The Journal of Open Source Software

partycls: A Python package for structural clustering

Joris Paret[1] **and Daniele Coslovich**[2]

1 Laboratoire Charles Coulomb (L2C), Université de Montpellier, CNRS, Montpellier, France **2** Dipartimento di Fisica, Università di Trieste, Italy

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

Summary

partycls is a Python framework to perform spatio-temporal clustering of systems composed of interacting particles. It provides descriptors suitable for applications in condensed matter physics and integrates the necessary tools of unsupervised learning into a streamlined workflow. Through a simple and expressive interface, partycls allows one to open a trajectory file, perform a clustering based on the selected structural descriptor, and analyze and/or save the results with only few lines of code. Additional pre-processing steps such as feature scaling and dimensionality reduction are organically integrated into the workflow and make it easy to assess the robustness of the results.

Statement of need

Analysis of the local arrangements of atoms and molecules in dense liquids and solids is crucial to understand their emergent physical properties. This is particularly important in systems whose local structure is heterogeneous, which include polycrystalline materials and partially ordered systems, like semi-crystalline polymers (C. & P., 2017) or metastable liquids during crystal nucleation (Russo & Tanaka, 2016). Even more challenging is the case of glass-forming liquids and glasses (Royall & Williams, 2015), which often display locally stable arrangements, known as locally favored structures, whose symmetry and local chemical concentration differ in a subtle way from the bulk. In the glassy systems, structure-property relationships have been long sought, but are difficult to identify in general (Hocky et al., 2014) and require tailored local structural descriptors (Richard et al., 2020).

Traditional methods to classify particles according to local arrangements of their neighbors include the Voronoi tessellation (Tanemura et al., 1977) and common neighbor analysis (CNA) (Honeycutt & Andersen, 1987). More recent approaches provide detailed insight into the topology of the particles' arrangements (Lazar et al., 2015; Malins et al., 2013). Many of these methods are implemented in open source code and can be directly applied to trajectories produced by computer simulations, but also to experimental data of colloidal suspensions analyzed using confocal microscopes (Royall & Williams, 2015). One of the shortcomings of these approaches, however, is that they tend to produce a very large number of distinct signatures, especially in disordered systems. Moreover, small distortions of the local environments can substantially affect the structural fingerprint of the particles.

Recently, unsupervised learning has emerged as an alternative approach to characterize the local structure of disordered materials (Boattini et al., 2019; Reinhart et al., 2017). In particular, clustering methods based on simple observables, such as radial distribution functions, bond angle distributions, and bond orientational parameters (BOP), can provide useful insight into the structural heterogenity of glassy systems (Boattini et al., 2020; Paret et al., 2020). The idea is to differentiate the particles' structural environments through the prism of clustering, i.e. by grouping the particles according to the similarity of their local structure. With the present code, we aim to provide a coherent framework to facilitate unsupervised learning of structural and dynamical features of condensed matter systems. Through a variety of structural descriptors, dimensionality reduction methods, clustering algorithms and filtering

options, partycls makes it possible to customize these steps to study specific aspects of the structure and to assess the robustness of the results. Future versions of the code will also implement clustering in space *and* time, to learn about the dynamics of the system as well.

Design

partycls is mostly written in Python, with a few parts coded in Fortran 90 for efficiency. It provides a simple and configurable workflow, from reading the input trajectory, through the pre-processing steps, to the final clustering results. It is designed to accept a large variety of formats for trajectory files, by relying on optional third-party packages such as `MDTraj` (McGibbon et al., 2015), which supports several well-known trajectory formats, and `atooms` (Coslovich, 2018), which makes it easy to interface custom formats often used by in-house simulation codes. Thanks to a flexible system of filters, it is possible to compute the structural descriptors or perform the clustering on restricted subsets of particles of the system, based on arbitrary particle properties. In addition to its native descriptors, partycls also supports additional structural descriptors via DScribe [dscribe]. A substantial fraction of the code acts as a wrapper around functions of the machine learning package `scikit-learn` (Pedregosa et al., 2011). This allows non-experienced users to rely on the simplicity of partycls's interface without any prior knowledge of this external package, while experienced users can take full advantage of the many options provided by `scikit-learn`. In addition, the code integrates the relevant tools for distributional clustering, such as a community inference method tailored to amorphous materials (Paret et al., 2020), and several helper functions, e.g. for merging mixture models (Baudry et al., 2010) and consistent centroid-based cluster labeling. A simple diagram of the different steps and combinations to create a custom workflow is shown in Figure 1. A collection of notebooks, with various examples and detailed instructions on how to run the code, is available in the partycls's repository.

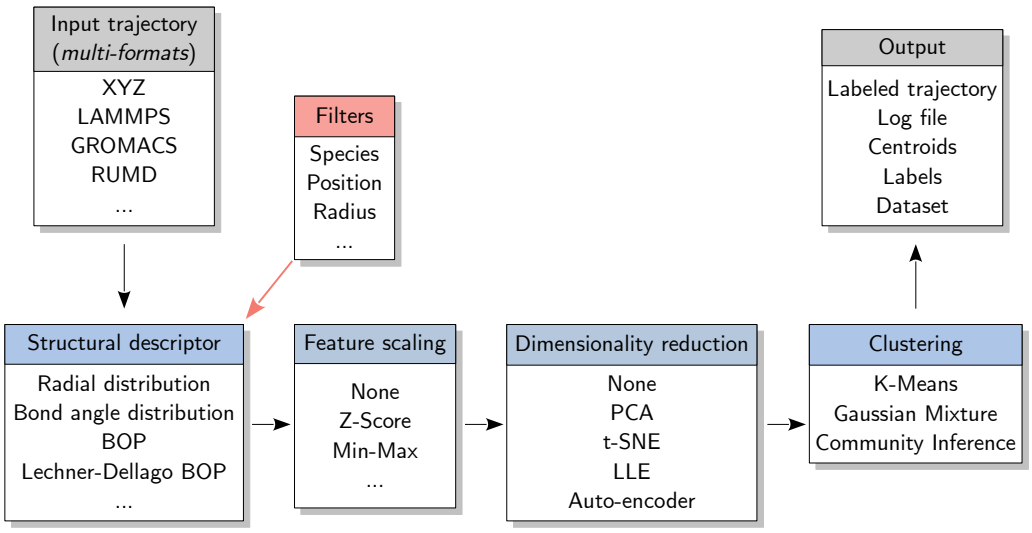

Figure 1: The different steps to perform a structural clustering. The input must be trajectory a file with a supported format. After selecting the type of structural descriptor (and optional filters) to use for the clustering, optional steps for pre-processing the data are possible: feature scaling and dimensionality reduction. Finally, a clustering is performed using the selected algorithm. Output files are written (unless disabled by the user), such as a labeled trajectory file (*i.e.* containing a row with cluster labels, to facilitate visualization) or the dataset used by the clustering algorithm.

Examples

As a simple example, we consider the detection of the grain boundaries in a polycrystal formed by differently oriented FCC crystallites. This is easily achieved even with a simple radial descriptor, since that the average radial distribution of particles at the boundaries is different than the one of the crystal in the bulk. The following short piece of code opens the input trajectory stored in the file `grains.xyz`, computes the local radial distribution functions of the particles, applies a standard Z-Score normalization on the data, and finally performs a clustering using the Gaussian mixture model (GMM) with $K = 2$ clusters (default):

```
from partycls import Workflow

wf = Workflow('grains.xyz',
              descriptor='gr',
              scaling='zscore',
              clustering='gmm')
wf.run()
```

Each of these steps is easily tunable, so as to change the workflow with little effort. The labels are available as a simple attribute of the `Workflow` instance. Optionally, a set of output files can be produced for further analysis, including a trajectory file with the cluster labels. Quick visualization of the clusters, as in Figure 2, is possible within partycls through optional visualization backends.

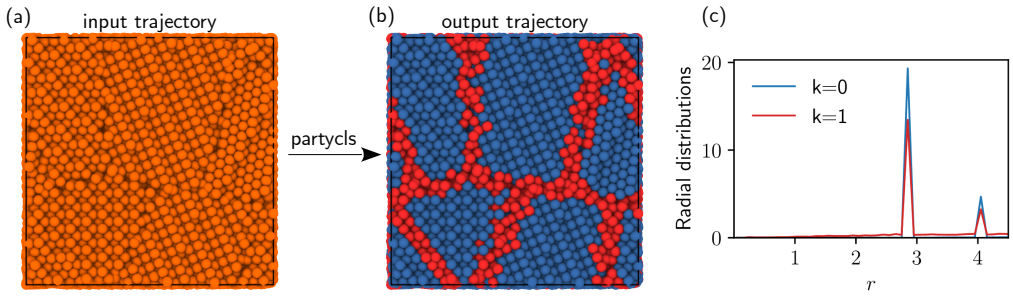

Figure 2: (a) A polycrystalline material with differently oriented FCC crystallites. (b) Using the individual radial distributions of the particle, we can distinguish between the crystalline particles (blue, $k = 0$) and particles at the boundaries (red, $k = 1$). (c) The radial distribution functions restricted to the clusters show a clear difference between the two local environments, with higher peaks for the crystals. All 3D visualizations were rendered in OVITO (Stukowski, 2009).

The local structure of a glass-forming liquid provides a more challenging bench-case, since the system is amorphous overall, but subtle structural features emerge at low temperature. Here, we consider a binary metallic alloy $Cu_{64}Zr_{36}$, which shows a tendency for local icosahedral arrangements around copper atoms (Soklaski et al., 2016). The fraction of atoms that form such locally favored structures increases markedly when the system is cooled at low temperature. We use LAMMPS (Plimpton, 1995) to perform a molecular dynamics simulation using an embedded atom potential. After a rapid quench from high temperature, the supercooled liquid is annealed at $T = 900$K. In the following piece of code, we open a LAMMPS trajectory using `atooms` as backend, we restrict the analysis to the copper atoms and use bond-angle correlations and the K-Means algorithm to form the clusters:

```
from partycls import Trajectory, Workflow
from partycls.descriptor import BondAngleDescriptor

trajectory = Trajectory('cuzr_900K.dat', fmt='lammps', backend='atooms')
descriptor = BondAngleDescriptor(trajectory)
```

```
descriptor.add_filter("species == 'Cu'")

wf = Workflow(trajectory,
              descriptor=descriptor,
              scaling='zscore',
              clustering='kmeans')
wf.run()
```

Here, we directly access classes for the trajectory and the structural descriptor, and then pass them to the `Workflow` instance. Every step of the workflow can also be performed manually by directly instantiating the desired classes, without creating an instance of `Workflow`.

In Figure 3, we see that the distribution of the cluster $k = 1$ is similar to what is expected for icosahedral structural environments, whereas that of the cluster $k = 0$ is flatter and thus more disordered. This provides evidence of the local structural heterogeneity of the system. Similar results have been obtained using related clustering algorithms for simpler models of glass-forming liquids based on Lennard-Jones interactions (Boattini et al., 2020; Paret et al., 2020).

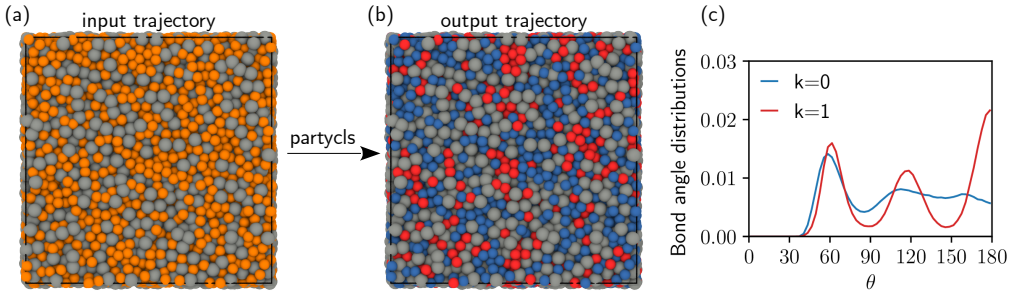

Figure 3: (a) Sample of a copper-zirconium mixture at $T = 900$K. Copper atoms are colored orange and zirconium atoms are colored grey. We look at the angular correlations around the copper atoms only (orange). (b) Copper atoms are now colored blue ($k = 0$) and red ($k = 1$) based on their cluster membership. Zirconium atoms (grey) are discarded from the analysis. (c) Bond angle distributions of the clusters.

Acknowledgements

We thank Robert Jack for his contribution to the community inference method, the merging of mixture models, and for his helpful comments.

References

Baudry, J.-P., Raftery, A. E., Celeux, G., Lo, K., & Gottardo, R. (2010). Combining mixture components for clustering. *Journal of Computational and Graphical Statistics*, *19*(2), 332–353. https://doi.org/10.1198/jcgs.2010.08111

Boattini, E., Dijkstra, M., & Filion, L. (2019). Unsupervised learning for local structure detection in colloidal systems. *The Journal of Chemical Physics*, *151*(15), 154901. https://doi.org/10.1063/1.5118867

Boattini, E., Marín-Aguilar, S., Mitra, S., Foffi, G., Smallenburg, F., & Filion, L. (2020). Autonomously revealing hidden local structures in supercooled liquids. *Nature Communications*, *11*(1), 5479. https://doi.org/10.1038/s41467-020-19286-8

C., M., & P., O. (2017). *Polymer*.

Coslovich, D. (2018). *atooms: A python framework for simulations of interacting particles* (Version 1.3.3) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.1183302

Hocky, G. M., Coslovich, D., Ikeda, A., & Reichman, D. R. (2014). Correlation of local order with particle mobility in supercooled liquids is highly system dependent. *Physical Review Letters*, *113*(15), 157801. https://doi.org/10.1103/PhysRevLett.113.157801

Honeycutt, J. Dana., & Andersen, H. C. (1987). Molecular dynamics study of melting and freezing of small lennard-jones clusters. *The Journal of Physical Chemistry*, *91*(19), 4950–4963. https://doi.org/10.1021/j100303a014

Lazar, E. A., Han, J., & Srolovitz, D. J. (2015). A topological framework for local structure analysis in condensed matter. *Proceedings of the National Academy of Sciences*, *112*(43), E5769–E5776. https://doi.org/10.1073/pnas.1505788112

Malins, A., Williams, S. R., Eggers, J., & Royall, C. P. (2013). Identification of structure in condensed matter with the topological cluster classification. *The Journal of Chemical Physics*, *139*(23), 234506. https://doi.org/10.1063/1.4832897

McGibbon, R. T., Beauchamp, K. A., Harrigan, M. P., Klein, C., Swails, J. M., Hernández, C. X., Schwantes, C. R., Wang, L.-P., Lane, T. J., & Pande, V. S. (2015). MDTraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal*, *109*(8), 1528–1532. https://doi.org/10.1016/j.bpj.2015.08.015

Paret, J., Jack, R. L., & Coslovich, D. (2020). Assessing the structural heterogeneity of supercooled liquids through community inference. *The Journal of Chemical Physics*, *152*(14), 144502. https://doi.org/10.1063/5.0004732

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Plimpton, S. (1995). Fast parallel algorithms for short-range molecular dynamics. *Journal of Computational Physics*, *117*(1), 1–19. https://doi.org/10.1006/jcph.1995.1039

Reinhart, W. F., Long, A. W., Howard, M. P., Ferguson, A. L., & Panagiotopoulos, A. Z. (2017). Machine learning for autonomous crystal structure identification. *Soft Matter*, *13*(27), 4733–4745. https://doi.org/10.1039/C7SM00957G

Richard, D., Ozawa, M., Patinet, S., Stanifer, E., Shang, B., Ridout, S. A., Xu, B., Zhang, G., Morse, P. K., Barrat, J.-L., & al., et. (2020). Predicting plasticity in disordered solids from structural indicators. *Physical Review Materials*, *4*(11), 113609. https://doi.org/10.1103/PhysRevMaterials.4.113609

Royall, C. P., & Williams, S. R. (2015). The role of local structure in dynamical arrest. *Physics Reports*, *560*, 1–75. https://doi.org/10.1016/j.physrep.2014.11.004

Russo, J., & Tanaka, H. (2016). Crystal nucleation as the ordering of multiple order parameters. *The Journal of Chemical Physics*, *145*(21), 211801. https://doi.org/10.1063/1.4962166

Soklaski, R., Tran, V., Nussinov, Z., Kelton, K. F., & Yang, L. (2016). A locally preferred structure characterises all dynamical regimes of a supercooled liquid. *Philosophical Magazine*, *96*(12), 1212–1227. https://doi.org/10.1080/14786435.2016.1158427

Stukowski, A. (2009). Visualization and analysis of atomistic simulation data with OVITOthe open visualization tool. *Modelling and Simulation in Materials Science and Engineering*, *18*(1), 015012. https://doi.org/10.1088/0965-0393/18/1/015012

Tanemura, M., Hiwatari, Y., Matsuda, H., Ogawa, T., Ogita, N., & Ueda, A. (1977). Geometrical analysis of crystallization of the soft-core model*). *Progress of Theoretical Physics*,

58(4), 1079–1095. https://doi.org/10.1143/PTP.58.1079