




# A Python Library for Pre- and Post-Processing of DAMASK Simulations

Daniel Otto de Mentock <sup>1</sup>, Sharan Roongta <sup>1</sup>, Franz Roters <sup>1</sup>, Philip Eisenlohr <sup>2</sup>, and Martin Diehl <sup>3,4</sup>

<sup>1</sup> Max Planck Institute for Sustainable Materials, 40237 Düsseldorf, Germany <sup>2</sup> Chemical Engineering and Materials Science, Michigan State University, East Lansing, MI 48824, United States of America <sup>3</sup> Department of Materials Engineering, KU Leuven, Kasteelpark Arenberg 44, 3001 Leuven, Belgium <sup>4</sup> Department of Computer Science, KU Leuven, Celestijnenlaan 200a, 3001 Leuven, Belgium

DOI: [10.21105/joss.07164](https://doi.org/10.21105/joss.07164)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

---

Editor: [Bonan Zhu](#) 

## Reviewers:

- [@hakonanes](#)
- [@mkuehbach](#)

Submitted: 11 July 2024

Published: 28 January 2025

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

DAMASK, the Düsseldorf Advanced Material Simulation Kit, is a modular multi-physics crystal plasticity simulation package distributed as free and open-source software under the GNU Affero General Public License (AGPL) ([Roters et al., 2019](#)). To facilitate easy pre- and post-processing of DAMASK simulations and simplify the creation of custom workflows for Integrated Computational Materials Engineering (ICME), an accompanying Python library has been developed. This library, which is also AGPL-licensed, is introduced here.

## Statement of need

Multi-physics-enriched crystal plasticity simulations are used to establish processing–structure–property relationships of crystalline materials at engineering length and time scales. Setting up a simulation requires parameterization of the constitutive models, description of microstructure and geometry, and definition of boundary and initial conditions. The interpretation of the resulting data requires tools for statistical analysis, plotting, and 3D visualization. Moreover, the design and study of complex materials using various computational techniques requires interoperability between different software packages in ICME workflows ([Shah et al., 2022](#)). These needs are best addressed by a modular set of routines for pre- and post-processing that integrate seamlessly into an existing ecosystem.

## Features

The materialpoint models implemented in DAMASK can be used in conjunction with different solvers: DAMASK\_grid, DAMASK\_mesh, DAMASK\_Marc. The grid solver (DAMASK\_grid) operates on periodically repeated hexahedral domains discretized by a structured grid, whereas the two other solvers are based on the finite element method and can be used for unstructured meshes, thus allowing for more complex geometries. Hence, the definition of the geometry together with boundary and initial conditions depends on the selected solver and requires different pre-processing tools. In contrast, the configuration of the materialpoint model and the DAMASK-specific HDF5 file format that is used to store the simulation results ([Diehl et al., 2017](#)) are solver-agnostic.

The DAMASK Python library is a package called damask and contains functionality for pre-processing tasks, such as the generation and modification of microstructures, load cases, material definition, or numerical parameters, as well as functionality for post-processing that enables analysis and visualization of DAMASK results. A particular focus is laid on finite-strain

continuum mechanics and crystallography. The routines for conversion between the different kinds of orientation representations, such as Euler angles, rotation matrices, unit quaternions, or axis-angle pairs, are based on a consistent set of conventions (Rowenhorst et al., 2015). The provided routines and data structures interoperate seamlessly with other libraries from the Python ecosystem, such as NumPy (Harris et al., 2020), pandas (McKinney, 2010), Matplotlib (Hunter, 2007), SciPy (Virtanen et al., 2020), VTK/PyVista (Ahrens et al., 2005) (Sullivan & Kaszynski, 2019), PyYAML, h5py (Collette, 2013) (Folk et al., 2011), and orix (Johnstone et al., 2020) to facilitate the definition of custom ICME workflows.

Apart from its DAMASK-specific processing capabilities, many routines of the library can be used in other materials science and continuum mechanics applications.

## Pre-Processing

The main goal of the pre-processing tools is to enable users to incorporate data from various sources into their simulation setup. For example, the users can integrate domain-specific software such as Neper (Quey et al., 2011), DREAM.3D (Groeber & Jackson, 2014), and Gmsh (Geuzaine & Remacle, 2009) to define microstructures and employ the provided Python routines to convert them into DAMASK-compatible input files. Similarly, the materialpoint configuration and load case definition are internally represented as particular Python classes that simplify the creation, modification, and export to YAML files. In addition to the three mandatory input files, i.e., geometry definition, material configuration, and the load case description, the creation and modification of an optional YAML file to fine-tune numerical parameters is also supported.

## Post-Processing

The post-processing tools are centered around the HDF5 output file resulting from a DAMASK simulation. This flexible file format is called DADF5, short for “DAMASK HDF5” (Diehl et al., 2017), which allows data storage according to the FAIR principles (Wilkinson et al., 2016). All data in DADF5 is enriched with metadata, such as the date of creation, a human-understandable description, and the physical unit, to ensure that the data is findable by humans and computers. The Result class provides custom views on the hierarchical data layout of the DADF5 file for easy accessibility. Moreover, it provides routines for the computation of derived quantities that are stored alongside with automatically created metadata in the output file. The data can be further processed within Python or exported to various file formats for analysis using third-party tools such as DREAM.3D, ParaView, or MTEX (Bachmann et al., 2010). This, together with the fact that the complete simulation setup is stored in the DADF5 file, makes it possible to re-use and re-evaluate the data.

## Alternatives

Some of the functionality provided by the damask package overlaps with other packages, such as orix (crystallography and rotations), PyVista (3D visualization), and SciPy (rotations). Because the orix and PyVista packages are currently in an early development status and not available via native package managers on popular Linux distributions, the corresponding damask functionality has been implemented independently of both. The rotation functionality of the damask package does not rely on SciPy in order to reproduce one-to-one the conventions used in the main simulation code of DAMASK written in Fortran, despite using SciPy routines for other purposes.

## Availability

The damask Python package is developed within the DAMASK main repository, but is also available as a separate package via multiple channels. For documentation and installation options we refer to the [DAMASK website](#).

## Acknowledgements

This research was financially supported by Internal Funds KU Leuven.

## References

- Ahrens, J., Geveci, B., & Law, C. (2005). ParaView: An end-user tool for large data visualization. In C. D. Hansen & C. R. Johnson (Eds.), *Visualization handbook* (pp. 717–731). Elsevier. <https://doi.org/10.1016/B978-012387582-2/50038-1>
- Bachmann, F., Hielscher, R., & Schaeben, H. (2010). Texture analysis with MTEX – free and open source software toolbox. *Solid State Phenomena*, 160, 63–68. <https://doi.org/10.4028/www.scientific.net/SSP.160.63>
- Collette, A. (2013). *Python and HDF5*. O'Reilly. ISBN: 9781449367831
- Diehl, M., Eisenlohr, P., Zhang, C., Nastola, J., Shanthraj, P., & Roters, F. (2017). A flexible and efficient output file format for grain-scale multiphysics simulations. *Integrating Materials and Manufacturing Innovation*, 6(1), 83–91. <https://doi.org/10.1007/s40192-017-0084-5>
- Folk, M., Heber, G., Koziol, Q., Pourmal, E., & Robinson, D. (2011). An overview of the HDF5 technology suite and its applications. *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, 36–47. <https://doi.org/10.1145/1966895.1966900>
- Geuzaine, C., & Remacle, J.-F. (2009). Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11), 1309–1331. <https://doi.org/10.1002/nme.2579>
- Groeber, M. A., & Jackson, M. A. (2014). DREAM.3D: A digital representation environment for the analysis of microstructure in 3D. *Integrating Materials and Manufacturing Innovation*, 3, 56–72. <https://doi.org/10.1186/2193-9772-3-5>
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- Johnstone, D. N., Martineau, B. H., Crout, P., Midgley, P. A., & Eggeman, A. S. (2020). Density-based clustering of crystal (mis)orientations and the orix Python library. *Journal of Applied Crystallography*, 53(5), 1293–1298. <https://doi.org/10.1107/S1600576720011103>
- McKinney, W. (2010). Data Structures for Statistical Computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61). <https://doi.org/10.25080/Majora-92bf1922-00a>
- Quey, R., Dawson, P. R., & Barbe, F. (2011). Large-scale 3D random polycrystals for the finite element method: Generation, meshing and remeshing. *Computer Methods in Applied Mechanics and Engineering*, 200(17-20), 1729–1745. <https://doi.org/10.1016/j.cma.2011.01.002>

- Roters, F., Diehl, M., Shanthraj, P., Eisenlohr, P., Reuber, C., Wong, S. L., Maiti, T., Ebrahimi, A., Hochrainer, T., Fabritius, H.-O., Nikolov, S., Friak, M., Fujita, N., Grilli, N., Janssens, K. G. F., Jia, N., Kok, P. J. J., Ma, D., Meier, F., ... Raabe, D. (2019). DAMASK – the Düsseldorf Advanced Material Simulation Kit for modeling multi-physics crystal plasticity, thermal, and damage phenomena from the single crystal up to the component scale. *Computational Materials Science*, 158, 420–478. <https://doi.org/10.1016/j.commatsci.2018.04.030>
- Rowenhorst, D., Rollett, A. D., Rohrer, G. S., Groeber, M., Jackson, M., Konijnenberg, P. J., & De Graef, M. (2015). Consistent representations of and conversions between 3D rotations. *Modelling and Simulation in Materials Science and Engineering*, 23(8), 083501. <https://doi.org/10.1088/0965-0393/23/8/083501>
- Shah, V., Sedighiani, K., Van Dokkum, J. S., Bos, C., Roters, F., & Diehl, M. (2022). Coupling crystal plasticity and cellular automaton models to study meta-dynamic recrystallization during hot rolling at high strain rates. *Materials Science and Engineering: A*, 143471. <https://doi.org/10.1016/j.msea.2022.143471>
- Sullivan, B., & Kaszynski, A. (2019). *PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK)*. <https://doi.org/10.21105/joss.01450>
- Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17, 261–272. <https://doi.org/10.1038/s41592-019-0686-2>
- Wilkinson, M. D., Dumontier, M., Aalbersberg, IJ. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Bonino da Silva Santos, L., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., ... Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1). <https://doi.org/10.1038/sdata.2016.18>