MASARYK
UNIVERSITY

FACULTY OF SCIENCE

Machine learning-assisted atomistic modeling of amorphous materials

Bachelor's Thesis

TOMÁŠ ROTTENBERG

Brno, Spring 2023

MASARYK UNIVERSITY

FACULTY OF SCIENCE

Machine learning-assisted atomistic modeling of amorphous materials

Bachelor's Thesis

TOMÁŠ ROTTENBERG

Advisor: Mgr. Pavel Ondračka, Ph.D.

Department of Physical Electronics

Brno, Spring 2023


UNIVERSITAS MASARYKIANA BRUNENSIS
SCIENTIA EST POTENTIA
FACULTAS RERUM NATURALIUM

Bibliographic record

Author:	Tomáš Rottenberg Faculty of Science Masaryk University Department of Physical Electronics
Title of Thesis:	Machine learning-assisted atomistic modeling of amorphous materials
Degree Programme:	Nanotechnology
Field of Study:	Physics
Supervisor:	Mgr. Pavel Ondračka, Ph.D.
Academic Year:	2022/2023
Number of Pages:	9 + 26
Keywords:	machine learning, computational physics, atomistic modeling, molecular dynamics, neural networks, DeePMD

Abstract

We explore the methodology of machine learning-assisted molecular dynamics simulations for quantum chemistry and solid state physics and demonstrate that by using the DeePMD method, it is possible to achieve linear scaling with system size while maintaining computational accuracy comparable to that of *ab initio* calculations. To show this, we use the Deep Potential Molecular Dynamics scheme to calculate various material properties, such as relaxation energy, relaxation volume, and bulk modulus, for select atomic structures and equate the precision of our results with their *ab initio* counterparts. The implementation of our approach uses the DeePMD-kit software suite with the LAMMPS molecular dynamics simulator and leverages active learning to train deep potential neural networks.

Acknowledgements

These are the acknowledgements for my thesis, which can span multiple paragraphs.

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Tomáš Rottenberg

Contents

List of Tables

List of Figures

1 Introduction

Molecular dynamics (MD) is a computer simulation method for calculating the physical movements of atoms and molecules. MD simulations work by numerically solving Newton's equations to obtain the trajectories of atoms and molecules. The forces acting on the simulated particles, along with their potential energies, are calculated using interatomic potentials or molecular mechanical force fields. MD is an invaluable tool used in many disciplines, including physics, chemistry, biology, and material science, as it enables running complex experiments virtually, examining their results on atomary scales, which might be unreachable even with costly laboratory equipment.

The accuracy of MD simulations is largely dependent on its model for atomic interactions. There are two prominent approaches to modeling these interactions. The *ab initio* calculations determine the electronic structure of a system from first principles using a quantum mechanical theory. *Ab initio* molecular dynamics (AIMD) is considered to be highly accurate, however, due to treating the electronic degrees of freedom, its computational burden is significant, and its usage is generally limited to smaller systems with hundreads of atoms and time scales of $\pm 100\,\mathrm{ps}$. Applications requiring larger cells and longer simulations are currently accessible only with empirical interatomic potentials, also called force fields (FFs). These models, while having much lower computational requirements, suffer from decreased precision and are not very transferable.

In recent years, there's been a steady surge of breakthrough applications of machine learning (ML) and neural networks. From beating human experts in games like chess and go to almost perfectly solving the problem of protein folding, artificial intelligence seems to be an ideal fit for finding stochastic heuristics and patterns in complicated problems, given a large dataset of learning data. This wave of transition toward machine learning-based algorithms has affected even MD software packages and given rise to a very successful class of machine learning potentials and force fields.

The goal of my thesis is to evaluate these machine learning potentials and demonstrate that they are capable of producing results that are on par with quantum mechanical methods while scaling in a linear

manner with the size of a system. My application uses the DeePMD-kit code, which is an implementation of the Deep Potential Molecular Dynamics (DeePMD) protocol. Furthermore, the LAMMPS Molecular Dynamics Simulator was used to run the machine learning potentials on concrete molecular dynamics ensembles. Last but not least, the OpenMX simulator was used to provide quantum mechanical data for training and evaluating the neural network models.

2 Molecular Dynamics

Molecular dynamics simulations is an umbrella term for a class of computational methods used to model and analyze the physical behavior of systems of atoms and molecules. MD allows one to monitor the full time evolution of a system, allowing for deep examination of the dynamics of atomic-level phenomena that cannot be observed directly. Computer simulations applied to condensed matter systems began their development as early as the 1950s, when two of the pillars of molecular simulation were introduced, namely the Monte Carlo (MC) sampling technique and the molecular dynamics method. In 1964, the first realistic MD simulation was developed by Rahman, who came up with a realistic model of liquid Argon. Rahman used the Lennard–Jones pair-wise additive potential and showed, that MD simulations with smooth potentials were possible.

Around the same time, Verlet proposed a stable numerical integration algorithm that is still very popular in modern MD software. We will discuss the Verlet integration algorithm in further chapters of this thesis. Verlet also invented a time-saving algorithm, the Verlet neighbour list.

A great leap forward in the MD methology happened in 1971, when Rahman and Stillinger published an MD study on modeling a realistic system of liquid water, a system composed of molecules, not just individual atoms. The significant results of their work prompted a multinational group of scientists centred around Berendsen at CECAM to try using MD simulations for examining biomolecules. The first MD simulation of a simple protein was due to Karplus and collaborators, and appeared shortly after, in 1977. In 2013, the Nobel Prize for Chemistry was awarded to Warshel, Levitt, and Karplus for their work on computer simulations in biochemistry, which was built upon the efforts of many researchers who had previously worked on simulating biomolecules.

Another inportant development took place in 1980. In this year, Anderson published a paper that described how to extend MD to enable it to sample the isoenthalpic (constant pressure) ensemble. The standard molecular dynamics algorithm was designed to simulate the behavior of a system of particles at constant energy, or in the mi-

crocanonical ensemble, because the Newton's equations of motion conserve energy. It was not straightforward to modify the MD algorithm to sample systems under different, more experimentally relevant conditions. Andersen's extensions for sampling the isoenthalpic ensemble inspired the question of whether it was possible to use MD to sample the canonical ensemble as well. Nosé, building on Andersen's work, introduced a new variable that linked the kinetic energy of the atoms to the external temperature, resulting in dynamics that sample the desired ensemble. This approach is known as the Nosé–Hoover thermostat, which is often used in a modified form called the Hoover thermostat.

In 1985, Car and Parrinello published a groundbreaking paper in Physical Review Letters that described a method for combining MD with density functional theory (DFT) calculations of electronic structure. This approach eliminated the need for a potential model, as energy, forces, and stress could be calculated directly from the electronic structure. The Car–Parrinello method allowed for the simulation of processes that involve bond formation or breaking and was the first to demonstrate that it is possible to combine finite temperature simulations with ground-state electronic structure calculations. This method also served as a bridge between the simulation community, which typically has a background in statistical mechanics, and the solid-state physics and quantum chemistry communities, which focus on electronic structure calculations at zero temperature.

During the 1980s and 1990s, the use of molecular simulations in condensed matter research became more widespread, due in part to previous successes in this field and also to the increasing availability and power of computers.

Modern MD methodology is frequently used to refine the three-dimensional structures of proteins and other large molecules, to study atomic-level phenomena that cannot be directly observed, such as thin-film growth and ion implantation, and to investigate the physical properties of nanotechnological devices that cannot yet be manufactured. In 2015, for example, MD simulation has been reported for pharmacophore development and drug design.

2.0.1 Classical Methods

The classical MD implementation uses the so-called "ball and sticks" model, where atoms and molecules are treated as soft balls and their bonds are represented by elastic sticks. The laws of classical mechanics define the dynamics of the entire system.

Each particle in an MD simulation has its own position vector $\mathbf{r}_i(t) = (x_i(t), y_i(t), z_i(t))$. A particle usually corresponds to an atom, although it may represent any simulable entity of interest that can be conveniently described by an interaction law. By Newton's second law the motion of each particle must obey the following relation

$$\mathbf{F}_i = m_i \frac{\mathrm{d}^2 \mathbf{r}_i}{\mathrm{d}t^2}, \tag{2.1}$$

where m_i is the mass of i-th particle and $\mathbf{F}_i$ is the force acting upon i-th particle. Interaction laws are usually specified by a potential function $U(\mathbf{r}_1, \ldots, \mathbf{r}_N)$, which represents the potential energy of N interacting particles as a function of their positions. Given the potential, the force acting upon i-th atom is determined by the gradient with respect to particle displacements

$$\mathbf{F}_i = -\nabla_{\mathbf{r}_i} U(\mathbf{r}_1, \ldots, \mathbf{r}_N) = -\left(\frac{\partial U}{\partial x_i}, \frac{\partial U}{\partial y_i}, \frac{\partial U}{\partial z_i} \right). \tag{2.2}$$

Let's briefly talk about the meaning and form of the potential U in MD simulations. Any quantum-chemistry textbook would insist, that in order to appropriately examine a behavior of molecule, we can't just look at its individual atoms. The quantum-mechanical lense reveals, that when atoms bond into molecules, their electron orbitals interact in complex ways, giving rise to non-trivial molecule orbitals. These electronic clouds that span multiple atoms then determine molecule's interactions with other particles. This paints molecules as a very complicated quantum systems, where electrons and nuclei are interacting together in an intricate manner. It turns out, however, that to a very good approximation, known as the Born–Oppenheimer adiabatic approximation and based on the difference in mass between nuclei and electrons, the electronic and nuclear problems can be separated. According to this approximation, we can presume that the electron clouds equilibrate quickly for each instanteneous configuration of the

heavy nuclei. The nuclei then move in the field created by the average electron densities. This allows us to consider the concept of a potential energy surface, which controls the movement of the nuclei without taking explicit account of the electrons. Given the potential energy surface, we may use classical mechanics to follow the dynamics of the nuclei. Rather than solving the quantum-mechanical problem, we can solve a classic-mechanical problem, in which the effect of the electrons on nuclei is expressed by en empirical potential. It can be very challenging to identify a potential function that accurately represents an energy surfaces of a system, but doing so greatly simplifies the computational process. Atomic force field models and the classical MD are based on empirical potentials with a specific functional form, representing the physics and chemistry of the systems of interest. The following equation is an example of such a force field, used in biosystem simulations

$$\begin{aligned} U(\mathbf{r}_1, \ldots, \mathbf{r}_N) = & \sum_{\text{bonds}} \frac{a_i}{2}(l_i - l_{i0})^2 + \sum_{\text{angles}} \frac{b_i}{2}(\theta_i - \theta_{i0})^2 \\ & + \sum_{\text{torsions}} \frac{c_i}{2}\left[1 + \cos(n\omega_i - \gamma_i)\right] \\ & + \sum_{\text{atom pairs}} 4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] \\ & + \sum_{\text{atom pairs}} k\frac{q_i q_j}{r_{ij}}. \end{aligned} \tag{2.3}$$

The covalent character of the system is defined by the first three terms of the system, where the summation indices run over all the bonds, angles and torsions. In contrast, the last two terms are only defined by atom pairs, with $q_i q_j$ being the product of their charges and $r_{ij} = |r_i - r_j|$ is the distance between the two atoms in the pair. The first two terms give energies of deformations of the bond lengths l_i and bond angles θ_i from their respective equilibrium values l_{i0} and θ_{i0} with force constants a_i and b_i. These two terms model the correct chemical structure, but prevent more complicated chemical phenomena like bond breaking. Rotations around the chemical bond are described by the third therm, which is priodic with periodicity determined by n and

heights of rotational barriers defined by c_i. The forth term represents the van der Waals repulsive and attractive interatomic forces in the form of the Lennard–Jones 12-6 potential. The last term is the Coulomb electrostatic potential. Some effects due to specific environments can be accounted for by properly adjusting partial charges q_i and effective value of the constant k as well as the van der Waals parameters ε_{ij} and σ_{ij}.

We now have a full mathematical description of the problem at hand. Due to the many-body nature of the problem, it is out of question to solve it analytically, thus it has to be discretized and solved numerically with a computer. First, we need to specify the initial conditions of the system, that is, the initial positions $\mathbf{r}_{i0}$ and initial velocities $\mathbf{v}_{i0}$ of the particles in the system. Then we have to use a numerical integrator to continually make finite time interval steps and find the successive values of positions $\mathbf{r}_i(t)$ and velocities $\mathbf{v}_i(t)$ in the time evolution of the system.

2.0.2 *Ab Initio* Methods

2.0.3 Density Functional Theory

2.0.4 Machine Learning

2.0.5 Deep Potential Molecular Dynamics

The Deep Potential Molecular Dynamics (DeepMD) (1) method uses neural networks for modeling many-body potentials and interatomic forces to drive classical molecular dynamics. The neural network architectures used by the DeepMD method are designed so that they preserve all the natural symmetries in the problem. These models are trained on *ab initio* data and are capable of producing results that are essentially indistinguishable from the original data while scaling linearly with the system size.

One of the most notable challenges in developing an efficient NN schema for molecular dynamics is devising an input format that would preserve the translational, rotational, and permutational symmetry of the system. The raw atomic coordinates from MD simulations cannot be used directly, as they do not exhibit these symmetries. Different ML models were proposed to address this problem. For example, the

Behler–Parrinello neural network (BPNN) (2) maps the coordinates onto a large set of two- and three-body symmetry functions. Another proposed model, gradient-domain machine learning (GDML) (3), maps the coordinates onto the eigenvalues of the Coulomb matrix. Both of these protocols are successful, but they are also needlessly complicated, with their use-cases being rather limited, as they do not come from a first-principles analysis of the modeling problem, and it is not straightforward to extend them beyond simple systems. The DeepMD methodology attempts to provide a more first principle-based approach to overcome the limitations associated with auxiliary quantities like the symmetry functions or the Coulomb matrix.

Consider a system of N atoms, where the coordinates of these atoms can be represented as a set of position vectors $\{\mathbf{R}_1, \ldots, \mathbf{R}_N\}$, with each $\mathbf{R}_i \in \mathbb{R}^3$. DeepMD decomposes the total system energy E into a sum of energy contributions from individual atoms,

$$E = \sum_i^N E_i, \tag{2.4}$$

where i is an index of an individual atom. Atomic energy E_i is fully determined by the position of the ith atom and by the positions of its near neighbours,

$$E_i = E_{s(i)}(\mathbf{R}_i, \{\mathbf{R}_j \mid j \in N_{R_C}(i)\}), \tag{2.5}$$

where $N_{R_C}(i)$ denotes the index set of the neighbour atoms of atom i within the cut-off radius R_C. $s(i)$ is the chemical species of atom i. For reasons discussed above, it is less than optimal to use the position vector data $\mathbf{R}_i, \{\mathbf{R}_j \mid j \in N_{R_C}(i)\}$ when modeling the function $E_{s(i)}$ with DNN. Thus, the DeepMD method introduces a mapping from position vectors to "descriptors" of atomic chemical environment, that better capture the underlying symmetries.

To construct the descriptor for atom i, we first calculate the relative positions of its neighbouring atoms,

$$\mathbf{R}_{ij} = \mathbf{R}_j - \mathbf{R}_i. \tag{2.6}$$

The coordinate of the relative position R_{ij} under the lab reference frame $\{\mathbf{e}_x^0, \mathbf{e}_y^0, \mathbf{e}_z^0\}$ is denoted by $(x_{ij}^0, y_{ij}^0, z_{ij}^0)$, such that

$$\mathbf{R}_{ij} = x_{ij}^0 \mathbf{e}_x^0 + y_{ij}^0 \mathbf{e}_y^0 + z_{ij}^0 \mathbf{e}_z^0. \tag{2.7}$$

Both representations $\mathbf{R}_{ij}$ and $(x^0_{ij}, y^0_{ij}, z^0_{ij})$ preserve the translational symmetry. The rotational symmetry is captured by constructing a local frame of reference and using it to express the coordinates of neighbouring atoms. We first pick atoms with indices $a(i)$ and $b(i)$ from the neighbours $N_{R_C}(i)$ by certain user-specified rules. The local reference frame $\{\mathbf{e}_{i1}, \mathbf{e}_{i2}, \mathbf{e}_{i3}\}$ of atom i is then constructed by

$$\mathbf{e}_{i1} = \mathbf{e}(\mathbf{R}_{ia(i)}), \tag{2.8}$$

$$\mathbf{e}_{i2} = \mathbf{e}\left(\mathbf{R}_{ib(i)} - (\mathbf{R}_{ib(i)} \cdot \mathbf{e}_{i1})\mathbf{e}_{i1}\right), \tag{2.9}$$

$$\mathbf{e}_{i3} = \mathbf{e}_{i1} \times \mathbf{e}_{i2}, \tag{2.10}$$

where $\mathbf{e}(\mathbf{R})$ denotes the normalized vector of $\mathbf{R}$, such that $\mathbf{e}(\mathbf{R}) = \mathbf{R}/|\mathbf{R}|$. The local coordinate (x_{ij}, y_{ij}, z_{ij}) is then calculated from the lab coordinate $(x^0_{ij}, y^0_{ij}, z^0_{ij})$ through the transformation

$$(x_{ij}, y_{ij}, z_{ij}) = (x^0_{ij}, y^0_{ij}, z^0_{ij}) \cdot \mathcal{R}(\mathbf{R}_{ia(i)}, \mathbf{R}_{ib(i)}), \tag{2.11}$$

where

$$\mathcal{R}(\mathbf{R}_{ia(i)}, \mathbf{R}_{ib(i)}) = [\mathbf{e}_{i1}, \mathbf{e}_{i2}, \mathbf{e}_{i3}] \tag{2.12}$$

is the rotation matrix with the columns being the local reference frame vectors. The descriptive information of atom i given by neighboring atom j is then obtained by using either both the radial and angular information or only the radial information

$$\{D_{ij}\} = \begin{cases} \left\{\frac{1}{R_{ij}}, \frac{x_{ij}}{R_{ij}}, \frac{y_{ij}}{R_{ij}}, \frac{z_{ij}}{R_{ij}}\right\}, & \text{full information;} \\ \left\{\frac{1}{R_{ij}}\right\}, & \text{radial-only information.} \end{cases} \tag{2.13}$$

The order of the neighbour indices j in $\{D_{ij}\}$ is fixed by sorting them first by their chemical species and then, within each chemical species, according to their inverse distances to the atom i, i.e., $1/R_{ij}$. The permutational symmetry is naturally preserved in this way. This is the full procedure for constructing the mapping from atomic positions to descriptors, which is denoted by

$$\mathbf{D}_i = \mathbf{D}_i(\mathbf{R}_i, \{\mathbf{R}_j \mid j \in N_{R_C}(i)\}). \tag{2.14}$$

The descriptors $\mathbf{D}_i$ preserve the translational, rotational, and permutational symmetries and are passed to a DNN to evaluate the atomic energies. This process can be mathematically expressed as

$$E_{s(i)} = \mathcal{N}_{s(i)}(\mathbf{D}_i), \tag{2.15}$$

The DNN used by DeepMD method is a feed forward neural network with multiple hidden layers, where each layer transforms the input data $\mathbf{d}_i^{p-1}$ from the previous layer into $\mathbf{d}_i^p$ and passes them as an input to the next layer. The transformation consists of a linear and a non-linear step, i.e.

$$\mathbf{d}_i^p = \varphi\left(\mathbf{W}_{s(i)}^p \mathbf{d}_i^{p-1} + \mathbf{b}_{s(i)}^p\right), \tag{2.16}$$

where φ represents the non-linear function and $\mathbf{W}_{s(i)}^p$ and $\mathbf{b}_{s(i)}^p$ are free parameters of the linear transformation to be optimize by the training process. In order to determine the unknown parameters $\mathbf{W}s^p, \mathbf{b}s^p$ of the linear transformation, a loss function L is minimized during the training process. This loss function is calculated by taking a weighted sum of mean square errors of the predictions made by the DNN. Specifically, the loss function $L(p\epsilon, p_f, p\xi)$ is defined as follows:

$$L(p_\epsilon, p_f, p_\xi) = \frac{p_\epsilon}{N}\Delta E^2 + \frac{p_f}{3N}\sum_i |\Delta \mathbf{F}_i|^2 + \frac{p_\xi}{9N}||\Delta \Xi||^2, \tag{2.17}$$

where ΔE, $\Delta \mathbf{F}_i$ and $\Delta \Xi$ denote root mean square (RMS) error in energy, force, and virial, respectively. The weights p_ϵ, p_f and p_ξ are varying during the learning process. Their dependence on the learning step t is defined as

$$p(t) = p^{\text{limit}}\left[1 - \frac{r_l(t)}{r_l^0}\right] + p^{\text{start}}\left[\frac{r_l(t)}{r_l^0}\right], \tag{2.18}$$

where $r_l(t)$ and r_l^0 are the learning rate at training step t and the learning rate ate the beginning, respectively. The prefactors p^{limit} and p^{start} are specified by the user configuration. It is easy to observe that as the learning process starts the factor $\frac{r_l(t)}{r_l^0}$ tends to 1 and thus the

value of $p(t)$ tends to p^{start}. When the learning process ends, the factor $1 - \frac{r_l(t)}{r_l^0}$ tends to 1 and thus the value of $p(t)$ tends to p^{limit}. By tuning these parameters a user can precisely specify what physical properties of the system should the model be learning to predict at different stages of the learning process. The learning rate $r_l(t)$ function is in our case given by

$$r_l(t) = r_l^0 \times d_r^{t/d_s}, \tag{2.19}$$

where d_r and d_s are the decay rate and decay steps, respectively. The decay rate d_r is required to be less than 1.

3 Equations of State

3.0.1 Birch–Murnaghan Equation of State

4 Implementation

4.0.1 LAMMPS

4.0.2 DeePMD-kit

4.0.3 Active Learning

5 Results and Evaluation

6 Conclusion

Bibliography

1. ZHANG, Linfeng; HAN, Jiequn; WANG, Han; CAR, Roberto; E, Weinan. Deep Potential Molecular Dynamics: A Scalable Model with the Accuracy of Quantum Mechanics. *Physical Review Letters*. 2018, vol. 120, no. 14. Available from DOI: 10.1103/physrevlett.120.143001.
2. BEHLER, Jörg; PARRINELLO, Michele. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Phys. Rev. Lett.* 2007, vol. 98, p. 146401. Available from DOI: 10.1103/PhysRevLett.98.146401.
3. CHMIELA, Stefan; TKATCHENKO, Alexandre; SAUCEDA, Huziel E.; POLTAVSKY, Igor; SCHÜTT, Kristof T.; MÜLLER, Klaus-Robert. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*. 2017, vol. 3, no. 5, e1603015. Available from DOI: 10.1126/sciadv.1603015.

A An appendix

Here you can insert the appendices of your thesis.