Recent developments in PseudoDojo, and AbiPy

M. Giantomassi

Université Catholique de Louvain
Louvain-la-Neuve, Belgium

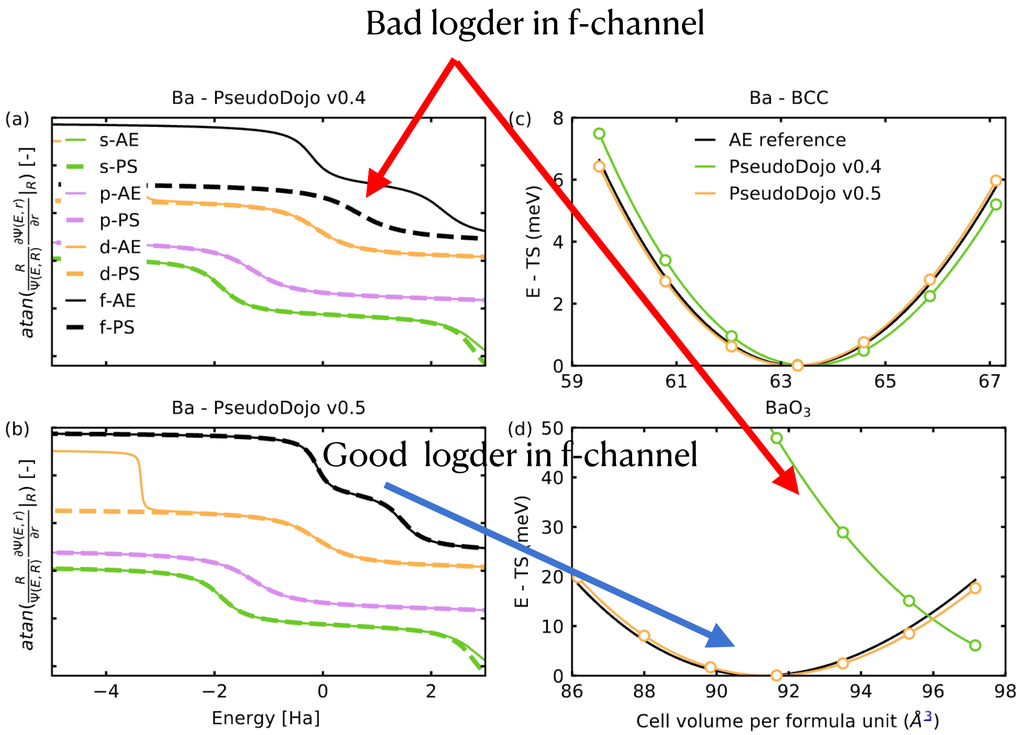
ABIDEV 2024
11th International ABINIT Developer Workshop
Saint-Paulin, Québec, Canada, June 30th – July 4th, 2024

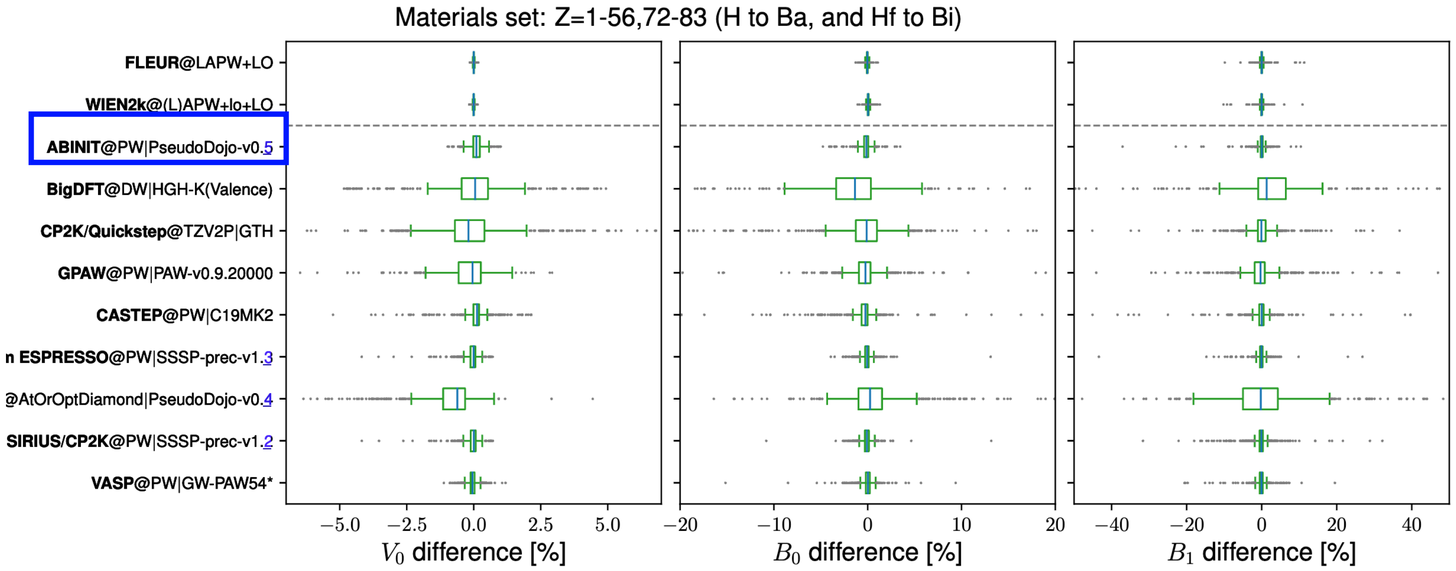
News from the PseudoDojo

- New version 0.5 with improved NC pseudos
- Two post-docs in Gian-Marco's group (C. Hargreaves, W. Jing) working on:
  - new website generated automatically with python from GitHub repos
  - generation, optimisation and validation of new pseudos
  - machine learning techniques applied to pseudo potential optimisation

PseudoDojo v0.5

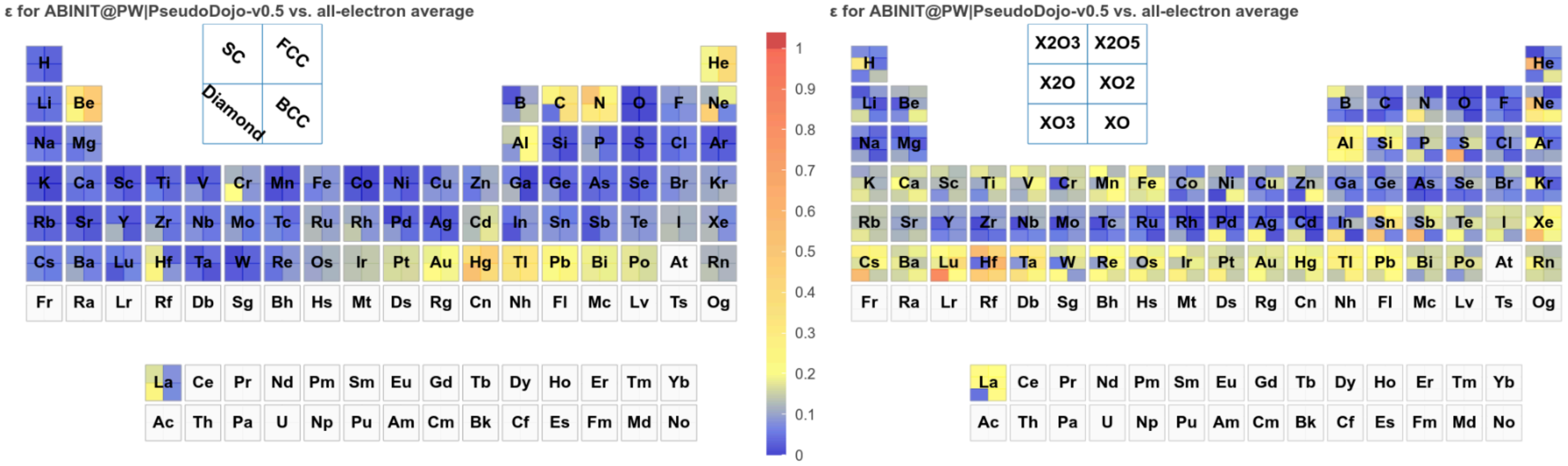
- NC pseudos validated against AE equations for states (EOS): four elemental solids and six oxides
- In v0.5, we have new pseudos including *f*-projectors for unbound states (*e.g.* Ba pseudo)
- Accuracy comparable to VASP-PAW





Bosoni et al. Nature Reviews Physics, **6**, 45–58 (2024)

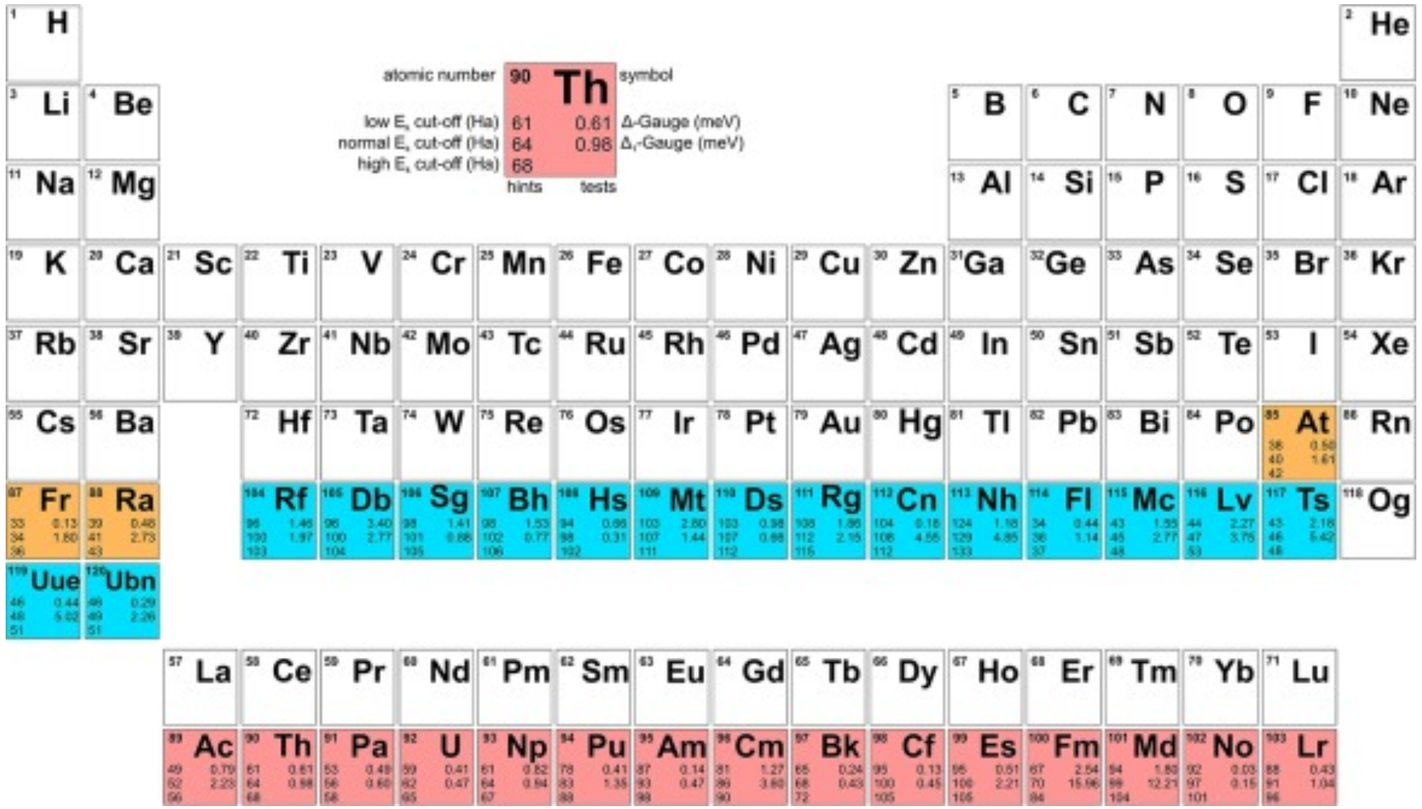
Error metrics for NC pseudos (v0.5)



- NC pseudos for lantanides, actinides and super-heavy elements (SHE) were not available at that time!
- But in the past months, we have been working hard on these elements …

NC pseudos for actinides and SHE

- Validated against AE EOS (one elemental solid, bands code as reference)
- Include semi-core states to improve transferability (ecut ~ 120/160 Ha)
- Will be made available when the new pseudodojo website will be up and running

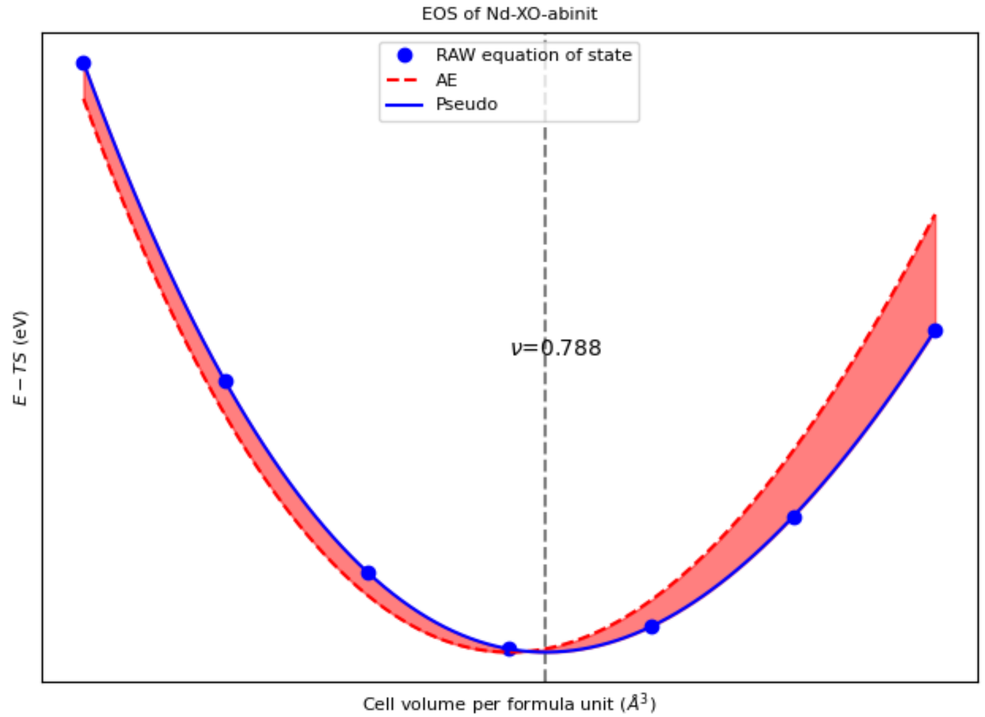


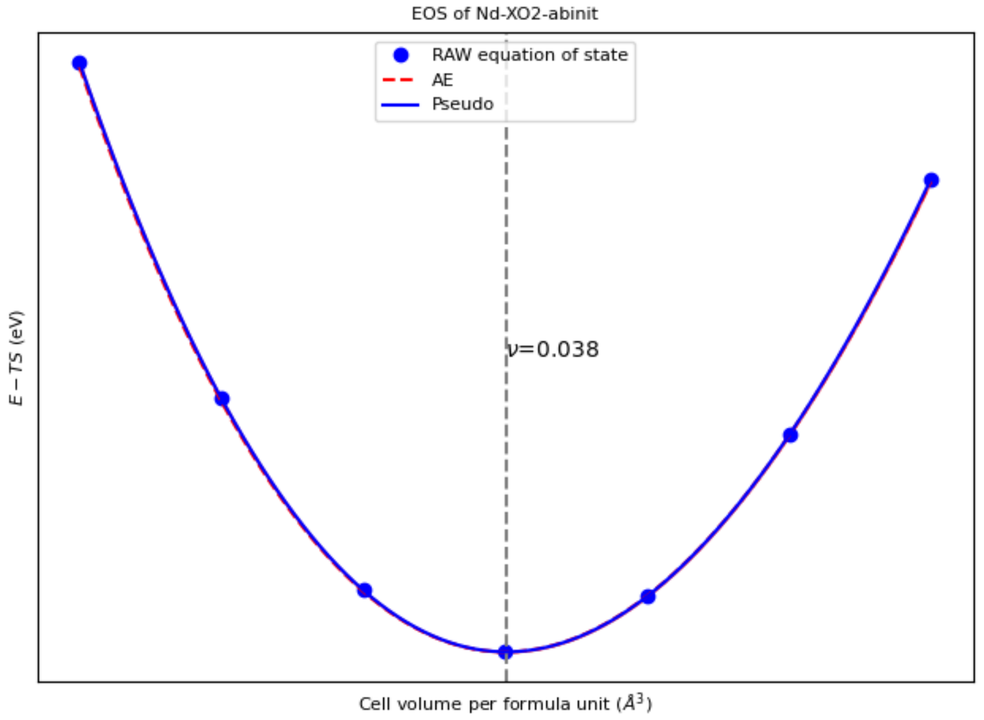
Tantardini *et. al.* Computer Physics Communications, 295, (2024), 109002

NC pseudos for lantanides

https://github.com/PseudoDojo/pseudodojo_experiments

- Preliminary pseudos available on GitHub (ecut ~ 70/90 Ha)
- Under validation using AE EOS from Bosoni *at al.* Initial results are encouraging but …
- AIIDA python workflow require some additional tuning:
  - Increase *nband* to convergence the SCF cycle
  - Read WFK from previous volume to avoid jumps in E(V)
  - New workflows based on ABINIT + jobflow to address these limitations





ABINIT with UPF2 pseudos (I)

- Only NC pseudos are supported (no PAW)
- Both scalar and fully relativistic (UPF2 uses $J = L + S$ while ABINIT uses $L \cdot S$)

Pros:

- Format supported by different *ab-initio* codes
- Useful for cross-validations studies
- Provides pseudized orbitals (needed to init the SCF cycle, KS+U, automatic Wannierization …)

Cons:

- UPF2 uses a much large radial mesh to represent orbitals wrt to psp8
- Discontinuities may appear at “large” r (numerical issues in oncvpsp?)
- For some elements, QE and Castep agree with ABINIT and AE only when the radial mesh is truncated at the same (low) r_{cut}

ABINIT with UPF2 pseudos (I)

- New input variable *wfinit* to initial wave functions at the beginning of the SCF cycle:

  0: Start from random wave-functions (**default**)

  1: Use atomic orbitals + random numbers.

  2: Use atomic orbitals without random numbers.

- New input variable *vloc_rcut* to cut the radial mesh, compute α-term and Bessel transforms
  - ABINIT with psp8 uses 6 Bohr
  - QE with UPF2 truncates at 10 Bohr

Technical issues that should be addressed before releasing new NC tables

- Make sure $V_{\rm loc}(r)$ does not present discontinuities at the oncvpsp level
- Generate UPF2 files with unbound states (useful for automatic wannierization)
- Meta-GGA support (more in Matthieu's talk)

News from AbiPy

- New API to use PseudoDojo pseudos from python (useful for scripts or HT workflows)
- One post-doc working in Gian-Marco's group (A. Akhtar) focusing on:
  - ML-techniques to accelerate ABINIT calculations
- One PhD student in Gian-Marco's group (H. Yu) working on:
  - Systematic validation of the accuracy of ML-potentials
  - Comparison between ABINIT phonons and finite displacement methods with ML

abipsp.py script

- AbiPy script to fetch pseudo potential tables from GitHub and install them in ~/.abinit/pseudos
- Provides versioning, checksum validation, json files with metadata and cutoff hints
- Usage: abipsp.py COMMAND

```
subcommands:
  Valid subcommands

  {avail,list,install,show,element,mkff}
                        sub-command help
    avail               Show available pseudopotential repos.
    list                List installed pseudopotential repos.
    install             Install pseudopotential repositories by name(s). Use `avail` command to get repo names.
    show                Show info on pseudopotential table(s).
    element             Find all pseudos in the installed tables for the given element (symbol or znucl).
    mkff                Call Abinit to compute PSPS.nc files from a list of pseudos and show results.
```

- abipsp.py install ONCVPSP-PBEsol-SR-PDv0.4

- The **avail** command shows the list of installed tables

```
abips.py avail
List of available pseudopotential repositories:

ps_generator    ps_type    xc_name    relativity_type    project_name      version  installed    name
--------------  ---------  ---------  -----------------  --------------  ---------  -----------  ----------------------
ONCVPSP         NC         PBEsol     SR                 PD                    0.4  True         ONCVPSP-PBEsol-SR-PDv0.4
ONCVPSP         NC         PBEsol     FR                 PD                    0.4  False        ONCVPSP-PBEsol-FR-PDv0.4
ONCVPSP         NC         PBE        SR                 PD                    0.4  False        ONCVPSP-PBE-SR-PDv0.4
ONCVPSP         NC         LDA        SR                 PD                    0.3  True         ONCVPSP-LDA-SR-PDv0.3
ONCVPSP         NC         LDA        SR                 PD                    0.4  True         ONCVPSP-LDA-SR-PDv0.4
ATOMPAW         PAW        LDA        SR                 JTH                   1.1  False        ATOMPAW-LDA-JTHv1.1
ATOMPAW         PAW        PBE        SR                 JTH                   1.1  False        ATOMPAW-PBE-JTHv1.1

References for the PD project:
        - https://doi.org/10.1016/j.cpc.2018.01.012
References for the JTH project:
        - https://www.sciencedirect.com/science/article/abs/pii/S0010465513004359?via%3Dihub
```

- Used in AbiPy and atomate2 to implement workflows

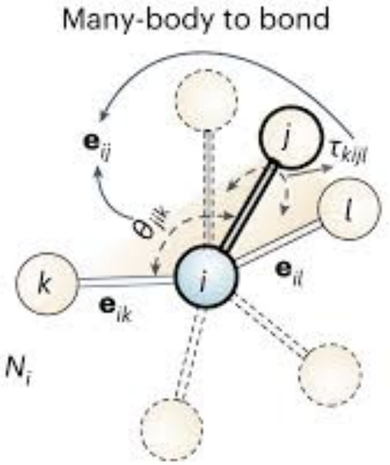
```python
from abipy.flowtk.psrepos import get_oncvpsp_pseudos
pseudos = get_oncvpsp_pseudos(xc_name="PBE", version="0.4")
```

- In the input file, one can use: pp_dirpath "$HOME/.abinit/pseudos/ONCVPSP-PBE-SR-PDv0.4/"

Machine learning potentials

+

+



- We do not deal with training or re-training of models
- The focus is on using *universal* ML potentials to accelerate *ab-initio* calculations
- Universal ML potentials are out-of-the box potentials covering most of the periodic table (MACE, CHGNET, M3GNET, …)
- We also support models trained from scratch such as nequip, allegro, deepmd …

abiml.py script

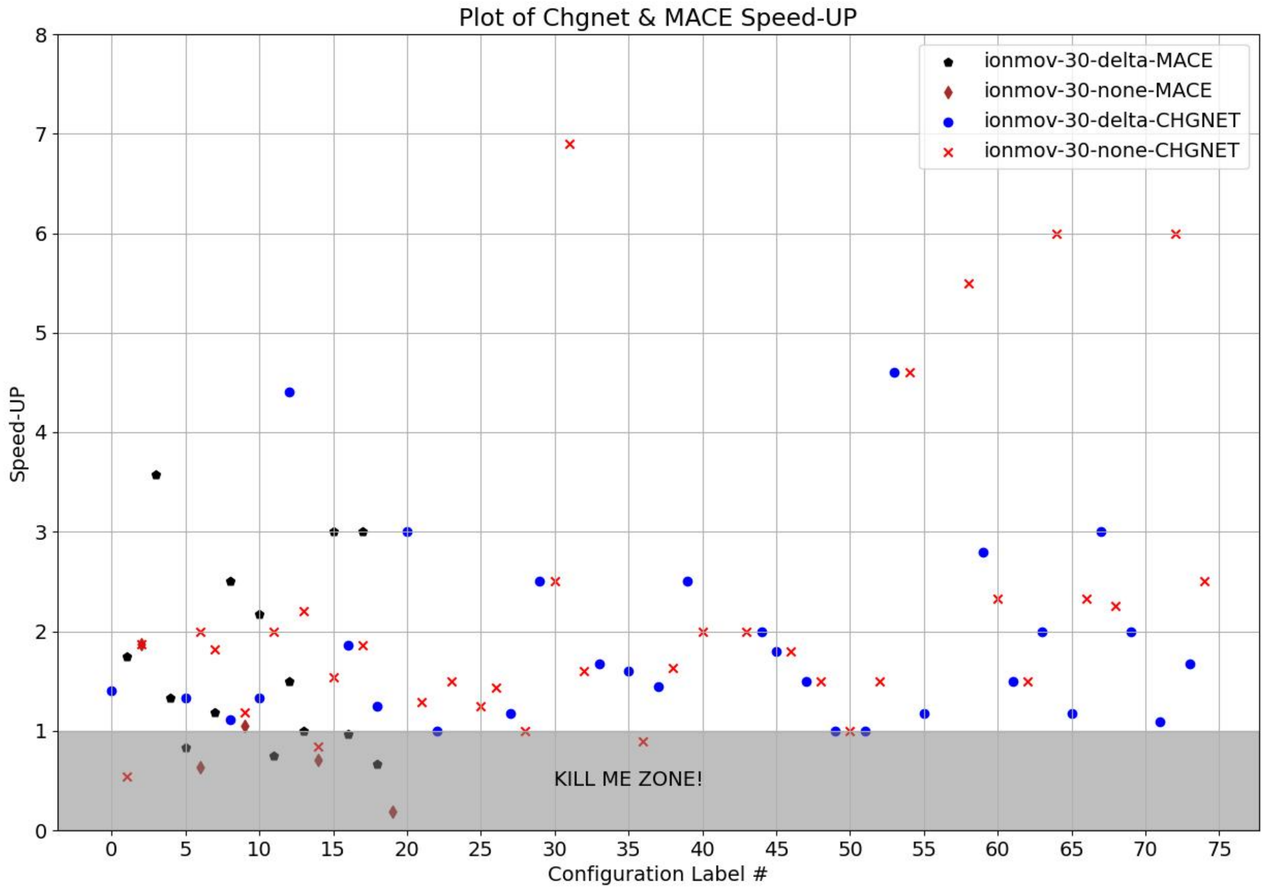
- Unified interface to perform calculations with different ML-potentials
- Relies on ASE and ASE calculators for structural relaxations, MD, NEB, …
- ML-phonons with phonopy
- Possible applications:
  - accelerate *ab-initio* calculations
  - validate ML-potentials with *ab-initio* data (parity plots for energies, forces, stresses)

```
Commands:
  abinit-relax  Interact with ABINIT in hybrid relaxation mode.
  compare       Compare different neural networks.
  cwf-eos       Compute CWF EOS with ML potentials.
  gs            Compute ground-state properties and magnetic moments with...
  install       Install NN potentials in the environment using pip.
  md            MD simulation with ASE and ML potential.
  mneb          Multi-NEB calculation with ASE and ML potential.
  neb           NEB calculation with ASE and ML potential.
  order         Generate ordered structures from CIF with partial...
  ph            Use phonopy and ML potential to compute phonons.
  phddb         Use phonopy and ML potential to compute phonons and...
  relax         Structural relaxation with ASE and ML potential.
  scan-relax    Generate 3D mesh of (nx,ny,nz) initial positions and...
  show          Show the NN potentials installed in the environment.
  validate      Compare ab-initio energies, forces, and stresses with...
```

abinit-relax

- New relaxation algorithm: *ionmov 30* (work done by A. Akhtar)
- ABINIT invokes *abiml.py* at runtime to relax the structure with the ML potential
- Need to correct for the diff. in forces/stresses between ML and ABINIT (Delta-Mode)

Speedup is computed as the ratio between the no. relaxation steps in fully ab-initio mode and the no. steps required in hybrid ABINIT + ML-potential



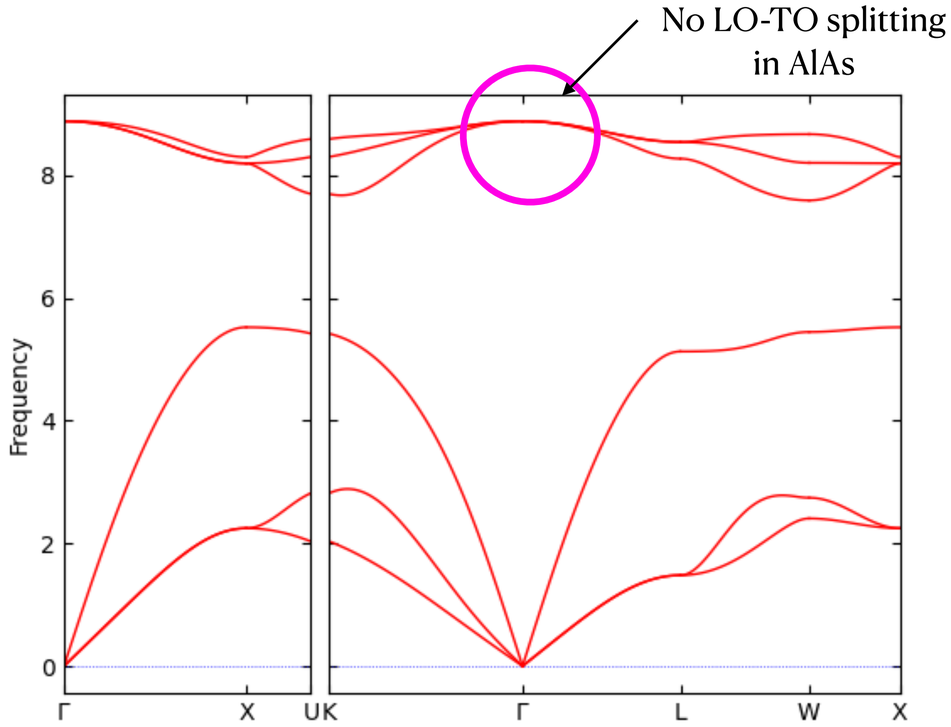
We are now performing a more systematic investigation using a dataset of ~100 structures (bulks, defects, surfaces ...)

Phonons with ML-potentials

- Read structure from FILE
- Relax structure with ML, use phonopy to build supercell and compute phonons
- Can compare with ABINIT phonons if DDB file is available (phdbb command)

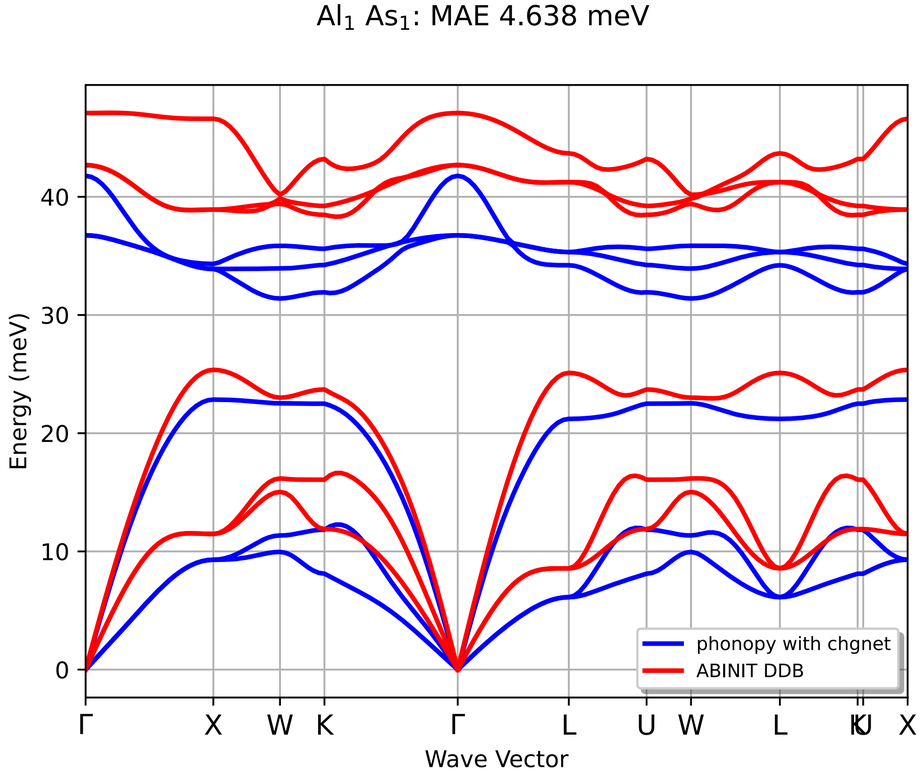
abiml.py ph FILE

Relax structure from FILE and use phonopy to compute phonons



abiml.py phddb DDB_FILE

Compare ABINIT phonons with ML starting from a DDB_FILE



Systematic assessment of various universal machine-learning interatomic potentials

- abiml.py has been used to analyse the accuracy of:
  - relaxations and formation energies (19998 unaries + binaries)
  - phonons (101 systems computed with ABINIT)

uMLIP	volume	a	b	c	α	β	γ
CHGNet	3.16	2.03	2.07	2.44	0.75	0.62	1.19
M3GNet	2.97	2.04	2.09	2.46	0.89	0.73	1.24
MACE	5.22	2.01	2.11	2.58	0.73	0.59	1.13
ALIGNN	7.85	3.42	3.42	3.61	0.94	0.86	1.32

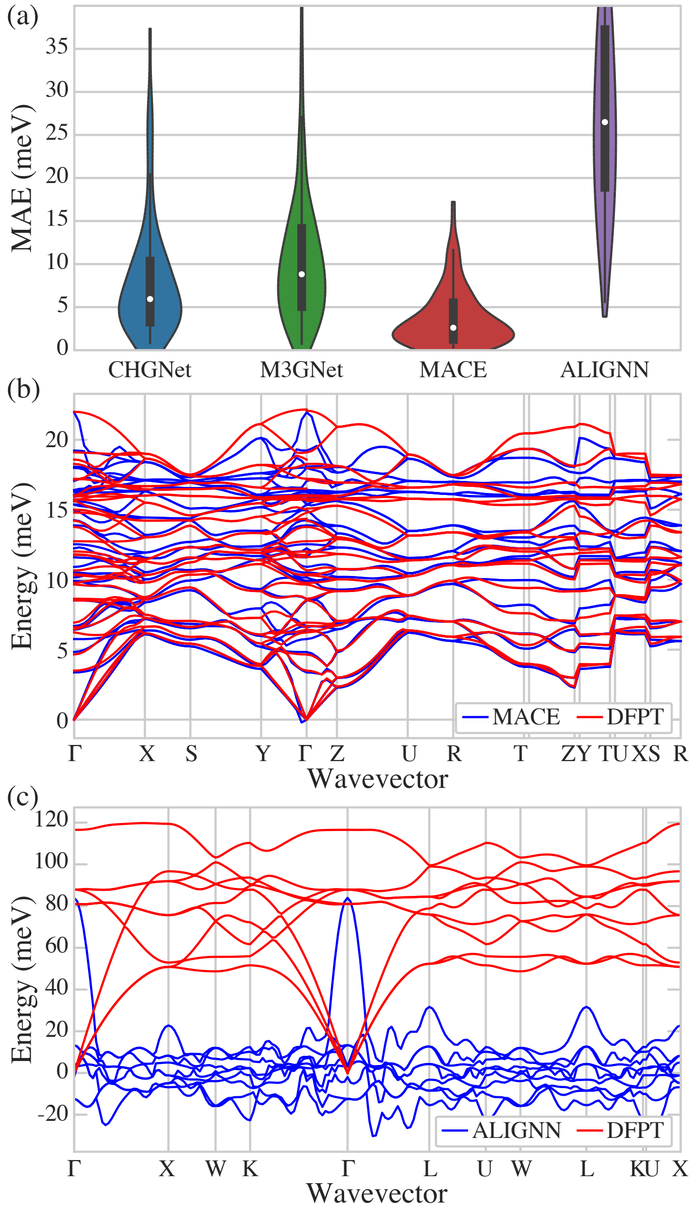
TABLE 3 Mean Absolute Relative Error (MARE in %) of the different uMLIPs in predicting the volume, lattice parameters, and angles.

uMLIP	MIN_MAE	MAX_MAE	MEAN_MAE
CHGNet	0.82	37.34	8.12
M3GNet	0.74	40.20	10.41
MACE	0.31	17.22	3.71
ALIGNN	5.60	75.38	29.36

TABLE 4 Minimum, maximum and average MAE in meV for the phonon band structures computed from different uMLIPs.

H. Yy et al. preprint arXiv:2403.05729

ML-phonons vs ABINIT



Thank you for your attention!

News from the EPH code

Already implemented:

- Interpolation of $\Delta_{\mathbf{q}\kappa\alpha} V$ in **q**-space including dipoles and quadrupoles
- e-ph self-energy and transport properties (SERTA, MRTA, IBTE)
- phonon line-widths and isotropic superconductivity
- ZPR with sum over bands plus Sternheimer to accelerate convergence
- cumulant expansion and Kubo-Greenwood (Joao + MG + Matthieu)

New developments:

- gstore_t object to store e-ph matrix elements (distributed over spins, **k**-points, **q**-points and ph-modes)
- Use eph_task = 11 to compute all the $g(\mathbf{k}, \mathbf{q})$ in parallel and save the results to GSTORE.nc
- gstore_* input variables to select:
    - subset of bands (energy window or range)
    - select **k/k+q** inside an energy window (useful for metals)
    - specify whether **k** or **q** should be in the full BZ or both

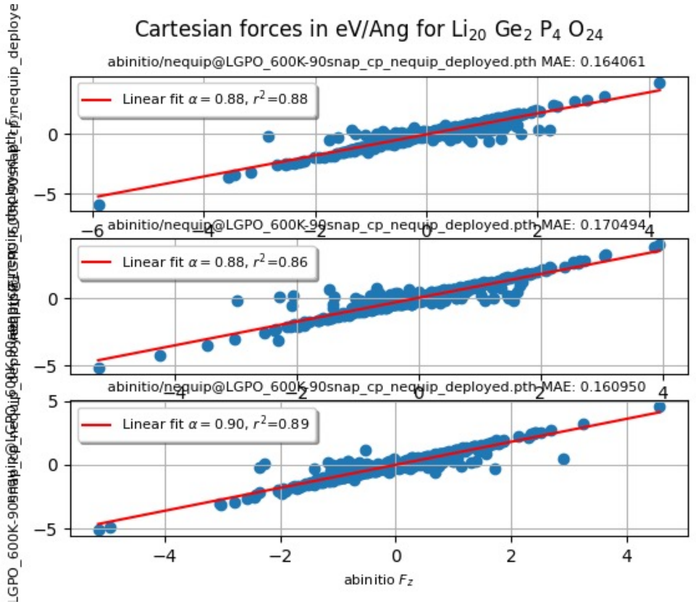
Why GSTORE?

- GSTORE.nc was initially developed to integrate ABINIT with Yambo but it is not just a file format
- One can reconstruct a *gstore_t* object from file, and this allows one to:
  - store all $g(\mathbf{k}, \mathbf{q})$ in memory (MPI-distributed)
  - implement "post-processing tools"
- This philosophy shines for algorithms in which the same $g(\mathbf{k}, \mathbf{q})$ is used many times e.g.:
  - iterative algorithms (see Vasilii's talk on variational polar on equations on Friday)
  - T-dependent equations (*e.g.:* anisotropic superconductivity)
- By default, GSTORE.nc stores e-ph matrix elements computed from the input WFK file but:
  - It is possible to densify the **q**-mesh for phonons (*eph_ngqpt_fine*)
  - In the forthcoming versions, it will be possible to use Wannier interpolation starting from GSTORE.nc
  - The issue with the wf-gauge is solved thanks to the new interface with *wannier90* coded by HeXu

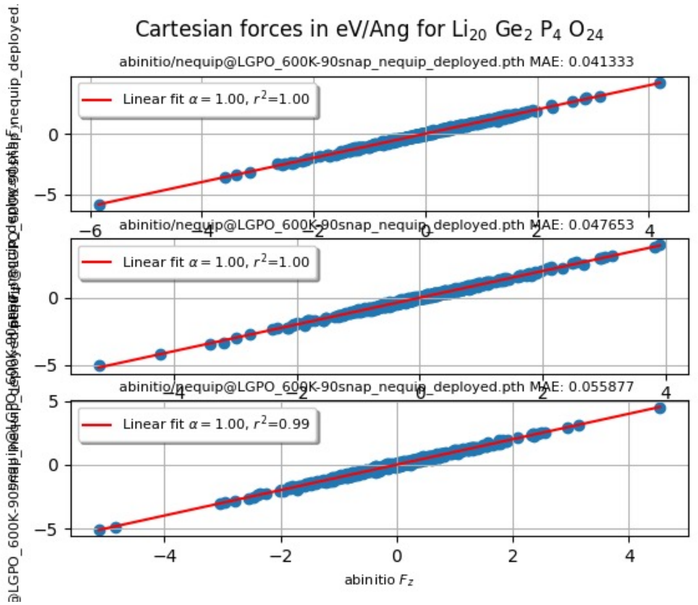
abiml.py validate

- Read configurations (structures/forces/stresses) from HIST.nc or XDATCAR
- Compare *ab-initio* results with ML-predicted ones
- Supports universal ML-potentials and models trained from scratch (*e.g.* allegro)

Allegro trained on 90 structures t-LGPO: validation



- Trained on the **CP forces** (90 structures) validated on the **CP forces** (10 structures).



- Trained on the **VASP-recalculated** forces (90 structures) validated on the **VASP-recalculated** forces (10 structures).