Pyrough: A tool to build 3D samples with rough surfaces for atomistic and finite-element simulations

Hugo Iteney, Javier Antonio Gonzalez Joa, Christophe Le Bourlot, Thomas W Cornelius, Olivier Thomas, Jonathan Amodeo

H. Iteney, J.A.Gonzalez Joa, C.L. Bourlot, T.W. Cornelius, O. Thomas, J. Amodeo, Pyrough: A tool to build 3D samples with rough surfaces for atomistic and finite-element simulations, Comput. Phys. Commun. (2023) 108958. https://doi.org/10.1016/j.cpc.2023.108958.

Pyrough: a tool to build 3D samples with rough surfaces for atomistic and finite-element simulations

Hugo Iteney, Javier Antonio Gonzalez Joa, Christophe Le Bourlot, Thomas W. Cornelius, Olivier Thomas, Jonathan Amodeo

Aix-Marseille Université, Université de Toulon, CNRS, IM2NP, 13013 Marseille, France

Université de Lyon, INSA-Lyon, MATEIS, UMR 5510 CNRS, 69621 Villeurbanne, France

Abstract

Natural samples are characterized by surface roughness which is intrinsically multi-scale as depicted by the well known concept of fractal dimension. Nevertheless, surface asperities are barely taken into account in simulations and modelling where flat surfaces and sharp corners or edges are generally preferred for the sake of simplicity. In this context, we propose here a versatile Python program called *Pyrough* that aims at building virtual samples characterized by configurable surface roughness for numerical applications such as atomistic and finite-element simulations. The program is open source and relies on the classical roughness theory that integrates the concept of self-affine surface. Several basic shapes including basic blocks, spheres, grains and wires with self-affine surface asperities are implemented and the object-oriented structure of the program simplifies the implementation of more complex objects. Virtual sample design is improved using Pyrough, which enables more realistic simulations to be made. Several application examples including *e.g.,* the design of wavy grain boundaries or nanoindentation testing using a roughened indenter tip are presented.

Keywords: Surface roughness, Modeling, Atomistic simulation, Finite-elements modeling, Computer programming

PROGRAM SUMMARY

Email address: jonathan.amodeo@cnrs.fr (Jonathan Amodeo)

Program title: Pyrough
Developer's repository link: `https://github.com/jamodeo12/Pyrough`
Licensing provisions: GNU General Public License 3 (GPL)
Programming language: Python
External routines/libraries: Gmsh, Meshio, Wulffpack, ASE, Atomsk, cv2
Nature of problem : 3D virtual samples used for atomistic or finite-element simulations generally rely on simplified geometries and surfaces for the sake of design simplicity. However, the influence of surface roughness play a crucial role in various fields of applications (*e.g.,* mechanics, catalysis, lubrication) and must be taken into account.
Solution method : *Pyrough* allows for the design of 3D virtual objects with rough surfaces by means of the classical roughness theory. The user can easily tune the morphology of surfaces and shape 3D objects. Output samples can be used in finite-element or atomistic simulations according to the user's needs.
Running time: spans from few seconds or less to several minutes depending on the size of the sample to be processed, the precision of the mesh, and the machine performance.
Additional comments: the program documentation is available at `https://jamodeo12.github.io/Pyrough/`

1. Introduction

Surface roughness has a fundamental impact on the mechanical, physical and chemical properties of bulk materials with implications in various fields such as *e.g.*, friction [1], lubrication [2, 3], machining [4] or polishing [5]. It is also known to play a key role at small scales as in the field of catalysis where surface morphology is a key factor for chemical reaction efficiency [6, 7] or in nanomechanical engineering applications where peaks and valleys associated with surface roughness act as stress concentrators which are natural sites for dislocation[1] nucleation [8, 9, 10, 11, 12, 13]. Indeed, these studies have emphasized the crucial role of surface steps and roughness on the mechanical behavior of crystalline materials and, especially, on the dislocation nucleation process which governs the strength of materials at the nanoscale.

[1]Dislocations are linear defects responsible for the irreversible deformation of crystalline materials.

Although shape optimization has recently emerged as a first step towards a more realistic, *in silico*, sample design [14, 15, 16], several mathematical formulations including linear transformation [17], 2D filters [18] and time series methods [19] were already used in the past to consider non-equilibrium *i.e.*, *mathematical*, surface roughness. Furthermore, simple theoretical approaches based on the fractal theory and more common observations of natural surfaces were made to investigate adhesion or friction at the nanoscale shedding light on the influence of surface details in mechanical testing [20, 21, 22]. Despite these approaches, the majority of atomistic and macroscopic numerical simulations usually rely on simple sample designs (using *e.g.,* flat surfaces, sharp angles and edges) which prevents studying the influence of surface details and confirms that an user-friendly tool is missing to generate more realistic virtual samples.

In this study, we describe the implementation of a numerical and open-source tool called *Pyrough* that is developed to design virtual samples with rough surfaces for atomistic and finite-element modeling (FEM) simulations based on the classical roughness theory [20, 23]. *Pyrough* gives the user the ability to manage the surface roughness and height distribution of isotropic and Gaussian random surfaces for the generation of rough virtual objects. Unlike existing (often closed-source) solutions in the fields of tribology and contact engineering [24, 25, 26, 27], *Pyrough* can serve as a pre-processor for an easy mapping of the generated rough topography for flat and curved samples. In what follows, we first recall the main concepts of the classical roughness theory and how to numerically generate rough surfaces. Then, details on *Pyrough*'s workflow, input/output file management and the implementation of the rough surface generation for several 3D shapes are provided. Finally, examples of *Pyrough*'s applications that emphasize the role of surface roughness are presented.

2. Mathematical roughness theory

A broad-spectrum of natural surfaces such as sea-floor [28], rock surfaces [29, 30, 31], clouds [32, 33], machined or fractured surfaces [34, 35] show an intrinsically multi-scale roughness [36]. This scaling phenomenon is known as *self-affine* and differs from the self-similar fractal geometry [20] by the anisotropic character of the scaling transformation. The self-affine scaling transformation for a height distribution $h(x, y)$ can be described by

the Equation 1,

$$\lambda^H h(\frac{x}{\lambda}, \frac{y}{\lambda}) = h(x, y) \tag{1}$$

where H is the Hurst exponent that scales between 0 and 1 and characterizes the surface roughness and λ is a scalar.

The roughness characterization of a surface S relies on the description of representative spatial frequencies. The power spectral density (PSD) for a 2D area $C^{2D}(\mathbf{q})$ is used to analyze the contribution of the various spatial frequencies *i.e.*, the wave vectors $\mathbf{q} = (q_x, q_y)$ [37, 23, 38]. It is expressed as the fast Fourier transform of the auto-correlation function $<h(\mathbf{x})h(0)>$ (with $\mathbf{x} = (x, y)$) as shown in Equation 2,

$$C^{2D}(\mathbf{q}) = \frac{1}{(2\pi)^2} \int_S \langle h(\mathbf{x})h(\mathbf{0})\rangle e^{-i\mathbf{q}.\mathbf{x}} d\mathbf{x} \tag{2}$$

Here $C^{2D}(\mathbf{q})$ has the m^4 unit but note that the PSD unit relies on its own definition. Equations 1 and 2 show that the PSD has a power-law dependence on the spatial frequency in the case of self-affine scaling such as,

$$C^{2D}(\mathbf{q}) \propto \mathbf{q}^{-2-2H} \tag{3}$$

where H is related to the surface fractal dimension D_f by the relation $D_f = 3 - H$.

Rough surfaces generated by *Pyrough* rely on the sum (in the real space) of cosine functions where each term of the sum represents the contribution of a spatial frequency. When compared to reciprocal space approaches (see *e.g.*, [23]), this comes to account for the sole real part of the discrete Fourier transform. In Cartesian coordinates, the mathematical expression of spatial oscillations is given by Equation 4,

$$cos(\mathbf{q}.\mathbf{x} + \phi) = cos[2\pi(\nu_x x + \nu_y y) + \phi] \tag{4}$$

where ν_x and ν_y are the the spatial frequencies along the x and y directions respectively and ϕ is the phase.

A discrete set of spatial frequencies $\nu_x = a$ and $\nu_y = b$ (where a and b are integers) is used to rationalize the range of investigated frequencies. *A*

and B are defined as the respective high-frequency cutoffs for a and b so that $a \in [\![-A; A]\!]$ and $b \in [\![-B; B]\!]$. Thus, the shortest wavelengths are $\lambda_{x,min} = \frac{1}{A}$ and $\lambda_{y,min} = \frac{1}{B}$ along the x and y directions, respectively. a and b can be positive or negative to ensure oscillations in both directions. Thus, a rough surface $h(x, y)$ can be described by a sum of elementary waves as in Equation 5,

$$h(x,y) = \sum_{a=-A}^{A} \sum_{b=-B}^{B} \alpha(a,b) cos[2\pi(ax + by) + \phi] \tag{5}$$

where $\alpha(a, b)$ is the associated amplitude of each elementary wave.

Based on Equation 5, two more contributions are made in order to allow *Pyrough* to generate self-affine rough surfaces that are randomly perturbed. First, the phase is randomly perturbed using $\phi = U(a, b)$ where U states for a uniform distribution on an interval of length π. Also, random perturbations and self-affine aspects are implemented within $\alpha(a, b)$. In *Pyrough*, $\alpha(a, b)$ is a zero-centered Gaussian distribution defined to get a smooth but random variation in amplitudes without constraining the magnitude *i.e.*, $\alpha(a, b) = G_{a,b}(a^2 + b^2)^{-(1+H)}$ where $G_{a,b}$ is a scalar randomly extracted from a reduced centered normal distribution for each (a, b) pair and $(a^2+b^2)^{-(1+H)}$ expresses the self-affine aspect of the surface, in line with Equation 3. Finally, the construction of a randomly perturbed self-affine and periodic surface can be modeled using Equation 6 in which H allows to monitor the roughness, as illustrated in Figure 1.

$$h(x,y) = C_1 \sum_{a=-A}^{A} \sum_{b=-B}^{B} G_{a,b}(a^2 + b^2)^{-(1+H)} cos[2\pi(ax + by) + U(a,b)] \tag{6}$$

where C_1 is a normalization factor introduced to fit the surface heights to the sample dimensions.

Additional theoretical details about how to model fractal surfaces can be found in ref. [39]. Note that a similar approach based on scalar spherical harmonics (Equation 7) can be used to describe the surface of spherical-shaped objects as done to model particle systems [40, 41], brain sections [42]

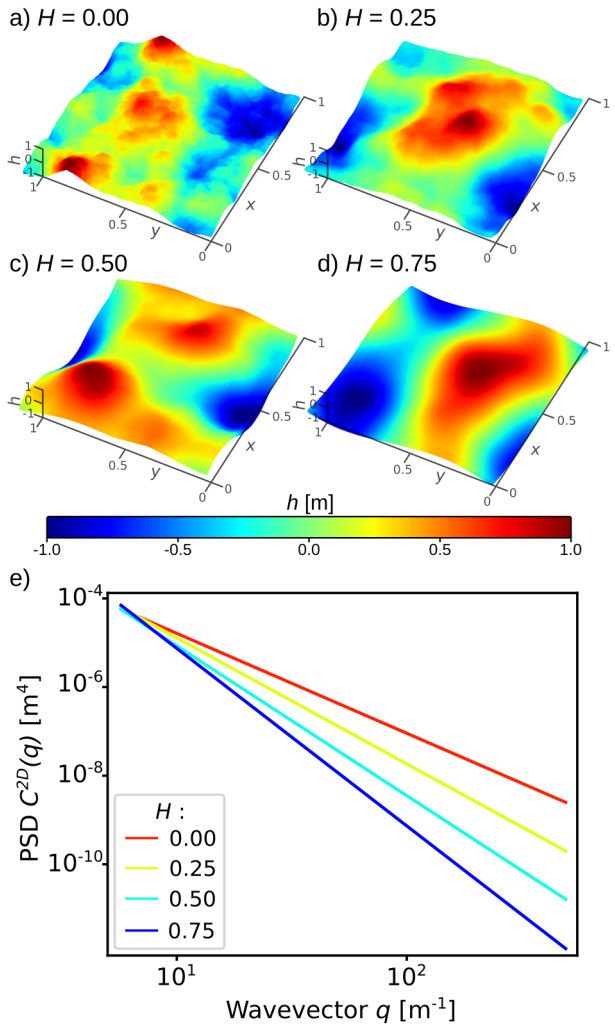

Figure 1: Random self-affine rough surfaces modeled using Equation 6 for various Hurst exponents. a) H=0.00, b) H=0.25, c) H=0.5, d) H=0.75, e) PSD of the various h profiles computed as $C^{2D}(\mathbf{q})=\sum_{x,y} h_{\mathbf{q}}(x,y)^2$ with $h_{\mathbf{q}}(x,y)$ the height matrix corresponding to the wave vector $\mathbf{q}$. For all calculations, A=50, B=50 and $h(x,y)$ values are normalized between 1 and -1.

or other 3D objects [43].

$$r(\theta,\phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} c_l^m Y_l^m(\theta,\phi) \tag{7}$$

where $r(\theta, \phi)$ is the polar radius of the sphere at coordinates $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi]$, c_l^m is the associated spherical harmonic coefficient and $Y_l^m(\theta, \phi)$ is the spherical harmonic function given by Equation 8,

$$Y_l^m(\theta, \phi) = \sqrt{\frac{2l+1}{4\pi} \frac{(m-l)!}{(m+l)!}} P_l^m(cos\theta) e^{im\phi} \tag{8}$$

where l and m are the degree and order of the associated Legendre function $P_l^m(x)$ expressed by,

$$P_l^m(x) = (1-x^2)^{\frac{|m|}{2}} \frac{d^{|m|}}{dx^{|m|}} \left[\frac{1}{2^l l!} \frac{d^l}{dx^l} (x^2-1)^l \right] \tag{9}$$

According to Equation 7, a spherical harmonic function representing the morphology of a rough sphere is characterized by its number of degrees and orders. Their amplitudes determine the intensity of the morphological characteristics that is expressed by,

$$L_l = \sqrt{\sum_{m=-l}^{l} ||c_l^m||^2} \tag{10}$$

Wei *et al.* [40] showed that the spherical harmonic descriptor D_l has a power-law dependence on the spherical harmonic degree l such as,

$$D_l = \frac{L_l}{L_0} \propto l^{-2H} \tag{11}$$

Statistical analyses as illustrated in Figure 2 allow to characterize every unique height distribution related to the random aspect of the rough surface generation process. Several statistical parameters that vary with the moment μ_ν of the height function h can be used to define the ID-card of the rough surface,

$$\mu_\nu = \frac{1}{n} \sum_{i=1}^{n} h_i^\nu \tag{12}$$

where ν states for the order of the moment, n is the number of points of the height distribution and h_i is the height value at point i.

The root mean square (RMS) or quadratic mean refers to the square root of the second moment μ_2 [44, 45]. It provides information on the variability of the heights data set and is calculated using Equation 13. Note that the standard deviation σ is equal to the RMS in the case of a zero-centered height distribution as shown in Figure 2a and b.

$$RMS = \sigma = \sqrt{\mu_2} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} h_i^2} \tag{13}$$

Statistical inputs derived from higher moments provide a more complete description of the rough surface. For example, the skewness sk allows to measure the symmetry of the statistical height distribution [46],

$$sk = \frac{\mu_3}{{\mu_2}^{\frac{3}{2}}} = \frac{1}{n\sigma^3}\sum_{i=1}^{n} h_i^3 \tag{14}$$

A symmetrical distribution with evenly distributed height peaks and valleys leads to sk tending to 0 as shown in Figure 2c. Last, the kurtosis K described by Equation 15 traduces the spiky nature of the height distribution [47, 48]. A Gaussian distribution has $K \sim 3$ whereas lower values correspond to flatter distributions. This is illustrated in Figure 2d where K decreases as the surface flattens.

$$K = \frac{\mu_4}{{\mu_2}^2} = \frac{1}{n\sigma^4}\sum_{i=1}^{n} h_i^4 \tag{15}$$

3. Pyrough implementation

3.1. General workflow

In a nutshell, *Pyrough* relies on integrating the roughness approach (Equation 6 and 7) and the statistical analysis tools presented above to design *in silico* 3D samples with rough surfaces to be used as input files for FEM or atomistic simulations. *Pyrough* is written in Python 3 and is developed under the GNU General Public License version 3 or later [49]. The code is object-oriented and is easy-to-use or improve even for non-coding users. *Pyrough* is executed using the following command line,

```
python Pyrough.py input.json
```

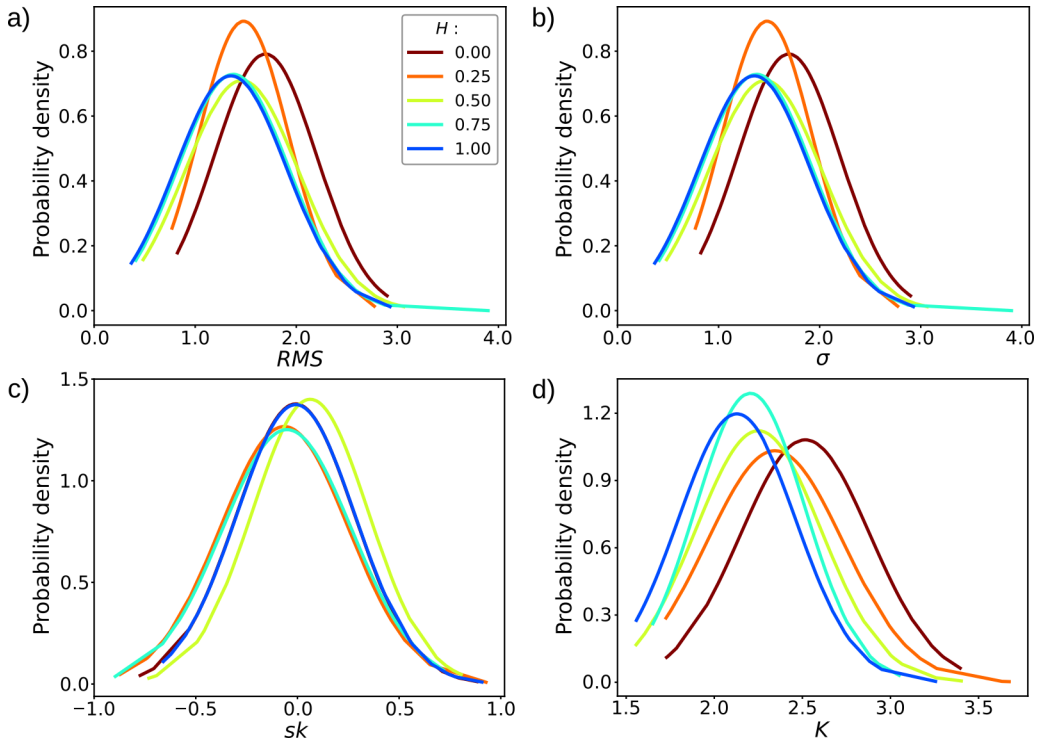

Figure 2: Statistical analyses of rough surfaces. a) Root mean square RMS, b) standard deviation σ, c) skewness sk, and d) kurtosis K. The sampling consists in a hundred rough surfaces generated for each value of H using C_1=1.

where *Pyrough.py* is the main python file that calls for the additional classes and functions existing in the *src/* folder during the process and *input.json* is the input file that informs the code about the sample geometry and roughness parameters as well as material and crystal structure. Several examples of *.json* files are provided in the *examples/* folder.

The default workflow of *Pyrough* is presented in Figure 3. It can be subdivided into four domains: *Input*, *Process*, *Output* and *External interface*. After the load of the *.json* input file, a pristine mesh is generated based on the shape of the targeted rough sample (*e.g.,* film, wire, sphere, *etc.*) using the *gmsh* library [50]. This section of the code is summarized as the *Param* class. Then, self-affine rough surfaces are generated using Equation 6 (or 7 in case of spherical samples) according to the roughness parameters contained in the input file. This is implemented in the *Sample* class. At this stage, the

full ID-card of the sample containing statistical information such as RMS, kurtosis, *etc.*, is generated as an output. The numerical rough surface is used to displace each surface node of the mesh to generate the main output *i.e.,* an *.stl* file containing the description of the rough samples (nodes, facets, *etc.*). Optional, the roughened sample mesh can be used as a mould to be filled with atoms, relying on the material lattice and crystallographic properties provided in the input file, using external interfacing with the open source program *ATOMSK* [51]. Finally, atomic positions are saved as additional output files using various formats including *.xyz*, *.lmp*, *.cfg*, *etc.* depending on user's requirements.

3.2. Input file

The *.json* input file of *Pyrough* is subdivided into four sections (see Table 1) that are processed within the *Param* class. First, the object type section informs the code on the sample shape to design. Each shape is associated to an object keyword in *Pyrough*. Currently, the available shapes are the film (keyword *Box*), cube (*Cube*), circular-based wire (*cWire*), faceted wire (*fWire*), sphere (*Sphere*) and faceted particles (*Wulff*). In addition, the user can easily implement additional shapes and associated keywords thanks to the object-oriented architecture of the code. Additional objects can be derived from aforementioned mother shapes as *e.g.*, the *Grain* object that will be described later. The second section in Table 1 describes the sample dimensions and surface roughness including mesh precision information (n_S and α) and roughness parameters such as the Hurst exponent H, the normalization factor C_1, the desired RMS or the frequency ranges N and M used to generate the rough surfaces. n_S is the characteristic size of the element before roughening and α is a scaling factor for mesh refinement in the roughened region (this will be discussed further in a later section). Additional information required for specific cases (*e.g.,* surface energies in case of Wulff-shaped systems) are also provided here. Then, the crystal structure (optional) section provides atomic-scale information on the material such as lattice parameter, crystal structure and crystalline orientation required if building the atomic structure. Finally, a last section relies on the output files generated by *Pyrough*. Associated with the FEM and ATOM parameter lists, several roughened mesh and atomic position files with various formats can be generated by the user including *.stl*, *.msh*, *.inp*, *.obj*, *.vtk*, *etc.* for FEM and *.xyz*, *.lmp*, *.cfg*, *.POSCAR*, *etc.* for atomic position files (see *gmsh*

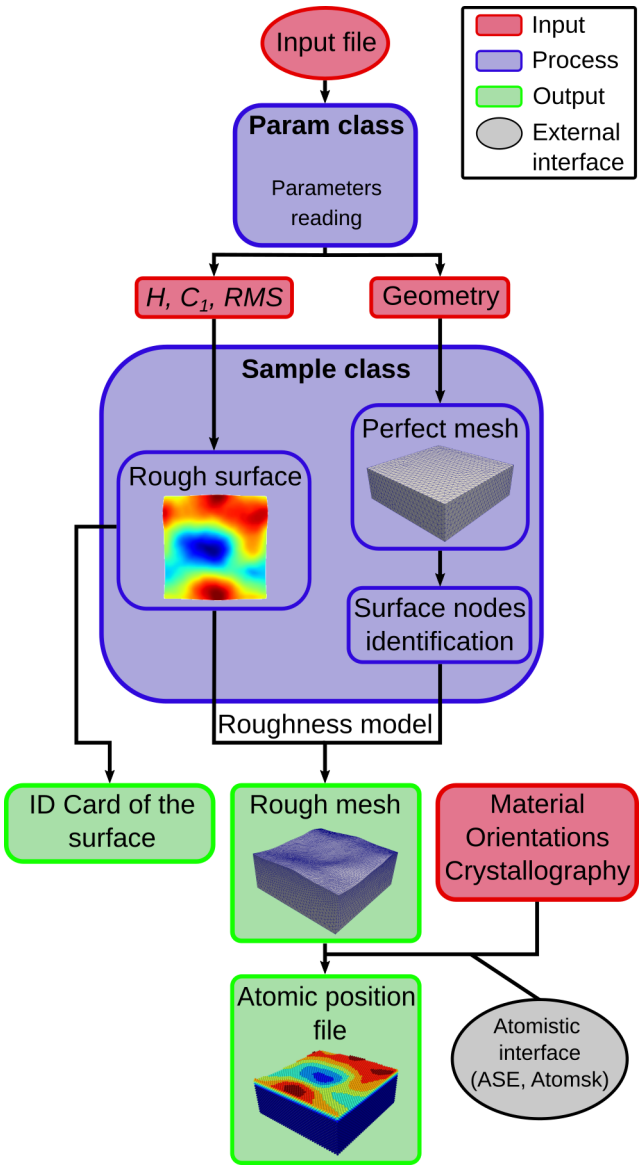

Figure 3: *Pyrough* default workflow categorized in four domains: *Input*, *Process*, *Output* and *External interface*.

and *ATOMSK* documentations for more details on output formats). Empty FEM and/or ATOM lists will not generate output files except the processed *.stl* file.

Note that it is also possible to directly provide a pristine mesh file on which

the user wishes to apply surface roughness still using the *.stl* format (particularly useful for users with their own *.stl* mesh). In this latter case, surfaces are identified based on the type of object (*Box*, *cWire*, *etc.*) the user chooses. All the inputs are read and interpreted by the *Param* class, *i.e. Pyrough* generates a param object after the input file reading in order to gather all the variables required for further processing. For instance, a simple call to the *param.width* variable is done to access the width of the sample within *Pyrough*. This class is used to distribute variables towards the different internal processes of *Pyrough* what simplifies the implementation of new sample shapes.

Table 1: Description of *Pyrough*'s input file.

Parameter	**Type**	**Description**	**Example**
Type of object	*str*	Shape of the sample	*Box, cWire, Cube, etc.*
Sample and roughness			
radius/length/width/height	*float*	Sample dimensions (nm×nm×nm)	*100, 100, 40*
H	*float*	Hurst exponent	*0.75*
RMS	*float*	Root mean square	*1.9*
C_1	*float*	Normalization factor	*1.2*
A, B	*int*	Frequency range	*30, 30*
n_S	*int*	Mesh elements size (before roughness)	*10*
α	*int*	Mesh refinement scaling factor $\in]0;1]$ *(optional)*	*0.33*
Raw stl	*str*	Initial object's mesh *(optional)*	*mymesh.stl*
fWire case :			
n_f	*int*	Number of wire facets	*5*
Wulff case :			
Surfaces	*list*	Surface list (Miller indices)	*[1,0,0], [1,1,0], [1,1,1]*
Energies	*list*	Surface energies [mJ/m^2]	*1297, 1531, 1196*
n_{At}	*int*	Number of atoms	*277389*
Crystal structure			
Material	*str*	Chemical formulation	*Au*
Lattice parameter	*float*	Lattice parameter of the crystal [Å]	*4.08*
Crystal structure	*str*	Lattice structure of the crystal	*fcc*
Orientation	*list*	Orientations along x, y and z-axis	*[1,0,0], [0,1,0], [0,0,1]*
Output parameters			
ATOM	*list*	Formats of the output atomic files	*["lmp", "xyz"]*
FEM	*list*	Formats of the output FEM files	*["stl", "msh", "inp"]*

3.3. In silico roughness design (Sample class)

After reading the input file, *Pyrough* generates a rough sample following a two-step process. Firstly, a pristine sample with dimensions imposed by the user is generated using the *Sample* class that gathers most of the processes

up to the end of *Pyrough* execution. To this aim and for the sake of computational time optimization, *Pyrough* relies on *gmsh* to generate 3D surface meshes out of triangular elements. Depending on the targeted sample shape, the appropriate function is chosen and the mesh nodes as well as corresponding facets are stored respectively in the *vertices* and *faces* variables. Below is an example of the function used for the generation of a sphere, centered at (0,0,0) with a radius of 10 nm.

```
gmsh.initialize()
gmsh.model.add("sphere")
gmsh.model.occ.addSphere(0, 0, 0, 10)
gmsh.model.occ.synchronize()
gmsh.option.setNumber("Mesh.MeshSizeMin", ns)
gmsh.option.setNumber("Mesh.MeshSizeMax", ns)
gmsh.model.mesh.generate(3)
node_tags, node_coords, _ = gmsh.model.mesh.getNodes()
element_types, element_tags, element_nodes = gmsh.model.mesh.getElements
vertices = node_coords.reshape(-1, 3)
triangle_idx = np.where(element_types == 2)[0][0]
faces = element_nodes[triangle_idx].reshape(-1, 3)-1
gmsh.finalize()
```

While a similar approach is used for several *Pyrough* shape presets, the process is slightly different for faceted systems that rely on the Wulff theory [52]. In this case, the Wulffpack library [53] is used for the generation of facets and nodes, according to the surfaces and energies parameters provided in the input file (see Table 1). Note that using *Pyrough* with the Wulff option requires ASE [54] as an additional package. Once the Wulff-shaped structure is generated, each facet is constructed using the *addLine*, *addCurveLoop* and *addPlaneSurface* functions of the *gmsh* library.

Secondly, *Pyrough* generates roughness displacing mesh surface nodes perpendicularly to the original surface. To this aim, a numerical rough surface $h(x, y)$ is generated using the roughness model (Equation 6 or 7) for each facet while surface nodes to be displaced are identified. Depending on the targeted sample shape, various protocols are used to identify surface nodes and apply roughness. In the following, we provide details for the various sample shapes already implemented in *Pyrough*.

Film

The *Box* object can be used to design films *i.e.,* 2D-periodic samples with a single rough surface. *Box* is particularly useful *e.g.,* to model the indentation of rough surfaces or perform contact friction simulations [55]. A cell of dimensions $length \times width \times height$ and mesh size n_S is first constructed. In this case, only the top surface is going to be roughened and surface nodes are originally located at the maximum height of the pristine sample mesh. A numerical grid is then generated from the surface nodes with x and y coordinates : each grid point refers to a surface node on the original mesh. Because of the periodicity of Equation 6 on the $[0, 1]$ interval, the grid is normalized and the numerical surface roughness function h is applied to its normalized coordinates, *i.e.* no repetition of the pattern appears along the x and y axis. The roughening procedure simply consists in displacing each surface node perpendicularly to the (x,y) surface (*i.e.*, along the z-axis) according to $h(x, y)$. The generation of a rough film is illustrated in Figure 4a.

Wire with circular cross-section

The second type of samples available in *Pyrough* is the *cWire* object (wire with circular cross-section) that can be useful to model rough cylinders in FEM [56] or nanowires for MD applications [57, 58, 59]. As illustrated in Figure 4b, a cylinder of radius $radius$ and length $length$ is constructed using the *extrude* function of *gmsh*, *i.e.* we first generate a circle in the xy-plane centered in (0,0) with n_S points along its circumference that we extrude along the z-axis with a regular mesh. This allows to consider vertices at the wire surface to be those positioned on the external radius r of the cylinder. In this case, the idea is to roll a rough surface around the wire in order to apply roughness. To do so, a numerical grid is generated on the $[0, 1]$ interval in the same way as in the case of a *Box* object except that the number of points along the x-axis is equal to the number of mesh points dividing the circular base and the number of points along the y-axis corresponds to the number of layers along the length of the wire. The numerical surface roughness function h is then applied to the numerical grid normalized on $[0, 1]$ interval for the sake of periodicity. Thus, each surface node from the wire mesh is moved along the r-axis by its corresponding displacement vector on the generated rough surface.

Faceted-wire

The *fWire* object allows for the design of a second type of wire made of faceted surfaces which is commonly observed at the nanoscale [60, 61, 62]. In this case, the initial regular mesh of length *length* along the z direction is constructed with the *extrude* function of *gmsh*. But in this case the base is formed of a (0,0) centered regular polygon with n_f sides inscribed in a circle of radius *radius* in the xy-plane. Then the same rolling approach as for a wire with circular cross-section is used to build the rough wire except that the surface nodes are here translated along the outward facet normal $\vec{n}$ they belong to. An additional step is required to identify nodes on surface edges. These latter are displaced along $\vec{n}$=$\vec{n_1}$+$\vec{n_2}$ where $\vec{n_1}$ and $\vec{n_2}$ are the normal directions of the two facets the node belongs to. This process is described in Figure 4c.

Wulff shapes and cube

As described in Figure 5, fully-rough 3D cubes and Wulff shapes can also be designed using *Pyrough* in accordance with equilibrium shapes observed experimentally [63, 64]. In these cases, one rough surface per facet is generated as illustrated in Figure 5. A specific process for the identification of each facet nodes is used in the *node_surface* function of the *src/Func_pyrough.py* file. Here, a node M is labelled as belonging to a given surface described by points A,B and C if $det(\vec{AB}, \vec{AC}, \vec{AM}) = 0$. Once the list of nodes for each facet is generated, each flat surface is rotated around $\vec{v} = \vec{n} \wedge \vec{u_z}$ by the angle $\theta = \arccos(\frac{\vec{n}.\vec{u_z}}{\|\vec{n}\|\|\vec{u_z}\|})$ formed between the facet outward normal $\vec{n}$ and $\vec{u_z}$ to align the isolated surface with the xy-plane. Then coordinates are normalized on the $[0, 1]$ interval in line with the periodicity of Equation 6 along the x and y-axis. Finally, each node is translated along its facet normal $\vec{n}$ using $h(x, y)$ to build the rough surface. The process is repeated for each facet of the sample. Using this method, all facets are generated independently but with the same roughness parameter set (H, C_1, A, B, *etc.*). As for the faceted wires, nodes localized on surface edges and corners are treated using facets-associated averaged displacements.

Sphere

Pyrough allows for the generation of spherical objects of radius *radius*,

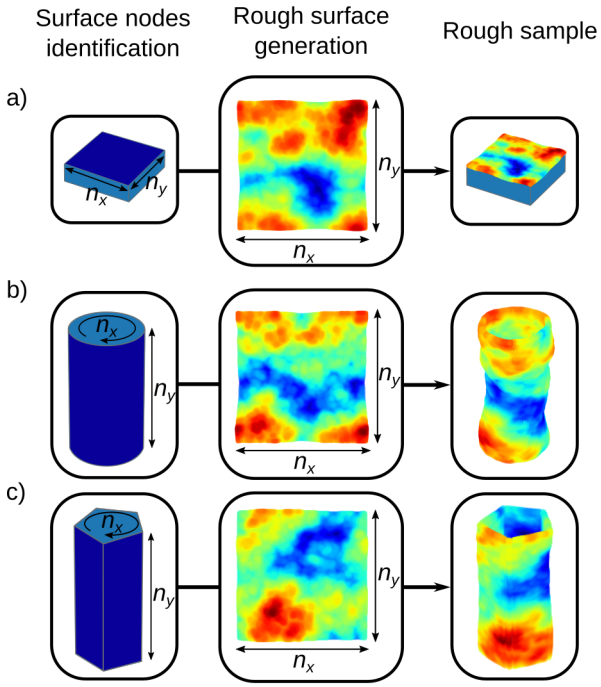

Figure 4: Construction of various rough samples, a) film, b) wire with circular cross-section and c) faceted wire. For each sample, the surface nodes are colored in dark blue and n_x and n_y correspond to the number of points along x and y-axis respectively in the original 3D mesh. A jet color map is used for the representation of height distributions, red is the maximum value whereas blue is the minimum one.

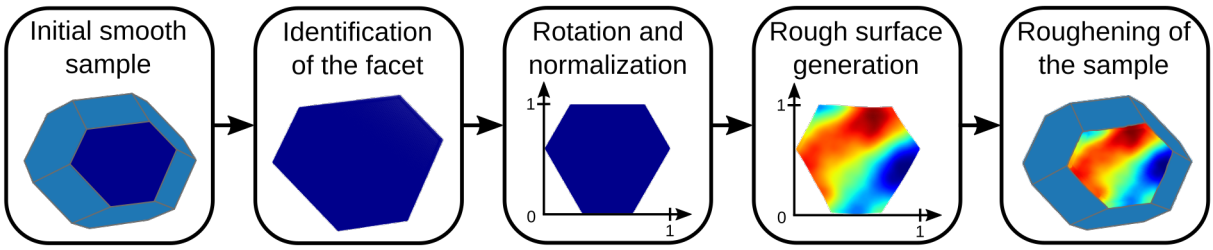

Figure 5: Construction of Wulff-faceted samples including the identification of one facet and relative nodes before the use of Equation 6. The process is repeated for each facet of the sample using the same roughness parameter set.

centered in (0,0,0), with roughened surfaces using the *addSphere* function of *gmsh*. In this case, roughness is applied to each surface node of the sphere by summing spherical harmonics based on Equation 7. A maximum har-

monic degree of 15 is used to correctly represent the sphere morphology as suggested by Wang *et al.* [65] but it can be easily modified according to the user's requirements. The self-affine aspect of sphere surfaces is integrated through the auto-correlation function l^{-2H} (Equation 11) so that the user can easily monitor the roughness degree of rough spheres.

Note that for any shape generated by *Pyrough*, the minimum value of $h(x, y)$ is added to every node-associated displacement while translations are performed along the outward normals of the sample, both in order to prevent element interpenetration issues.

3.4. Refinement and volumetric 3D mesh

Surface roughening can induce significant element distortions into the sample. Thus, mesh refinement is used to mitigate this aspect using the *gmsh* library and applying a mathematical field that maps the element size throughout the sample. *Pyrough* relies on a linear scaling of the element size that ranges from maximum n_S to minimum $\alpha.n_S$, the latter characterizing the element size right under the rough surface. Also, a specific field is defined for each *Pyrough* shape object as follow,

- Film : $n_S + n_S(\alpha - 1)\frac{z - z_{min}}{z_{max} - z_{min}}$ with z_{min} and z_{max} respectively the minimum and maximum node z-value from the film mesh.
- Wire with circular or faceted cross-section : $n_S + n_S(\alpha - 1)\frac{\sqrt{x^2+y^2}}{r_{max}}$ with r_{max} the maximum radial coordinate.
- Sphere, Wulff shape, Cube : $n_S + n_S(\alpha - 1)\frac{\sqrt{x^2+y^2+z^2}}{r_{max}}$ with r_{max} the maximum radial coordinate.

Then, a 3D volumic mesh is derived using the *gmsh* classical procedure and first-order tetrahedron elements. Examples for the mesh refinement process are shown in Figure 6

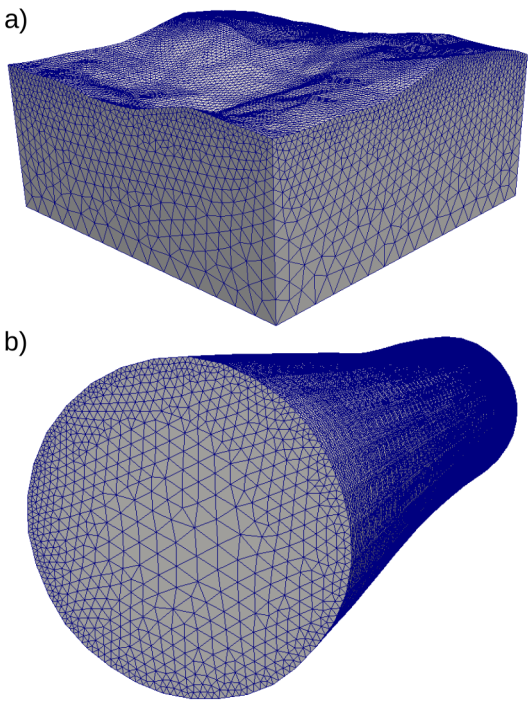

Figure 6: Pyrough refined mesh process using element size fields, (a) application to a *Box* type object where the size of the elements is reduced getting closer to the surface, (b) application to a circular wire with a radial decrease of the mesh size.

3.5. Outputs

The name of each output file generated by *Pyrough* relies on the prefix of the *.json* input file. Thus, considering the *object.json* input file, *Pyrough* systematically generates an *object.stl* file for the refined rough 3D mesh and an *object_stat.txt* file for its corresponding ID-card. In addition, all the mesh files corresponding to the FEM list variable are generated *e.g.*, *object.msh*, *object.vtk*, *etc.* Atomic positions can be generated as well using formats specified in the $ATOM$ list variable *e.g.*, *object.lmp*, *object.xyz*, *etc.* Additional examples about currently available shapes in *Pyrough* are shown in Figure 7 using a one-to-one comparison between their FEM mesh and atomistic representations.

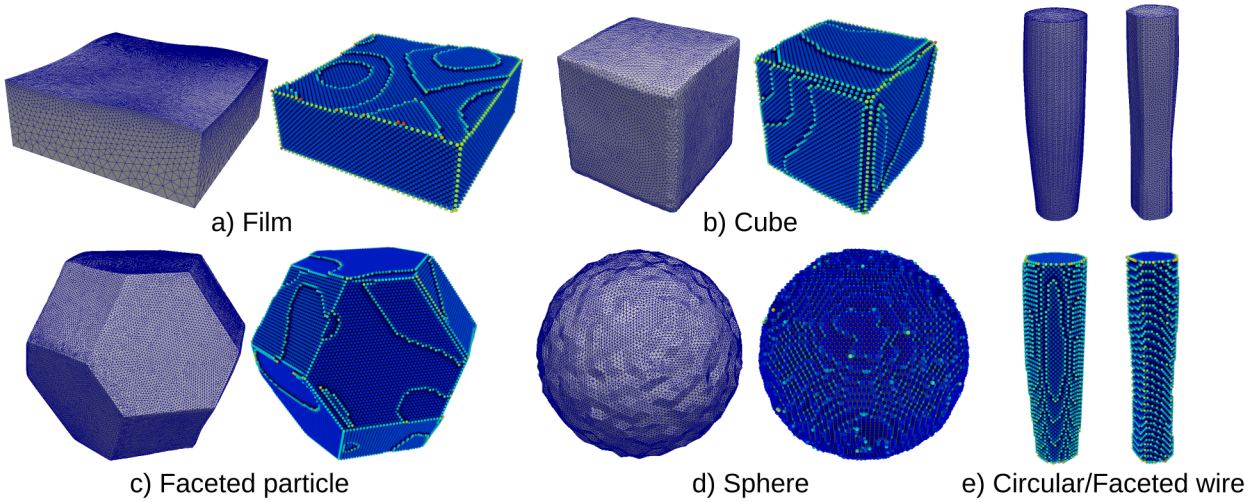

Figure 7: Examples of objects that can currently be generated using *Pyrough*. Each single shape is represented using its corresponding FEM mesh and atomistic representation.

4. Examples of application

4.1. Statistically-equivalent experimental surfaces

Pyrough can be used via a reverse processing approach to generate virtual surfaces with roughness properties similar to those measured in real-life experiments using *e.g.,* atomic force microscopy (AFM). With AFM, one can easily visualize the height distribution on a rough surface with a lateral resolution of few nanometers and the vertical one in the angström range. Figure 8 shows the 200×200 nm^2 top surface of a polystyrene particle [66] for which we mapped local heights using gray-scale analyses (*imread* function, *cv2* Python package). The resolution of the image is 800×800 pixels. Associated statistical parameters are presented in Table 2.

When analyzing a finite-length portion of an aperiodic rough surface, the abrupt truncation at the boundaries introduces high-frequency leakage that distorts the true spectral representation. To remove this effect, window functions can be applied to the surface tapering it towards zero at the window edges [67, 68]. For this specific application, we rely on the radially symmetric

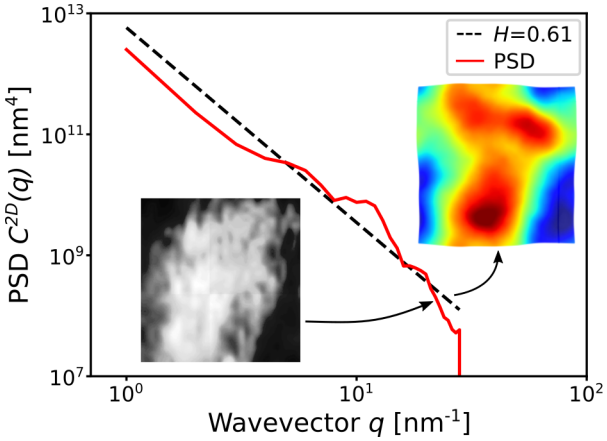

Figure 8: Comparison of PSDs $C^{2D}(q)$ as extracted from an experimental polystyrene particle surface [66] using grey-scale surface analyses (red curve) and a *Pyrough*-generated surface with measured roughness parameters (H and RMS) as inputs (black dashed curve). The Hurst exponent H is calculated from the power-law Equation 3.

Table 2: Statistical analysis of a roughened surface obtained in (i) the experiment, the analysis of the top surface of a polystyrene particle [66], and in (ii) *Pyrough*. The experimental measured parameters are used as *Pyrough* inputs to generate a surface characterized by similar statistical parameters.

Parameter	Experiment	*Pyrough*
σ	4.62	4.65
RMS	4.65	4.65
sk	-0.24	-0.20
K	2.27	2.40

Hann window as described in the work of Jacobs *et al.* [23],

$$w^{2D}_{Hann}(x,y) = \left(\frac{3\pi}{8} - \frac{2}{\pi}\right)^{-\frac{1}{2}} \times \left(1 + cos\left(\frac{2\pi\sqrt{X^2+Y^2}}{min(L_x,L_y)}\right)\right) \quad \text{for } X^2+Y^2 < \left(\frac{min(L_x,L_y)}{2}\right)^2 \tag{16}$$

where L_x and L_y are the original dimensions of the image, $X = x - L_x/2$ and $Y = y - L_y/2$.

One can note that w^{2D}_{Hann}=0 if the condition on $X^2 + Y^2$ is not fulfilled. Thus, the PSD associated to the particle surface is computed owing the pixel size as a reference for the largest $\mathbf{q}$ to compute $C^{2D}(\mathbf{q})$=$\sum_{x,y}[w^{2D}_{Hann}(x,y)h_{\mathbf{q}}(x,y)]^2$, the latter's slope allowing to derive the value of H associated with the surface (Figure 8). The H and RMS parameters as derived from the experimental surface can then be used as *Pyrough* inputs to model several numerical rough surfaces with a similar statistical description. The normalization factor C_1 in Equation 6 is here adjusted so that the RMS of the numerical surface fits with the experiment. One ends up with a random rough surface as shown in Figure 8 with a Hurst exponent and statistical distribution similar to the experimental data set (see Table 2). Such approach can be used to perform statistical analyses on a wide variety of samples with similar surface roughness properties. The protocol for the construction of rough surfaces based on experimental images outlined above is fully implemented within *Pyrough* as an extension and can be called using the following command line,

```
python Pyrough.py -surface image.png size zmin zmax
```

where *-surface* is the keyword to call for the experimental image processing, *image.png* is the imaged surface to analyze, *size* is the lateral dimension of the surface in nm and *zmin* and *zmax* are respectively the surface minimum and maximum heights in nm.

4.2. Indentation with a roughened spherical tip

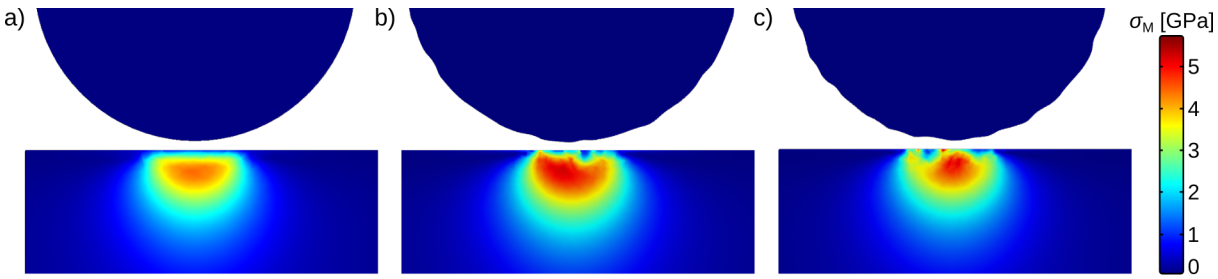

Figure 9: Von Mises stress distribution during the indentation of a gold sample with a roughened spherical tip, a) perfect sphere, b) $H = 0.75$, c) $H = 0.50$. The von Mises stress σ_M states is recorded for an indentation depth of 0.6 nm but the tip is moved back on the illustration for the sake of visibility.

When performing nanoindentation, both the sample and indenter tip surface roughness are supposed to have a severe impact on the local stress distribution in the sample subsurface [21, 69, 70]. Figure 9 illustrates how the roughness of a spherical indenter designed using *Pyrough* influences the local stress state during nanoindentation using FEM. Three simulations are performed including one with a perfectly smooth indenter and two with roughened tips characterized by $H = 0.50$ and 0.75, respectively. The FEM simulations are performed using the COMSOL Multiphysics$^{\text{TM}}$ software [24]. The indenter is modeled as a 15 nm diameter diamond sphere characterized by a Young's modulus E=1050 GPa, a Poisson's ratio $\nu = 0.1$ and a density ρ = 3515 kg.m^{-3}. The indented surface is made of gold with E = 70 GPa, ν = 0.44, ρ = 19300 kg.m^{-3} and dimensions 30×30×7.5 nm^3. The probe is loaded at the center of the sample top surface with a displacement of the indenter along the z-axis defined as $\delta l_z(t) = -\dot{\delta l} \times t$ where t is the time and $\dot{\delta l}$ is the indenter displacement rate. The displacements of the indenter along the x and y-axis are set to zero. The side and bottom surfaces of the sample are kept fixed during the simulation whereas the top surface is unconstrained but associated in a contact pair with the indenter. The contact is solved using the Penalty method with the offset null penalty function [71]. Figure 9 shows the distribution of the von Mises stress inside the gold surface after a displacement of the indenter of 0.6 nm. For the perfectly spherical indenter, the stress is homogeneously distributed due to the spherical symmetry of the contact with a maximum value of 4.6 GPa z-axis aligned and located 5.0 Å below the sample surface. In contrast, an asymmetric distribution of the von Mises stress is noticed for the two cases relying on roughened indenters. The two simulations are characterized by a maximum stress of 5.9 GPa (H=0.50, Figure 9b) and 5.7 GPa (H=0.75, Figure 9c), respectively. The impact of roughness is particularly significant here where small asperities on the indenter surface behave like super-small indenters that induce stress localization within the sample. The indenter's roughness also breaks the contact symmetry as the contact zone is not continuous anymore.

4.3. Groovy grain boundaries and interfaces

Accounting for the detailed shape of interfaces such as grain boundaries (GBs) is crucial to understand thermal [72], electrical [73], magnetic [74] or mechanical [75] properties evolution near interfaces [76, 77]. Here we rely on the *Grain* object that allows the construction of a rough interface between

two grains as shown in Figure 10. The *Grain* object is built sticking together two *Film* objects characterized by h and its opposite -h roughness functions, respectively. This process is illustrated in Figure 10 where it is applied to a coherent <001> Ag/Au interface. Note that the *Grain* object can handle grains with various kinds of materials characterized by various orientations and crystal structures.

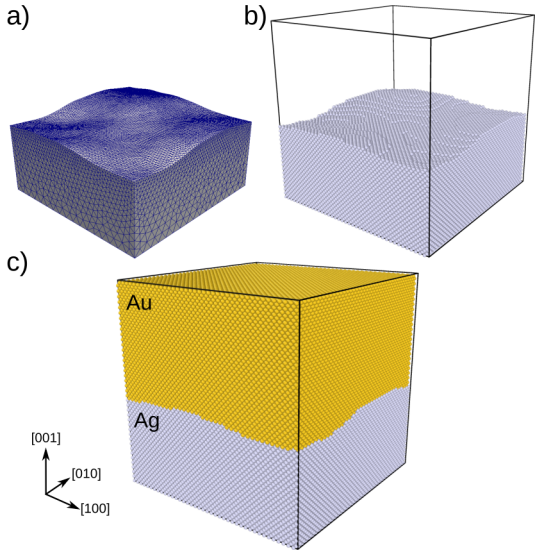

Figure 10: Generation of a wavy grain boundary using *Pyrough*, (a) single-grain FEM mesh with a rough surface characterized by h, (b) same single grain filled with Ag atoms, (c) Coupling of the complementary Au grain characterized by $-h$ leading to an Ag/Au <001> coherent interface.

4.4. Nanowire shape distribution

The mechanical properties of nanocrystals strongly depend on their shape [14, 15]. In order to generate more realistic virtual objects in MD simulations, one can apply roughness to their surfaces. Here we use *Pyrough* to generate 5000 Au nanowires with 3 nm radius pseudo-circular cross-section and 20 nm length (object $cWire$) with various H and normalization factors C_1. Three categories of cross-section geometries are defined to classify the nanowire shapes including high amplitude (HA), low amplitude (LA) and quasi-circular (QC). Rougher geometries lead to unrealistic designs, whereas HA and LA geometries have credible roughness comparable with what is observed experimentally [78, 79, 80]. The QC geometry shows small variations

of the external radius distribution with a cross-section shape that is close to the perfect circle. Cross-sections geometries rely on the RMS of the sample with an increasing value corresponding to a HA shape tendency. The ranking of the RMS depending on H and C_1 is presented in Figure 11. Based on the data set, one can find where to get the required geometry. For an identical value of C_1, the shapes are always displayed in the same order with increasing H: HA, LA and QC. It is in accordance with the Hurst exponent's definition since it controls the bandwidth of the low-pass filter applied to the surface roughening process. This confirms that *Pyrough* is adequate for the construction of different rough 3D object topologies.

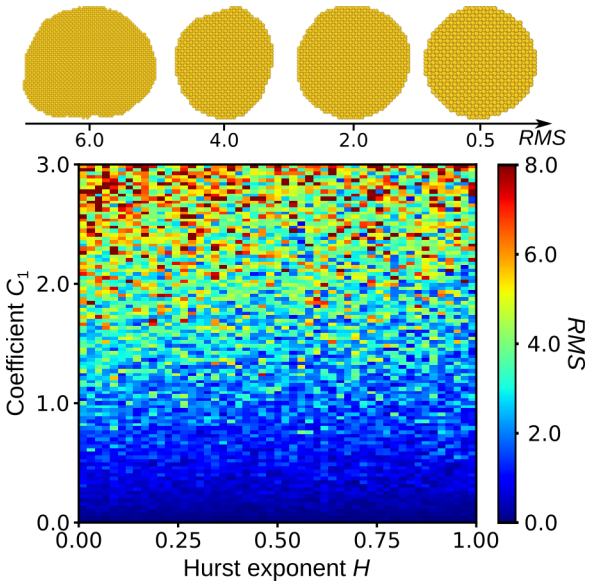

Figure 11: Nanowire cross-section geometry classification according to RMS values and RMS distribution map of Au nanowires as a function of H and C_1. In order of highest to lowest RMS: HA, LA and QC.

5. Conclusion

Pyrough is a Python tool that enables the design of 3D virtual samples with roughened surfaces for application to atomistic or FEM simulations. The object-oriented aspect of *Pyrough* makes it user-friendly and its usage consists in few simple command lines. The objects catalog of *Pyrough* allows to construct simple 3D shapes (particles, wires, GBs, *etc.*) and can easily be

upgraded by the user thanks to a free access via GitHub. The aim of *Pyrough* is to provide a better representation of 3D virtual objects when compared to the experiment by considering surface roughness and providing the user ways to manage it. The multi-scale diversity of outputs from atomistic to macroscopic FEM simulations makes *Pyrough*'s extremely useful for a large range of applications improving the accuracy of material property calculations.

6. Acknowledgements

This work was supported by the Agence National de Recherche, grant no. ANR-20-CE09-0015 (ANR SASHA) and the HPC resources of both the Fédération Lyonnaise de Modélisation et Sciences Numériques (FLMSN) and the Aix-Marseille unversity mesocentre. The authors acknowledge B. Goument and J. Izaguirre for their contributions to the project.

References

[1] Sedlaček, M., Podgornik, B., and Vižintin, J., Wear **266** (2009) 482, doi:10.1016/j.wear.2008.04.017.

[2] Jeng, Y.-R., Tribology Transactions **33** (1990) 402, doi:10.1080/10402009008981970.

[3] Menezes, P. and Kailas, S., Biosurface and Biotribology **2** (2016) 1, doi:10.1016/j.bsbt.2016.02.001.

[4] Khorasani, A. M., Yazdi, M. R. S., and Safizadeh, M. S., International Journal of Computational Materials Science and Surface Engineering **5** (2012) 68, doi:10.1504/IJCMSSE.2012.049055.

[5] Özgünaltay, G., Yazici, A. R., and Görücü, J., Journal of Oral Rehabilitation **30** (2003) 218, doi:10.1046/j.1365-2842.2003.01022.x.

[6] Lin, B. et al., Industrial & Engineering Chemistry Research **57** (2018) 9127, doi:10.1021/acs.iecr.8b02126.

[7] Gao, Y., Wang, W., Chang, S., and Huang, W., ChemCatChem **5** (2013) 3610, doi:10.1002/cctc.201300709.

[8] Brochard, S., Beauchamp, P., and Grilhé, J., Physical Review B **61** (2000) 8707, doi:10.1103/PhysRevB.61.8707.

[9] Gouldstone, A., Vliet, K. J. V., and Suresh, S., Nature **411** (2001) 656, doi:10.1038/35079687.

[10] Mitlin, D., Misra, A., Mitchell, T. E., Hirth, J. P., and Hoagland, R. G., Philosophical Magazine **85** (2005) 3379, doi:10.1080/14786430500145271.

[11] Navarro, V., de La Fuente, O. R., Mascaraque, A., and Rojo, J. M., Physical Review Letters **100** (2008) 105504, doi:10.1103/PhysRevLett.100.105504.

[12] Brochard, S., Hirel, P., Pizzagalli, L., and Godet, J., Acta Materialia **58** (2010) 4182, doi:10.1016/j.actamat.2010.04.009.

[13] Bel Haj Salah, S., Gerard, C., and Pizzagalli, L., Computational Materials Science **129** (2017) 273, doi:10.1016/j.commatsci.2016.12.033.

[14] Amodeo, J. and Lizoul, K., Materials & Design **135** (2017) 223, doi:10.1016/j.matdes.2017.09.009.

[15] Kilymis, D., Gérard, C., Amodeo, J., Waghmare, U., and Pizzagalli, L., Acta Materialia **158** (2018) 155, doi:10.1016/j.actamat.2018.07.063.

[16] Zimmerman, J., Bisht, A., Mishin, Y., and Rabkin, E., Journal of Materials Science **56** (2021) 18300, doi:10.1007/s10853-021-06435-7.

[17] Patir, N., Wear **47** (1978) 263, doi:10.1016/0043-1648(78)90157-6.

[18] Hu, Y. and Tonder, K., International Journal of Machine Tools and Manufacture **32** (1992) 83, doi:10.1016/0890-6955(92)90064-N.

[19] Watson, W., King, T., Spedding, T., and Stout, K., Wear **57** (1979) 195, doi:10.1016/0043-1648(79)90152-2.

[20] Mandelbrot, B. B., Physica Scripta **32** (1985) 257, doi:10.1088/0031-8949/32/4/001.

[21] Pastewka, L. and Robbins, M. O., Proceedings of the National Academy of Sciences **111** (2014) 3298, doi:10.1073/pnas.1320846111.

[22] Weber, B. et al., Nature Communications **9** (2018) 888, doi:10.1038/s41467-018-02981-y.

[23] Jacobs, T. D. B., Junge, T., and Pastewka, L., Surface Topography: Metrology and Properties **5** (2017) 013001, doi:10.1088/2051-672X/aa51f8.

[24] COMSOL Multiphysics^{TM}, www.comsol.com, COMSOL AB, v.5.4.

[25] Röttger, M. C. et al., Surface Topography: Metrology and Properties **10** (2022) 035032, doi:10.1088/2051-672x/ac860a.

[26] Bin, L. Y., Generating Rough surface for Abaqus using MATLAB, MATLAB Central File Exchange.

[27] Kanafi, M. M., Surface generator: artificial randomly rough surfaces, MATLAB Central File Exchange.

[28] Mareschal, J.-C., Pure and Applied Geophysics PAGEOPH **131** (1989) 197, doi:10.1007/BF00874487.

[29] Brown, S. R. and Scholz, C. H., Journal of Geophysical Research **90** (1985) 12575, doi:10.1029/JB090iB14p12575.

[30] Candela, T. et al., Journal of Geophysical Research: Solid Earth **117** (2012), doi:10.1029/2011JB009041.

[31] Wong, P.-z., Howard, J., and Lin, J.-S., Physical Review Letters **57** (1986) 637, doi:10.1103/PhysRevLett.57.637.

[32] Rys, F. S. and Waldvogel, A., Physical Review Letters **56** (1986) 784, doi:10.1103/PhysRevLett.56.784.

[33] Marshak, A., Davis, A., Wiscombe, W., and Cahalan, R., Journal of Geophysical Research **100** (1995) 26247, doi:10.1029/95JD02895.

[34] Hasegawa, M., Liu, J., Okuda, K., and Nunobiki, M., Wear **192** (1996) 40, doi:10.1016/0043-1648(95)06768-X.

[35] Mandelbrot, B. B., Passoja, D. E., and Paullay, A. J., Nature **308** (1984) 721, doi:10.1038/308721a0.

[36] Mandelbrot, B. B. and Mandelbrot, B. B., The fractal geometry of nature, volume 1, WH freeman New York, 1982.

[37] Persson, B. N. J., Albohr, O., Tartaglino, U., Volokitin, A. I., and Tosatti, E., Journal of Physics: Condensed Matter **17** (2005) R1, doi:10.1088/0953-8984/17/1/R01.

[38] Youngworth, R. N., Gallagher, B. B., and Stamper, B. L., Optical manufacturing and testing VI (2005) 58690, doi:10.1117/12.618478.

[39] Barnsley, M. F. et al., The Science of Fractal Images, Springer New York, 1988, doi:10.1007/978-1-4612-3784-6.

[40] Wei, D., Wang, J., Nie, J., and Zhou, B., Computers and Geotechnics **104** (2018) 1, doi:10.1016/j.compgeo.2018.08.002.

[41] Feinauer, J. et al., Materials Characterization **106** (2015) 123, doi:10.1016/j.matchar.2015.05.023.

[42] Ruiz de Miras, J., Martínez-Lledó, G., Orwig, W., and Sepulcre, J., Computer Physics Communications **254** (2020) 107381, doi:10.1016/j.cpc.2020.107381.

[43] Mousa, M.-H., Chaine, R., Akkouche, S., and Galin, E., 15th Pacific Conference on Computer Graphics and Applications (PG'07) (2007) 248, doi:10.1109/PG.2007.39.

[44] Bakolas, V., Wear **254** (2003) 546, doi:10.1016/S0043-1648(03)00133-9.

[45] Duparré, A. et al., Applied Optics **41** (2002) 154, doi:10.1364/AO.41.000154.

[46] Doane, D. P. and Seward, L. E., Journal of Statistics Education **19** (2011) 3, doi:10.1080/10691898.2011.11889611.

[47] DeCarlo, L. T., Psychological Methods (1997) 17, doi:10.1037/1082-989X.2.3.292.

[48] Ruppert, D., The American Statistician **41** (1987) 1, doi:10.1080/00031305.1987.10475431.

[49] GNU general public license, version 3, 2007, http://www.gnu.org/licenses/gpl.html.

[50] Geuzaine, C. and Remacle, J.-F., International Journal for Numerical Methods in Engineering **79** (2009-09-10) 1309, doi:10.1002/nme.2579.

[51] Hirel, P., Computer Physics Communications **197** (2015) 212, doi:10.1016/j.cpc.2015.07.012.

[52] Barmparis, G. D., Lodziana, Z., Lopez, N., and Remediakis, I. N., Beilstein Journal of Nanotechnology **6** (2015) 361, doi:10.3762/bjnano.6.35.

[53] Rahm, J. and Erhart, P., Journal of Open Source Software **5** (2020) 1944, doi:10.21105/joss.01944.

[54] Hjorth Larsen, A. et al., Journal of Physics: Condensed Matter **29** (2017) 273002, doi:10.1088/1361-648X/aa680e.

[55] Kalker, J. J., Dekking, F. M., and Vollebregt, E. A. H., Journal of Applied Mechanics **64** (1997) 361, doi:10.1115/1.2787315.

[56] Somadder, S. and Islam, M. S., IOP Conference Series: Materials Science and Engineering **438** (2018) 012036, doi:10.1088/1757-899X/438/1/012036.

[57] Deb Nath, S., Computational Materials Science **87** (2014) 138, doi:10.1016/j.commatsci.2014.02.013.

[58] Byggmästar, J., Granberg, F., Kuronen, A., Nordlund, K., and Henriksson, K. O. E., Journal of Applied Physics **117** (2015) 014313, doi:10.1063/1.4905314.

[59] Weinberger, C. R. and Cai, W., Journal of Materials Chemistry **22** (2012) 3277, doi:10.1039/c2jm13682a.

[60] Bernal, R. A. et al., Nano Letters **11** (2011) 548, doi:10.1021/nl103450e.

[61] Hanrath, T. and Korgel, B. A., Small **1** (2005) 717, doi:10.1002/smll.200500033.

[62] Roos, B., Kapelle, B., Richter, G., and Volkert, C. A., Applied Physics Letters **105** (2014) 201908, doi:10.1063/1.4902313.

[63] Kitayama, M. and Glaeser, A. M., Journal of the American Ceramic Society **85** (2004) 611, doi:10.1111/j.1151-2916.2002.tb00140.x.

[64] Crosby, L., Enterkin, J., Rabuffetti, F., Poeppelmeier, K., and Marks, L., Surface Science **632** (2015) L22, doi:10.1016/j.susc.2014.10.014.

[65] Wang, X., Yin, Z., Xiong, H., Su, D., and Feng, Y., International Journal for Numerical Methods in Engineering **122** (2021) 5626, doi:10.1002/nme.6766.

[66] Yamamoto, T., Fukushima, T., Kanda, Y., and Higashitani, K., Journal of Colloid and Interface Science **292** (2005) 392, doi:10.1016/j.jcis.2005.05.095.

[67] Prabhu, K. M., Window functions and their applications in signal processing, Taylor & Francis, 2014.

[68] Elson, J. M. and Bennett, J. M., **34** (1995-01-01) 201, doi:10.1364/AO.34.000201.

[69] Zhang, S., Li, D., and Liu, Y., Micromachines **13** (2022) 1907, doi:10.3390/mi13111907.

[70] Kim, T. and Olver, A., Tribology International **31** (1998) 727, doi:10.1016/S0301-679X(98)00085-1.

[71] Perić, D. and Owen, D. R. J., International Journal for Numerical Methods in Engineering **35** (1992) 1289, doi:10.1002/nme.1620350609.

[72] Merabia, S. and Termentzidis, K., Physical Review B **89** (2014) 054309, doi:10.1103/PhysRevB.89.054309.

[73] Foy, N., Chevallier, E., Zerari, H., Zehouani, D., and Favry, D., Tribology International **151** (2020) 106432, doi:10.1016/j.triboint.2020.106432.

[74] Fleischmann, C. et al., Journal of Applied Physics **107** (2010) 113907, doi:10.1063/1.3391470.

[75] Zhang, L., Lu, C., and Tieu, K., Computational Materials Science **118** (2016) 180, doi:10.1016/j.commatsci.2016.03.021.

[76] Rost, M. J., Quist, D. A., and Frenken, J. W. M., Physical Review Letters **91** (2003) 026101, doi:10.1103/PhysRevLett.91.026101.

[77] Prakash, A. et al., Acta Materialia **92** (2015) 33, doi:10.1016/j.actamat.2015.03.050.

[78] Kim, H. et al., Applied Physics A **104** (2011) 23, doi:10.1007/s00339-011-6475-0.

[79] Uesawa, N., Inasawa, S., Tsuji, Y., and Yamaguchi, Y., The Journal of Physical Chemistry C **114** (2010) 4291, doi:10.1021/jp909920d.

[80] Narayanan, S., Cheng, G., Zeng, Z., Zhu, Y., and Zhu, T., Nano Letters **15** (2015) 4037, doi:10.1021/acs.nanolett.5b01015.