SoftwareX 9 (2019) 238–243

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

ART_data_analyzer: Automating parallelized computations to study the evolution of materials

Liang Tian [a,b,*], Lin Li [a], Jun Ding [c], Normand Mousseau [d]

[a] Department of Metallurgical and Materials Engineering, University of Alabama, Tuscaloosa, AL, 35404, USA
[b] Department of Materials Science and Engineering, University of Michigan, Ann Arbor, MI, 48109, USA
[c] Materials Science Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[d] Département de physique, Université de Montréal, C.P. 6128, succ. Centre-ville, Montréal (Québec), Canada

ARTICLE INFO

Article history:
Received 10 December 2018
Received in revised form 1 March 2019
Accepted 1 March 2019

Keywords:
Activation and relaxation techniques
Kinetics
Automation and parallelization
Machine learning

ABSTRACT

The kinetics and dynamic evolution of material structures need a comprehensive understanding of the potential energy landscape at current sample state. The Activation–Relaxation Technique (ART) is an efficient way to probe the potential energy landscape by sampling a large amount of events (a single event involves initial, saddle and final state) from which a statistical distribution of activation energy barrier can be extracted. However, there has been a lack of a user-friendly toolkit to automate the parallelization of running of ART simulations and post-processing of data from ART simulations to extract useful physics information and insights. The ART_data_analyzer Python package has been developed to serve this purpose and fill in this gap for the broad community of scientific researchers interested in the kinetics and dynamic transitions of material structures. As a demo, we utilized this software package to demonstrate the user-friendly workflow of studying ZrCuAl metallic glass sample prepared by molecular dynamics.

Code metadata

Current Code version	*V1.1*
Permanent link to code / repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX_2018_251
Legal Code License	*GNU General Public License v3.0*
Code Versioning system used	*git*
Software Code Language used	*python*
Compilation requirements, Operating environments & dependencies	*Linux, OS X, Microsoft Windows; numpy scipy matplotlib pandas scikit-learn pyvoro pathos tess*
If available Link to developer documentation / manual	https://github.com/liangtianumich/ART_data_analyzer/blob/master/readme.txt
Support email for questions	liangtianisu@gmail.com

Software metadata

Current software version	*V1.1*
Permanent link to executables of this version	https://github.com/liangtianumich/ART_data_analyzer
Legal Software License	*GNU General Public License v3.0*
Computing platform/Operating System	*Linux, OS X, Microsoft Windows.*
Installation requirements & dependencies	*numpy scipy matplotlib pandas scikit-learn pyvoro pathos tess*
If available Link to user manual - if formally published include a reference to the publication in the reference list	https://github.com/liangtianumich/ART_data_analyzer/blob/master/readme.txt
Support email for questions	liangtianisu@gmail.com

* Correspondence to: 1024 North Engineering Research Center, Tuscaloosa, AL, 35404, USA.
E-mail address: liangtianisu@gmail.com (L. Tian).

https://doi.org/10.1016/j.softx.2019.03.002

1. Motivation and significance

How atomic structure evolves inside any condensed matter physical system depends on their potential energy landscape. The potential energy landscape is a mapping between the high-dimensional spatial coordinates of all atoms and their potential energy. For atomic structures to evolve, it involves a series of events, where each event consists of three critical states (i.e. initial state, saddle state and final state). A complete description of potential energy landscape contains all possible states of the samples, and can provide all possible events that could occur in the sample. The activation energy barrier (i.e. potential energy difference between initial state and saddle state) determines the transition rate of an event according to the transition state theory. Therefore, determination of all possible events and their corresponding activation energy barrier by calculating potential energy landscape will lead to a thorough understanding of the kinetics and dynamics of structural evolution of a condensed matter sample [1].

The activation and relaxation technique ARTn software developed by Mousseau et al. [2–5] is one of the most widely used tools to find the open-ended saddle states by initially triggering a cluster of atoms surrounding the central atom. For many complex physical systems, the local atomic environment shows strong spatial dependency. In order to obtain the correct statistical distribution of activation and relaxation events, it is necessary to perturb the atoms at various locations of the atomic sample (e.g. molecular dynamics sample) to converge to the correct distribution while minimizing computational expense. However, the ARTn software has not implemented the parallelization of each ARTn simulation, which implies that considerable manipulation is required from users. Therefore, one of the purposes of this ART_data_analyzer package is to address these issues to allow users to sample sufficient events in parallel. This process repeats iteratively until the activation energy of all events reach convergence verified by the implemented student t statistical convergence test. The other advantage of this ART_data_analyzer package is to achieve the necessary user-friendliness through automation while still maintaining the essential physics workflow for user to understand various task processes they are working on. The automation of various post-processing tasks in this ART_data_analyzer package includes filtering events, calculating, visualizing, correlating various physical quantities and/or their changes.

This ART_data_analyzer package can be broadly applied in studying the kinetics of various material classes and molecular samples, such as crystalline materials and their defects (grain boundaries and interfaces), amorphous and molecular materials due to the generality nature of activation and relaxation technique to find open-ended saddle states. This ART_data_analyzer package allows user customization through an easy-to-use input setting file, which organizes various input parameters to customize either ART running or ART data post-processing. For example, not every event found by ART is a successful event. This ART_data_analyzer package provides the option for the user to specify their own customized filtering criteria to decide whether an event is successful. This ART_data_analyzer package is also very robust to any interruption during calculation due to either machine failure or human intervention by providing various well-documented and simple-to-use command options. This ART_data_analyzer package provides the option to perform correlation analysis between various physics data using machine learning models. The package can also manage data by archiving most necessary data files or deleting unused data files to consolidate data for saving disk space, which will facilitate the sharing of research data between research teams.

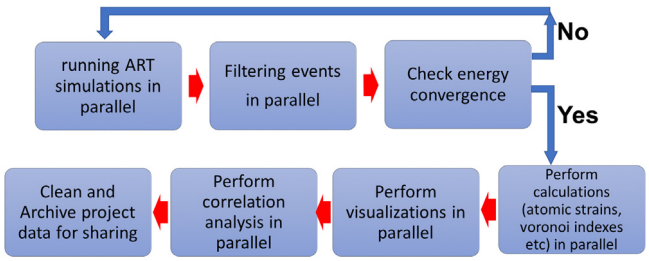

Fig. 1. The pipeline software architecture of this ART_data_analyzer package allows user customization in running various module tasks in a standard workflow.

2. Software description

This ART data analyzer package is a Python package containing an easy-to-use command line tool art_data. This art_data tool integrates with activation and relaxation technique software ARTn package developed by Mousseau et al. [2–4]. This ART_data_analyzer package is now part of ARTn repository in gitlab. The ARTn package in gitlab is freely available by contacting Normand Mousseau at normand.mousseau@umontreal.ca. This ART_data_analyzer package automates the parallel computations of the following aspects in a user workflow: (1) running the Activation and Relaxation Technique ARTn simulations to generate raw events data by the Python ART wrapper; (2) running post-processing tasks of ART data (i.e., filtering, calculating, visualizing, correlating various physical quantities change for large amount of found events)

The automation processes include but not limited to: save all necessary calculations in files, extract, transform and load data for automated visualization, perform correlation analysis between physical quantities.

2.1. Software architecture

The software package provides a pipeline architecture for various modules to communicate through various results files created by each module to allow the user to perform various tasks in a standard workflow. In addition, the package allows the user to customize the input parameters for each module by modifying the input setting file before running each module. Fig. 1 shows the pipeline software architecture of this ART_data_analyzer package in a user workflow.

2.2. Software usage

Before running a large amount of ART simulations and ART data post-processing tasks in parallel, it is worthwhile to note that this automation processes will generate from tens of GBs to a few hundred of GBs data, depending on the nature of each atomic sample. User needs to ensure that ARTn is installed properly and their environment is set up properly, such as ARTn built with lammps library is installed correctly (including loading all necessary lammps package before building lammps); PATH and LD_LIBRARY_PATH for linux, DYLD_LIBRARY_PATH for Unix has been set up in your current terminal session in order for ARTn and lammps to find the openmpi, openblas library. These environmental variables should be the same as those when building lammps and ARTn. A detailed ART installation guide, located in /ART_installation_guide/readme.txt, indicates how to install ARTn with lammps.

After installing ARTn and lammps, the installation and usage of ART_data_analyzer package can be as simple as downloading

the package to desired location and sourcing the modified environmental file. The following things need to be checked when modifying the environmental file.

- Modify DATA_DIR to your desired directory path to save the raw ART simulation data.
- Generally, other environmental variables can be set up the same as the default values.
- Currently, the default settings are that the ART_INPUT storing input files directory path is the same as the DATA_DIR so that all ART input files should be located under DATA_DIR.
- ART_SAMPLE contains the sample name, whose default sample name is conf.lammps and sample type is lammps_data. These have been set up by export ART_SAMPLE=conf.lammps and export SAMPLE_TYPE=lammps_data. This conf.lammps lammps data file can be converted from lammps dump file by Ovito visualization software easily [6]. In some occasions, if user need to use the lammps dump file as the sample, he can modify the sample name to the dump file name (e.g., conf.dump) and sample type to dump, e.g., by export ART_SAMPLE=conf.dump and export SAMPLE_TYPE=dump.
- MY_ART needs to be set up as the path to point to the ART_data_analyzer package to use all features of the package including the package API art_data command line tool.

3. Software functionalities

A list of software functionalities can be checked by prompting the art_data -h in the command line to see a list of options that this ART_data_analyzer package can perform. A detailed guide of using various functionalities of this ART_data_analyzer package can be checked by art_data --desc. Fig. 2 shows the main content of this ART_data_analyzer package on GitHub https://github.com/liangtianumich/ART_data_analyzer.

This ART_data_analyzer package has been developed in a general way to be robust for a wide range of homogeneous and heterogeneous materials, e.g., crystalline materials, interfaces, grain boundaries, amorphous materials. By default, this ART_data_analyzer package will analyze the whole sample. If a user is only interested in a sub-section of the sample, e.g., grain boundary region in a bi-crystal sample, then the user needs to create a file called interested_atom_list.json in user specified data directory, which contains a list of atoms' id in all interested sub-sections. This data directory has been set up by the user in environment.sh as DATA_DIR. If this interested_atom_list.json file has not been created by the user, then interested_atom_list.json file will be automatically created as the list of all atoms in the whole sample. Even though the user saves all atoms of the interested sub-sections into the interested_atom_list.json file, he can always choose to perturb part of the atoms in the file as central atoms, which will be saved into central_atom_list.json file. These two files will be created when using art_data --example command argument. It is worthwhile to mention that we should not use the only subsection of sample configuration, but need to use the whole sample configuration as the input sample for ART. The purpose is to ensure that the subsection configuration of initial minimized whole sample does not substantially deviates from the configuration of the only subsection sample after going through ART's initial minimization step, so that we are truly probing the Potential Energy Landscape (PEL) of the initial whole sample.

3.1. Running ART simulations in parallel

After modifying environment.sh and source it, the following steps are needed to run ARTn simulations in parallel.

First, create and move 4 input files of ARTn bart.sh, conf.lammps, in.lammps, interatomic potential file into directory $ART_INPUT, default is $DATA_DIR.

refconfig file will be automatically created by the package later based on lammps_data file conf.lammps or lammps dump file conf.dump. When sample_type is dump, user need to pay attention if the atomic coordinates in lammps dump file is fractional or absolute, now the default is using fractional coordinates. For bart.sh, users need to have the correct input parameters. The in.lammps is the lammps input script file, which default read lammps_data file conf.lammps. It is worthwhile to note that some potential need specific lammps packages when building lammps. Otherwise, in.lammps file cannot read this potential.

Second, use --example command to generate input setting file for art_data.

As mentioned above, if users are only interested in a subsection (e.g., grain boundary or interface) of the whole sample, he needs to create a file called interested_atom_list.json under $DATA_DIR to specify the interested sub-section. If this file does not pre-exist, –example will automatically generate the file interested_atom_list.json based on all atoms in the whole sample. The usage of this command can be, e.g., art_data --example 2000 > input.json.

Third, run the command: art_data -s input.json --art --run, where input.json is the same input file created in the previous step. This command will automatically set up the input files for running ARTn by running ./mod_bart.sh inside each test dir in parallel. The input.json contains the key num_of_proc, which specify the number of cores to run in parallel. Default setting uses all cores in the local machine.

3.2. Running ART post-processing tasks in a user workflow in parallel

The previously created input file, e.g., input.json, is needed for post-processing of raw ART data to run the following tasks. However, users should manually edit the parameters related to ART post-processing tasks in this input file. A list of these parameters can be checked by art_data --settings-format. The text below shows the major post-processing tasks currently implemented. More task features will be implemented in the future.

(a) Filter events:
User can filter out unsuccessful events defined by the filtering criteria saved in the input file by art_data -s input.json --filter. The three filtering criteria are implemented as: (1) the energy of saddle state must be higher than that of both the initial state and the final state (2) the final state should not be identical to the initial state (3) comparing event pairs to remove redundant events [7].

(b) Perform activation and relaxation energy convergence tests:
After filtering events, users can calculate energy by art_data -s input.json --eng --calc. The user can check the statistical convergence of filtered events by checking the convergence of activation energy and relaxation energy using student t statistical test. The current convergence tests have two modes:

(1) art_data -s input.json --eng --ttest OPTION PATH_1 PATH_2 , where OPTION is 'ind' or 'rel', PATH_1 is the path string to the ART data directory 1 and PATH_2 is the path string to the ART data directory 2.

(2) art_data -s input.json --eng --ttest_kfold OPTION k n will randomly divide the ART data (saved in the "path_to_data_dir" key of input.json) into k folds for n different times and perform t test on each data fold pair to ensure all fold pairs are convergent.

The user needs to test which ttest mode is a good convergence criteria for their systems. For example, a good starting convergence criteria is art_data -s input.json --eng --ttest_kfold ind 2

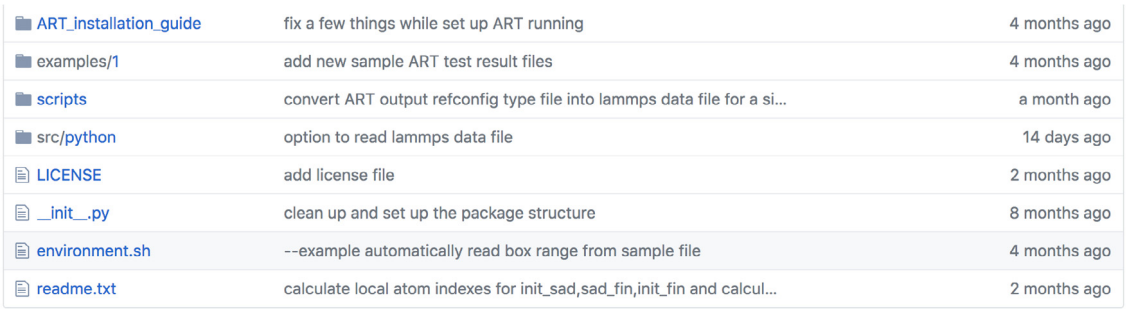

Fig. 2. The main content of the ART_data_analyzer Python package shown in github.

3. If not converged, users can run more ART simulations until convergence by art_data -s input.json --art --run_more N_TESTS, where N_TESTS is the number of tests to be calculated more. This command will update the central_atom_list.json file as well. Users need to update the input file by art_data -s input.json --update_input . Users need to add --re_calc when redo the event filtering and energy calculations, such as art_data -s input.json --filter --re_calc .

(c) Run calculations and visualizations:
art_data -s input.json --strain --calc will invoke the calculations of atomic strains [8] and displacement and automatically plots results for individual events and statistics of all events. art_data -s input.json --strain -v will plot the quantities at event level after strain calculation. art_data -s input.json --strain --stats will plot the statistics of all events after strain calculation.

(d) Data-driven correlation analysis on physical processes
An example is to find the locally involved atoms for all ARTn local events by a machine learning outlier detection algorithm. This local atoms finding algorithm currently support training LinearSVR model [9] between atomic displacement versus atomic shear strains, using a user customized residual threshold to identify outliers as locally involved atoms. This critical residual threshold can be obtained by performing a parameter sweep on the residual threshold with a double slope stopping convergence criteria as the convergence criteria, which is implemented as art_data -s input.json --find_residual. The critical slope could vary for different types of samples, which can be customized by the user in the input file as the key critical_local_atoms_slope. A reasonable value of this critical residual threshold to start with is usually around 0.54.

After the critical residual threshold (e.g., 0.54) is found, we can identify all local atoms' indexes for all filtered events by art_data -s input.json --find_local_index 0.54. This will save local atoms' indexes into a file called local_atoms_index.json.

(e) Find user customized list of atoms
To find the central atom of each initial triggered event, art_data -s input.json --find_central_index will find central atom index and save it into a file called central_atom_index.json for each event.

To find the atoms of initial triggered clusters, art_data -s input.json --find_triggered_cluster_atoms_index will find initial triggered atoms' indexes and save them into a file called initial_cluster_atoms_index.json for each event.

To find the maximum displaced atom, art_data -s input.json --find_max_disp_index will find the index of maximum displaced atom during the initial to saddle process and save them into a file called max_disp_atom_index.json for each event.

In addition to the argument above, users can also customize the list of atoms by specifying the key atom_list in input file input.json. For example, if atom_list is None or 'all', calculations will operate on all atoms in the sample; if atom_list is [2,3,5,8], calculations will operate on atoms whose indexes is 2,3,5,8

(f) Run calculations for interested list of atoms:

(1) atomic strain and atomic displacement:
art_data -s input.json --local --strain --calc will invoke the calculations of atomic strains and displacements for only identified local atoms of all filtered events. Users can replace --local with --central if they are interested in the strains and displacements of the central atom. Similarly, users use --initial for initial triggered cluster atoms, --max_disp for the maximum displaced atom.

(2) voronoi index calculation:
art_data -s input.json --local --voro --calc will read local_atoms_index.json and calculate the voronoi indexes for these local atoms of all filtered events. Voronoi index calculations are performed by using the pyvoro python package to provide an interface to voro++ C++ library developed by Chris Rycroft [10].

(3) voronoi index classification
The voronoi index classifications [11,12] and visualizations are done by art_data -s input.json --voro --classify, which will classify voronoi indexes and plot the classification results for all filtered events; calculate and plot the dynamic transition probability matrix.

4. Tenary ZrCuAl metallic glass example demo

Fig. 3 demonstrates some results of using this ART_data_analyzer package to study the kinetics and dynamic evolution of the Zr46Cu46Al8 metallic glass (MG) sample prepared by LAMMPS molecular dynamics package [13]. The addition of Al leads to more Cu and Al-centered full icosahedral clusters and more icosahedral medium-range orders in the ternary metallic glass, which resembles nanostructured composites [14,15]. The increase of full icosahedral clusters and the enhancement of the atomic packing density are responsible for the higher activation energy of ZrCuAl metallic glasses (mean 1.06 eV, higher than calculated 1.02 eV of Cu50Zr50 prepared with the same cooling rate) [16]. The shape of relaxation energy distribution spectrum is very similar to those of Cu50Zr50 at various cooling rates, implying that the saddle states of various MGs is indeed resembling the melting liquid state of the MGs, decoupling the activation and relaxation processes [17]. Based on Ref [18], Al addition also increases the resistance to the initiation of plastic flow and the propensity for strain localization, which generally might lead to a smaller ductility .

5. Impact and conclusions

We have developed a very user-friendly, parallelized, automated scalable Python software package to study the kinetics and dynamic evolution for a broad range of material classes, such as crystalline materials and their defects (e.g., grain boundaries and interfaces) and amorphous materials.

Its impact and applications lie in:

(1) calculate the distributions of activation energy barrier and relaxation energy

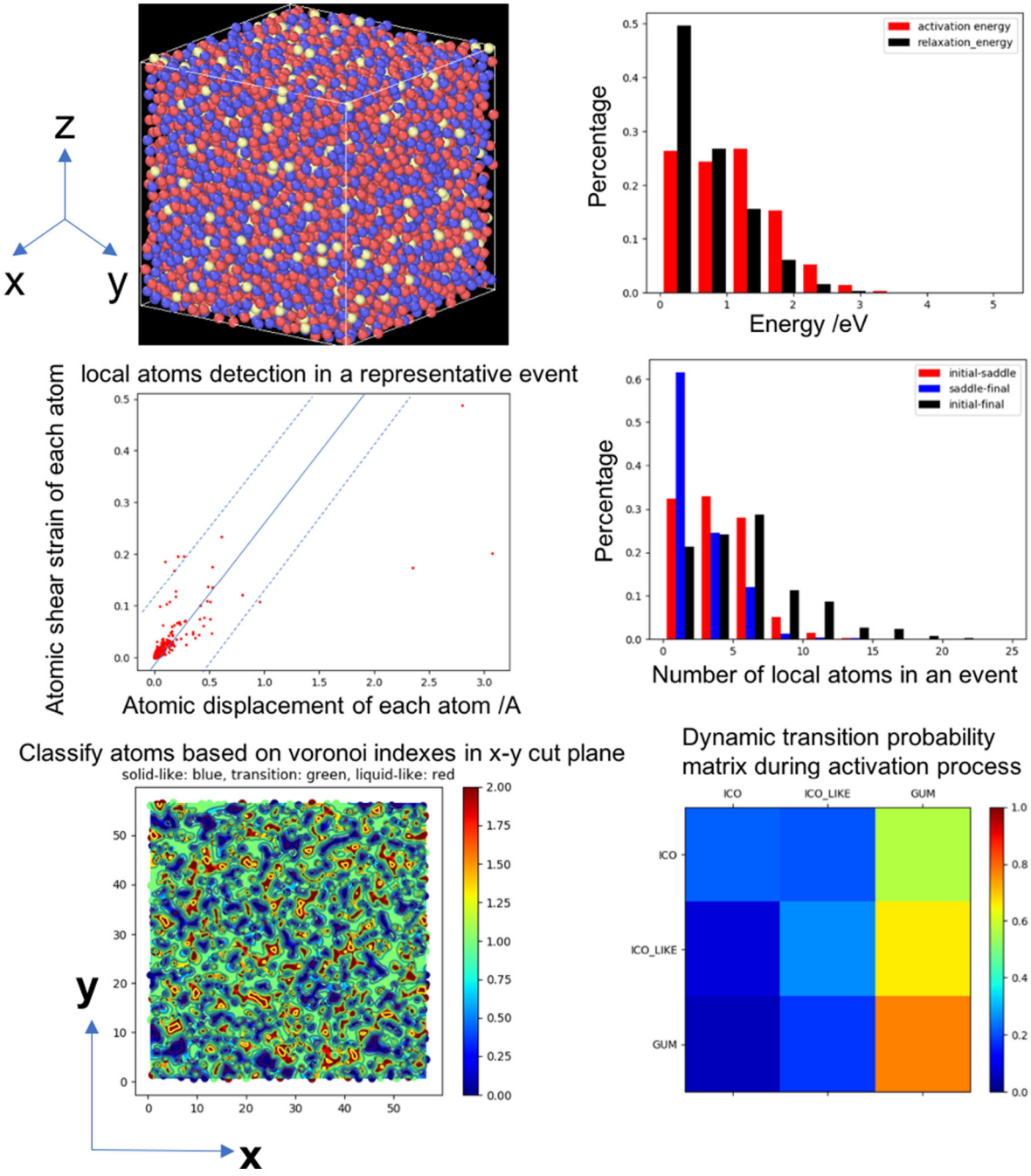

Fig. 3. Some demo results of using this ART_data_analyzer package on the Zr46Cu46Al8 metallic glass sample prepared under 10E10 cooling rate. Top left image shows the molecular dynamics (MD) sample of Zr46Cu46Al8 metallic glass. Top right image shows the activation energy and relaxation energy distribution spectrums for the MD sample. Middle left image shows atomic shear strain versus atomic displacement for each atom in a single representative event. The solid line and dashed lines demonstrated using the LinearSVM machine learning method to find locally involved atoms in the representative event. Middle right shows the statistical distribution of the number of locally involved atoms for all events. Bottom left shows a spatial map of voronoi index in x–y plane cutting through half way of z axis. The voronoi indexes are classified into solid (blue), transition (green) and liquid (red) based on ref [11,12]. The bottom right shows the probability matrix for dynamic transition between various type of atoms. ICO stands for atoms with local icosahedral ordering (solid type), ICO-LIKE stands for atoms with local icosahedral-like ordering (transition type), GUM stands for atoms with local disorder (liquid type) . (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

(2) correlating local structural characteristics with local properties from large amount of event data to derive dynamical evolution of local structure–properties relation.

Future versions of the Python software package will integrate with other software packages, such as LAMMPS, Ovito and pymatgen [19].

Acknowledgments

LT and LL acknowledge the financial support by the U.S. Department of Energy, Office of Science, Basic Energy Sciences (BES), USA, by Award no. DE-SC0016164. LT and LL appreciate the valuable discussions with Dr Yue Fan at University of Michigan, Ann Arbor.

References

[1] Andreas H. Exploring the potential energy landscape of glass-forming systems: from inherent structures via metabasins to macroscopic transport. J Phys: Condens Matter 2008;20(37):373101.

[2] Barkema GT, Mousseau N. Event-based relaxation of continuous disordered systems. Phys Rev Lett 1996;77(21):4358–61.

[3] Malek R, Mousseau N. Dynamics of lennard-jones clusters: a characterization of the activation-relaxation technique. Phys Rev E 2000;62(6):7723–8.

[4] Machado-Charry E, Beland LK, Caliste D, Genovese L, Deutsch T, Mousseau N, Pochet P. Optimized energy landscape exploration using the ab initio based activation-relaxation technique. J Chem Phys 2011;135(3):034102.

[5] Athènes M, Marinica M-C, Jourdan T. Estimating time-correlation functions by sampling and unbiasing dynamically activated events. J Chem Phys 2012;137(19):194107.

[6] Alexander S. Visualization and analysis of atomistic simulation data with ovito–the open visualization tool. Modelling Simulation Mater Sci Eng 2010;18(1):015012.

[7] Fan Y, Iwashita T, Egami T. How thermally activated deformation starts in metallic glass. Nature Commun 2014;5:5083.

[8] Li J, Shimizu F. Least-square atomic strain. 2005, http://li.mit.edu/A/Graphics/A/annotate_atomic_strain/Doc/main.pdf.

[9] Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-Learn: Machine Learning in Python. 2011.

[10] Rycroft CH. Voro++: a three-dimensional voronoi cell library in c++. Chaos 2009;19(4):041111.

[11] Ma E. Tuning order in disorder. Nature Mater 2015;14:547.

[12] Wang B, Luo L, Guo E, Su Y, Wang M, Ritchie RO, Dong F, Wang L, Guo J, Fu H. Nanometer-scale gradient atomic packing structure surrounding soft spots in metallic glasses. npj Comput. Mater. 2018;4(1):41.

[13] Plimpton S. Fast parallel algorithms for short-range molecular dynamics. J Comput Phys 1995;117(1):1–19.

[14] Tian L, Li L. A review on the strengthening of nanostructured materials. Int J Curr Eng Technol 2018;8(2):236–49.

[15] Khoddam S, Tian L, Sapanathan T, Hodgson PD, Zarei-Hanzaki A. Latest developments in modeling and characterization of joining metal based hybrid materials. Adv Energy Mater 2018;20(9):1800048.

[16] Li F, Zhang H, Liu X, Dong Y, Yu C, Lu Z. Effects of al addition on atomic structure of cu-zr metallic glass. J Appl Phys 2018;123(5):055101.

[17] Fan Y, Iwashita T, Egami T. Energy landscape-driven non-equilibrium evolution of inherent structure in disordered material. Nature Commun 2017;8:15417.

[18] Cheng YQ, Cao AJ, Sheng HW, Ma E. Local order influences initiation of plastic flow in metallic glass: effects of alloy composition and sample cooling history. Acta Mater 2008;56(18):5263–75.

[19] Ong SP, Richards WD, Jain A, Hautier G, Kocher M, Cholia S, Gunter D, Chevrier VL, Persson KA, Ceder G. Python materials genomics (pymatgen): A robust, open-source python library for materials analysis. Comput Mater Sci 2013;68:314–9.