The universal interface for testing atomistic potentials

Bohumir Jelinek
Center for Advanced Vehicular Systems
Mississippi State University
Testing of atomistic potentials

Task at hand

• validate *Modified Embedded Atom Method (MEAM)* interatomic potentials for Al, Si, Mg, Cu, Fe, and their alloys

Method

1. calculate basic structural properties of single crystals, formation energies of defects, and structural and elastic properties of simple compounds using MEAM

2. compare with other interatomic potentials and *ab-initio* methods
Issues

Need to learn formats of input parameter files, atomic configuration files, and output files of

- classical MD code implementing MEAM (LAMMPS)
- *ab-initio* code (VASP)

Need to create atomic configurations for

- single crystal structures, basic crystallic compounds, point defects (vacancies, interstitials, substitutions), planar defects (varying surfaces, stacking faults), and strained structures
What would help

A tool applicable to quickly evaluate basic properties from classical interatomic potentials and \textit{ab-initio} methods.

Ideally, a single universal tool would be able to

- create basic atomic configurations and manipulate them
- serve these atomistic configurations as inputs to a variety of methods/simulation codes and obtain energies

Anything like that available?
Atomistic Simulation Environment (ASE)\textsuperscript{1}

- universal Python interface to many DFT codes (calculators), with visualization, simple GUI, documentation, and tutorials
- creates molecules, crystal structures, surfaces, nanotubes, analyzes symmetry and spacegroups
- provides support for Equation of state, structure optimization, dissociation, diffusion, constrains, NEB, vibration analysis, phonon calculations, infrared intensities, MD in NVE, NVT, and NPT ensembles, STM, and electron transport
- recent support for LAMMPS by Jörg Meyer (TU München)

### Calculators working with ASE

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>vasp</td>
<td>Planewave PAW code</td>
<td>DFT</td>
</tr>
<tr>
<td>abinit</td>
<td>A planewave pseudopotential code</td>
<td>DFT</td>
</tr>
<tr>
<td>siesta</td>
<td>LCAO pseudopotential code</td>
<td>DFT</td>
</tr>
<tr>
<td>exciting</td>
<td>Full Potential LAPW code</td>
<td>DFT, LAPW</td>
</tr>
<tr>
<td>jacapo</td>
<td>ASE interface to Dacapo, planewave ultra-soft pseudopotentials</td>
<td>DFT</td>
</tr>
<tr>
<td>dftb</td>
<td>DftbPlus DFT based tight binding</td>
<td>DFT</td>
</tr>
<tr>
<td>turbomole</td>
<td>Fast atom orbital code <a href="https://wiki.fysik.dtu.dk/ase/ase/calculators/calculators.html">Turbomole</a></td>
<td>DFT, HF</td>
</tr>
<tr>
<td>FHI-aims</td>
<td>Numeric Atomic Orbital, full potential code</td>
<td>DFT, HF</td>
</tr>
<tr>
<td>fleur</td>
<td>Full Potential LAPW code</td>
<td>DFT, LAPW</td>
</tr>
<tr>
<td>emt</td>
<td>Effective Medium Theory calculator</td>
<td>EMT</td>
</tr>
<tr>
<td>Asap</td>
<td>Highly efficient EMT code (written in C++)</td>
<td>EMT</td>
</tr>
<tr>
<td>GPAW</td>
<td>Grid-based real-space PAW code</td>
<td>DFT, HF</td>
</tr>
<tr>
<td>Dacapo</td>
<td>Old interface to Dacapo. Requires Numeric python and ASE2</td>
<td>DFT</td>
</tr>
<tr>
<td>lammps</td>
<td>Classical molecular dynamics code</td>
<td>CMD</td>
</tr>
</tbody>
</table>

---

1. [https://wiki.fysik.dtu.dk/ase/ase/calculators/calculators.html](https://wiki.fysik.dtu.dk/ase/ase/calculators/calculators.html)

Bohumir Jelinek, CAVS, Mississippi State University
Transformation of general simulation cell into LAMMPS specific coordinate system

- important issue in making LAMMPS work with ASE

- LAMMPS supports non-orthogonal (triclinic) simulation boxes, but **triclinic box vectors cannot be arbitrarily oriented**

- transformation is needed from ASE general coordinate system to LAMMPS specific coordinate system and back

- implemented in **LAMMPS calculator** by Jörg Meyer
Examples from ASE documentation

Calculators

For ASE, a calculator is a black box that can take atomic numbers and atomic positions from an Atoms object and calculate the energy and forces and sometimes also stresses.

In order to calculate forces and energies, you need to attach a calculator object to your atoms object:

```python
>>> a = read('molecule.xyz')
>>> e = a.get_potential_energy()
```

Here, we used the set_calculator() method to attach an instance of the Abinit class and then we asked for the energy:

```python
>>> from ase.calculators import Abinit
>>> calc = Abinit(...) >>> a.set_calculator(calc)
>>> e = a.get_potential_energy()
```

Alternatively, a calculator can be attached like this:

```python
atoms = Atoms(..., calculator=Siesta())
```
Please have a look at the following script doc/tutorials/N2Cu.py:

```python
from ase import Atoms
from ase.visualize import view
from ase.calculators.emt import EMT
from ase.constraints import FixAtoms
from ase.optimize import QuasiNewton
from ase.lattice.surface import fcc111, add_adsorbate

h = 1.05
d = 1.10

slab = fcc111('Cu', size=(4, 4, 2), vacuum=10.0)

slab.set_calculator(EMT())
e_slab = slab.get_potential_energy()

molecule = Atoms('2N', positions=[(0., 0., 0.), (0., 0., d)])

molecule.set_calculator(EMT())
e_N2 = molecule.get_potential_energy()

add_adsorbate(slab, molecule, h, 'ontop')
constraint = FixAtoms(mask=[a.symbol != 'N' for a in slab])
slab.set_constraint(constraint)
dyn = QuasiNewton(slab, trajectory='N2Cu.traj')
dyn.run(max=6.05)

print 'Adsorption energy:', e_slab + e_N2 - slab.get_potential_energy()
#view(slab)
```
ASE requirements

For basic ASE functionality:

- **Python**
- **NumPy**

Highly recommended (to create png and eps files, and for ASE GUI)

- **matplotlib**
- **PyGTK**
Conclusions

- ASE provides a universal interface to many electronic-structure codes and LAMMPS
- ASE interface for LAMMPS and VASP was utilized in testing Al-Si-Mg-Cu-Fe MEAM alloy potentials
- Following the LAMMPS example, ASE can provide support to other classical MD codes
- ASE simplifies and increases efficiency of atomistic simulation research
Copenhagen, Denmark

Christiansborg Palace, amusement park Tivoli, City Hall Square, ferry