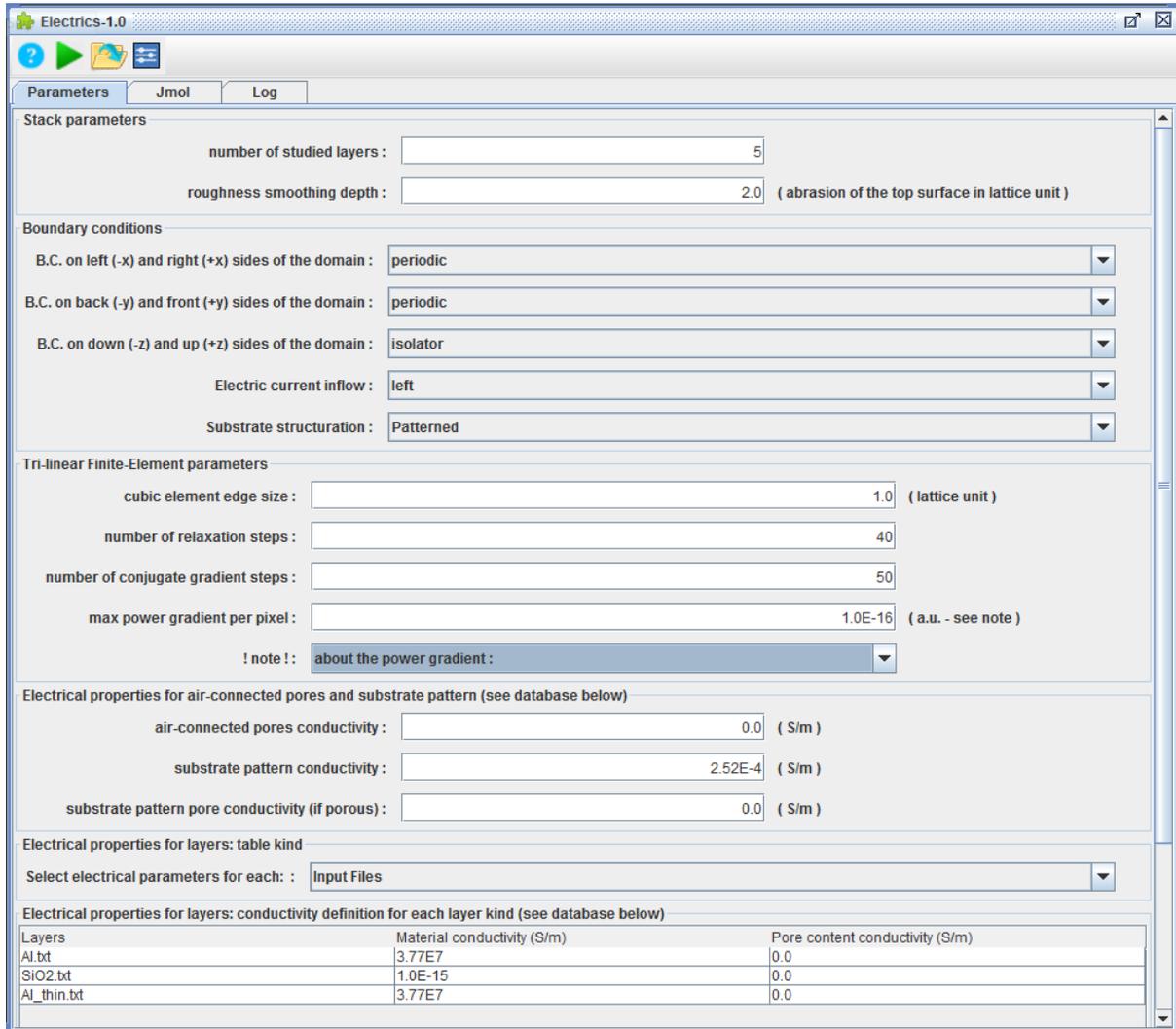
where N_i , Δ_i and ΔV_i are respectively the number of elements in the i direction, the size of an element in the i direction and the electric potential difference in the i direction (set by the boundary conditions).

10.2 NASCAM GUI example

10.2.1 Input parameters

The *Electrics* parameter window pops up by clicking on Tools→Electrics. The *Electrics* parameters defined in Figure 52 are:

- the stack parameters:
 - the choice to study the whole structure only, or to do a loop by increasing the number of layers
 - the number of studied layers asked by the user (can be adapted automatically if too high)
 - the n last z -levels of atoms to be removed on the top surface of the stack (to abrase and/or smooth the roughness)
- the boundary conditions:
 - the boundary condition on the left ($-x$) and right ($+x$) sides of the domain: can be periodic, insulating or conductive
 - the boundary condition on the back ($-y$) and front ($+y$) sides of the domain: can be periodic, insulating or conductive
 - the boundary condition on the down ($-z$) and up ($+z$) sides of the domain: can be periodic, insulating or conductive
 - the electric current inflow side: be careful not to put an insulating boundary condition on this side!!!
 - at the substrate side ($-z$): the substrate can be patterned or flat



The screenshot shows the 'Electrics-1.0' software interface with the following parameters:

- Stack parameters:**
 - number of studied layers: 5
 - roughness smoothing depth: 2.0 (abrasion of the top surface in lattice unit)
- Boundary conditions:**
 - B.C. on left (-x) and right (+x) sides of the domain: periodic
 - B.C. on back (-y) and front (+y) sides of the domain: periodic
 - B.C. on down (-z) and up (+z) sides of the domain: isolator
 - Electric current inflow: left
 - Substrate structuration: Patterned
- Tri-linear Finite-Element parameters:**
 - cubic element edge size: 1.0 (lattice unit)
 - number of relaxation steps: 40
 - number of conjugate gradient steps: 50
 - max power gradient per pixel: 1.0E-16 (a.u. - see note)
 - ! note!: about the power gradient:
- Electrical properties for air-connected pores and substrate pattern (see database below):**
 - air-connected pores conductivity: 0.0 (S/m)
 - substrate pattern conductivity: 2.52E-4 (S/m)
 - substrate pattern pore conductivity (if porous): 0.0 (S/m)
- Electrical properties for layers: table kind**
- Select electrical parameters for each: Input Files
- Electrical properties for layers: conductivity definition for each layer kind (see database below)**

Layers	Material conductivity (S/m)	Pore content conductivity (S/m)
Al.bt	3.77E7	0.0
SiO2.bt	1.0E-15	0.0
Al_thin.bt	3.77E7	0.0

Figure 52: *Electrics* input parameters.

- the Tri-linear Finite-Element parameters:
 - the cubic element edge size (in lattice unit): the structure is discretized by using a cubic mesh
 - the number of relaxation steps
 - the number of conjugate gradient calculation steps for each relaxation step
 - the maximal power gradient per pixel: due to the normalization of variables during the calculation, the power gradient is dimensionless. By default, its maximal value is set to 1.E-16
- Electrical properties for air-connected pores and substrate pattern:
 - the air-connected pores electrical conductivity (S/m)
 - (if the substrate is structured) substrate pattern electrical conductivity (S/m)
 - (if the substrate is structured) substrate pattern pore electrical conductivity (S/m)
- Electrical properties - table kind: method to define electrical properties for each layer (per layer or per input file)

- Electrical Properties for layers: choice of the electrical conductivity of host and pore content for each layer or input file (in function of the table kind choice)

10.2.2 Results

The first results you can observe during the computation is in the log window. As shown in Figure 53, you can access the effective electric conductivity computed by finite element.

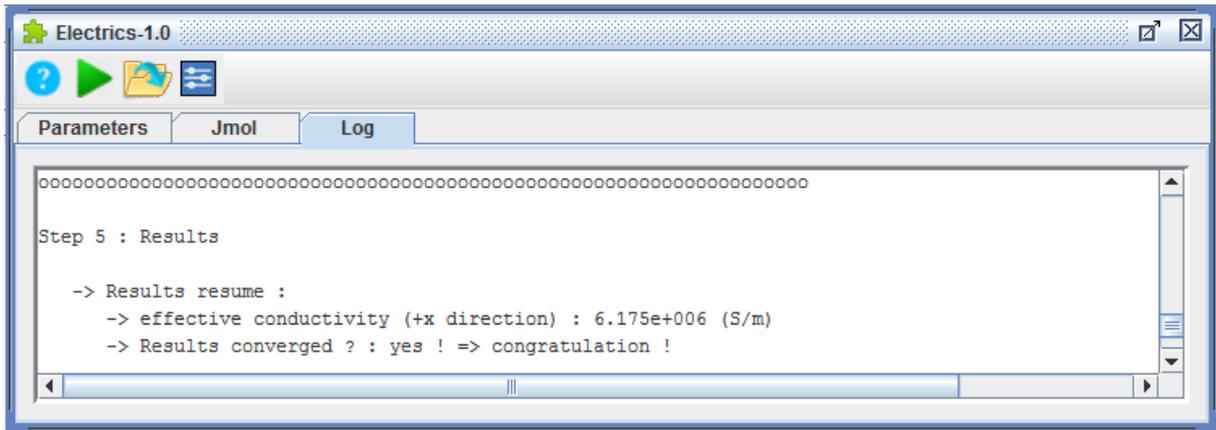


Figure 53: log display: final effective electrical conductivity

The second available results are 3D JMOL maps of the electric conductivity (Figure 54 - left), the electric potentials (Figure 54 - right) and of the electric currents (Figure 55).

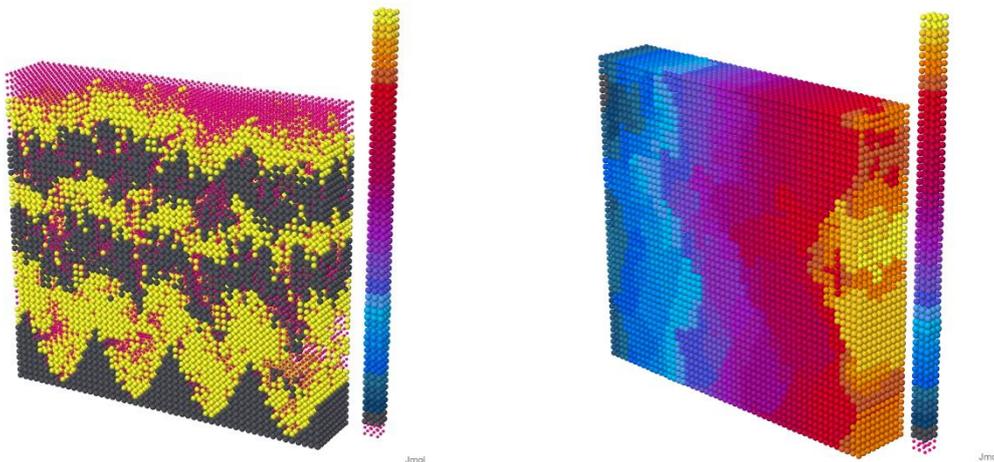


Figure 54: JMOL 3D display. Left: electrical conductivities (a.u.). Right: electric potentials (a.u.)

As shown in Figure 55, it is possible to display the current intensities (left - automatic) or the current vectors (right) by typing the “spacefill 3%;vector scale 3.0;vector 0.05;” command.

Moreover, it is possible to display an xz cross-section by typing the “display y==0” command (the value of y depends on the coordinates of the structure).

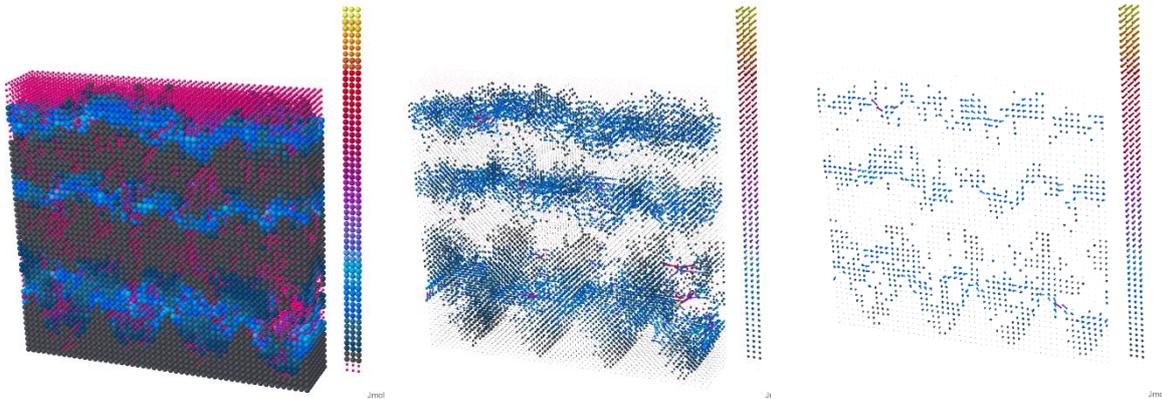


Figure 55: JMOL display of electric current. Left: electric current magnitude (a.u.). Middle: electric current vectors. Right: electric current vectors xz cross-section (a.u.).

Finally, if the “loop on the number of layers” option is checked, the Results tab displays the evolution of effective conductivity (and resistivity) in function of the thickness or the number of layers (see Figure 56).

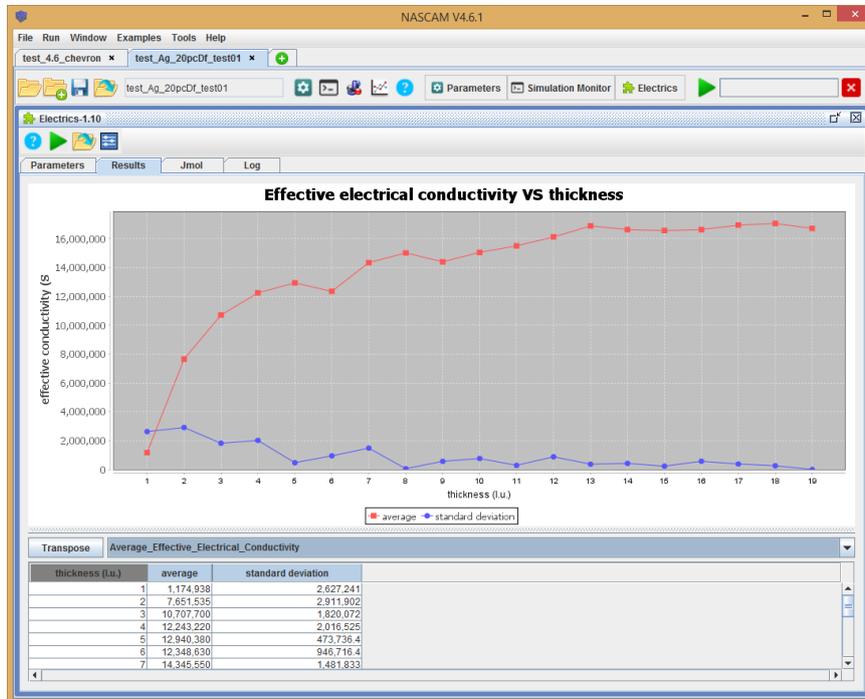
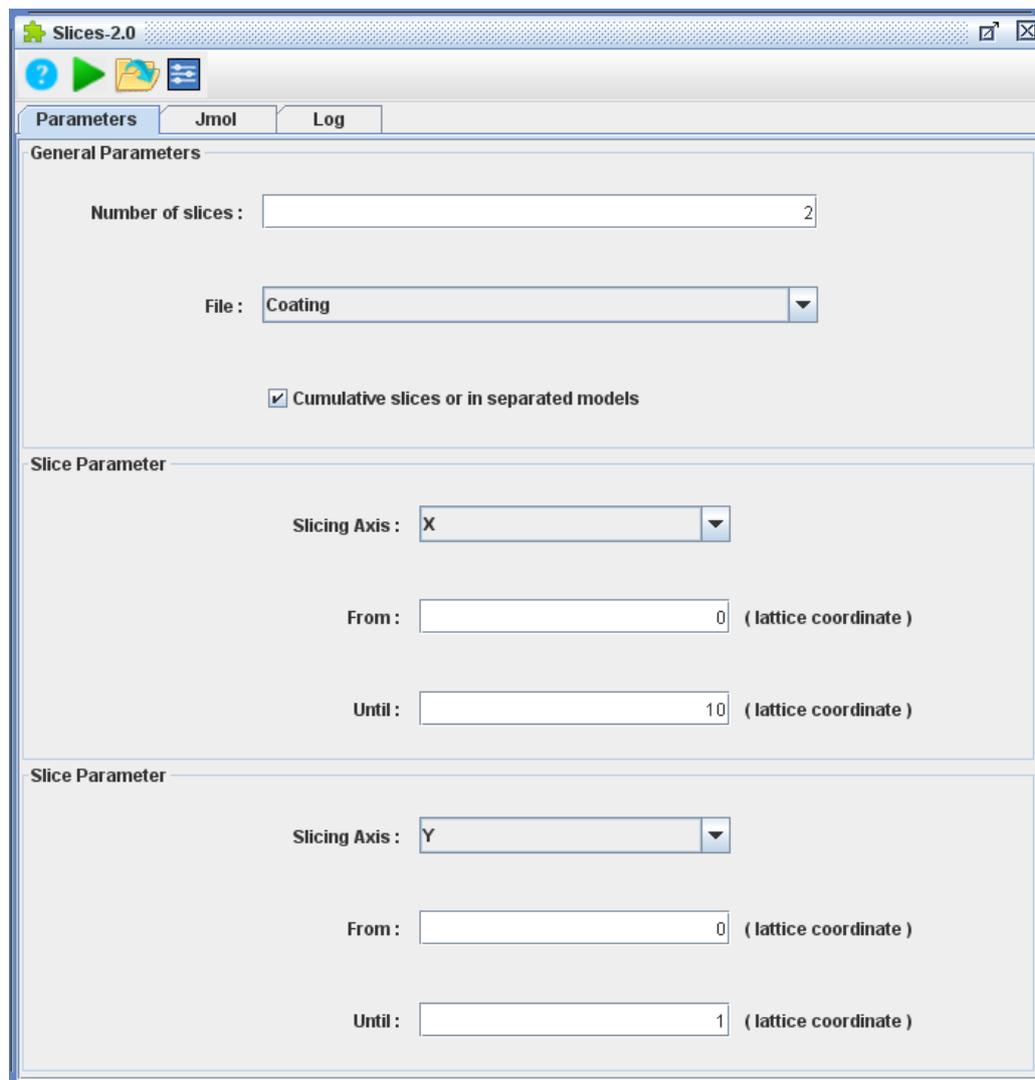


Figure 56: log display: final effective electrical conductivity

11 Slices

This plugin creates slices of a final structure "*coating.xyz*", substrate or user defined file. It is possible to make one or several slices of a desired thickness perpendicular to the X, Y, or Z-axis. For each slice, it is necessary to set the axis, the starting and the ending coordinates of the slice. One can produce either a set of slices (checkbox is not checked) or an intersection of slices. For the example shown in the figure below, two slices are produced; the first one contains atoms with x coordinate from $x=0$ to $x=10$, the second slice contains atoms with coordinate $y=0$ to $y=1$, then the intersection of these sets will be done.

In case the checkbox is not checked, two output files are produced, the first file contains atoms with x coordinate from $x=0$ to $x=10$, the second file contains atoms with coordinate $y=0$ to $y=1$.



The screenshot shows the 'Slices-2.0' software interface. It has three tabs: 'Parameters', 'Jmol', and 'Log'. The 'Parameters' tab is active and is divided into three sections:

- General Parameters:**
 - Number of slices: 2
 - File: Coating
 - Cumulative slices or in separated models
- Slice Parameter (1st slice):**
 - Slicing Axis: X
 - From: 0 (lattice coordinate)
 - Until: 10 (lattice coordinate)
- Slice Parameter (2nd slice):**
 - Slicing Axis: Y
 - From: 0 (lattice coordinate)
 - Until: 1 (lattice coordinate)

12 Thermal Conductivity

Thermal conductivity plugin is used to calculate a thermal conductivity of a single layer or multi-layer films and evaluate the temperature profile in the film. It is applicable to flat substrates or to featured substrates with aspect ratio of the features less than 1. The plugin is based on an effective medium theory that's why it is supposed that the coating is more or less uniform in the directions parallel to the substrate.

12.1 Algorithm

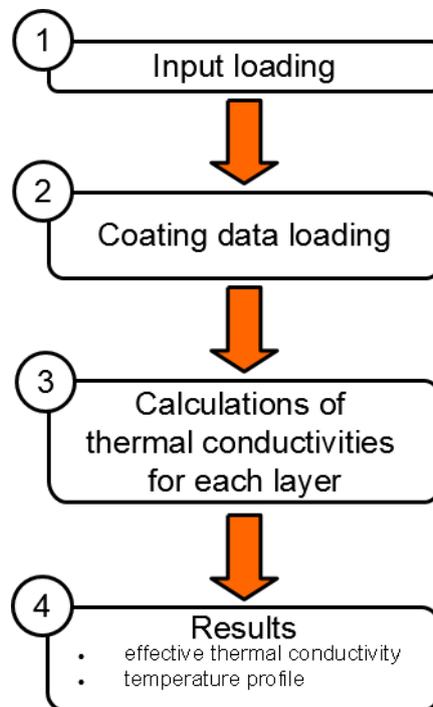


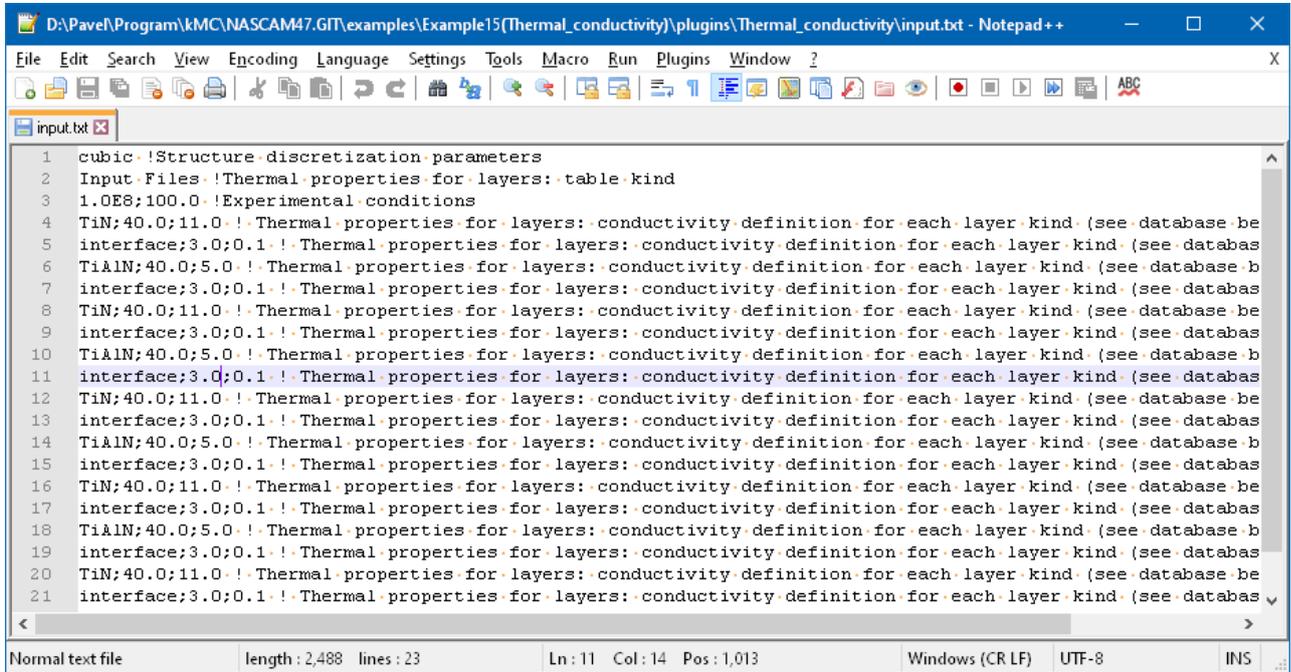
Figure 57 : Flowchart of the Thermal conductivity plugin.

Different steps of the *Thermal conductivity* plug-in are shown in Figure 57 .

12.1.1 Input and Coating data loading

During this step, two kinds of files are loaded. The first one is the *input.txt* file where all the parameters set by the user are stored, see Figure 58. Detailed description of the input is given below. The second data file is the simulation result of the coating deposition, *coating.xyz*, that

contains all the information on the coating. The porosities of sub-layers are calculated based on the data.



```

1 cubic !Structure discretization parameters
2 Input Files !Thermal properties for layers: table kind
3 1.0E8;100.0 !Experimental conditions
4 TiN;40.0;11.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database be
5 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
6 TiAlN;40.0;5.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database b
7 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
8 TiN;40.0;11.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database be
9 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
10 TiAlN;40.0;5.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database b
11 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
12 TiN;40.0;11.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database be
13 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
14 TiAlN;40.0;5.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database b
15 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
16 TiN;40.0;11.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database be
17 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
18 TiAlN;40.0;5.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database b
19 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas
20 TiN;40.0;11.0 ! Thermal properties for layers: conductivity definition for each layer kind (see database be
21 interface;3.0;0.1 ! Thermal properties for layers: conductivity definition for each layer kind (see databas

```

Figure 58 : Input file for Thermal conductivity plug-in.

12.1.2 Calculation of a thermal conductivity of a single deposited layer.

Calculation of a thermal conductivity of a layer k is based on two parameters. The first parameter is a thermal conductivity of a bulk material, k_0 . The second parameter is a pore distribution in the deposited film.

While k_0 is an input parameter for the calculations of total thermal conductivity and one have to find it value somewhere, the porosity of the film is taken from kMC simulation of the film growth. The thermal conductivity of a film with pores is calculated by using Landauer relation based on effective medium theory (R. Landauer, The electrical resistance of binary metallic mixtures, J. Appl. Phys. 23, 779-784, 1952).

$$k = 1/4 \left[k_p(3n_p - 1) + k_0(2 - 3n_p) + \left\{ [k_p(3n_p - 1) + k_0(2 - 3n_p)]^2 + 8k_0k_p \right\}^{1/2} \right], (1)$$

where n_p and k_p are porosity of the film and thermal conductivity of material which fills the pores, normally it equals to 0. Interesting, that the relation is the same as Bruggeman's relation for effective dielectric constant. If porosity is not constant across the layer thickness, then the

layer is divided into sub-layers so that the change in the porosity in the sub-layer is less than 10% and the thermal conductivity is calculated for each sub-layer. The total thermal conductivity then calculated as follow:

$$\frac{1}{k} = \sum_{i=1}^N \frac{l_i/l}{k_i}, \quad (2)$$

where k_i is a thermal conductivity of a sub-layer “ i ”, l_i is a thickness of the sub-layer “ i ”, l is the total thickness of the layer, and N is the number of sub-layers. This equation has simple physical sense, in case of a multiple layers the total thermal resistivity of the film, $1/k$, is a sum of thermal resistivities of sub-layers.

12.1.3 Calculation of thermal conductivity of a film consisted of a multiple layers

For the case of a multi-layer film one can use a simple generalization of Equation (2) to calculate the total thermal conductivity of the whole stack of individual layers. In this case, it is also necessary taking into account thermal resistivity of the interfaces between the layers. So the equation for the total thermal conductivity has the form:

$$\frac{1}{k} = \sum_{i=1}^N \frac{l_i/l}{k_i} + \sum_{j=1}^{N-1} r_j, \quad (3)$$

where r_j are thermal resistivities of the interfaces.

12.2 NASCAM GUI example

12.2.1 Input parameters

One can access input parameters for the plug-in by clicking on Tools → Thermal_Conductivity, see Figure 59. The input parameters are:

- Number of layers, should be the same as number of deposition stages (in case of multi-layer deposition).
- Materials for each layer should be setup manually in accordance to the deposited material at each stages.
- Values for thermal conductivities should be setup manually.

- Thermal load.
- Temperature at the film surface.

One can increase an effective thickness of the coating by changing parameter "distance between 2 mono-layers". If one changes the value from the default value 3.0 to 30.0, then each atom will represent a "super particle" with a diameter of 30 Angstroms.

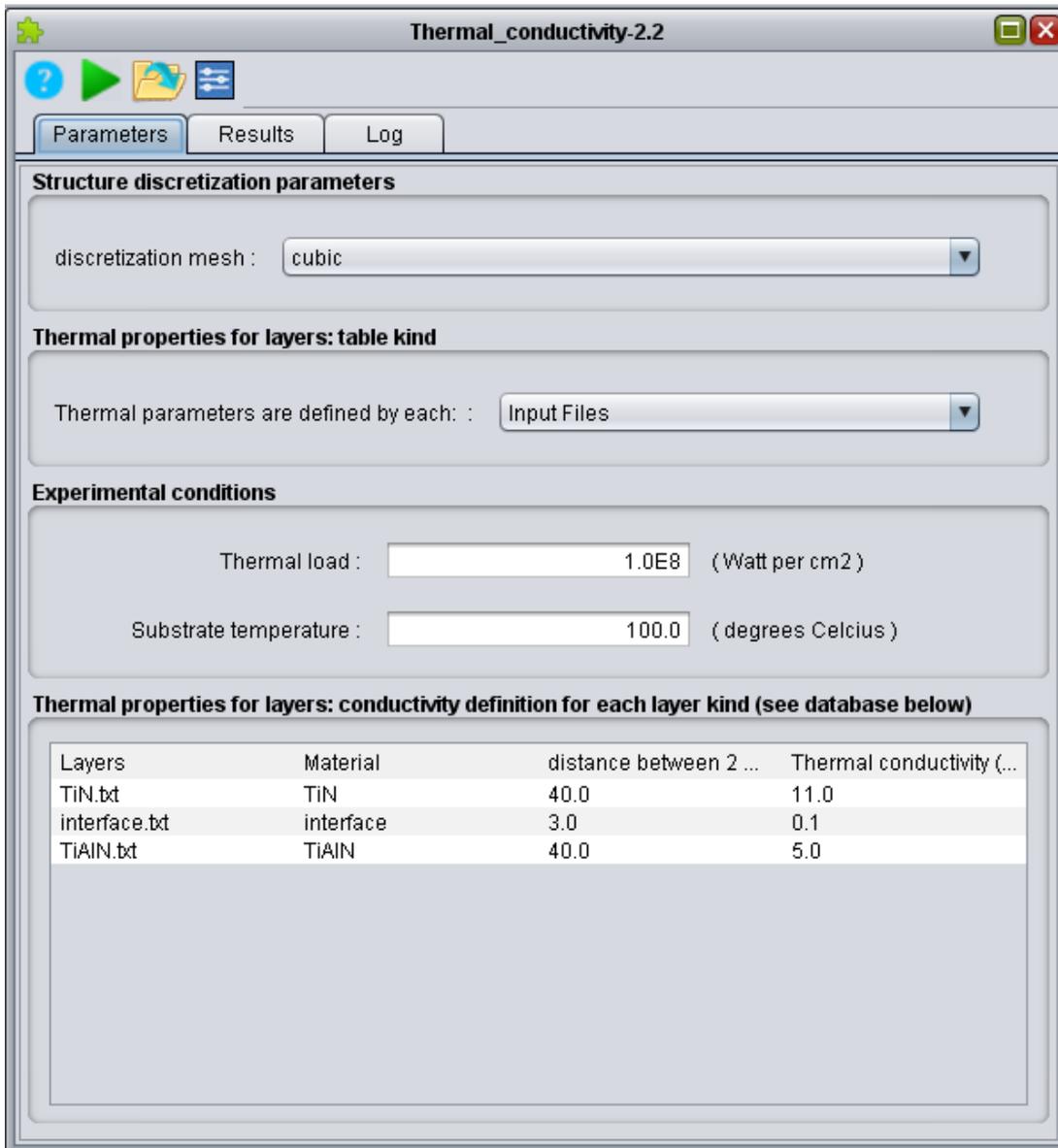


Figure 59 : Input parameters for Thermal conductivity plug-in

12.2.2 Results

For the illustration we will give examples of calculations of the thermal conductivity for multi-layer TiN/TiAlN film (experimental details see M.K. Samani et al, Thermal conductivity of titanium nitride/titanium aluminum nitride multilayer coatings deposited by lateral rotating cathode arc, Thin Solid Films 578 (2015) 133–138). Experimental results are given in Figure 60.

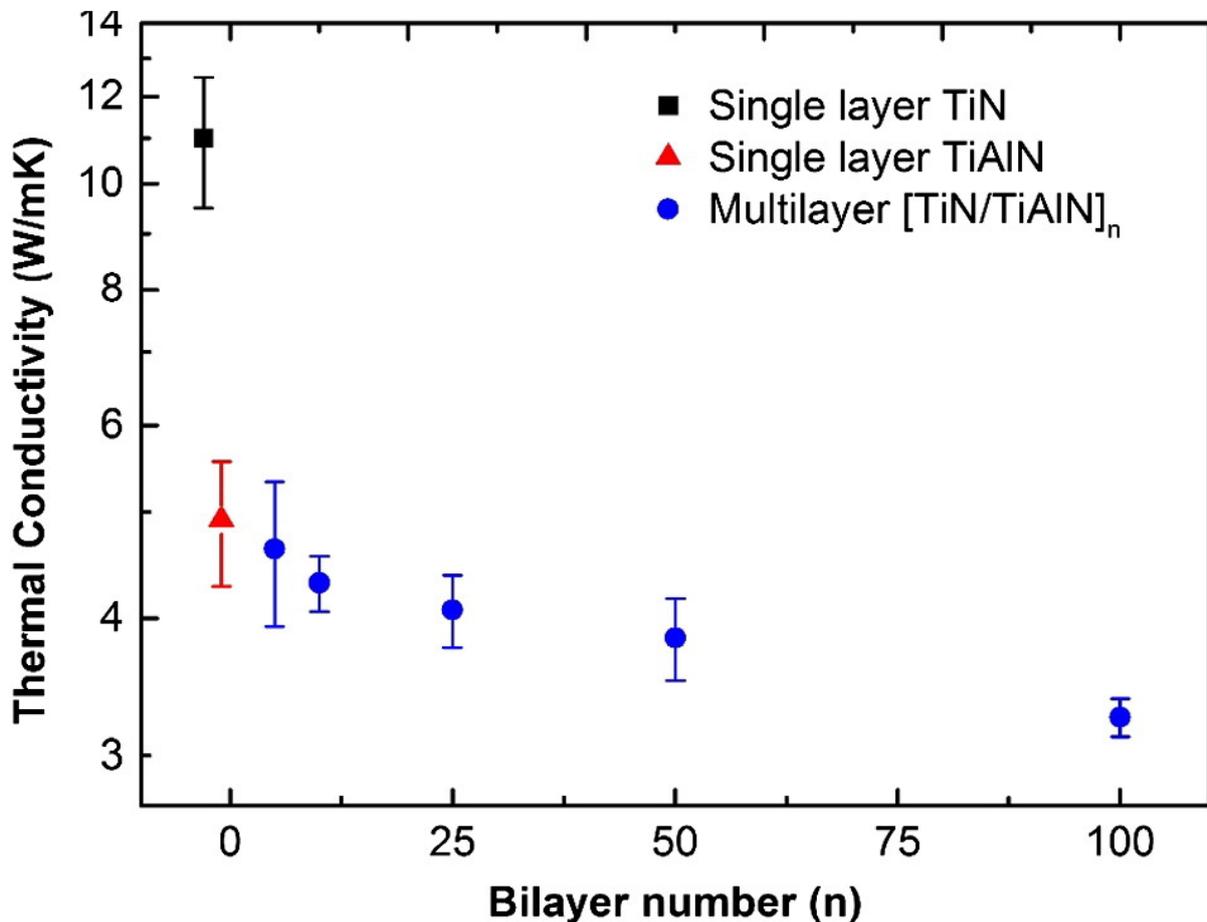


Figure 60 : Thermal conductivity of the as-deposited TiN, TiAlN single layers and [TiN/TiAlN]_n multilayer coatings.

A case with number of bilayers n=5 was chosen for the example. Firstly, it is necessary to setup NASCAM simulation for cases of multilayer deposition. To simulate such a film one has to

simulate deposition $N = 4 \cdot n - 1$ single layers. This number is easy to understand if one takes into account that the structure of the film is as follow:

TiN/Interface/TiAlN/interface/
 TiN/interface/TiAlN/interface/
 .../
 TiN/interface/TiAlN/interface/
 TiN/interface/TiAlN/

Therefore, for each bilayer it is necessary to simulate 4 single layers, except for the last bilayer, where there is no last interface layer. Figure 61 illustrates the simulation setup and results of the simulations.

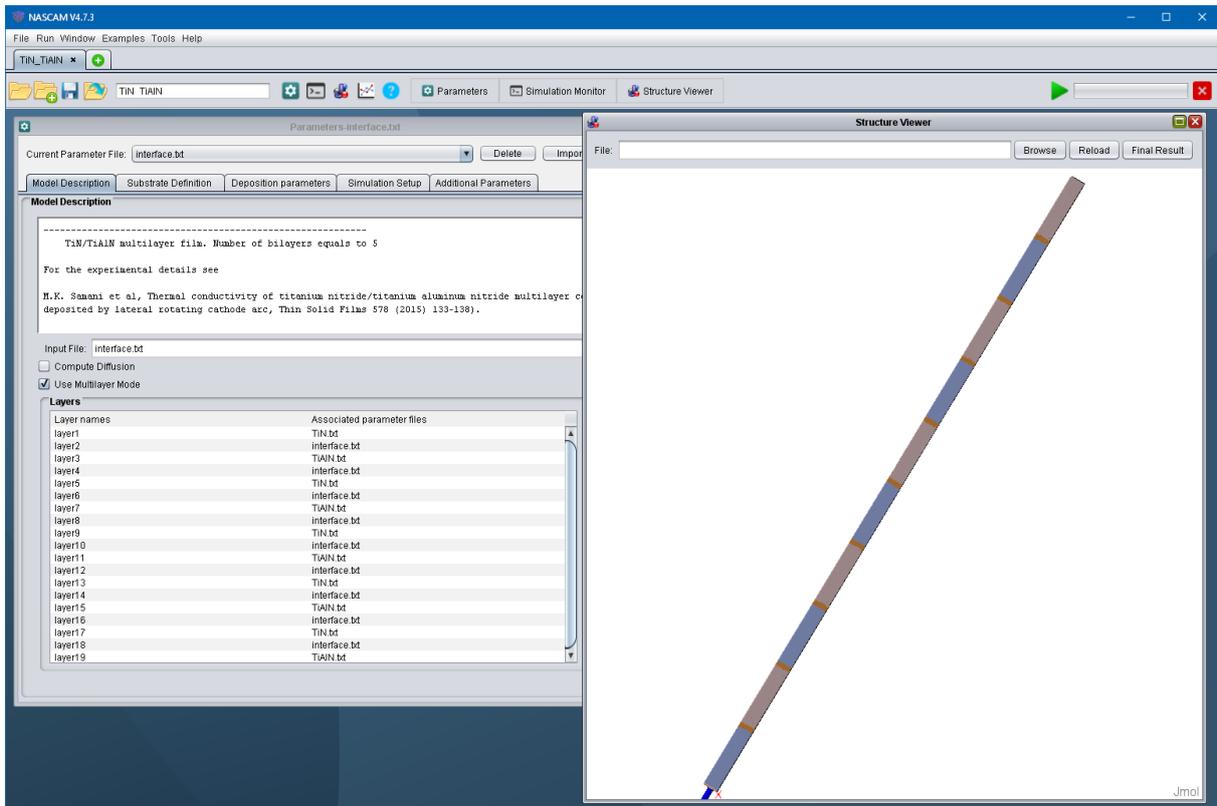


Figure 61 : Simulation of deposition of 5 bilayers of TiN/TiAlN.

On completing film growth simulation one can start calculation of the thermal conductivity of the film.

To start the calculation it is necessary to set up thermal conductivity of each layer and thickness of each layer. In addition, to get a temperature profile it is necessary to set the thermal load and substrate temperature.

Although there are 19 deposited layers in the example it is not necessary to set the data for each layer as the data for all TiN layers, TiAlN layers and interface layers are the same. For this reason, one should set up the data only three times, for each kind of deposited layers. Obviously, there are 3 different kind of deposited layers in the example, that's why there are only three lines in the input table for the thermal properties of the materials.

Setting up of the layer thickness should be done as follow. The idea is to scale up the thickness of a simulated layer to the real thickness of a sample. The scaling is a necessary step as usually the simulation are not capable to reproduce experiment 1:1. Therefore, it is necessary to find the ratio of experimental thickness in nm to simulated thickness, which is expressed in number of deposited monolayers. This value should be used as “distance between two monolayers” for the corresponding simulated deposited layer.

12.2.3 Temperature profile across the coating.

The results of the calculation of the thermal conductivity of the coating one can get by clicking the *Results* tab of the plug-in window, the tab presents the temperature profile across the coating, see Figure 62.

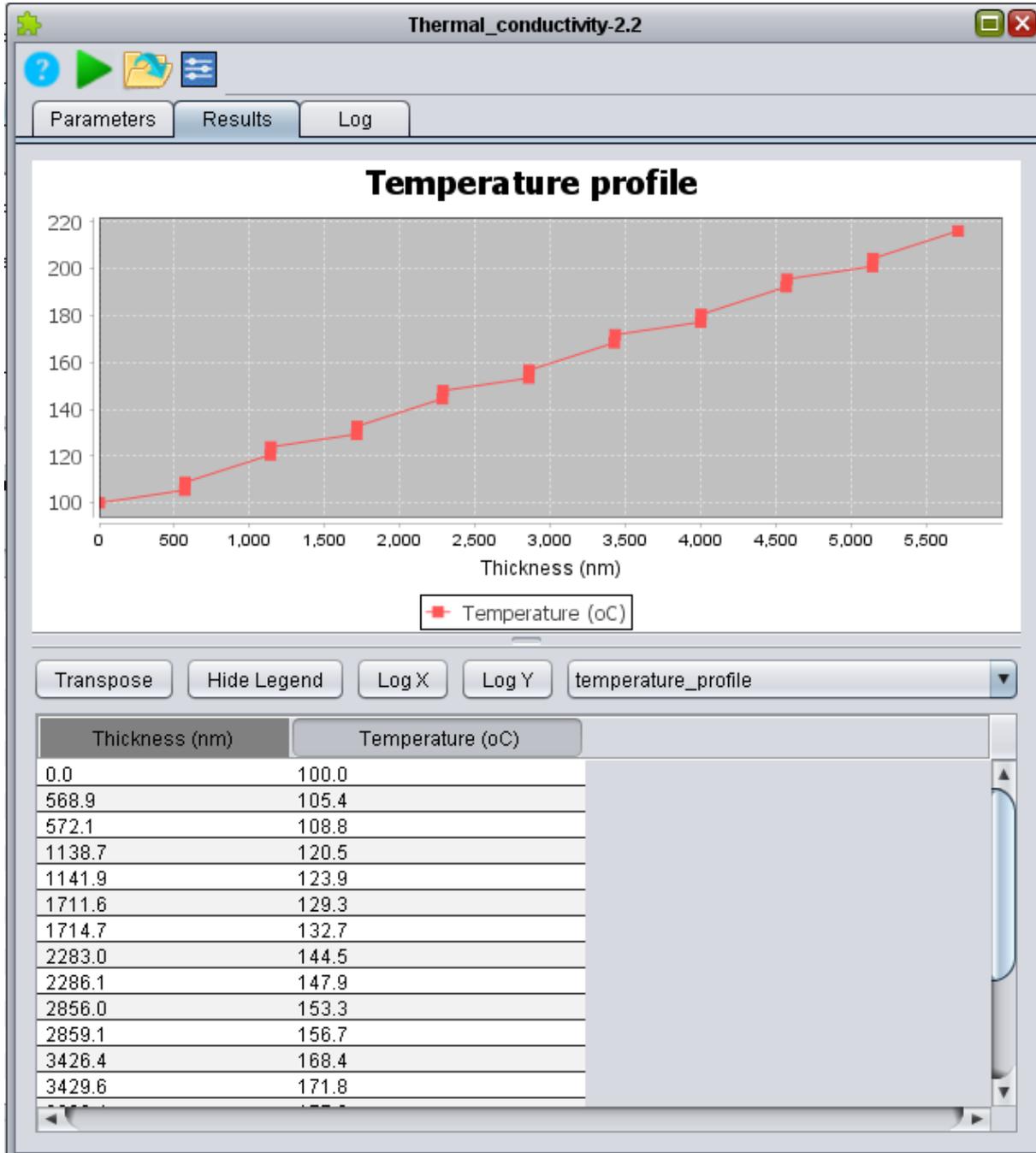


Figure 62 : Temperature profile across the coating calculated by the plugin.

Equation 3 may be used to estimate thermal resistance of the interfaces. The value of the thermal resistance may be evaluated to get the best fit to the experimental data. This procedure gives $r = 3.66 \cdot 10^{-4} \text{ m} \cdot \text{K} / \text{W}$. It has to be noted that it is the thermal resistance of the interfaces that

results in the decrease of the thermal conductivity of the film with the increase of the number of bi-layers TiN/TiAlN. In addition, we have to mention the best agreement to the experimental data was obtained when the values of thermal conductivities of TiN and TiAlN monolayers were taken about 1.5 times less than they were measured, see Figure 63. The correction to the experimental values required additional considerations.

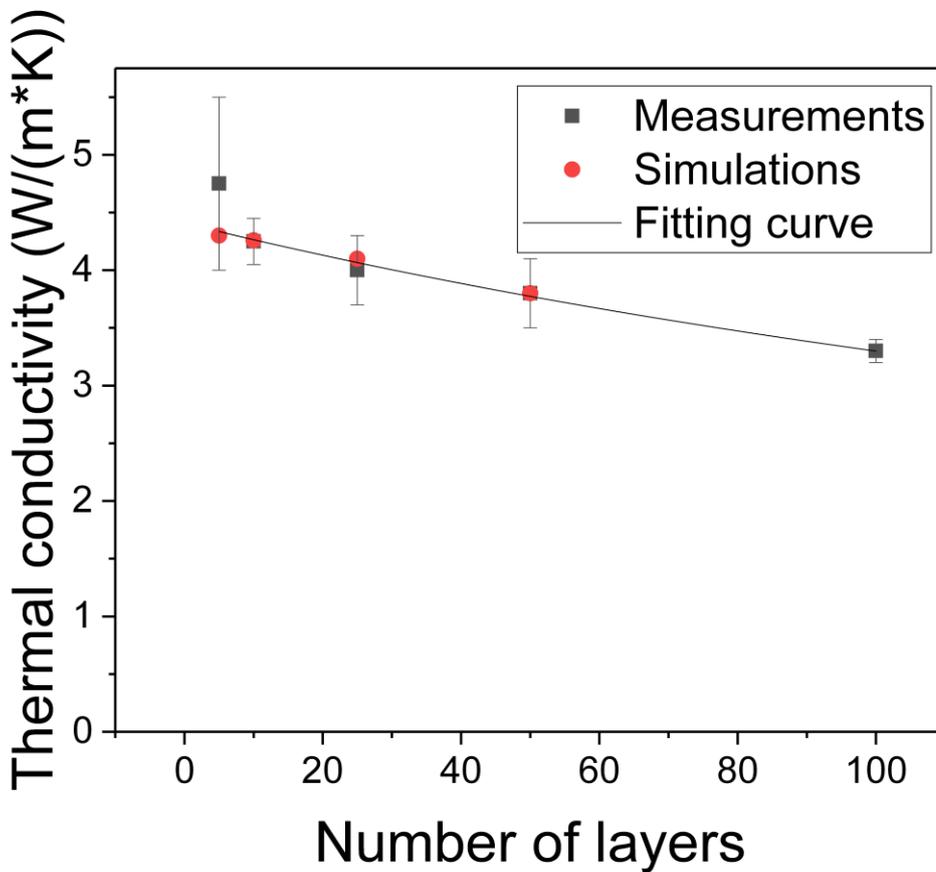


Figure 63 : Experimental values of thermal conductivity of TiN/TiAlN multi layer film and values simulated by Thermal plugin and calculated by Eq. 3

13 References

- Bosch, S. (2000). Effective dielectric function of mixtures of three or more materials: a numerical procedure for computations. *Surface Science*, 453(1-3), 9-17. doi:10.1016/S0039-6028(00)00354-X
- Garboczi, E. (1998, 12 01). Finite Element and Finite Difference Programs for Computing the Linear Electric and Elastic Properties of Digital Images of Random Materials. *NISTIR*. Retrieved from http://ws680.nist.gov/publication/get_pdf.cfm?pub_id=860168
- Heavens, O. S. (1955). *Optical Properties of Thin Films*. London: Butterworth.
- Luersen, M. (2004). A constrained, globalized, and bounded Nelder–Mead method for engineering optimization. *Structural and Multidisciplinary Optimization*, 27(1), 43-54. doi:10.1007/s00158-003-0320-9
- Scocchi, G. (2013). Evaluation of a simple finite element method for the calculation of effective electrical conductivity of compression moulded polymer–graphite composites. (elsevier, Ed.) *Composites: Part A*(48), pp. 15-25. doi:<http://dx.doi.org/10.1016/j.compositesa.2012.12.013>
- Troparevsky, M. C. (2010). Transfer-matrix formalism for the calculation of optical response in multilayer systems: from coherent to incoherent interference. *18*(24), pp. 24715-24721. doi:<https://doi.org/10.1364/OE.18.024715>
- Vedam, K. (1994). *Optical Characterization of Real Surfaces and Films* (Vol. 19). Pennsylvania: Physics of thin films - Advances in research and development.