Materials Genome Software Framework: Scalable Parallel Simulation, Virtual Reality Visualization and Machine Learning

Ankit Mishra[a], Nitish Baradwaj[a], Lindsay Bassman[a], Brandon K. Horton[a], Subodh Tiwari[a], Sungwook Hong[a], Aravind Krishnamoorthy[a], Erick Moen[b], Pankaj Rajak[c], Rajiv Kalia[a], Aiichiro Nakano[a], Ken-ichi Nomura[a], Fuyuki Shimojo[d], Priya Vashishta[a]

[a] MAterials Genome Innovation for Computational Software (MAGICS), Collaboratory of Advanced Computing and Simulations, Department of Computer Science, Department of Physics & Astronomy, Department of Chemical Engineering & Materials Science, Department of Biological Sciences, University of Southern California, Los Angeles, CA 90089-0242, USA

[b] Broad Center for Biological Sciences, California Institute of Technology, Pasadena, CA 91125, USA

[c] Argonne Leadership Computing Facility, Argonne National Laboratory, Chicago, IL 60439, USA

[d] Department of Physics, Kumamoto University, Kumamoto 860-8555, Japan

(ankitmis, baradwaj, bassman, bkhorton, sctiwari, sungwooh, kris658, rkalia, anakano, knomura, priyav@usc.edu, emoen@caltech.edu, prajak@anl.gov, shimojo@kumamoto-u.ac.jp

***Abstract*—With the advent of the national materials genome initiative in the U.S., software tools and environments for scientific computing have become imperative for computational design, synthesis and characterizing new materials. In the U.S. Department of Energy center called MAterials Genome Innovation for Computational Software (MAGICS), we have developed open-source high performance computing software frameworks for (1) scalable quantum and reactive molecular dynamics simulations; (2) virtual-reality visualization of simulations; and (3) machine-learning tools for big data analytics of simulation data. These frameworks have been used in a series of comprehensive, three-day workshops to train 106 people from 55 universities. Moreover, the tutorial modules and open-source software are available at the MAGICS software portal, for modeling, simulation, and visualization for materials studies.**

Keywords–materials genome software; high performance computing; quantum molecular dynamics; reactive molecular dynamics; virtual reality; machine learning; .

I. INTRODUCTION

Advances in computational power and materials modelling have enabled fundamental understanding of various processes and properties at the molecular level [1]. First principles-informed models can successfully predict a wide range of thermomechanical, electromagnetic and optical properties of materials [2, 3]. This forms the basis of the U.S. materials genome initiative (MGI) that aims to greatly accelerate the design of new materials, guided by computational design, synthesis and characterization [4].

Central to MGI are predictive atomistic simulations that are derived from first principles. These include quantum molecular dynamics (QMD) [5] and reactive molecular dynamics (RMD) [6] simulations. The ever increasing computational power of parallel computers, combined with the availability of popular software such as LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) [7] and VASP (Vienna Ab initio Simulation Package) [8], provides a versatile framework for performing various computations for MGI on parallel computers. However, the interdisciplinary nature of this field makes it often difficult for a student or scientist to learn necessary concepts and available methods for solving a problem at hand. Hence, an organized and structured presentation of essential topics and methods has become very important.

In the U.S. Department of Energy-supported centre called Materials Genome Innovation for Computational Software (MAGICS), we have developed a suite of tutorials and associated open-source software to train students for key components of MG research (https://magics.usc.edu/). The MAGICS software courseware is aimed at giving a researcher broad understanding of materials simulations using different software. The tutorial modules are built on our QMD and RMD simulation engines, respectively named QXMD and RXMD, which are scalable from desktop to current multi-petaflop/s [9] and future exaflop/s [10] parallel supercomputers. On top of these simulation engines, the tutorials teach plug-ins for the prediction of materials properties such as thermal conductivity. Furthermore, students are trained in on-the-fly visualization of simulations in commodity virtual-reality (VR) platforms, as well as machine-learning (ML) tools for big data analytics of simulation data. These tutorial modules are designed to concisely cover the necessary theory underlying the simulation methods as well as to provide effective hands-on exercises. The whole contents can be covered in 22 hours of lectures and hands-on sessions, including the setting up of a computational environment. This provides an end-to-end experience starting from software installation to running and analyzing simulations. The key contents are carefully split into three sessions each of roughly six hours. We have tested our framework of software distribution and its effectiveness in teaching essential concepts and methods in three MAGICS software workshops over the past one-and-half years. We have successfully taught 106 people from 55 different universities in the U.S. (Fig. 1). Our participants have diverse backgrounds ranging from undergraduate, graduate, post graduate and young faculties. We distribute the necessary plug-ins and software required for performing these simulations to allow the participant to use them for solving various scientific problems. This allows creation of a vibrant community of users who are not only theoretically sound but also adept at using these software to their research.

The MAGICS software courseware is modular, and its self-contained modules can be used as programming assignments in a classroom setting. We have used subsets of the modules in graduate courses at the University of Southern California (USC), including CSCI 699 (Extreme-scale quantum simulations) in computer science and MASC 575 (Basics of atomistic simulation of materials) in materials science. The tutorial modules include lecture slides on basic concepts and software overview, Python notebooks on specific procedures and comprehensive readme files of the software. The courseware is available at the MAGICS software workshop Web site (https://magics.usc.edu/workshop/courseware), while the associated open-source software is available at the MAGICS software portal (https://magics.usc.edu/software/), making them widely usable in various workshop and classroom settings.

II. MAGICS Software Tutorial Modules

This section describes individual software and plug-in that are covered by MAGICS software tutorials.

A. *Quantum Dynamics Software with Nonadiabatic Extension*

QXMD is a scalable parallel program for Quantum Molecular Dynamics (QMD) simulations with various eXtensions such as nonadiabatic dynamics. QMD is a widely

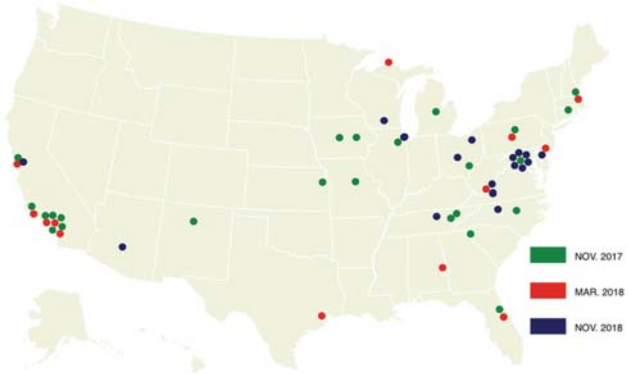


Fig. 1. Distribution of the 106 workshop participants from 55 different universities distributed throughout the U.S

used simulation method to study dynamic behaviors of materials as shown in Fig. 2 [5]. QMD follows the trajectories of all atoms, while computing interatomic forces quantum mechanically in the framework of density functional theory (DFT) [11]. In particular, nonadiabatic QMD (NAQMD) describes electronic excitations and transitions between excited electronic states assisted by atomic motions, thereby describing photo-excitation dynamics involving electrons and nuclei [12, 13].

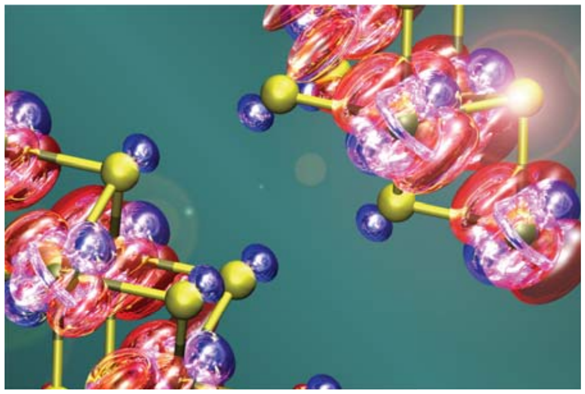


Fig. 2. Computation of charge distribution using QXMD software [14, 15].

QXMD is a software platform for developing new computational techniques to make QMD simulations metascalable (*i.e.*, "design once, scale on emerging computing [10] architectures") [9, 16, 17]. The QXMD simulation engine is parallelized based on hybrid space-band decomposition [16]. The program uses the message passing interface (MPI) library [18] to perform internode communication. Also required is fast Fourier transform (FFT) library, such as FFTW [19], to perform computations involving plane-wave basis.

The MAGICS tutorial uses elementary features of the QXMD software to train students and researchers on the fundamental basics of QMD and NAQMD simulations. After basic lectures, students are given access to QXMD GitHub repository (https://github.com/USCCACS/QXMD). In addition to a detailed readme file to describe the installation and execution of the software, the repository contains an example folder containing four tutorial modules: (1) structural optimization of a water molecule; (2) adiabatic QMD of water, where the electrons stay in the ground state; (3) NAQMD of photo-excited electron-hole pairs in an atomically-thin layered material [14]; and (4) multi-scale shock theory (MSST) [20] to simulate shock-front dynamics [21].

The QXMD tutorial module has also been used in one-semester advanced graduate course at USC in Spring 2018: CSCI 699, "Extreme-scale quantum simulations" (http://cacs.usc.edu/education/cs699.html). In this course, lectures provided fundamental knowledge in order to: (1) reduce the intractable quantum many-body problem to lower-complexity problems, while retaining the essential physics; (2) design scalable parallel algorithms for linear-scaling QMD simulations; (3) and develop metascalable QMD software on current and future computer architectures. In this classroom setting, the QXMD tutorial modules were used as programming assignments to gain hands-on experience on QMD simulations.

B. *Reactive Molecular Dynamics Software*

RXMD is a scalable parallel software for reactive molecular dynamics (RMD) simulations (Fig. 3) based on first principles-informed reactive force-fields (ReaxFF) [6, 22].

RMD follows the time evolution of atomic trajectories, whereas ReaxFF describes chemical bond breakage and formation based on a reactive bond-order concept and charge transfer based on a charge-equilibration (QEq) scheme [23-25].

Computations in RXMD are parallelized using spatial decomposition, where the simulated system is decomposed into spatially localized subsystems and each processor is assigned computations associated with one subsystem [26]. Message passing is used to exchange necessary data for the computations utilizing the MPI library. Specifically, before computing the forces on atoms in a subsystem, atomic positions within the interaction cutoff radius within the boundaries of the neighboring subsystems are cached from the corresponding processors. After updating the atomic positions according to a time stepping procedure, some atoms may have moved out of its subsystem. These moved-out atoms are migrated to the proper neighbor processors. The RXMD program is written in Fortran 90.

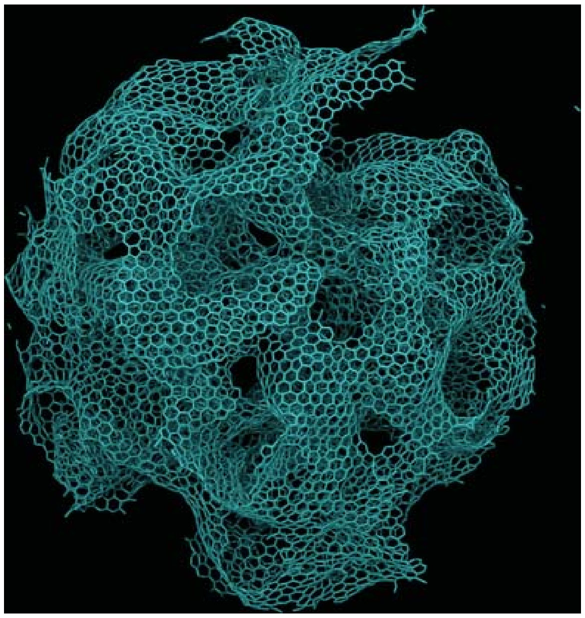


Fig. 3. Nano-porous meta-carbon simulated using RXMD [27].

RXMD supports a new polarizable charge-equilibration (PQEq) scheme and the resulting polarizable reactive force-field (ReaxPQ) method [28], as well as its extension, ReaxPQ+, to describe dielectric responses more accurately [29]. In PQEq, each atom consists of a positively-charged core and a negatively-charged shell. Accordingly, PQEq is implemented using a different parallelization scheme compared to the original QEq. Message passing is now required for the positions of the charge shells, which are treated as additional atomic attributes. Namely, they are cached from neighbor processors along with atomic positions and charges in order to compute interatomic forces. In addition, shell positions are migrated to appropriate processors whenever the corresponding atoms move across spatial-subsystem boundaries.

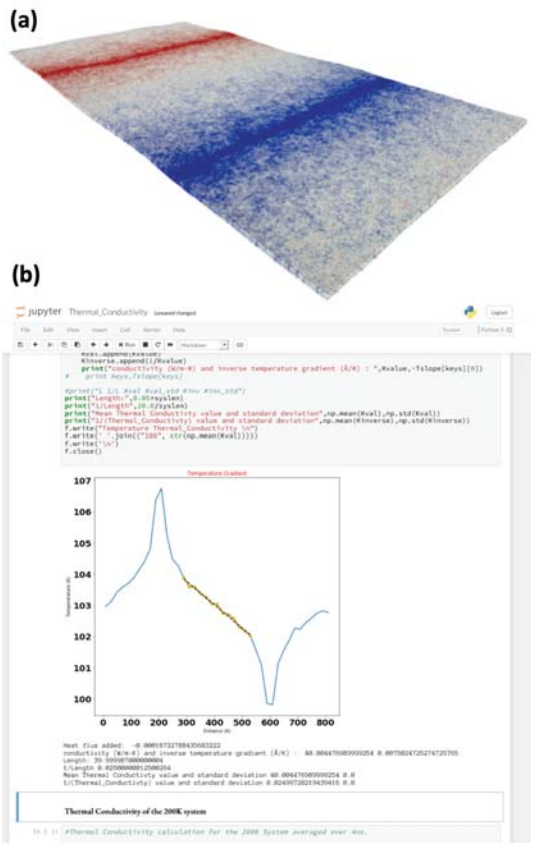
For large granularity (the number of atoms per spatial subsystem, $N/D > 10^2$), simple spatial decomposition (*i.e.*, each processor is responsible for the computation of the forces on the atoms within its subsystem) suffices, whereas for finer granularity ($N/D \sim 1$), neutral-territory or other hybrid decomposition schemes can be incorporated into the framework [30-32]. Our parallelization framework also includes load-balancing capability. For irregular data structures, the number of atoms assigned to each processor varies significantly, and this load imbalance degrades the parallel efficiency. Load balancing can be stated as an optimization problem as well. We minimize the load-imbalance cost as well the size and the number of messages. Our topology-preserving spatial decomposition allows message passing to be performed in a structured way in only 6 steps, so that the number of messages is minimized. To minimize the load imbalance cost and the size of messages, we have developed a computational-space decomposition scheme [33]. The main idea is that the computational space shrinks in a region with high workload density, so that the workload is uniformly distributed. The sum of load-imbalance and communication costs is minimized as a functional of the computational space using simulated annealing. We have found that wavelets allow compact representation of curved partition boundaries and thus speed up the optimization procedure [34].

Within each compute node, we introduce an additional layer of shared-memory parallelism using the Open Multi-Processing (OpenMP) application programming interface [35]. OpenMP allows RXMD to take advantage of the simultaneous multithreading support provided by modern processors, such as the PowerPC A2 architecture of IBM Blue Gene/Q and the second-generation Intel Xeon Phi, to achieve better utilization of the computing capacity within each core [29]. Multithreading the most computationally expensive bond-order and force computations within RXMD has greatly accelerated these calculations and served to reduce overall time-to-solution (T2S). A secondary benefit of multithreading is that it allows MPI processing elements to be exchanged for local threads, thereby reducing the total number of processing elements in the MPI job and similarly reducing the communication and atom-caching overheads at large scale.

RXMD software has been used in the MAGICS software workshops to train students and researchers on (1) basics of RMD simulations, and (2) massively-scalable capacity of the RXMD code. Students are given access to RXMD GitHub repository (https://github.com/USCCACS/RXMD). The repository contains detailed description of software usage through a readme file, as well as an example folder constituting of following tutorial modules: (1) sulfidation of MoO_3 to form MoS_2 [36]; (2) SiC oxidation to form SiO_2 (Fig. 3) [27].

C. ThermoPlugins – Software Plugins for LAMMPS to Calculate Thermal Properties of Materials

ThermoPlugins are a set of Python- and C-based utilities that extend the functionality of the popular LAMMPS molecular dynamics (MD) program [7] to enable the calculation of more accurate thermal conductivity and vibrational properties. ThermoPlugins consists of two modules, *caltc* and *caldos*. The first module provides functionality for performing length-scaling and temperature scaling simulations to compute the intrinsic thermal conductivity of materials by overcoming size-effects inherent in MD simulations of thermal transport (Fig. 4). The second module implements functions for the calculation of system-level quantum corrections to the lattice specific heat of the simulated system, which corrects for the purely classical specific heat intrinsic to all MD simulations.



```
    Kval.append(Kvalue)
    Kinverse.append(1/Kvalue)
    print("conductivity (W/m-K) and inverse temperature gradient (Å/K) : ",Kvalue,-Tslope[keys][0])
#    print keys,Tslope[keys]

#print("L 1/L Kval Kval_std Kinv Kinv_std")
print("Length:",0.05*syslen)
print("1/Length",20.0/syslen)
print("Mean Thermal Conductivty value and standard deviation",np.mean(Kval),np.std(Kval))
print("1/(Thermal_Conductivty) value and standard deviation",np.mean(Kinverse),np.std(Kinverse))
f.write("Temperature Thermal_Conductivity \n")
f.write(' '.join(["100", str(np.mean(Kval))]))
f.write('\n')
f.close()
```

```
Heat flux added:  -0.000187327884356832222
conductivity (W/m-K) and inverse temperature gradient (Å/K) :  40.0044769899999254 0.00750247252747252765
Length: 39.9999870000000004
1/Length 0.0250000000812500264
Mean Thermal Conductivty value and standard deviation 40.0044769899999254 0.0
1/(Thermal_Conductivty) value and standard deviation 0.0249972802194394l8 0.0
```

Thermal Conductivity of the 200K system

```
#Thermal Conductivity calculation for the 200K System averaged over 4ns.
```

Fig. 4. (a) Thermal-transport calculation using ThermoPlugins. The colored blue region shows the cold strip on the sheet while the red region shows the hot region. This enables creation of thermal gradient necessary for computation of transport properties. (b) Snapshot of Python notebook.

Both modules implement strategies for higher sampling of thermal profiles and dynamical correlations leading to a more accurate estimation of thermal conductivity and vibrational properties. The utilities also provide an estimation of systematic errors in estimation of thermal transport and are useful as a training tool to understand the effect of these corrections on calculated thermal conductivity the size of the simulated system leading to computed thermal conductivity values significantly different than experimental values. Further, classical MD simulations ignore quantum-mechanical effects related to the quantization of lattice vibrational energy, which is responsible for sub-classical specific heat and correspondingly suppressed thermal conductivity at low temperatures**[37].**

The ThermoPlugins tutorial module provides hands-on training on how research-level materials simulations are performed using a specific example of thermal conductivity. To make complex theoretical concepts and computational procedures accessible, we have developed a comprehensive Python notebook. The notebook explains basic concepts and equations, as well as every step from performing simulations and plotting the results.

D. Game Engine Assisted Platform for Scientific Computing in Virtual Reality

The Game-Engine-Assisted Research platform for Scientific computing (GEARS) is a collaborative visualization framework developed at the MAGICS center to perform simulations and on-the-fly data exploration in VR environments (Fig. 5) [38]. This hardware-agnostic framework accommodates multiple programming languages and game engines in addition to supporting integration with a widely-used materials simulation engine, LAMMPS [7]. GEARS also features a novel data exploration tool called virtual confocal microscopy, which endows scientific visualization with enhanced functionality.

The GEARS engine provides the capability of interactive data visualization. It makes use of commodity game engines, like Unity and Unreal Engine, to simplify access to VR headsets. The most straightforward application of VR to data visualization is interactive viewing of pre-computed results for data exploration. To realize this feature, GEARS employs a simple workflow, in which external 3D modelers, such as VMD and Blender, are used to create a 3D object that can be added to the scene along with the appropriate script (for example, LeapRTS.cs for the Leap Motion controller [39]).
This aspect of GEARS allows users a quick, straightforward outlet for immediate visualization of snapshots of data from materials simulations or molecular structures. Though the current interactive visualization mode supports single-frame data, we plan to expand this feature by creating multiple scenes containing different simulation time steps for more dynamical data exploration.

The engine also supports real-time visualization by plugging into an external simulator. GEARS also takes advantage of the programing capabilities provided by game engines, such as C# and JavaScript supported by Unity and C++ for Unreal Engine, to facilitate reuse of existing simulation codes. This mode of GEARS is suitable to explore simulation results in real-time, rendered entirely within the game engine. To realize this real-time rendering, it is critically important to design an efficient data-bridging method between the game engine and user-developed simulation program.

GEARS employ two approaches, called *Run-when-Ready* and *Render-when-Ready*, depending on the size of the data and the complexity of the simulation engine. *Run-when-Ready* calls the simulation engine upon a frame update to advance the state of the simulation (for example particle positions) of the frame by one timestep. *Render-when-Ready* makes use of the multi-threading optimization technique to offload the simulation engine computation onto a new thread while the main thread is only responsible for handing the render state of the game engine. In this approach, when the simulation engine finishes

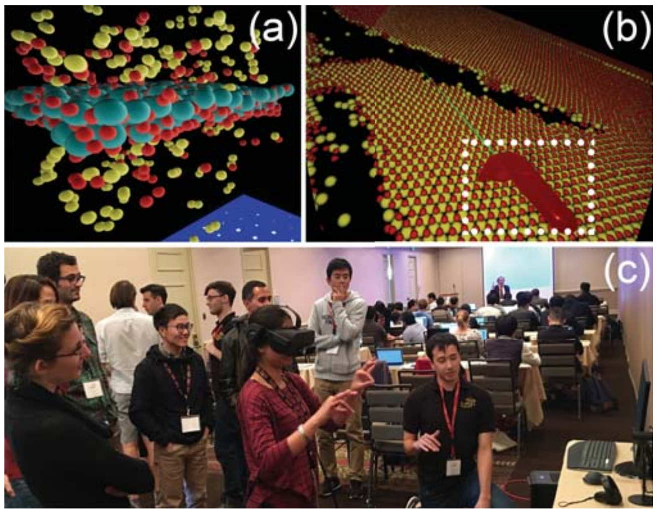


Fig. 5. (a) An example of visualizing a chemical vapor deposition (CVD) simulation through GEARS' LAMMPS integration. (b) Simulating and exploring data related to a molybdenum disulfide (MoS_2) fracture via the LAMMPS integration. Using the controllers indicated by white lines, researchers can freely navigate through the structure, pause and resume the animation, and even highlight regions of interest with the provided laser attachment (shown in green). (c) MAGICS software workshop participant interacting with GEARS.

one timestep computation, the main thread either updates the frame state or stores the simulation states for rendering later while the worker thread continues to produce new states as background. Using this technique as well as other optimizations, like impostors and graphics processing unit (GPU) instancing [40], we have demonstrated real-time simulation sizes reaching up to 500,000 particles. To minimize the amount of coding necessary for GEARS users, we provide two demos — Lennard-Jones MD [41] and electron-transfer simulation by kinetic Monte Carlo [42] — implementing the *Run-when-Ready* and *Render-when-Ready* approaches in the GEARS GitHub repository (https://github.com/USCCACS/GEARS). This is provided as Demo3 of UnityGEARS on the GEARS repository. In an effort to make our immersive scientific computing suite accessible to a broad research community, we have also interfaced GEARS with one of the most widely used MD simulation engine called LAMMPS [7]. LAMMPS was developed at Sandia National Laboratory and supports a variety of interatomic potentials, statistical ensembles, and flexible simulation setups. The LAMMPS interface enables users to visualize their MD simulations in VR environment without any coding. A "How-to" for setting up the LAMMPS integration environment and an example demo are documented in detail in the LammpsCompliation section of UnrealGEARS on the GEARS repository. Examples of the LAMMPS integration are shown in Fig. 5 (a) and (b). The GEARS software was presented in the MAGICS software workshop, where participants were able to interact with VR headsets to visualize any simulation immersively in the system.

E. Machine Learning Software Code – Classification of defects in materials with damage

Machine learning (ML) tutorial module is designed to classify defects in materials with damage. The module uses code written in C language and iPython [43] notebook interactive shell to help participants learn simple ML concepts. The module uses support vector machine (SVM), which is a classic linear classifier, in order to identify defects in nickel (Ni) crystal. While simple, this tutorial contains the entire end-to-end procedures that are common to most ML tasks, using an example familiar to most materials scientists.

The interactive shell creates an abstraction that enables understanding of the core concepts without getting into details of the implementation. The initial part of the module involves creation of the training data, feature-set generation and labelling of atoms for efficient understanding of the underlying material's geometry. The feature-generation code takes the initial geometry as input and generates a feature set for each atom's local environment. Feature representation for each atom consists of 17 statistical properties: number of nearest-neighbor (NN) atoms, average distance of NN, minimum distance of NN, maximum distance of NN, average distance between NN, minimum distance between NN, maximum distance between NN, NN's average numbers of neighbors, NN's neighbor's average distance, NN's neighbor's minimum distance, NN's neighbor's maximum distance, number of neighbors between distance 3-4, 4-5, 5-6 Å, average distance of neighbors between 3-4, 4-5, 5-6 Å. This model is used to predict defects in pure Ni crystal, before and after indentation [44], to classify defects in the system. This labelled representation corresponding to every atom helps classify the atomic environment with great accuracy. The trained model has an accuracy of 96.83% on training data set and 98.96% in test data set. To visualize the model's prediction, we provide an interface by using molecular visualization software OVITO (Fig. 6) [45].

Students are given access to ML GitHub repository (https://github.com/USCCACS/ML-defect-analysis). The repository contains all the codes required for carrying out the analysis. Detailed instructions are also included, which guide the user on downloading, compiling and running the module.

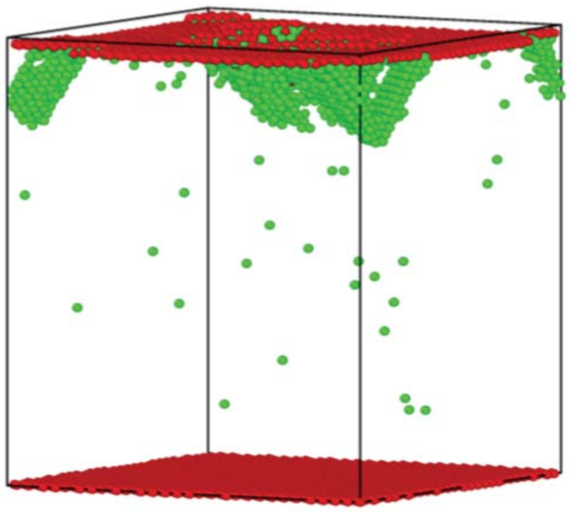


Fig. 6. Visualization of defects labelled by machine learning based on a linear-classifier model. The atoms shown in green have defects, while those shown in red are surface atoms.

III. Conclusion

We have developed a suite of modular tutorials and associated open-source software to train students for key components of computational materials research, *i.e.*, QMD and RMD simulations on parallel computers, thermal-conductivity plug-in, VR visualization and ML tools for data analysis. The tutorial modules and open-source software are available at the MAGICS software portal, making them widely usable in various workshop and classroom settings.

Acknowledgments

This work was supported as part of the Computational Materials Sciences Program funded by the U.S. Department of Energy, Office of Science, \Basic Energy Sciences, under Award Number DE-SC00014607. MAGICS Workshops were performed using computing facilities at the Center for High Performance Computing of the University of Southern California.

References

[1] C. R. A. Catlow and G. D. Price, "Computer modelling of solid-state inorganic materials," *Nature,* vol. 347, p. 243, 1990.
[2] E. A. Carter, "Challenges in modeling materials properties without experimental input," *Science,* vol. 321, p. 800, 2008.
[3] S. Yip and M. P. Short, "Multiscale materials modelling at the mesoscale," *Nature Materials,* vol. 12, p. 774, 2013.
[4] A. Jain, K. A. Persson, and G. Ceder, "Research update: the materials genome initiative: data sharing and the impact of collaborative ab initio databases," *APL Materials,* vol. 4, p. 053102, 2016.
[5] M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos, "Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients," *Reviews of Modern Physics,* vol. 64, pp. 1045-1097, 1992.
[6] T. P. Senftle, S. Hong, M. M. Islam, S. B. Kylasa, Y. Zheng, Y. K. Shin*, et al.*, "The ReaxFF reactive force-field: development, applications and future directions," *npj Computational Materials,* vol. 2, p. 15011, 2016.
[7] S. Plimpton, "Fast parallel algorithms for short-range molecular dynamics," *Journal of Computational Physics,* vol. 117, pp. 1-19, 1995.
[8] G. Kresse and J. Furthmuller, "Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set," *Physical Review B,* vol. 54, pp. 11169-11186, 1996.
[9] N. A. Romero, A. Nakano, K. Riley, F. Shimojo, R. K. Kalia, P. Vashishta*, et al.*, "Quantum molecular dynamics in the post-petaflops era," *IEEE Computer,* vol. 48, pp. 33-41, 2015.
[10] R. F. Service, "Design for US exascale computer takes shape," *Science,* vol. 359, pp. 617-618, 2018.
[11] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Physical Review,* vol. 136, pp. B864-B871, 1964.
[12] C. F. Craig, W. R. Duncan, and O. V. Prezhdo, "Trajectory surface hopping in the time-dependent Kohn-Sham approach for electron-nuclear dynamics," *Physical Review Letters,* vol. 95, p. 163001, 2005.
[13] F. Shimojo, S. Ohmura, W. Mou, R. K. Kalia, A. Nakano, and P. Vashishta, "Large nonadiabatic quantum molecular dynamics simulations on parallel computers," *Computer Physics Communications,* vol. 184, pp. 1-8, 2013.
[14] M. F. Lin, V. Kochat, A. Krishnamoorthy, L. Bassman, C. Weninger, Q. Zheng*, et al.*, "Ultrafast non-radiative dynamics of atomically thin $MoSe_2$," *Nature Communications,* vol. 8, p. 1745, 2017.
[15] L. Bassman, A. Krishnamoorthy, H. Kumazoe, M. Misawa, F. Shimojo, R. K. Kalia*, et al.*, "Electronic origin of optically-induced sub-picosecond lattice dynamics in $MoSe_2$ monolayer," *Nano Letters,* vol. 18, pp. 4653-4658, 2018.
[16] F. Shimojo, R. K. Kalia, M. Kunaseth, A. Nakano, K. Nomura, S. Ohmura*, et al.*, "A divide-conquer-recombine algorithmic paradigm for multiscale materials modeling," *Journal of Chemical Physics,* vol. 140, p. 18A529, 2014.
[17] K. Nomura, R. K. Kalia, A. Nakano, P. Vashishta, K. Shimamura, F. Shimojo*, et al.*, "Metascalable quantum molecular dynamics simulations of hydrogen-on-demand," *Proceedings of Supercomputing, SC14,* pp. 661-673, IEEE/ACM, 2014.
[18] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI*, Third ed. Cambridge, MA: MIT Press, 2014.
[19] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proceedings of IEEE,* vol. 93, pp. 216-231, 2005.
[20] E. J. Reed, L. E. Fried, and J. D. Joannopoulos, "A method for tractable dynamical studies of single and double shock compression," *Physical Review Letters,* vol. 90, pp. 235503-235503, 2003.
[21] K. Shimamura, M. Misawa, S. Ohmura, F. Shimojo, R. K. Kalia, A. Nakano*, et al.*, "Crystalline anisotropy of shock-induced phenomena: omni-directional multiscale shock technique," *Applied Physics Letters,* vol. 108, p. 071901, 2016.
[22] A. C. T. van Duin, S. Dasgupta, F. Lorant, and W. A. Goddard, "ReaxFF: a reactive force field for hydrocarbons," *Journal of Physical Chemistry A,* vol. 105, pp. 9396-9409, 2001.
[23] A. K. Rappe and W. A. Goddard, "Charge equilibration for molecular dynamics simulations," *Journal of Physical Chemistry,* vol. 95, pp. 3358-3363, 1991.
[24] A. Nakano, "Parallel multilevel preconditioned conjugate-gradient approach to variable-charge molecular dynamics," *Computer Physics Communications,* vol. 104, pp. 59-69, 1997.
[25] K. Nomura, P. E. Small, R. K. Kalia, A. Nakano, and P. Vashishta, "An extended-Lagrangian scheme for charge equilibration in reactive molecular dynamics simulations," *Computer Physics Communications,* vol. 192, pp. 91-96, 2015.
[26] K. Nomura, R. K. Kalia, A. Nakano, and P. Vashishta, "A scalable parallel algorithm for large-scale reactive force-field molecular dynamics simulations," *Computer Physics Communications,* vol. 178, pp. 73-87, 2008.
[27] K. Nomura, R. K. Kalia, Y. Li, A. Nakano, P. Rajak, C. Sheng, *et al.*, "Nanocarbon synthesis by high-temperature oxidation of nanoparticles," *Scientific Reports,* vol. 6, p. 24109, 2016.

[28] S. Naserifar, D. J. Brooks, W. A. Goddard, and V. Cvicek, "Polarizable charge equilibration model for predicting accurate electrostatic interactions in molecules and solids," *Journal of Chemical Physics,* vol. 146, p. 124117, 2017.

[29] K. Liu, S. Hong, R. K. Kalia, A. Nakano, K. Nomura, P. Rajak*, et al.*, "Shift-collapse acceleration of generalized polarizable reactive molecular dynamics for machine learning-assisted computational synthesis of layered materials," *Proceedings of Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems, ScalA18,* pp. 41-48, IEEE, 2018.

[30] D. E. Shaw, "A fast, scalable method for the parallel evaluation of distance-limited pairwise particle interactions," *Journal of Computational Chemistry,* vol. 26, pp. 1318-1328, 2005.

[31] M. Kunaseth, R. K. Kalia, A. Nakano, K. Nomura, and P. Vashishta, "A scalable parallel algorithm for dynamic range-limited n-tuple computation in many-body molecular dynamics simulation," *Proceedings of Supercomputing, SC13,* ACM/IEEE, 2013.

[32] M. Kunaseth, S. Hannongbua, and A. Nakano, "Shift/collapse on neighbor list (SC-NBL): fast evaluation of dynamic many-body potentials in molecular dynamics simulations," *Computer Physics Communications,* vol. 235, pp. 88-94, 2019.

[33] A. Nakano and T. J. Campbell, "An adaptive curvilinear-coordinate approach to dynamic load balancing of parallel multiresolution molecular dynamics," *Parallel Computing,* vol. 23, pp. 1461-1478, 1997.

[34] A. Nakano, "Multiresolution load balancing in curved space: the wavelet representation," *Concurrency: Practice and Experience,* vol. 11, pp. 343-353, 1999.

[35] B. Chapman, G. Jost, and R. van der Pas, *Using OpenMP*. Cambrige, MA: MIT Press, 2007.

[36] S. Hong, A. Krishnamoorthy, P. Rajak, S. Tiwari, M. Misawa, F. Shimojo*, et al.*, "Computational synthesis of MoS_2 layers by reactive molecular dynamics simulations: initial sulfidation of MoO_3 surfaces," *Nano Letters,* vol. 17, pp. 4866-4872, 2017.

[37] A. Krishnamoorthy*,* P. Rajak, P. Norouzzadeh, D.J. Singh, R.K. Kalia, A. Nakano et al., "Thermal conductivity of MoS2 monolayers from molecular dynamics simulations," *AIP Advances, vol. 9(3). p. 035042,* 2019.

[38] B. K. Horton, R. K. Kalia, E. Moen, A. Nakano, K. Nomura, M. Qian*, et al.*, "Game-engine-assisted research platform for scientific computing (GEARS) in virtual reality," *SoftwareX,* vol. 9, pp. 112-116, 2019.

[39] *Leap Motion Developers*. Available at https://developer.leapmotion.com/.

[40] M. Tarini, P. Cignoni, and C. Montani, "Ambient occlusion and edge cueing to enhance real time molecular visualization," *IEEE Transactions on Visualization and Computer Graphics,* vol. 12, pp. 1237-1244, 2006.

[41] D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed. San Diego, CA: Academic Press, 2001.

[42] H. S. Byun, M. Y. El-Naggar, A. Nakano, and P. Vashishta, "A derivation and scalable implementation of the synchronous parallel kinetic Monte Carlo method for simulating long-time dynamics," *Computer Physics Communications,* vol. 219, pp. 246-254, 2017.

[43] F. Perez and B. E. Granger, "IPython: A System for Interactive Scientific Computing," *Computing in Science & Engineering,* vol. 9, pp. 21-29, 2007.

[44] I. Szlufarska, A. Nakano, and P. Vashishta, "A crossover in the mechanical response of nanocrystalline ceramics," *Science,* vol. 309, pp. 911-914, 2005.

[45] S. Alexander, "Visualization and analysis of atomistic simulation data with OVITO–the Open Visualization Tool," *Modelling and Simulation in Materials Science and Engineering,* vol. 18, p. 015012, 2010.