J. Appl. Comput. Mech., 12(4) (2026) 1348-1358
DOI: 10.22055/jacm.2025.48554.5322

ISSN: 2383-4536
jacm.scu.ac.ir

Shahid Chamran University of Ahvaz

Journal of Applied and Computational Mechanics

Research Paper

Neper-3D-CAE: The Interface Script between Neper and Abaqus for Polycrystalline Modeling Used in Crystal Plasticity Simulation

Mehran Araghi[1], Ali Nayebi[1], Hojjatollah Rokhgireh[2]

[1] Solid Mechanics Department, Mechanical Engineering School, Shiraz University, Shiraz, Iran, Email: araghimehran@gmail.com (M.A.); nayebi@shirazu.ac.ir (A.N.)
[2] Department of Mechanical Engineering, University of Larestan, Lar, Iran, Email: rokhgireh@gmail.com

Received June 29 2025; Revised September 29 2025; Accepted for publication November 08 2025.
✉ Corresponding author: A. Nayebi (nayebi@shirazu.ac.ir)
© 2025 Published by Shahid Chamran University of Ahvaz

Abstract. Neper-3D-CAE is a Python script that seamlessly converts polycrystalline tessellations from Neper into fully partitioned Abaqus CAE models for crystal plasticity finite-element simulations (CPFEM). The tool accommodates any number of grains, arbitrary grain-face topologies, and curved external boundaries without manual intervention. Its versatility is demonstrated using representative volume elements: a cubic sample containing a central cylindrical void with 41 Voronoi grains, and microstructures comprising 10, 50, and 100 grains. Meshes produced in Abaqus CAE faithfully reproduce Neper's reference geometries, enabling straightforward stress–strain analyses under complex loading. Neper-3D-CAE bridges a critical gap in CPFEM preprocessing, streamlining microstructure-informed simulations and accelerating research in crystal plasticity and additive manufacturing.

Keywords: Crystal plasticity; Polycrystalline microstructure; Python interface; CPFEM; Voronoi tessellation.

1. Introduction

Plastic deformation modeling of engineering materials has gained increasing attention in recent decades due to growing demands for more functional products. Since plastic deformation is an irreversible, history-dependent phenomenon, an incremental formulation must be used to solve plasticity problems. In the small-deformation regime, strains can be approximated by linear terms only. However, this approximation leads to large errors in the large-deformation regime because plastic strains accumulate over time [1]. Phenomenological models are commonly used to describe general material behavior [2]. Crystal plasticity (CP) focuses on the deformation mechanisms of crystals and serves as a phenomenological model for metal plasticity. CP modeling reduces numerical errors and provides deeper insight into inelastic deformation in metals.

Generating a polycrystalline material texture is required to simulate large deformations with crystal plasticity finite-element methods (CPFEM). Tools such as Neper [3] and DREAM.3D [4] produce these textures, but their outputs are not directly compatible with commercial software like Abaqus CAE and must be converted into Abaqus-readable .inp files. Other packages—OptiMic [5], KANAPY [6], and MICROSTRUCTIPY [7]—face similar interoperability challenges.

Abaqus CAE offers advanced capabilities for creating arbitrary geometries and internal voids, applying complex boundary conditions, and leveraging a broad element library with sophisticated meshing techniques [8]. It supports CP simulations through customized user-material subroutines (UMAT/VUMAT) [9], which numerous researchers have implemented for crystal plasticity modeling [10–13]. Demir et al. [14] validated their UMAT-based CPFEM using Neper-generated textures. Rosenbusch [15] integrated the open-source FEniCSx solver into Abaqus to solve the underlying partial differential equations for CPFEM, again using Neper data.

Many CPFEM studies adopt parallelepiped geometries with cubic grains or raster tessellations [16–19], while others use polygonal grains [20, 21]. Three-dimensional microstructures for CP simulations in Abaqus CAE have been generated using Neper [22, 23]. Wang et al. [24] scripted a 36-columnar-grain microstructure in Abaqus CAE to model the mechanical behavior of 316L stainless steel under tension and compression. Fu et al. [25] examined fatigue-crack initiation in 2D CP models using Neper outputs in Abaqus CAE. Wu et al. [26] studied damage-growth mechanisms in TiAl alloys at elevated temperatures and multiple phases, again leveraging Neper textures. Similarly, Zhou et al. [27] developed a Python script in Abaqus CAE to generate polycrystalline thin-sheet geometries for ductility studies.

Although many researchers have focused on numerical methods for solving CP problems, there remains a strong need for seamless interfaces between crystallographic-texture generators (e.g., Neper [3]) and Abaqus CAE. Grilli et al. [28] addressed this by creating Neper2CAE, which imports 2D microstructures into Abaqus CAE. In the present study, we introduce a Python code that generates both 2D and 3D crystalline textures for CPFEM in Abaqus CAE directly from Neper outputs. Theoretically, this code supports arbitrary 3D shapes and, with minor modifications, can produce grains with any number of edges—enabling the modeling of more rounded grain morphologies.

Published online: November 11 2025

Table 1. Procedure steps for cell partitioning by sketch in the Abaqus CAE software for creating the polycrystalline microstructure by using of meta-data of microstructure generated by Neper.

Step	Description
Step1:	Creating geometry
Step2:	Creating datum plane for vertices
Step3:	Creating datum axis for definition of the new coordinate system
Step4:	Make sketch profile for each grain faces
Step5:	Cell partitioning by sketch planar partitioning option in Abaqus CAE
Step6:	Creating sets for each grain
Step7:	Assign grain orientations to the associated sets

Table 2. Description and the structure of the input file of Neper-3D-CAE script saved as text files.

Text file name	Description	Structure of the text file
Datum_Points	The ID number and coordinates of the texture vertices saved in rows	$[ID \quad X \quad Y \quad Z]$
Faces	The number of vertices in each face n_v, along with vertices ID's of each grain face saved in rows	$[n_v \quad ID_1 \quad ID_2 \quad \ldots \quad ID_{n_v}]$
Inner_Faces	A portion of Faces text file that contain only inner grain faces	$[n_v \quad ID_1 \quad ID_2 \quad \ldots \quad ID_{n_v}]$
Polygons	Contains grain ID p, number of faces in each grain n_f, and grain faces ID's ID_{fi}. Each row is associated to one grain	$[p \quad n_f \quad ID_{f1} \quad ID_{f2} \quad \ldots \quad ID_{fn_f}]$
Orientations	Contains the Euler angels for defining grain orientation	$[\theta \quad \varphi \quad \psi]$

2. Software Description

The developed Python program for generating polycrystalline textures leverages the classes and dictionaries defined in Abaqus CAE. It employs the sketch-based planar partitioning method to create grains within prescribed geometries. Named Neper-3D-CAE, the tool requires five input text files extracted from the tessellation output of Neper. Table 1 summarizes the cell-partitioning steps in the program, each of which uses one or more of these input files.

2.1. Input Files

Table 2 outlines the text files utilized by Neper-3D-CAE and their internal structures. Each row in these files defines a single tessellation element—vertex, face, or grain—of the microstructure. It is important to note that all of these files are directly extracted from the tessellation output generated by Neper.

The coordinates of datum points are measured in the global coordinate system. These coordinates are designated as (X,Y,Z) with unit vectors $\mathbf{E}=(\mathbf{E}_1,\mathbf{E}_2,\mathbf{E}_3)$. The vertices in the Faces and Inner_Faces text file are arranged in an ordered manner which enables the creation of the sketch profile by sketching connected face edges. The length of each row in Faces, Inner_Faces, and Polygons text files may differ with other rows. That's because each face and grain have its own number of vertices n_v and number of faces n_f, respectively. Also, p is the Polygon or grain number. In the present study, three Euler angles representation is used for defining the grain orientation with respect to the loading direction. However, it's possible to use of other formulations such as Roe and Bunge Euler angles systems [29]. $\theta, \varphi,$ and ψ are Euler angles of crystal orientations with respect to the loading direction.

3. Methodology

3.1. Definition of the grain face new coordinate system

Abaqus CAE's sketch module uses a 2D coordinate system to create sketch profiles. Because grain faces in the tessellation may assume any orientation in 3D space, a proper coordinate system called "new coordinate system" with (x,y,z) coordinates and unit vectors $\mathbf{e}=(\mathbf{e}_1,\mathbf{e}_2,\mathbf{e}_3)$ is constructed, as follows: In order to create the microstructure tessellation, it is enough to partition the geometry by using of inner faces. Starting with the entries in the Inner_Faces text file, the origin of the new coordinate system is set at the first listed vertex. The non-regularized tessellation generated by Neper consists of grains with flat faces. So, all vertices of a given face lie within the same plane. The face plane is constructed using first three vertices in the Inner_Faces text file. The normal vector of the plane is aligned with the z direction. In order to define the unit normal of this plane, $\mathbf{e}_3$, one can write:

$$\begin{aligned}\mathbf{v}_1 &= \mathbf{P}_2 - \mathbf{P}_1 \\ \mathbf{v}_2 &= \mathbf{P}_3 - \mathbf{P}_1\end{aligned} \tag{1}$$

Equations (1) define two vectors in the face plane which has been created by the first three vertices $\mathbf{P}_1$, $\mathbf{P}_2$, and $\mathbf{P}_3$. The unit normal of the plane could be calculated by the cross product of these two vectors as:

$$\mathbf{e}_3 = \frac{\mathbf{v}_1 \times \mathbf{v}_2}{\|\mathbf{v}_1 \times \mathbf{v}_2\|} \tag{2}$$

To define the in-plane coordinates (x,y), a datum axis will be created using the first and second vertices in the Inner_Faces text file. This axis will be used to define the x direction of the new coordinate system. According to the Eq. (1), this axis is in the $\mathbf{v}_1$ direction. Equation (3) defines the x direction unit normal:

$$\mathbf{e}_1 = \frac{\mathbf{v}_1}{\|\mathbf{v}_1\|} \tag{3}$$

Since orthogonal coordinate systems is used, $\mathbf{e}_2$ will be calculated by using cross product of the $\mathbf{e}_3$ and $\mathbf{e}_1$ as follows:

$$\mathbf{e}_2 = \mathbf{e}_3 \times \mathbf{e}_1 \tag{4}$$

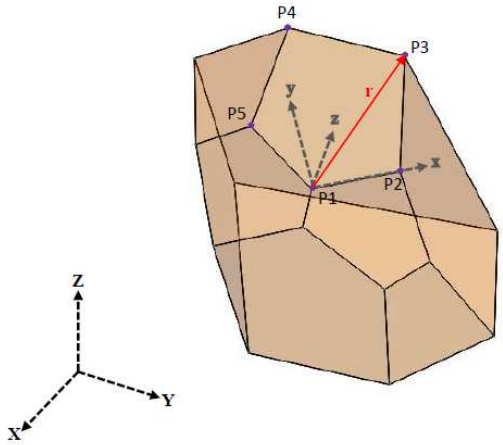


Fig. 1. The geometry of a grain and one of its new coordinate systems.

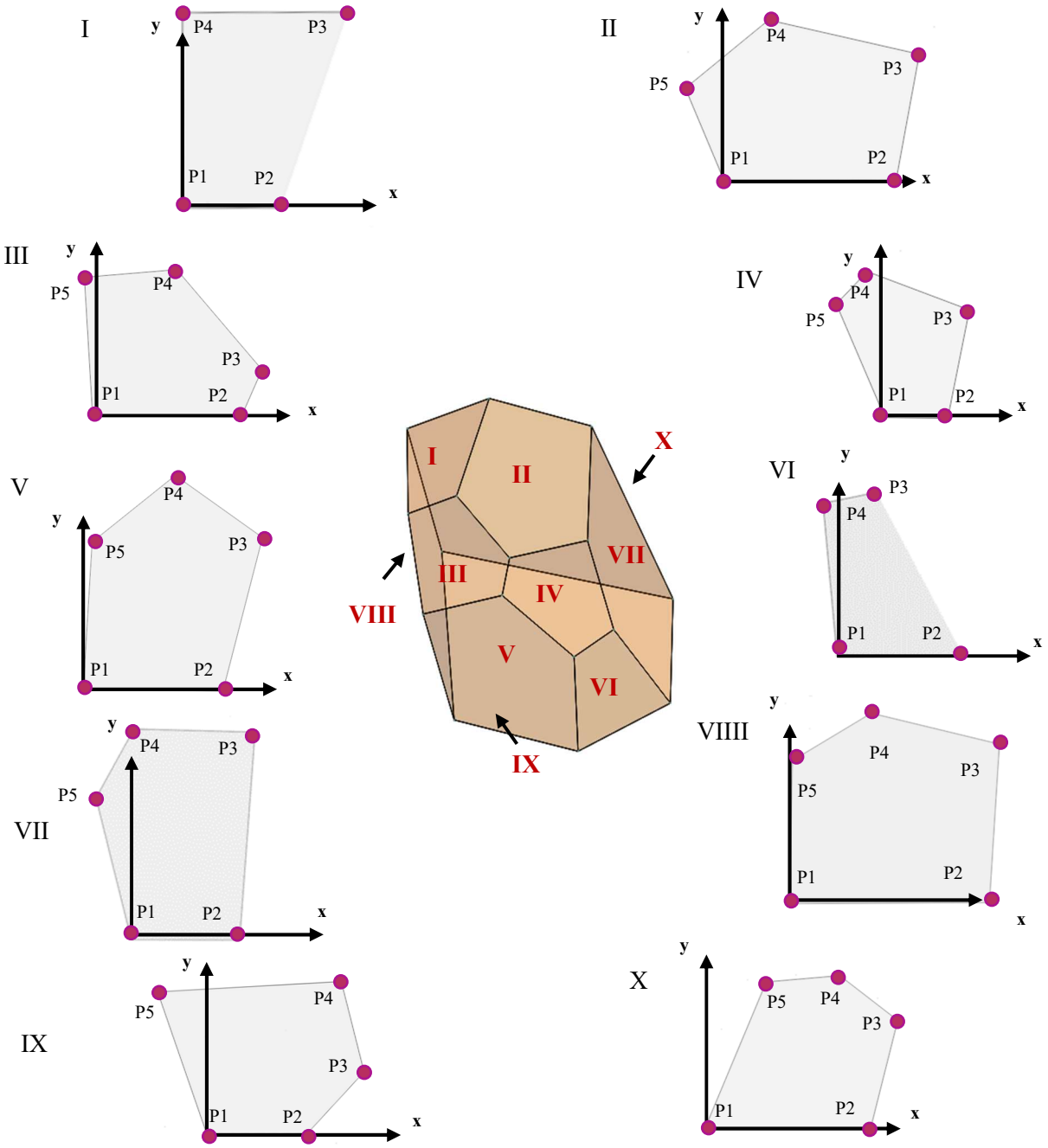


Fig. 2. The grain faces vertices coordinates in its new coordinate system.

Based on the above definitions, the local z components of all grain-edge vectors are zero. We then repeat this procedure for each row in the Inner_Faces file to establish a new coordinate system for every grain face.

3.2. Calculating position of the face vertices in new coordinate system

After creating the geometry, Abaqus CAE creates and assigns a global coordinate system to the model. All measurements in the Abaqus CAE modules are calculated in this global coordinate system. As described earlier, Abaqus CAE uses a 2D coordinate system to create sketch profiles in the sketch module. The vertex coordinates in this 2D system differ from those in the global coordinate system. Following the procedure in Section 3.1, a 2D coordinate system has been defined for each grain face. A rotation matrix **R** between the global and local coordinate systems can be defined as:

$$\mathbf{R} = \mathbf{E}_i.\mathbf{e}_j \quad \mathbf{i},\mathbf{j} = 1,2,3 \tag{5}$$

The vector that connects the origin of the new coordinate system and the k-th vertex of the grain face is represented in the new coordinate system as $\mathbf{r}$ and global coordinate system as $\mathbf{r}'$, respectively. The relation between the components of these vectors could be obtained by using of rotation matrix **R** as follows:

$$\mathbf{r} = \mathbf{R}.\mathbf{r}' \tag{6}$$

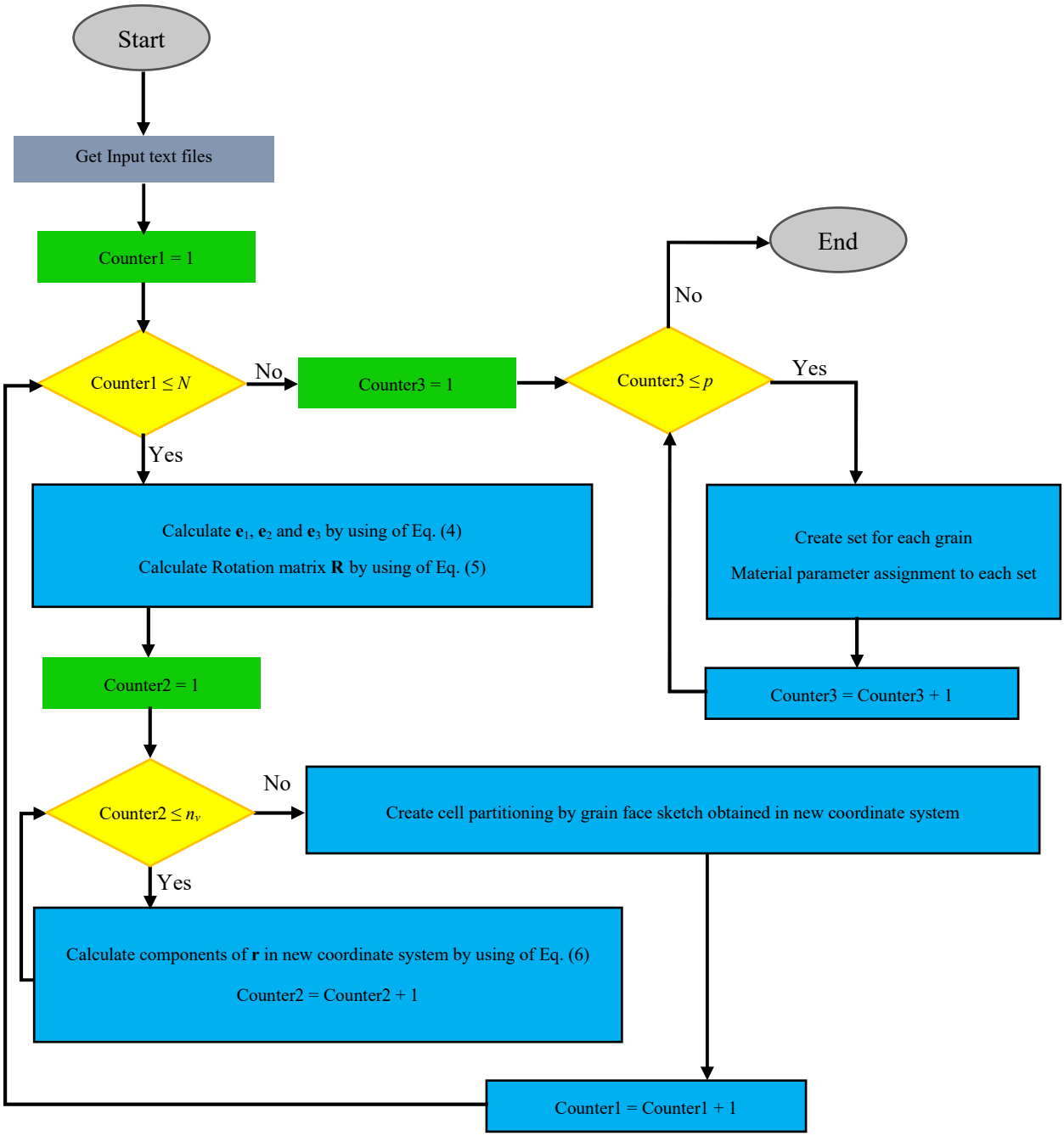


Fig. 3. Flowchart for generating texture in Abaqus by using of Neper meta-data.

By repeating this procedure for every vertex of that face, we obtain all vertex components in the new coordinate system, after which the cell-partitioning step can commence. Figure 1 shows the geometry of a typical grain and the new coordinate system defined for one of its faces. Also, Fig. 2 illustrates the coordinates of the grain faces vertices in the associated new coordinated system to each grain face.

3.3. Create cell partitioning, defining set for each cell, and material parameters assignment

In the new coordinate system, the coordinates of a grain face's vertices are calculated, and the sketch profile is then constructed by drawing lines between adjacent vertices. Repeating this procedure for all interior faces results in a fully partitioned geometry.

Defining a set for each grain requires point coordinates within the partitioned domains. Since Voronoi tessellation always produces convex grains, these coordinates can be obtained simply by averaging the vertex positions listed in the Polygons and Faces text files.

The final step in this process is the assignment of material parameters, which is implemented through multiple lines of code. The grain orientations defined in the Orientation text file are applied at this stage.

The presented flowchart for creating the microstructure tessellation is detailed in Fig. 3 where N stands for numbers of inner faces. Figure 4 illustrates the steps for generating the texture for an arbitrary geometry.

In order to demonstrate the capability of the Neper-3D-CAE workflow for generating 3D polycrystalline geometries, a cubic representative volume element (RVE) containing 41 grains and a central cylindrical void was created in Abaqus CAE (Fig. 5(a)). The RVE edge length and void radius were set to 0.1 mm and 0.05 mm, respectively. The grain morphology was generated by enforcing morphological columnar grain growth, consistent with the texture characteristics of additively manufactured samples.

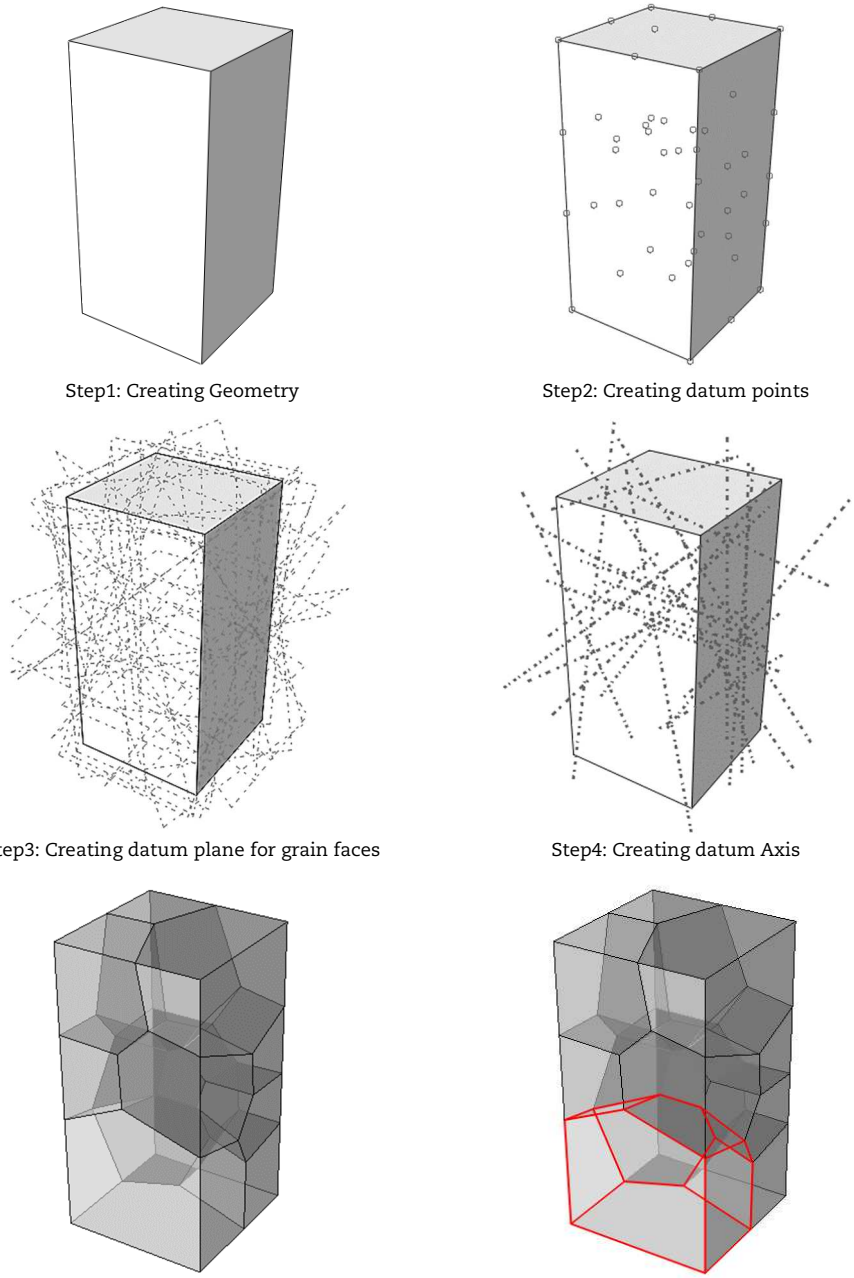


Fig. 4. Steps of creating texture in Abaqus CAE.

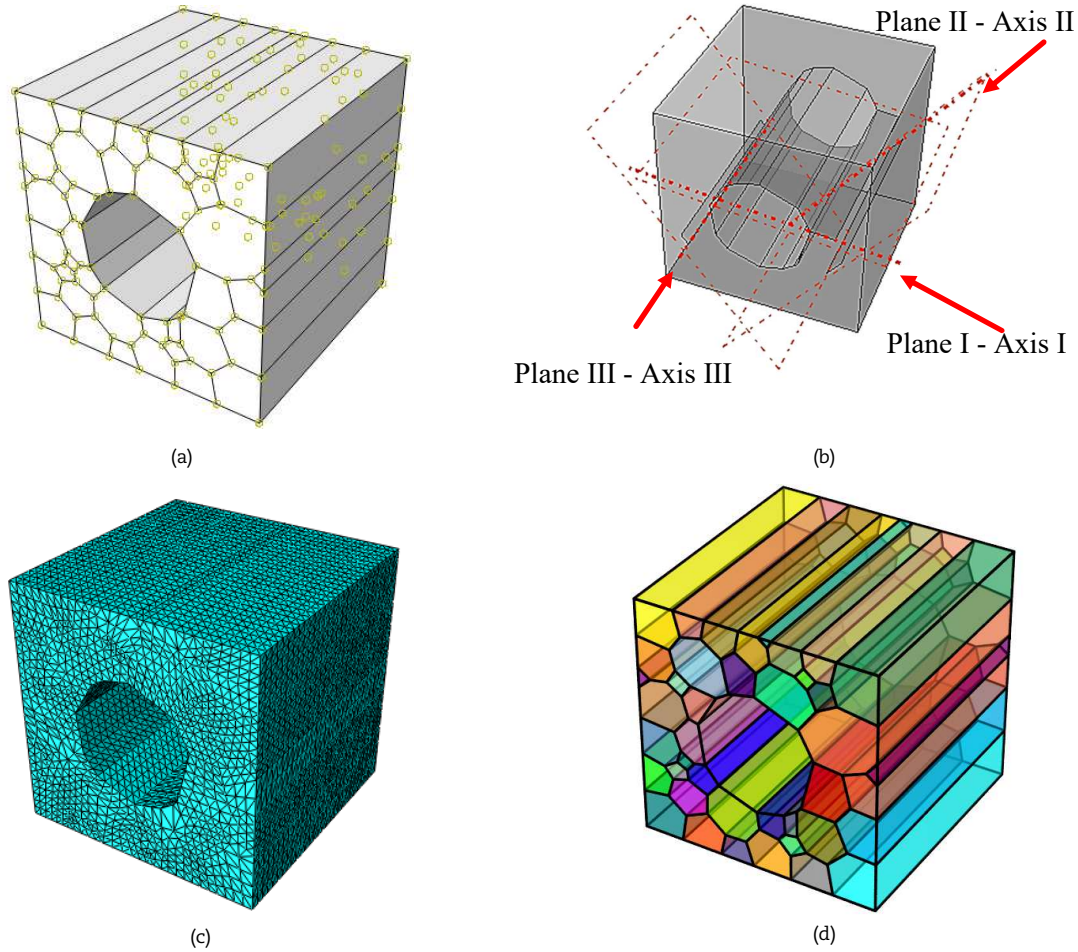


Fig. 5. (a) The generation of cubic RVE with cylindrical micro-void inside it in Abaqus CAE using Neper-3D-CAE, (b) datum planes and datum axes used for generating tessellation, (c) spatial discretization of the created model by using of tri elements in Abaqus CAE, and (d) the reference geometry created by Neper software.

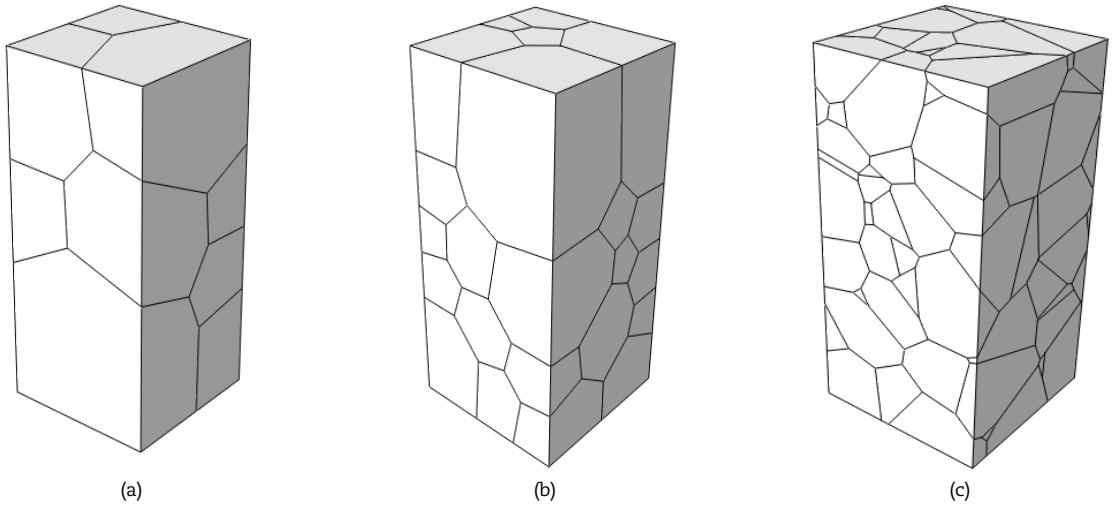


Fig. 6. A (20,40,20) cube with (a) 10 grains, (b) 50 grains, and (c) 100 grains generated in Abaqus CAE by using of Neper-3D-CAE script.

To illustrate the geometry-modeling procedure, Fig. 5(b) presents the first three datum planes and axes defined on the grain faces. The finite-element mesh produced by Abaqus CAE and the reference geometry generated by Neper are shown in Figs. 5(c) and 5(d), respectively.

The proposed method for generating microstructure geometry for CPFEM imposes no restrictions on mesh parameters. Both linear and quadratic tetrahedral and hexahedral elements can be employed. However, the method cannot be used to create microstructures containing non-planar internal grain faces, since Abaqus CAE relies on 2D sketches for cell partitioning. It should be noted that, Neper software create the geometry of the cylinder by approximating the circular shape with a polygon prism with flat faces as well as for other curved structures. As a result, the prismatic grains with flat faces would be created in this Software and Neper-3D-CAE program.

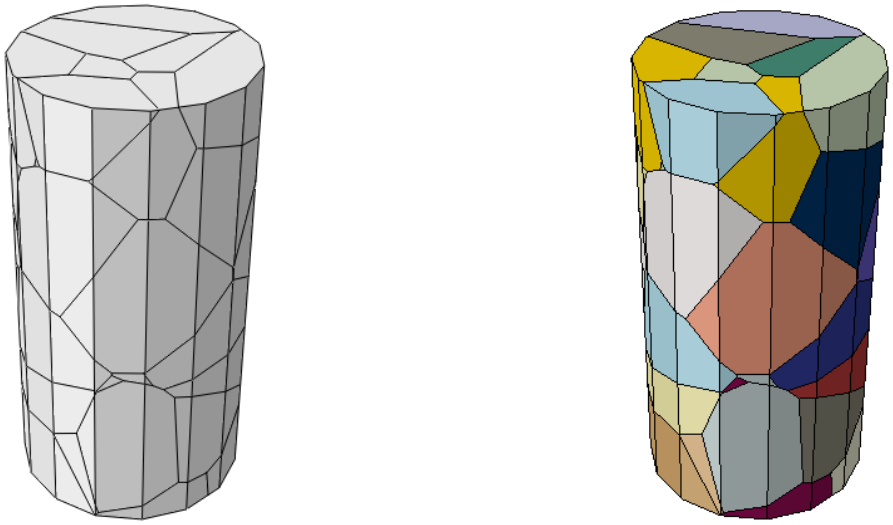


Fig. 7. Polycrystalline tessellation generated in Abaqus CAE by suing of Neper-3D-CAE script for cylindrical geometry with the diameter and height of 20 and 40, respectively.

4. Results

By performing the pre-mentioned procedure in a loop for all rows in the Inner_Faces text file, the polycrystalline texture could be generated for different geometries. A general geometry may contain curved boundaries. It's not possible to define a plane for a grain face contains more than three vertices on the curved boundaries. Only inner faces have been used in the developed program for cell partitioning. Therefore, any geometry, could be used for CPFEM modeling by using this program. Figure 6 demonstrates a cubic geometry with 10, 50, and 100 grains, respectively. Furthermore, Fig. 7 shows another example of the capability of program for generating textured cylindrical geometry for CPFEM modeling.

5. Crystal Plasticity Simulation of 316L SS Fabricated by SLM Method

5.1. Constitutive equations of CP

Linear anisotropic elastic behavior has been considered in the simulations, where λ and μ are lame constants and β is used for defining anisotropic behavior as proposed by [34]. Finite deformation was supposed in CP and its decomposition can be written as:

$$\mathbf{F} = \mathbf{F}^{e}\mathbf{F}^{p} \tag{7}$$

where $\mathbf{F}^{e}$ and $\mathbf{F}^{p}$ represent the elastic part and plastic part of the deformation gradient, respectively. Then, the velocity gradient can be obtained as:

$$\mathbf{L} = \dot{\mathbf{F}}\,\mathbf{F}^{-1} = \dot{\mathbf{F}}^{e}\left(\mathbf{F}^{e}\right)^{-1} + \dot{\mathbf{F}}^{e}\dot{\mathbf{F}}^{p}\left(\mathbf{F}^{p}\right)^{-1}\left(\mathbf{F}^{e}\right)^{-1} \tag{8}$$

which can be divided to the elastic ($\mathbf{L}^{e}$) and plastic parts ($\mathbf{L}^{p}$), where they can be written as:

$$\begin{cases} \mathbf{L}^{e} = \dot{\mathbf{F}}^{e}\left(\mathbf{F}^{e}\right)^{-1} \\ \mathbf{L}^{p} = \dot{\mathbf{F}}^{e}\,\mathbf{L}_{0}^{p}\left(\mathbf{F}^{e}\right)^{-1} = \sum_{\alpha=1}^{n}\dot{\gamma}^{\alpha}\mathbf{s}^{\alpha}\mathbf{m}^{\alpha} \end{cases} \tag{9}$$

where $\mathbf{s}^{\alpha}$ and $\mathbf{m}^{\alpha}$ are unit vectors along the slip direction and slip plane normal, respectively. The yield criterion of a single-crystal was defined by the critical shear stress for each slip system as:

$$\dot{\tau}_{c}^{\alpha} = \sum_{\beta=1}^{n} h_{\alpha\beta}\dot{\gamma}^{\beta} \tag{10}$$

where $\dot{\tau}_{c}^{\alpha}$ represents the critical shear stress rate of each slip system, $h_{\alpha\beta}$ coefficients are the components of the hardening modulus matrix, $\dot{\gamma}^{\beta}$ are the shear strain rate of the β slip system.

In order to describe the hardening behavior, it was supposed by [31-32] as:

$$h_{\alpha\beta} = h(\gamma)\left[q + (1-q)\delta_{\alpha\beta}\right] \tag{11}$$

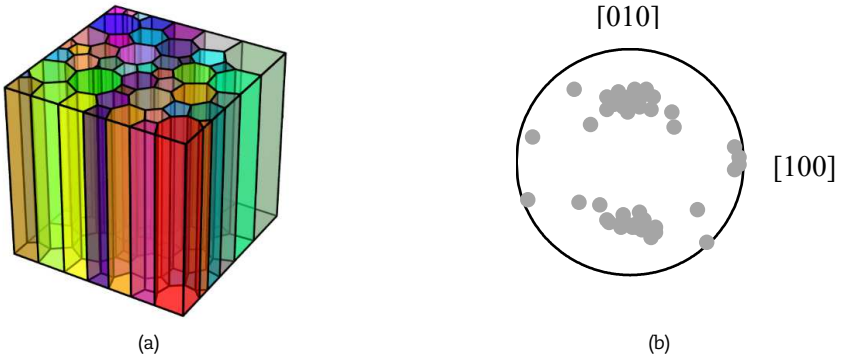
where q is the parameter associated with latent hardening effects. This parameter typically ranges between 1 and 1.4. $h(\gamma)$ is a critical stress function and can be defined as:

$$h(\gamma) = h_{0}\,\mathrm{sech}^{2}\left(\frac{h_{0}\gamma}{\tau_{c}^{\alpha} - \tau_{0}^{\alpha}}\right) \tag{12}$$

where h_{0}, τ_{0}^{α} and τ_{c}^{α} are hardening parameter, initial shear yield stress and ultimate yield stress of each slip system, respectively.

Table 3. Material parameters used for CPFEM simulation of 316L SS samples fabricated by SLM method.

λ MPa	μ MPa	β MPa	τ_0^α MPa	τ_c^α MPa	h_0 MPa	λ_p	$\dot{\gamma}_0$	m	q
175000	55000	5000	140	380	10	0	0.001	0.1	1



(a) (b)

Fig. 8. (a) Schematic representation of microstructure with columnar grains for CPFEM simulation of 316L SS sample under tension, (b) crystallographic orientation with respect to the loading direction.

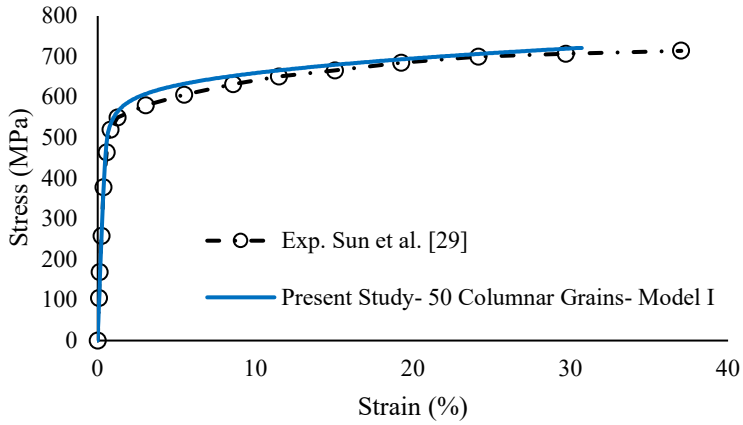


Fig. 9. Comparison of numerical stress-strain curve obtained in the present study with experimental results of [29] for 316L SS fabricated samples by using of SLM method.

Rate-dependent models are considered in this study, in which slip systems are defined independently. The shear rate for active slip systems is expressed via an exponential function of Hutchinson [33]:

$$\dot{\gamma}^\alpha = \dot{\gamma}_0 \operatorname{sgn}(\tau^\alpha)\left|\frac{\tau^\alpha}{\tau_c^\alpha}\right|^{\frac{1}{m}} \tag{14}$$

where $\dot{\gamma}_0$ and m are constants and when m tends to zero, the model becomes rate independent model.

5.2. Numerical solution of CP model

The method proposed by [34] were used for numerical integration of the CP modeling were plastic deformation gradient $\mathbf{F}^p$ is considered to be the fundamental variable. The plastic incompressibility can be assumed and plastic deformation gradient can be obtained by solving Eq. (14) at each time step:

$$\mathbf{f} = \sum_\alpha \dot{\gamma}^\alpha \left(\mathbf{P}^\alpha + \Omega^\alpha\right) - \frac{1}{\Delta t}\left(\mathbf{I} - \mathbf{F}_n^p \mathbf{F}_{n+1}^{p^{-1}}\right) + \lambda_p \left(\left|\mathbf{F}_{n+1}^p\right| - 1\right)\mathbf{I} = \mathbf{0} \tag{14}$$

In this equation, a penalty parameter, λ_p, is introduced to enforce the incompressibility condition. Newton-Raphson method was used to numerically solve Eq. (14).

5.3. Creating the geometry

The proposed method could be used for FE simulation in practical engineering applications such as CPFEM. In order to investigate the mechanical behavior of 316L SS under tension, a cubic RVE with dimensions of 0.1 mm containing 50 columnar grains which is consistent with microstructure of SLM fabricated samples has been created as shown in Fig. 8(a). Also, Fig. 8(b) illustrates the initial crystallographic orientations with respect to the loading direction. The initial orientation of the crystals relative to the [001] pole is plotted based on reference data [29].

5.4. Simulation Results of CP Model

The proposed model for CP simulation has been integrated into Abaqus software by using of UMAT subroutine. Table 3 represents the material parameters used for the CPFEM simulation. Most of these parameters have physical meaning and could be calculated by using of tensile and shear test results. The geometry has been discretized by using of 2365 quadratic tetrahedron C3D10 elements.

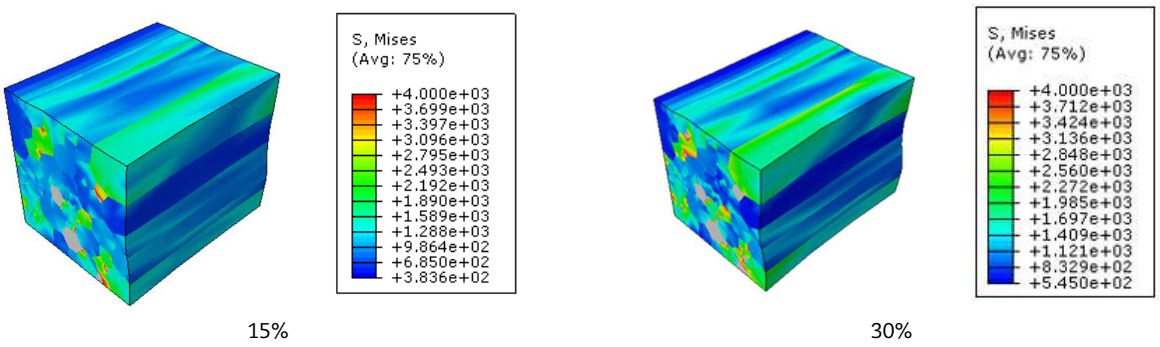


Fig. 10. Equivalent stress distribution at 15% and 30 % strains of CPFEM simulation for 316 LSS.

The obtained stress-strain curve has been compared with experimental results [29] in Fig. 9 for strains up to 30 percent. The obtained results are consistent with experimental results and the elastic modulus, yield strength, and hardening behavior of the 316L SS SLM samples under tension has been captured with acceptable accuracy by using of a geometry with minimum number of grains. The equivalent stress distribution of the simulations at 15% and 30 % strain were presented in Fig. 10. The simulation has captured the orientation dependent of stress distribution for grains in the model.

6. Conclusions

A python program called "Nerper-3D-CAE" used for generating 2D and 3D polycrystalline microstructure in Abaqus CAE has been developed. Using this program make it possible to solve CPFEM modeling with commercial finite element programs like Abaqus CAE by considering the grain shape effect. Also, the geometry could be adjusted. Since only inner faces of a polycrystalline texture are being used for cell partitioning of the geometry, this program can be used for generating texture for any geometry even with curved boundaries. Using this program with proper input files, provides all the required input files for solving CPFEM files with Abaqus CAE software. In order to investigate the engineering application of the proposed method for cell partitioning of a polycrystalline material, a UMAT subroutine has been implemented in Abaqus CAE software for CPFEM simulation of 316L SS samples fabricated by using of SLM method. It's observed that proposed method has captured the elastic and plastic behavior of the samples.

Author Contributions

The manuscript was written through the contribution of all authors. All authors discussed the results, reviewed, and approved the final version of the manuscript.

Acknowledgments

Not applicable.

Conflict of Interest

The authors declared no potential conflicts of interest concerning the research, authorship, and publication of this article.

Funding

The authors received no financial support for the research, authorship, and publication of this article.

Data Availability Statements

The datasets generated and/or analyzed during the current study are available from the corresponding author on reasonable request.

Nomenclature

$\mathbf{E}=(\mathbf{E}_1,\mathbf{E}_2,\mathbf{E}_3)$	Unit vectors of reference coordinate system	Subscript α	Counter index for slip systems
$\mathbf{e}=(\mathbf{e}_1,\mathbf{e}_2,\mathbf{e}_3)$	Unit vectors of new coordinate system	Subscript β	Counter index for slip systems
$\mathbf{F}$	Deformation gradient	β	Anisotropic elastic material constant
$\mathbf{F}^e$	Elastic deformation gradient	$\delta_{\alpha\beta}$	Kronecker delta
$\mathbf{F}^p$	Plastic deformation gradient	γ^β	Shear strain for β-th slip system
$\dot{\mathbf{F}}$	Rate of deformation gradient	$\dot{\gamma}^\beta$	Shear strain rate for β-th slip system
$h_{\alpha\beta}$	Components of hardening matrix	$\dot{\gamma}_0$	Shear strain rate for reference plane
h_0	Hardening material parameter	τ_0^α	Initial shear strength for α-th slip system

$h(\gamma)$	Hardening function	τ_c^α	Critical shear strength for α-th slip system
$\mathbf{I}$	Identity function	Ω^α	Anti-symmetric part of velocity gradient for α-th slip system
ID	Vertex ID	Δt	Time interval
ID_{fi}	Inner face ID	λ	Lame constant
$\mathbf{L}$	Velocity gradient	λ_p	Penalty parameter
$\mathbf{L}^e$	Elastic velocity gradient	μ	Lame constant
$\mathbf{L}^p$	Plastic velocity gradient	$\mathbf{P}^\alpha$	Symmetric part of velocity gradient for α-th slip system
m	Material parameter	q	Hardening parameter
$\mathbf{m}^0$	Initial unit normal vector of slip systems	$\mathbf{R}$	Rotation matrix
n	Number of steps	$\mathbf{r}$	Position vector for vertices in new coordinate system
n_f	Number of faces	$\mathbf{r}'$	Position vector for vertices in reference coordinate system
n_v	Number of vertices	$\mathbf{s}^0$	Initial unit direction vector of slip systems
p	Number of polygons		

References

[1] Khan, A.S., Huang, S., *Continuum theory of plasticity*, John Wiley & Sons, 1995.
[2] Simo, J.C., Hughes, T.J., *Computational inelasticity*, Vol. 7, Springer Science & Business Media, 2006.
[3] Quey, R., Dawson, P.R., Barbe, F., Large-scale 3D random polycrystals for the finite element method: Generation, meshing and remeshing, *Computer Methods in Applied Mechanics and Engineering*, 200(17-20), 2011, 1729-1745.
[4] Groeber, M.A., Jackson, M.A., DREAM. 3D: a digital representation environment for the analysis of microstructure in 3D, *Integrating Materials and Manufacturing Innovation*, 3(1), 2014, 56-72.
[5] Serrao, P.H., Sandfeld, S., Prakash, A., OptiMic: A tool to generate optimized polycrystalline microstructures for materials simulations, *SoftwareX*, 15, 2021, 100708.
[6] Prasad, M.R., Vajragupta, N., Hartmaier, A., Kanapy: A Python package for generating complex synthetic polycrystalline microstructures, *Journal of Open Source Software*, 4(43), 2019, 1732.
[7] Hart, K.A., Rimoli, J.J., MicroStructPy: A statistical microstructure mesh generator in Python, *SoftwareX*, 12, 2020, 100595.
[8] Smith M., *ABAQUS/Standard user's manual*, version 6.9. United States: Dassault Systèmes Simulia Corp, 2009.
[9] Systèmes, D., *Abaqus User Subroutines Reference Guide*, Version 6.14. Dassault Systemes Simulia Corp., Providence, RI, USA, 2014.
[10] Huang, Y., *A user-material subroutine incroporating single crystal plasticity in the ABAQUS finite element program*, Cambridge: Harvard University, 1991.
[11] Alankar, A., Mastorakos, I.N., Field, D.P., A dislocation-density-based 3D crystal plasticity model for pure aluminum, *Acta Materialia*, 57(19), 2009, 5936-5946.
[12] Alharbi, H.F., Kalidindi, S.R., Crystal plasticity finite element simulations using a database of discrete Fourier transforms, *International Journal of Plasticity*, 66, 2015, 71-84.
[13] Biswas, A., Prasad, M.R., Vajragupta, N., ul Hassan, H., Brenne, F., Niendorf, T., Hartmaier, A., Influence of microstructural features on the strain hardening behavior of additively manufactured metallic components, *Advanced Engineering Materials*, 21(7), 2019, 1900275.
[14] Demir, E., Martinez-Pechero, A., Hardie, C., Tarleton, E., OXFORD-UMAT: An efficient and versatile crystal plasticity framework, *International Journal of Solids and Structures*, 307, 2025, 113110.
[15] Rosenbusch, S.M., Diercks, P., Kindrachuk, V., Unger, J.F., Integrating custom constitutive models into FEniCSx: A versatile approach and case studies, *Advances in Engineering Software*, 206, 2025, 103922.
[16] Yan, W., Lin, S., Kafka, O.L., Lian, Y., Yu, C., Liu, Z., Liu, W.K., Data-driven multi-scale multi-physics models to derive process–structure–property relationships for additive manufacturing, *Computational Mechanics*, 61(5), 2018, 521-541.
[17] Cruzado, A., Lucarini, S., LLorca, J., Segurado, J., Crystal plasticity simulation of the effect of grain size on the fatigue behavior of polycrystalline Inconel 718, *International Journal of Fatigue*, 113, 2018, 236-245.
[18] Zhang, C., Zhang, L.W., Shen, W.F., Xia, Y.N., Yan, Y.T., 3d crystal plasticity finite element modeling of the tensile deformation of polycrystalline ferritic stainless steel, *Acta Metallurgica Sinica (English Letters)*, 30(1), 2017, 79-88.
[19] Moghaddam, M.G., Achuthan, A., Bednarcyk, B.A., Arnold, S.M., Pineda, E.J., Grain size-dependent crystal plasticity constitutive model for polycrystal materials, *Materials Science and Engineering: A*, 703, 2017, 521-532.
[20] Feather, W.G., Lim, H., Knezevic, M., A numerical study into element type and mesh resolution for crystal plasticity finite element modeling of explicit grain structures, *Computational Mechanics*, 67(1), 2021, 33-55.
[21] Hu, P., Liu, Y., Zhu, Y., Ying, L., Crystal plasticity extended models based on thermal mechanism and damage functions: Application to multiscale modeling of aluminum alloy tensile behavior, *International Journal of Plasticity*, 86, 2016, 1-25.
[22] Grilli, N., Tarleton, E., Cocks, A.C., Neper2CAE and PyCiGen: Scripts to generate polycrystals and interface elements in Abaqus, *SoftwareX*, 13, 2021, 100651.
[23] Roe, R.J., Inversion of pole figures for materials having cubic crystal symmetry, *Journal of Applied Physics*, 37(5), 1966, 2069-2072.
[24] Wang, Z., Jiang, B., Wu, S., Liu, W., Anisotropic tension-compression asymmetry in SLM 316L stainless steel, *International Journal of Mechanical Sciences*, 246, 2023, 108139.
[25] Fu, B., Wang, C., Dong, Y., Liu, X., Ke, Y., Wang, D., Wahab, M.A., Crystal plasticity modeling of fretting fatigue crack initiation behavior in Ti6Al4V, *Engineering Failure Analysis*, 168, 2025, 109074.
[26] Wu, H., Zhang, Y., Zou, T., Wang, Q., Zhang, H., Wang, T., ..., Wang, Q., Crystal plasticity analysis of tensile plastic behavior and damage mechanisms of additive manufactured TiAl alloy under elevated temperatures, *Journal of Materials Research and Technology*, 32, 2024, 2188-2199.
[27] Zhou, S., Bettaieb, M.B., Abed-Meraim, F., A physically-based mixed hardening model for the prediction of the ductility limits of thin metal sheets using a CPFE approach, *International Journal of Plasticity*, 176, 2024, 103946.
[28] Grilli, N., Tarleton, E., Cocks, A.C., Neper2CAE and PyCiGen: Scripts to generate polycrystals and interface elements in Abaqus, *SoftwareX*, 13, 2021. 100651.
[29] Sun, Z., Tan, X., Tor, S.B., Chua, C.K., Simultaneously enhanced strength and ductility for 3D-printed stainless steel 316L by selective laser melting, *NPG Asia Materials*, 10, 2018, 127-136.
[30] Hosford, W.F., *The Mechanics of Crystals and Textured Polycrystals (Oxford Engineering Science Series)*, Oxford University Press, New York, 1993.
[31] Isaenkova, M.G., Yudin, A.V., Rubanov, A.E., Osintsev, A.V., Degadnikova, L.A., Deformation behavior modelling of lattice structures manufactured by a selective laser melting of 316L steel powder, *Journal of Materials Research and Technology*, 9(6), 2020, 15177-15184.
[32] Kergaßner, A., Koepf, J.A., Markl, M., Körner, C., Mergheim, J., Steinmann, P., A Novel Approach to Predict the Process-Induced Mechanical Behavior of Additively Manufactured Materials, *Journal of Materials Engineering and Performance*, 30, 2021, 5235–5246.
[33] Hutchinson, J.W., Bounds and self-consistent estimates for creep of polycrystalline materials, *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 348(1652), 1976, 101-127.

[34] Ling, X., Horstemeyer, M.F., Potirniche, G.P., On the numerical implementation of 3D rate-dependent single crystal plasticity formulations, *International Journal for Numerical Methods in Engineering*, 63(4), 2005, 548-568.

ORCID iD

Mehran Araghi https://orcid.org/0000-0002-9894-1479
Ali Nayebi https://orcid.org/0000-0002-5732-8248
Hojjatollah Rokhgireh https://orcid.org/0000-0001-7551-7424

© 2025 Shahid Chamran University of Ahvaz, Ahvaz, Iran. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0 license) (http://creativecommons.org/licenses/by-nc/4.0/).

How to cite this article: Araghi M., Nayebi A., Rokhgireh H. Neper-3D-CAE: The Interface Script between Neper and Abaqus for Polycrystalline Modeling Used in Crystal Plasticity Simulation, *J. Appl. Comput. Mech.*, 12(4), 2026, 1348-1358. https://doi.org/10.22055/jacm.2025.48554.5322

Publisher's Note Shahid Chamran University of Ahvaz remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.