GRENOBLE INP Phelma UGA

Dialogue between molecular dynamics and continuous media: definition of a cohesive model based on atomic scale information

- 2nd YEAR INTERNSHIP REPORT -

Academic Year 2021-2022

INTERNSHIP SUPERVISOR
Noel JAKSE
0670482789
noel.jakse@grenoble-inp.org

Thibault MROZ
2nd Year - SIM
thibault.mroz@grenoble-inp.org

Table of Contents

Acknowledgments

Before starting this internship report, I would like to thank my supervisor, Noel JAKSE, and my co-internship tutor, Rafael ESTEVEZ, for accepting my application for this internship even though I had no qualifications in these fields. I was simply curious to learn a lot about the field of artificial intelligence and more precisely, about Machine Learning.

I also thank them for trusting me throughout this internship by giving me quite complex tasks that pushed me to improve. It is a subject of internship in a very recent field with little documentation.

I would like to thank the colleagues with whom I shared my office and who helped me in certain tasks and who were willing to give me their time to explain certain principles.

Finally, I warmly thank Noel JAKSE for this opportunity and for the next ones to come, especially at Georgia Institute of Technology.

Glossary

Adiabatic type of thermodynamic process that happends without transferring heat or mass between systems.. 13

Cohesive Zone fracture formation is regarded as a gradual phenomenon and separation of the crack surfaces takes place across an extended crack tip, or cohesive zone, and is resisted by cohesive tractions. 10–12

Molecular Dynamic a computer simulation method for analyzing the physical movements of atoms and molecules.. 12, 13, 17, 22

NPT corresponds to an isothermal-isobaric ensemble.. 13

NVE corresponds to a microcanonical ensemble.. 13

NVT corresponds to a canonical ensemble.. 13

SSH a cryptographic network protocol for operating network services securely over an unsecured network.. 9

Young modulus mechanical property that measures the tensile or compressive stiffness of a solid material when the force is applied lengthwise.. 15, 22, 24, 25

List of Figures

List of Tables

1 — Introduction

1.1 Context

Technological advances in Artificial Intelligence and Machine Learning (MLOps) have led to the development of many tools to facilitate the life of users and methods to advance research. A very recent phenomenon in the world of research, Machine Learning allows to save a lot of computing time to perform large scale simulations but also to make some predictions.

There are several branches of Machine Learning. The most classical one is the one where a program is given a lot of data and it learns from this data. The program can then provide a prediction based on the input parameters. However, by doing this, we lose the physical sense (if the data are physical simulations or experiments). Another branch is to sort the input data according to what is more likely to happen. The physical meaning is then preserved but the program will only be able to provide an estmiation of what could happen.

1.2 Internship's Problematic

This internship is part of a multiscale analysis of fracture and more precisely by conducting a diaglogue in molecular dynamics and description in continuous medium. More precisely, the aim is to identify a cohesive zone model, representing the fracture mechanism at the continuous scale through a "stress-vector" - "opening" relation. This model will be identifiable following calculations in molecular dynamics which produce the numerical experiments.

An important part of the work is to conduct Molecular Dynamics simulations on a Silicon (Si) crystal for which the fracture occurs by cleavage. However, since Si has anisotropic elastic properties, it is expected that the fracture properties are also anisotropic. Therefore, simulations for different orientations between the crack plane and the crystalline symmetry planes will also have to be carried out. A systematic approach can be conducted. Nevertheless, the methodology associating Machine Learning and Molecular Dynamics is to be exploited in order to gain in calculation time.

Once the cohesive model is identified, it is then possible to study and predict the intercations between cracking and microstructure (in a polycrystal for example), as well as between crack and cavity.

It is a 100% digital project with a strong interest in simulation methods and Machine Learning.

1.3 Laboratory Presentation

1.3.1 History

The Laboratory of Science and Engineering of Materials and Processes is the result of the merger of three units on January 1, 2007. It is a joint research unit: CNRS, Grenoble-INP, and IESA. It brings together an average of 220 people including 56 researchers and teacher-researchers, 37 engineers, technicians and administrative staff, 60 PhD students, post-doctoral fellows, guests and trainees.

1.3.2 Research Groups

The Laboratory relies on four research groups that perpetuate the basic sciences in physics and physical chemistry, thermodynamics and kinetics, solid and fluid mechanics:

- **EPM :** Elaboration by Magnetic Processes
- **GPM2 :** Physical and Mechanical Engineering of Materials
- **PM :** Metal Physics
- **TOP :** Thermodynamics, modeling, Process Optimization

This internship is placed between two divisions (PM and TOP) in a small team composed of :

- **Noel JAKSE :** Teacher-researcher in the TOP research group, Master and Supervisor of the internship
- **Rafael ESTEVEZ :** Researcher, Co-Supervisor of the internship
- **Thibault MROZ :** Intern Assistant Engineer

The TOP Research Group focuses on materials development, thermodynamic phenomena (stability and characterization) and atomistic, thermodynamic, kinetic and reactor modeling. This has applications in the fields of thin films, complex metal alloys and functional materials.

The PM Research Group focuses on the metallurgy of metals: atomic structure, mechanical and physical properties and oxidation. This has applications in the fields of materials for energy and microelectronics but also for structural materials.

The internship is in the field of atomistic modeling and atomic structure.

1.4 Report Outline

2 — Hardware and Methods

In this part, will be detailed all the hardware, software and methods used to carry out simulations.

2.1 Hardware and Software

As this internship is 100% digital, a good comuter is required. A personal computer (MacBook Air M1) as well as a computer provided by the laboratory (under Ubuntu) will be the main equipment for this internship.

The main softwares are the following ones:

- **Visual Studio Code (VS Code)**: a source-code editor developed by Microsoft.
- **Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS)**: a molecular dynamics programm (coded in `C++`) from Sandia National Laboratories.
- **Ovito**: a visualization and analysis software for output data generated in molecular dynamics.
- **Perseus GRICAD**: high performance computing and storage platforms.
- **GitLab**: open-source software based on `git` to host code and provides wiki and bugs tracking system. GitLab of the Project

2.1.1 LAMMPS

LAMMPS is an open-source molecular dynamic code with a focus on materials modeling. It provides potentials for solid-state materials (metals and semiconductors). It can be used to model atoms or, more generically, as a parallel particle simulator at the atomic, meso, or continuum scale.

LAMMPS does not have any graphic interface which makes the handholding not that easy. The input code is written is `.txt` files thats are compilled through a `Makefile` called with a `bash` command : `lmp_serial -in input.file.txt`.

LAMMPS provides a `log.lammps` file as output. All the behaviour of the script (output values, warnings, errors ...) is written in this file. However, with specific commands, this software can provide other outfile such as a `dump.test` file, which will be useful to have a visualization of the material behaviour.

Here is a quick recap :

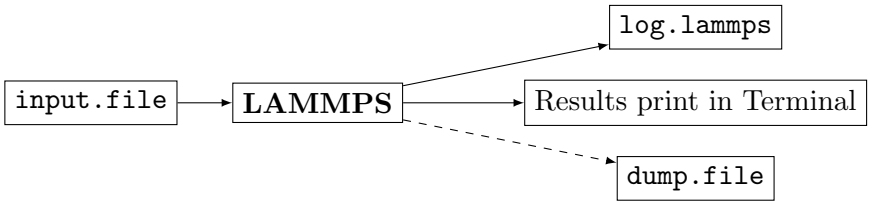


Figure 2.1: LAMMPS operation

2.1.2 Ovito

Ovito is a scientific visualization and data analysis tool for atomistic and other particle-based models. The community edition is free of charge under an open source license. Ovito has a Pro version which is a powerful extension with extended analysis toolset, visualization capabilities and automation with the `Python` integration. For this intership, the community edition is used.

Ovito will be used to visualize the behaviour of the atoms (mainly their position and velocity along the x,y and z axes). It will help to have a first sight of the simulation to see if there is no inconsitent behaviours before going deeper in the process.

The visualization is based on the `dump.file` that LAMMPS is producing. So here is the final scheme :

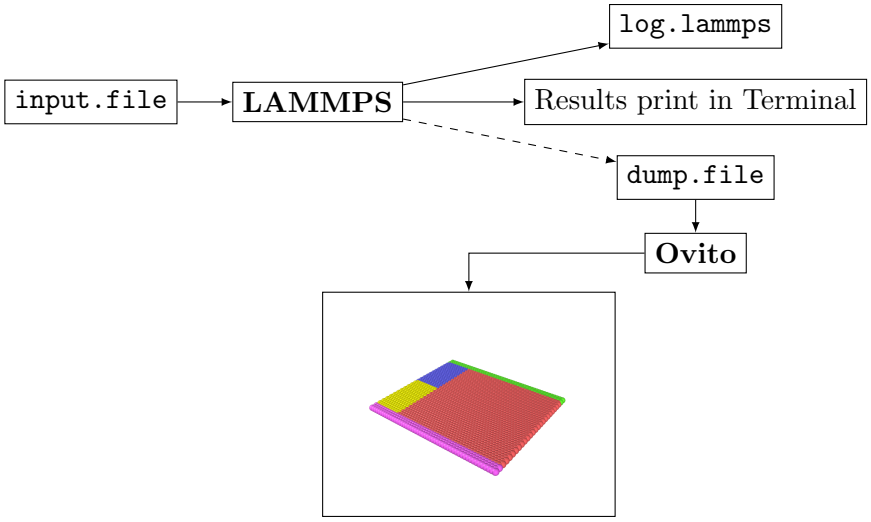


Figure 2.2: Final Operation Scheme

2.1.3 Perseus GRICAD

GRICAD offers intensive computing and data processing infrstructures to answer the needs of scientists. This tool provides an access to computing, grid, cloud, notebook and associated storage platforms. Moreover, an user support with assitance is opened. These infrastructures are open to all members of the scientific communities of the Grenoble site, as well as to their external collaborators. To have an access to this computing tool, a Perseus account is required. Once the Perseus account is created, you need to be member of the project to run your scripts. For this internship, the project is `pr-atosimul`.

GRICAD provides four computing culsters that are different (each cluster have their own hardware and configuration). Cluster access is normally done using a `SSH Client` (Secure Shell Protocol) [1]. However, SSH servers are vulnerable to scans and attacks so, for security reasons, it is not possible to let the clusters be directly accessed from the internet. GRICAD provides two SSH gateways that are more secure than the clusters. So the login method is to first, login to an SSH gateway and the login to the targeted cluster :

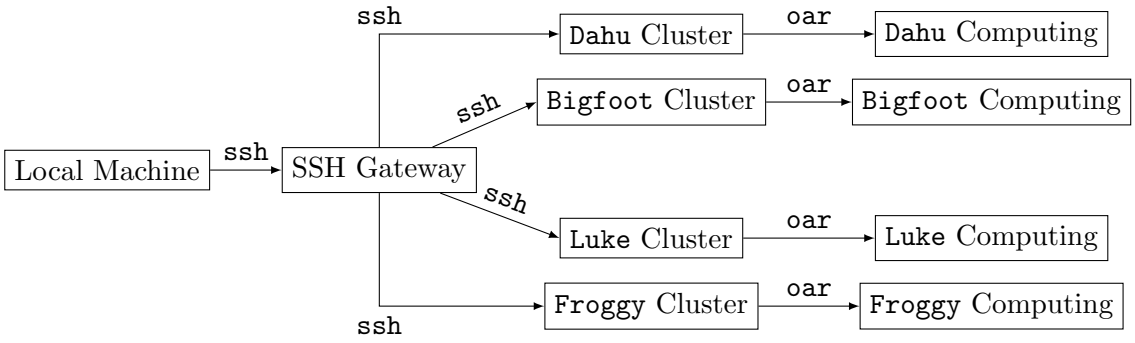


Figure 2.3: SSH cluster access schema

Those two SSH gateways (called `Rotule` and `Trinity`) are grouped under a single DNS : `access-gricad.univ-grenoble-alpes.fr`. This allows for load balancing on these two machines. Moreover, if one server came to fail, the other one is still available for computing.

The submission work for computing is made through a `run.oar` file. It is a `bash` script that provides the number of nodes and cores of the processor wanted by the user, the walltime (max time of computing), the name of some output files and then commands to run external scripts. A script is given in the A) appendix.

Then, all the output files are stored on the cluster. To visualize them through Ovito for instance, a File Transfer Protocol Secure (FTPS) is used.

2.2 Methods

The aim is to make predictions about the behavior of cracks using a Machine Learning program. The algorithm will have to learn from the numerous simulations. It is therefore crucial that these input data are correct in order to limit the error on the predictions at the output of the program. So the first part of this internship is to work on the limit and initial conditions in order to have correct simulations.

2.2.1 The Theory

As mentionned before, this internship is a multiscale analysis. So, it focuses on Molecular Dynamic for the small scale and Fracture Mechanic for the large scale.

a) Fracture Mechanics

Here is a simplified scheme of a fracture :

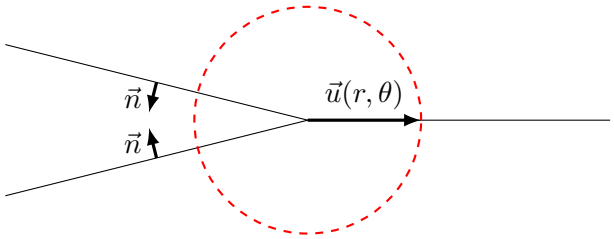


Figure 2.4: Simplified fracture scheme

A crack is defined by a discontinuity in the potential $\vec{u}$ and that there is no stress on the free surfaces ($\vec{T} = \vec{0}$). According to Fracture Mechanic, the dashed red circle corresponds to the cohesive zone model. Outside this area, the surfaces are free of effort: $\vec{T} = \underline{\underline{\sigma}} \cdot \vec{n} = \vec{0}$. This is not the case in the cohesive zone where we have an interatomic potential $\vec{u}$ defined as $\vec{u}(r,\theta) = f(r) \times g(\theta) \cdot \vec{u_r}$.

According to T.L. Anderson [2], locally (so when $r \to 0$), we have :

$$\vec{u} = \frac{K_1}{2\mu}\sqrt{r} f_{XY}(\theta) \quad \text{where} \quad K_1 = \text{a loading parameter}$$

K_1 depends mainly on the type of crack (on the side, central, in tension, in traction...) and f_{XY} is a function that depends on X and Y which are the plan coordinates.

The interatomic potential induces an interatomic force $\vec{F} = \nabla \vec{u}$ which (*on the figure below, T is the stress vector defined as $\vec{T} = \frac{\sum \vec{f_i}}{S}$ where S corresponds to the surface*).

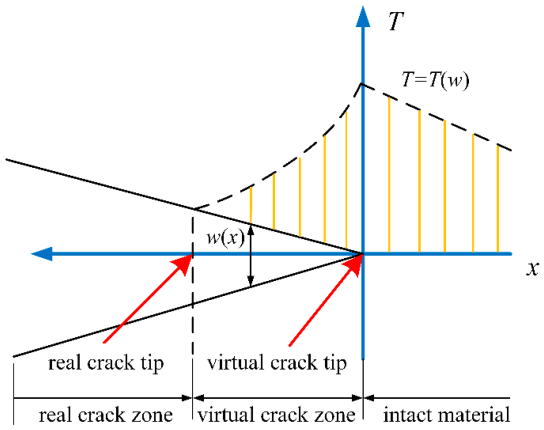


Figure 2.5: Interatomic force $\vec{F}$ [3]

We can define a_0 which is the distance from the virtual crack tip where the interatomic force is zero : $\Delta = r - a_0$.

So, we have the following criteria :

$$\begin{cases} \Delta < 0 \implies f < 0 \implies \text{atoms repulsion} \\ \Delta > 0 \implies f > 0 \implies \text{atoms attraction} \end{cases}$$

This cohesive zone implies a difficult problem to resolve during a tensile test. The solution is to split the problem into two independent and simpler to solve problems. The first problem is a tensile test with a crack on the side without taking into account the cohesive zone model. The second problem is to take the cohesive zone model but without traction. To make the tests independent, there are two different loading parameters K_1. In the case of the first test, it is a crack opening parameter $K_{1_{opening}}$. In the case of the second test, it is a crack closing parameter $K_{1_{closing}}$.

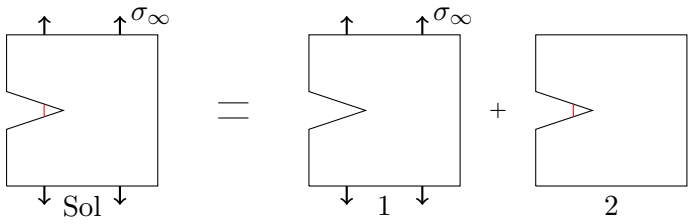


Figure 2.6: Problem splitting

Then, in order to calculate the crack propagation using fracture mechanics, a propagation criteria has been demonstrated by Alan Arnold Griffith [4] in his paper. He

demonstrated for a tensile test charged with σ_∞, with G the energy per unit of free area, A the free surface, W the crack work and a the crack lenght, that:

$$\frac{\partial W}{\partial A} = \frac{W\left[(2(a+da))\right] - W\left[(2a)\right]}{dA} = 2\gamma$$

Moreover, $\frac{\partial W}{\partial A}$ corresponds to the consumed energy to create an additional free surface. So, as long as $G < \frac{\partial W}{\partial A}$, there is no crack propagation. However, when $G = 2\gamma$, there is propagation iniation.

That is how we calculate fracture propagation in materials according to fracture mechanics. But, we cannot really go deeper in the virtual crack zone and understand all the atomic interactions in the cohesive zone. To have a better understanding of potentials and crack propgation, we need to look closer and use molecular dynamic.

b) Molecular Dynamics

Molecular dynamics is a numerical method that resolver Newton's equation of motion $\vec{F} = \frac{d}{dt}(m\vec{v})$ for system of interacting. As the atoms are allowed to interact thanks to interatomic potentials for a fixed period of time, it gives the dynamic evolution of the system.

Here is a simplified schema of a molecular dynamic progam (such as LAMMPS) :

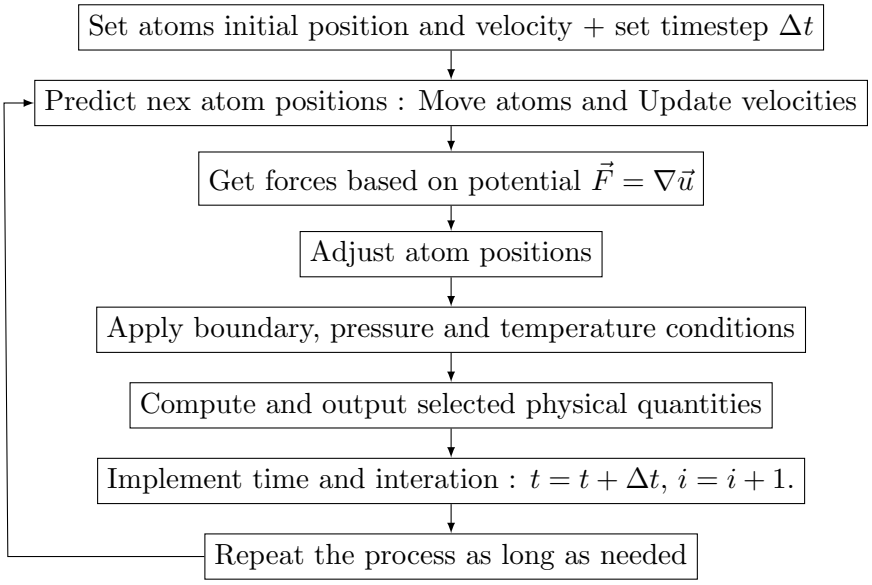


Figure 2.7: Simplified schema of a molecular dynamic program

The engineering of a molecular dynamic algorithm should account for the available computational power of the machine. That is why parameters such as timestep, simulation box size, number of atoms, potential... are wisely choose to have a correct computational time. Thoose simualtions requires sometimes high computational power to output some results. That is why a calculating computer is used during this internship. The type of system is also an important parameter to take into consideration. Only microcanonical (NVE), canonical (NVT) and isothermal-isobaric (NPT) ensembles are used for simulations.

In the microcanonical ensemble, the system is totally isolated from changes in moles (N), volumes (V) and energy (E). It is linked with an adiabatic process without heat exchange.

In the canonical ensemble, the amount of atoms (N), volume (V) and temperature (T) are conserved. In a NVT system, the energy is exchanged with a thermostat. There are several thermostat algorithm and methods.

In the isothermal-isobaric ensemble, the amount of atoms (N), pressure (P) and temperature (T) are conserved. Compared to the NVT ensemble, a barostat is needed in addition to a thermostat. This ensemble is getting closer to laboratory conditions.

In addition to the choice of the type of system to run a simulation, molecular dynamics require a potential function which is a mathematical description of particles interaction. There are plenty of potentials that can be defined at many levels of physical accuracy. It exists pair potential functions in which the total potential energy can be calculated from the sum of all interatomic pairs energy. An example of such potential is the Lennard-Jones potential:

$$U(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right]$$

However, empirical potentials (often called force fields) are frequently used in material physics. Force fields potentials consist of a summation of bonded forces associated with chemical bonds, electrostatic charges and Van der Waals interactions. These potentials contain a huge amount of free parameters (atomic charge and radius, bond length and angles...) that makes calibration complicated. Its calculation is the bottleneck in the speed of molecular dynamic simulations. But, they are more precise than pair potential functions. An example of such potential is the ReaxFF potential.

2.2.2 The Model used

To simulate the propagation of a crack in a material, the model used is based on the theory of molecular dynamics. The idea is to carry out a small-scale tensile test on a sample with the beginning of a crack on the side. LAMMPS will allow to calculate the position, the speed and the forces exerted on each atom of the box.

Here is the model that will be used for simulations :

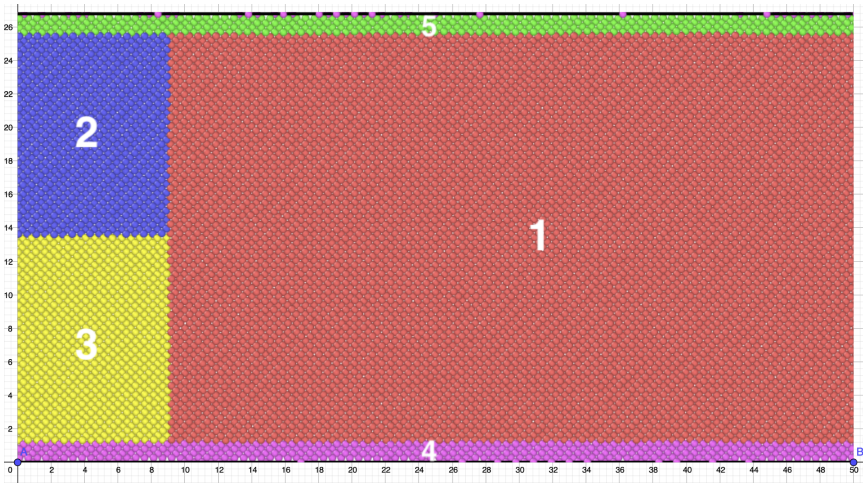


Figure 2.8: Model used for simulations

5 different regions are defined. There are 50 atoms along the x axis (horizontally) and 25 along the y axis (vertically). The dimensions of the box therefore vary according to the mesh parameter of the chosen material. During the tensile test, all regions are mobile except the 4th one. To simulate a tensile test in LAMMPS, we set the same velocity for all the mobile groups. To simulate the crack, regions 2 and 3 are not linked. An `input.file` is given in the B) appendix.

3 — A Good Simulation Visualisation

The goal now is to determine which parameters of the molecular dynamic algorithm to study to have correct simulations in order to compute Machine Learning algorithm on those simulations. To do so, it is necessary to compare the output values given by the algorithm to experienced data. As a tensile test is realised, a good value to check is the young modulus. The young modulus can be characterised with the follwing equation: $\sigma = E \cdot \epsilon$ where σ corresponds to the average strain along the tensile test direction (y here), E the young modulus and ϵ the elongation. The elongation is defined as $\frac{L-L_0}{L_0}$ where L is the length at a given timestep of the simulation and L_0 the length at the start of the simulation.

3.1 First Results

But at first, let's try to match the curve in the figure 2.5. The process is to watch the crack evolution in a small box at the tip of the virtual crack zone as the same time that the box length increases. The `avg_yy_strain-vs-separation` folder of the GitLab's project contains the `python` code to print the results of the simulation. Here is the result:

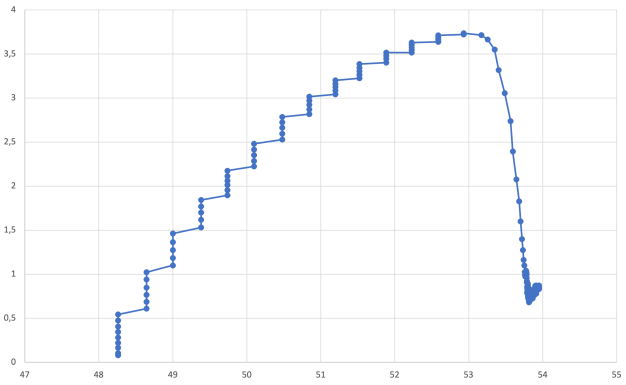


Figure 3.1: σ_{yy} *vs* l_y

The first part of the curve is not continuous. This might cause several problems for next simulations so fixing this is a priority. The curve is staircase: there are jumps in the values of the length of the box. To overcome this problem, the idea is to play on two main parameters: the pulling speed and the updating of the data in the algorithm.

3.2 Pulling speed impact

The elongation of the system increases in a non continuous way. One idea would be to decrease the velocity of the atoms in the upper part (atomic group 5) and in the moving

part (atomic group 1,2 and 3). This is called the opening speed. However, to have coherent simulations, it is necessary to keep the product **velocity** $\times$ **simulation time** constant.

But: **simulation time** = **number of steps** $\times$ **timestep**. In molecular dynamics, we cannot touch the timestep because it corresponds to the temporal discretization in the Taylor expansion:

$$\text{for} \quad \Delta t \ll t, \quad X(t+\Delta t) = X(t) + \Delta t \times \dot{X}(t) + \frac{\Delta t^2}{2} \times \ddot{X}(t) \quad \text{where} \quad \Delta t = \text{timestep}$$

So by decreasing the opening speed by a factor of 10, the number of steps must increase by a factor of 10. The calculations can quickly become quite large to perform on a normal machine. To perform the most demanding calculations, scripts must be run on the `dahu` computing cluster.

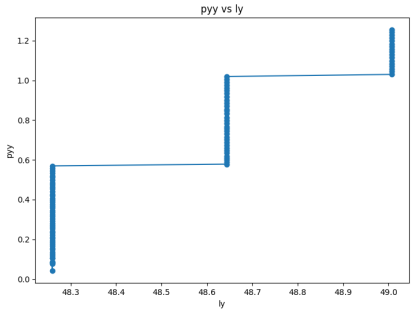
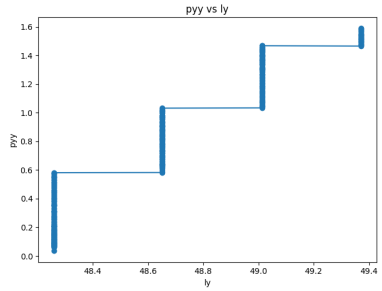
	100 steps - $V = 0,03$	**300 steps - $V = 0,01$**
Product = 3		

Table 3.1: Results for 100 and 300 steps

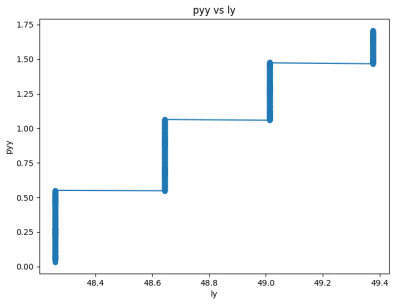
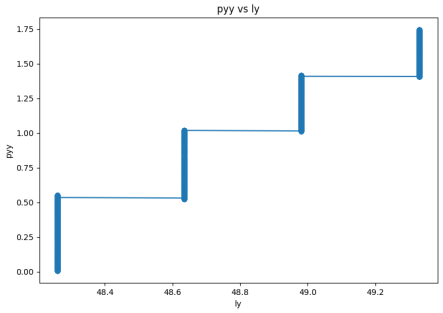
	1000 steps - $V = 0,003$	**3000 steps - $V = 0,001$**
Product = 3		

Table 3.2: Results for 1000 and 3000 steps

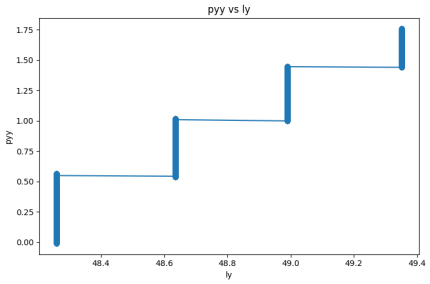
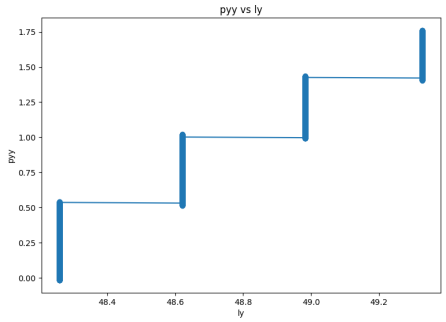
	10000 steps - $V = 0,0003$	**30000 steps - $V = 0,0001$**
Product = 3		

Table 3.3: Results for 10000 and 30000 steps

We can notice that the speed of traction has no influence on the smoothing of the curve (even for a large number of runs at a very low speed). It is therefore necessary to play on another parameter.

3.3 Neighbor List impact

The neighbor list is a major paramter in the actualisation of atoms' position and velocity.

One of the big problems in molecular dynamics is that the complexity of the calculations is in n^2 where n is the number of atoms in simulation.

For instance, let us take a system with 100 atoms in it. To calculate interatomic intercations, it is necessary for each atoms to calculate interactions with the 99 others. This leads to (100 atoms) $\times$ (99 calculations/atom) $\approx 10^4$ calculations. Now let us take a system with 1000 atoms: (1000 atoms) $\times$ (999 calculations/atom) $\approx 10^6$ calculations. By increasing the number of atoms by a factor of 10, we increase the number of calculations by $10^2 = 100$.

For much larger systems like the ones used in this internship, this can cause very heavy computations that can take several days or weeks. Several techniques have been experimented to reduce the computation time, like the "neighbor list".

3.3.1 What is the Neighbor List ? [5]

One way to manage the quadratic growth of the calculations is to set up a cutoff radius: all atoms beyond this cutoff radius are ignored during the calculations.

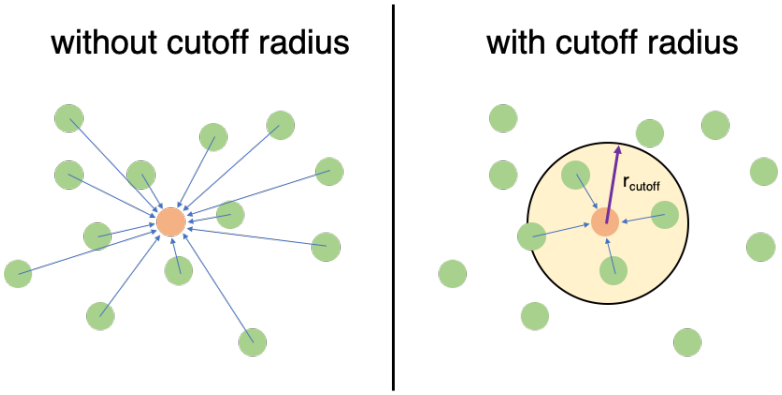


Figure 3.2: Cutoff Radius

However, the choice of the value of this radius is not trivial. First, the atoms must not interact with their periodic image in the system. Secondly, implementing a cutoff radius affects the interatomic potential: it cannot be defined at every point. This induces discontinuities in the potential and therefore unintended behaviors. To counter this, potentials taking into account the cut-off radius as a parameter have been developed. The Lennard Jones potential (LJ) as seen in the theory can be quoted.

Once the value of the cutoff radius is defined, all the neighboring atoms can be found by looping on the positions of each atom:

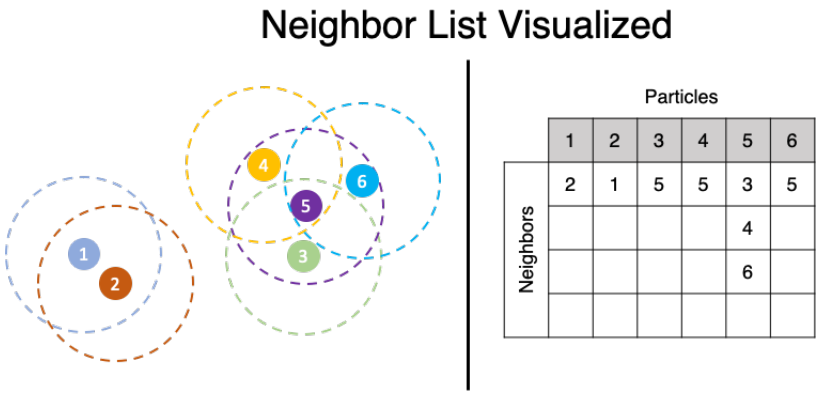


Figure 3.3: Neighbor atoms

Creating this list is always a problem in n^2 but it does not need to be updated at each iteration. One method to update this nearby list is to set a skin thickness. This distance is beyond the cutoff radius:

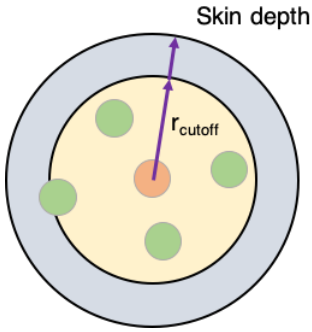


Figure 3.4: Skin Thickness

As soon as an atom leaves the cutoff radius and the skin thickness, it activates the update of the list of neighboring atoms. Moreover, by knowing the average or maximum velocity of the atoms, we can predict at which timestep the atom will leave this distance.

Even if this neighbor list method is easy to implement and allows to drastically reduce the computations, there is a big disadvantage. Indeed, this method does not take into account the long distance interactions. This makes the simulations much less accurate.

It is therefore necessary to choose carefully the cut-off radius so that the properties are not greatly affected. The approach to find this value is to decrease step by step the cut-off radius until the system returns values (diffusivity, elastic modulus, ...) outside the acceptable error range:

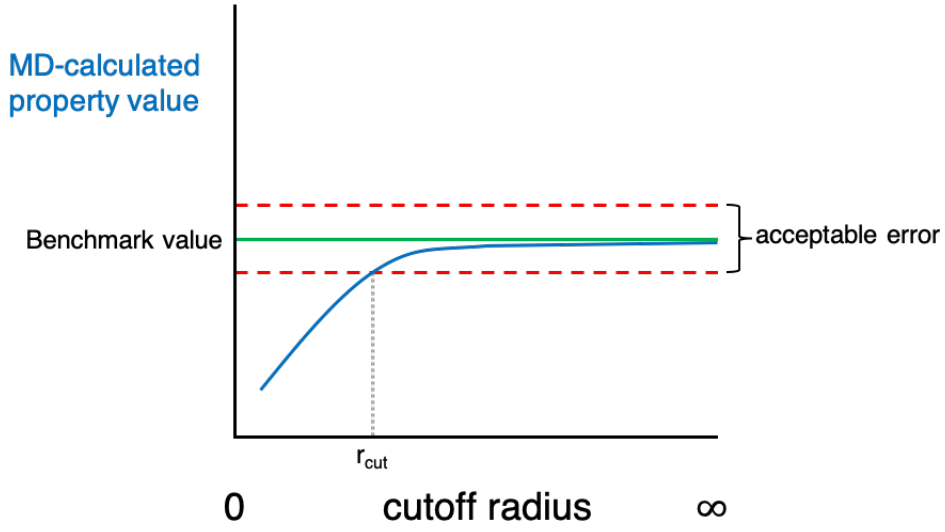


Figure 3.5: Acceptable error

So, decreaising the skin thickness will actualise more often the neighbor list and the box length as well. However, it increases the calculation time.

3.3.2 Results

Simulations with different skin thickness have been ran on `dahu`. The average strain along the y axis versus the box length (along the y axis too) for different skin thickness values is represented below:

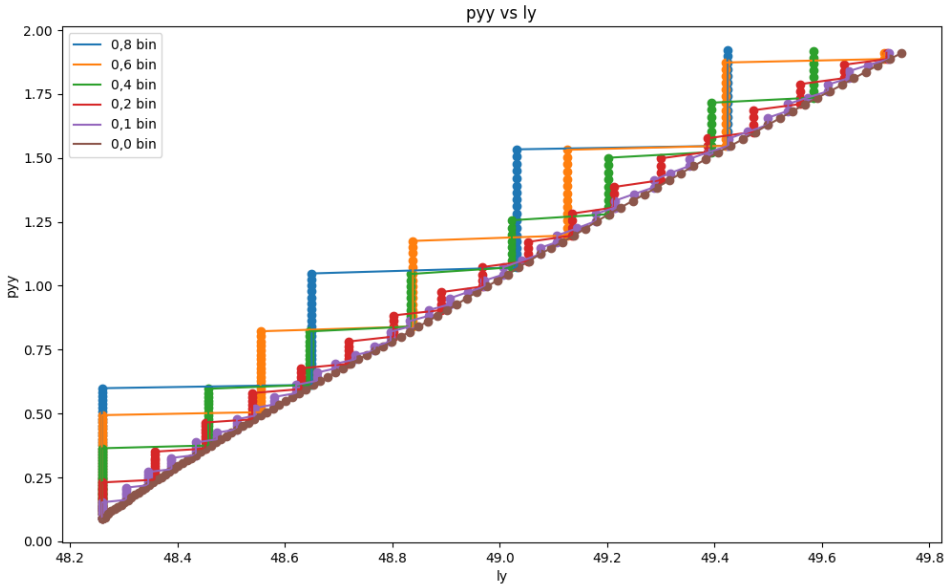


Figure 3.6: σ_{yy} *vs* l_y for different skin thickness values

The impact of the skin thickness is quite remarquable here. For the next simulations, a very low skin thickness will be chosen (0,1 bin or 0,0 bin).

3.4 Crack test

After the results found in this last section, these conditions will be tested with a crack propagation on the side of the simulation box. To do this, these calculations will be ran on a large number of timestep to observe the behavior for a fairly long time. Here is the result as an animation (to watch the animation, please use `Adobe Reader`):

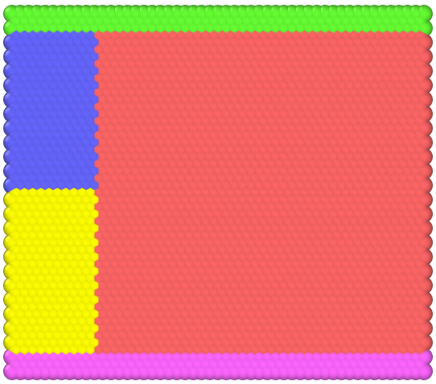


Figure 3.7: Crack behaviour animation

These calculations are made for a zero skin thickness (0.0 bin). The crack propagation starts cleanly at the beginning but stops suddenly. The material then explodes. By looking a little more closely at the animation, when the crack stops propagating, a creation of dislocations propagating at 45 degrees to the direction of propagation of the crack appears:

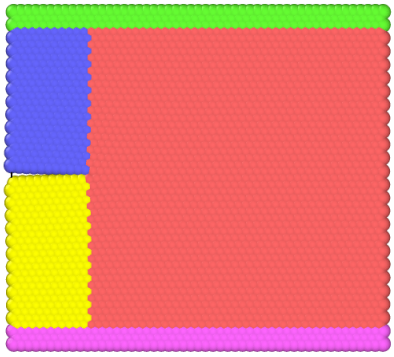


Figure 3.8: Dislocation animation

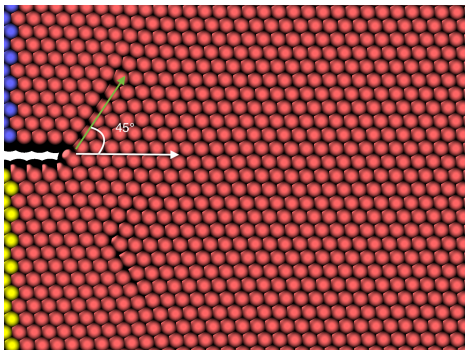


Figure 3.9: Dislocation propagation

When a material is subjected to a stress, it can deform, propagate dislocations or crack. It is all a question of energy. In this case, the propagation of the crack costs less energy. However, after a while, the material has stored enough energy to start the propagation of dislocations. The crack stops propagating.

However, in a silicon wafer, the atoms are all oriented in the [111] direction. There can therefore be no dislocations (no defects present in a wafer). So it is necessary to change the potential to have a fine cracking, without blunting during propagation. We will therefore use the ReaxFF (Reax For Field).

4 — The Potential Choice

In this section, we will discuss the influence of the potential on the simulation results.

4.1 Comparison method

To know if the results of a simulation are good, one compares the values of one or several mechanical properties at the output of LAMMPS to values calculated much more precisely. These data are available on the site materialsproject.org. It is normal that the results of LAMMPS do not correspond exactly to the values indicated in the databases of materialsproject.org: we base ourselves on an empirical potential which contains many parameters that can be adjusted. The data in the database are calculated using DFT (Density Functional Theory), a method using quantum mechanics to calculate forces in systems with very few atoms (≈ 300), which is not the case in molecular dynamics. So if the values calculated by LAMMPS are very close to the tabulated values, we can conclude that the results are good.

The idea is to compare 3 different potentials (Stillinger-Weber, ReaxFF and the ML potential) to see which one fits the materialsproject data better. Different simulations are performed on Bulk Silicon with a simple relaxation. We vary the lattice parameter to find the atomic volume corresponding to the minimum of the atomic potential energy. Thus we will have $\vec{F} = \vec{\nabla} E = \vec{0}$ and thus a zero pressure.

We will compare the young modulus provided by LAMMPS and the one available on materialsproject.org. LAMMPS does not carry out the value of the young modulus but the values of the stress tensor coefficient. It can be calculated with those coefficient. The demonstration is given in the appendix C).

4.1.1 Stillinger-Weber Potential

The Stillinger-Weber Potential has been designed for Silicone. The mathematical is complicated and without interest for the internship. It is composed of two terms: a two-body term (which corresponds to the interaction energy between two neighboring atoms) and a three-body term (specifically added to energetically promote the tetrahedral environment of the atoms).

4.1.2 ReaxFF Potential

ReaxFF, for Reactive Force Field is an empirical potential based on a force field, itself based on the order of interatomic bonds. It has a wide application in molecular dynamics. While traditional force fields are unable to model chemical reactions because of the need to form and break bonds, ReaxFF abandons explicit bonds in favor of bond orders, which allows for continuous bond formation and breaking. It is therefore a much finer potential than the Lennard-Jones potential. [6]

4.1.3 Machine Learning Potential

For a better understanding, please read the appendices D) and E) before.

This potential is a HDNNP that provides great performance for Silicone. It was released on openkim.org in 2020. Openkim is library that hosts interatomic potentials and analytics for molecular simulation. The ML Potential that we used was the `SNAP_ZuoChenLi_2019_Si__MO_869330304805_000`. It is a potential that is the result of a study that compares the performance of machine learning interatomic potentials. [8]

The potential chosen use the bispectrum descriptor (which will not be detailed here) that works really well with the Spectral Neighbor Analysis Potential (SNAP) (this is another f function).

According to the study [8], this potential has the highest root-mean-square errors (RMSEs) in predicted energies and forces.

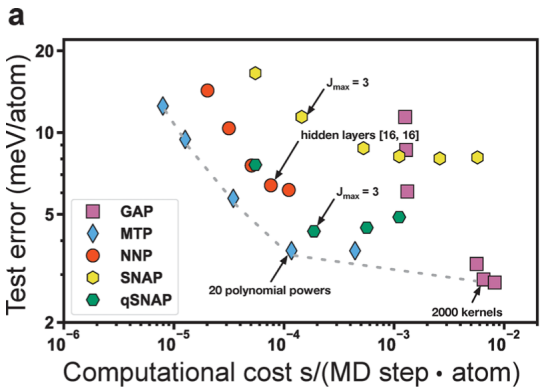


Figure 4.1: Test error for different ML Potentials

However, it shows the best extrapolability. This is really good for large systems like the one we are using. NNP are good for interpolation, it will render precise results but it is more computationally demanding. For large systems it is not possible. Moreover, we are studying solid diamond Silicone. So, we only have small atom displacement which is even better for extrapolation. This saves a lot of calculation time. Finally, the SNAP function does not require many runs to train.

4.2 Results

We varied the lattice parameter to find the atomic volume corresponding to the minimum of the atomic potential energy. Here are the results:

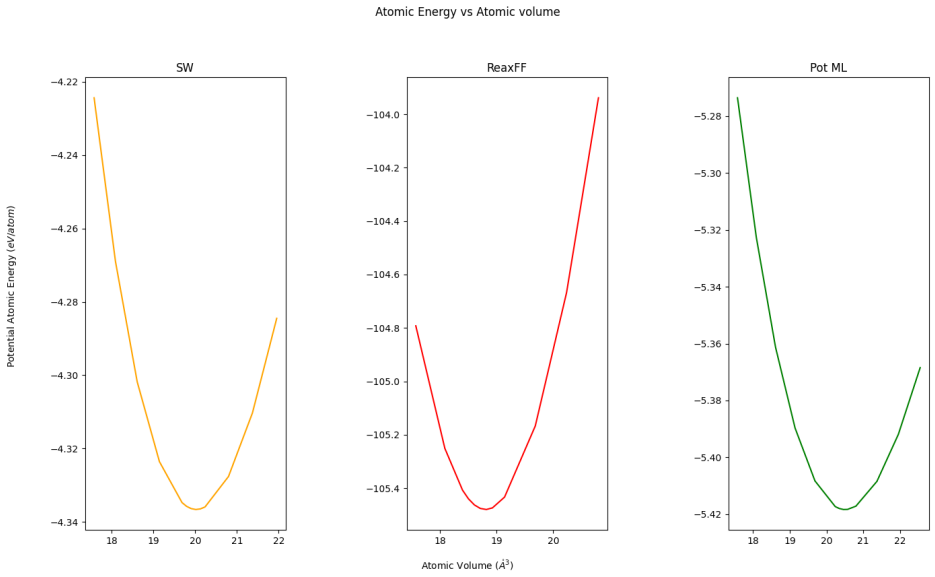


Figure 4.2: Atomic Energy VS Atomic Volume for different potentials

Then, with the minimum atomic volume, i.e. the minimum lattice parameter because of the cubic volume, we can calculate the young modulus for different potentials:

	Stillinger-Weber	**ReaxFF**	**ML Potential**	**Materials Project**
$a_{\textbf{équilibre}}$ $(\mathring{A})$	5.428	5.318	5.473	**3.867**
Bulk Modulus (GPa)	102.50	2238.15	106.82	**83**
Shear Modulus 1 (GPa)	50.35	1573.56	71.90	**63**
Shear Modulus 2 (GPa)	34.42	977.56	12.45	
Poisson Ratio	0.34	0.31	0.44	**0.20**
Young Modulus (GPa)	131.34	3937.10	185.38	**181.05**

Table 4.1: Mechanical constant results

It can be seen that the ML Potential results are very close to the Materials Project data, especially for the young modulus. The differences are explained by the lack of precision when training the machine learning algorithm. It has only been trained with forces and stresses values from DFT calculations [8]. Adding energy values might have increased the algorithm precision.

We have seen that the SNAP function has a large margin of errors. This may also explain the differences between the ML and Materials Project values. Using a more precise function would greatly increase the calculation time, which is already relatively long when the latest simulation is based on 512 atoms.

Conclusion and Future Prospects

The work carried out during this internship has allowed the analysis of important simulation parameters in order to constitute a good database for the implementation of a Machine Learning program later on to predict crack propagation in Silicon. A lot of work has been done on the interatomic potential which plays a very important role in molecular dynamics simulations. The identification and understanding of the Machine Learning Potential has taken us well forward in this project. We have seen that this type of potential is a very good compromise between accuracy and computation time. It can nevertheless be improved. A team at SIMaP is studying this type of interatomic potential. Collaborative work with this project could be very encouraging for the future.

At this date (13/07/2022), the project is far from being finished. The approach taken those 10 first weeks was to determine a good set of parameters to have correct simulations. It would be nice to continue this project and maybe going a bit deeper on the Machine-Learning Potential because results are good:

1. Check with Ovito on multiple timesteps to see if we have a fine cracking, without blunting during the propagation.
2. If ok, run a tensile test.
3. If ok, run a tensile test with a crack at the side.

If thoose steps still provides good results, the next step is to create a database in order to train a Machine-Learning algorithm as explain in the report's appendices.

All the work (source code, comments, results...) provided during this internship is accessible on the GitLab repository: `https://gitlab.ensimag.fr/mrozt/meca-ml`

Bibliography

[1] **Article**, Secure Shell
https://en.wikipedia.org/wiki/Secure_Shell

[2] **Book**, T.L. Anderson, *Fracture Mechanics : Fundamentals and Applications*, Boca Raton, 2017

[3] **Article**, Kaida Dai, Baodi Lu, Pengwan Chen and Jingjing Chen, *Modelling Microstructural Deformation and the Failure Process of Plastic Bonded Explosives Using the Cohesive Zone Model*, 2019
https://www.mdpi.com/1996-1944/12/22/3661

[4] **Book**, A.A. Griffith, *The Phenomena of Rupture and Flow in Solids*, Royal Aircraft Establishment, 1920

[5] **Article**, Cameron Mcelfresh, *Molecular Dynamics: Neighbor Lists in Python*, Medium, 2020

[6] **Article**, ReaxFF
https://en.wikipedia.org/wiki/ReaxFF

[7] **Article**, Jörg Behler, *Four Generations of High-Dimensional Neural Network Potentials*, Chemical Reviews, 2021
https://doi.org/10.1021/acs.chemrev.0c00868

[8] **Article**, Yunxing Zuo, Chi Chen,Xiangguo Li, Zhi Deng,Yiming Chen, Jörg Behler,Gabor Csanyi, Alexander V. Shapeev, Aidan P. Thompson, Mitchell A. Wood, and Shyue Ping Ong, *Performance and Cost Assessment of Machine Learning Interatomic Potentials*, The Journal of Physical Chemistry, 2020

APPENDICES

A) run.oar script

```
#!/bin/bash

folder=$(pwd | cut -d "/" -f 4)

#OAR -n lammps-reaxff-0,0
#OAR -l /nodes=1/core=32,walltime=10:00:00
#OAR --stdout lammps.%jobid%.out
#OAR --stderr pytest.%jobid%.err
#OAR --project pr-atosimul

# load environment
source /applis/site/guix-start.sh

# lancement du code

mpirun -np `cat $OAR_FILE_NODES|wc -l` lmp -in in_older.$folder-0,0.txt
```

B) input.file script example

```
# 2d LJ crack simulation

dimension        2
boundary        s s p

atom_style        atomic
neighbor        0.8 bin
neigh_modify        delay 10

# create geometry

lattice                hex 0.93
region                box block 0.0 50.0 0.0 25.0 -0.5 0.5
create_box        5 box
create_atoms        1 box

mass                1 1.0
mass                2 1.0
mass                3 1.0
```

```
mass                 4 1.0
mass                 5 1.0

# LJ potentials

pair_style         lj/cut 2.5
pair_coeff         * * 1.0 1.0 2.5
pair_modify         shift yes

# define groups

region                1 block INF INF INF 1.0 INF INF
group                lower region 1
region                2 block INF INF 24.0 INF INF INF
group                upper region 2
group                boundary union lower upper
group                mobile subtract all boundary

region                leftupper block INF 10.0 12.5 INF INF INF
region                leftlower block INF 10.0 INF 12.5 INF INF
group                leftupper region leftupper
group                leftlower region leftlower

set                group leftupper type 2
set                group leftlower type 3
set                group lower type 4
set                group upper type 5

# initial velocities

compute                  new mobile temp
velocity                mobile create 0.001 887723 temp new mom yes rot yes
velocity                upper set 0.0 0.003 0.0
velocity                mobile ramp vy 0.0 0.003 y 1.0 24.0 sum yes

# fixes

fix                1 all nvt temp 0.01 0.01 1.0
fix                2 boundary setforce NULL 0.0 0.0

# run - NE PAS TOUCHER AU TIMESTEP
```

```
timestep                0.001
thermo                  100
thermo_style            custom step temp pyy lx ly lz
thermo_modify           temp new

neigh_modify            exclude type 2 3

dump                    1 all custom 100 dump.test id type x y z fx fy fz

run                     10000
```

C) Young Modulus calculation

The stress matrix is the following one:

$$\underline{\underline{\sigma}} = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{pmatrix} \quad \text{where} \quad \sigma_{ij} = C_{ijkl} \cdot \epsilon_{kl} \tag{1}$$

So it comes that:

$$\begin{aligned} \sigma_{11} &= C_{11kl} \cdot \epsilon_{kl} \\ &= C_{1111} \cdot \epsilon_{11} + C_{1122} \cdot \epsilon_{22} + C_{1133} \cdot \epsilon_{33} \end{aligned} \tag{2}$$

Or, we know that (in an istrope case): $\epsilon_{22} = \epsilon_{33} = -\nu \cdot \epsilon_{11}$. So we have:

$$\begin{aligned} \sigma_{11} &= (\lambda + 2\mu) \cdot \epsilon_{11} - 2\nu\lambda \cdot \epsilon_{11} \\ &= [\lambda(1 - 2\nu) + 2\mu] \cdot \epsilon_{11} \end{aligned} \tag{3}$$

Then, we can identify the modulus: $E = [\lambda(1 - 2\nu) + 2\mu]$. With $\nu = \frac{\lambda}{2(\lambda+\mu)}$, it comes that:

$$E = \frac{\lambda \cdot \mu + 2\mu\lambda + 2\mu^2}{\lambda + \mu} = \frac{\mu(3\lambda + 2\mu)}{\lambda + \mu} \tag{4}$$

Thanks to the C_{ijkl} coefficients, we can find everything. But, it is necessary to realise 6 idependants tests to calculate every terms (in the general case): tensile teste along the x,y and z axis then shear test along the x,y and z axis.

D) Machine Learning explanation

Machine Learning is a scientific field, and more specifically a sub-category of artificial intelligence. It consists of letting algorithms discover patterns, i.e. recurring motifs, in data sets. Anything that can be stored digitally can be used as data for Machine Learning. By detecting patterns in this data, the algorithms learn and improve their performance in performing a specific task. There are several branches of machine learning. The one used for this internship is the supervised learning. The supervised learning is a machine learning task consisting of learning a prediction function from annotated examples. It means that that we provide the inputs and the results in the data set.

Let's call X the input values and Y the results of the X value. The Machine Learning algorithm can be sum up as a f function like so: $Y = f(X)$. The goal of this algorithm is learn the links between the input (X) and the result (Y). With a lot of training, this function will be able to provides the Y value of an unknown X value. That is what we call a prediction.

How does it work ?

The development of a Machine Learning model is based on five main steps:

1. **The preparation of the training data set.** This is the most crucial step because it will be the data that will be used to train the algorithm. If the data are incorrect, the algorithm will not make good predictions. That is the work that I have done during this internship.
2. **Data labelling.** This is a mandatory step for a supervised learning. We set the Y value to the correct X. It is also really important that every piece of data have the same structure, i.e. the same type of data.
3. **Algorithm identification.** We select the type of algorithm, i.e. the f function, that will be trained with the data set. The choice of the algorithm depends on the amount and the type of the data set.
4. **Algorithm training.** In this iterative process, variables are executed and the output result is compared to the one that might have occured (the Y value). Then, les poids et le biais peuvent être ajustés pour accroître la précision des résultats.
5. **Include the last step.** Include the last step on the website.

E) The elaboration of a Machine Learning Potential [7]

The elaboration of a Machine Learning Potential is now based on the fourth generation of High Dimensional Neural Network Potential (HDNNP). Here the first, second

and third ones will be detailed in order to have an overview of the method to elaborate such potentials. This appendix correspond to the f function that was introduced in the appendix D).

First Generation

The First Generation of Machine Learning Potentials is based on Feed Forward Neural Network (FFNN).

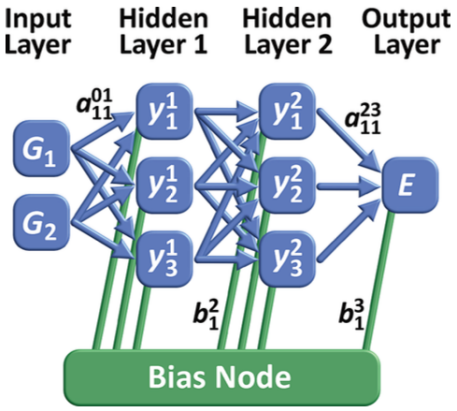


Figure 3: FFNN Structure

The G_1 and G_2 nodes are the entry point of the neural network (X value) and E the output one (Y value). For this example, $E = f(G_1, G_2)$. The values y_i^k of the neurons i in layer k and E depend on the connecting weights a_{ij}^{kl} and the bias weights b_i^k. The function f is the neural network of hidden layers that will be trained to fit the output energy value of the data set. So, the more hidden layer and neurons used, the higher the fitting capability of the FFNN for complicated functions is. Neurons are connected to each other by weights. They are fitting parameters that are optimised during the traing of the neural network. Bias nodes act like adjustable offset. The functional analytic form of the FFNN is the following one:

$$E = f_1^3 \left(b_1^3 + \sum_{l=1}^{3} a_{l1}^{23} f_l^2 \left(b_l^2 + \sum_{k=1}^{3} a_{kl}^{12} f_k^2 \left(b_k^1 + \sum_{j=1}^{2} a_{jk}^{01} G_j \right) \right) \right) \tag{5}$$

This is a linear combination of the two input nodes then shifted by bias. Moreover, a non linear activation function is applied to each shifted sum. This makes the FFNN a non linear method.

After applying the activation function, a number y_k^1 is obtained for each neuron in the first hidden layer. Then, the values of the second layer are calculated in the same way and

so on until the output y_1^3 is reached. This make the information flow unidirectional. That is why it is called a Feed Forward Neural Network.

The goal is to determined the weights and bias nodes values to fit the energy of the data set:

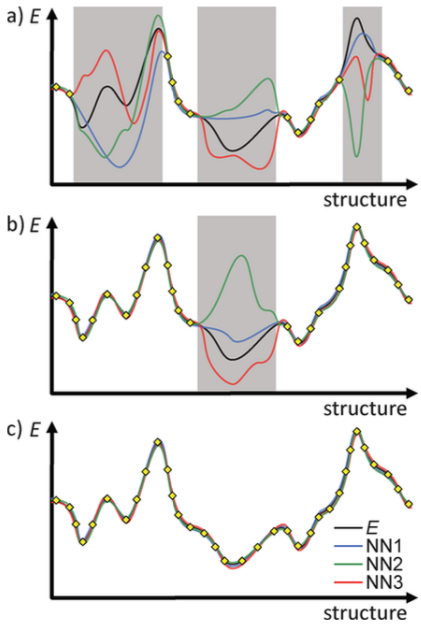


Figure 4: Different Neural Network fiting the data set outputs

The big advantage of FFNN is that they are very flexible with large numbers of fitting parameters. It provides accurate representations and no knowledge about physical principles is required. Energies and forces are calculated way much faster compared to DFT. However, there are important limitations:

- The size of the FFNN cannot be arbitrarily increased to adapt to large systems (the method becomes computationally more demanding).
- The dimension of inputs nodes is fixed.

Second Generation

The second generation is characterised by the transition of low-dimensional neural networks to high-dimensional ones. The limitations could be overcome by three steps:

1. Hypothesis that a large part of the atomic interactions can be described by interactions of the atoms with their local chemical environments.

2. A new type of descriptor with exact translational, rotational and permutation invariance.

3. The use of active learning for the construction of the training set in high-dimensional configuration spaces.

Single neural networks are abandonned and the exploitation of the locality of the atomic interaction energies works really well. This approximation was introduced in ML potentials and can be described like so:

$$E_{\text{short}} = \sum_{i=1}^{N_{\text{element}}} \sum_{j=1}^{N_{\text{atom}}^{i}} E_j^i \tag{6}$$

N_{atom}^{i} corresponds to the number of atoms of the respective element and E_j^i is the atomic energy contribution. The "short" term means that only short-range interactions are considered (atoms that are inside the cutoff radius (6 to 10 $\mathring{A}$)). The cutoff radius is a convergence parameter.

Once the cutoff radius has been chosen, the structural information has to be converted to a suitable input for the Neural Network Potential (NNP). This is what we call the descriptor environment. The choice of the descriptor is an important step. A type of descriptor is the Atom-Centered Symmetry Function (ACSF). It provides local structural fingerprints of the atomic environment.

So, the combination of the locality approximation with the ACSF descriptor makes avaible the use of a FFNN on each atom. each atom will have its own neural network that will predict the energy contribution of the atom E_j^i.

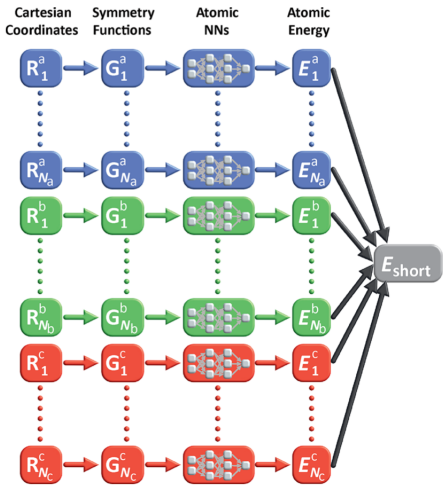


Figure 5: Structure of a second-generation high-dimensional neural network potential

Atom positions are transformed to a vector of symmetry function values G_i^u based on neighboring atoms coordinates. So, we have the same architecture for all atoms which is really good input for the neural network to save calculation time. Also, it ensures that atoms of the same element are chemically equivalent: the energy contribution is only a function of the atomic environment.

Then if an atom is added to the system, a new neural network of the respective atom is evaluated. So, the second-generation of NNP (which are HDNNP for High-Dimensional NNP) has flexible number of inputs.

Third Generation

We will briefly discuss on the third generation of NNP. This generation of NNP use even bigger HDNNP than the second one. In this generation, long-range electrostatic interactions are taken into account.

$$E_{\text{total}} = E_{\text{short}} + E_{\text{elec}} \tag{7}$$

E_{elec} is calculated the same way as E_{short}:

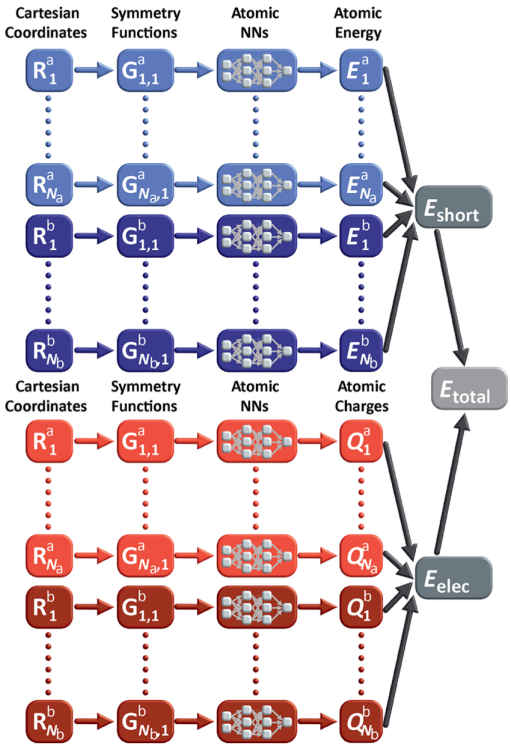


Figure 6: 3G-HDNNP Structure