Facilitating finding and selecting potentials from the Interatomic Potentials Repository

Lucas M Hale
Zachary T Trautt
Chandler A Becker
Rebekah Newby

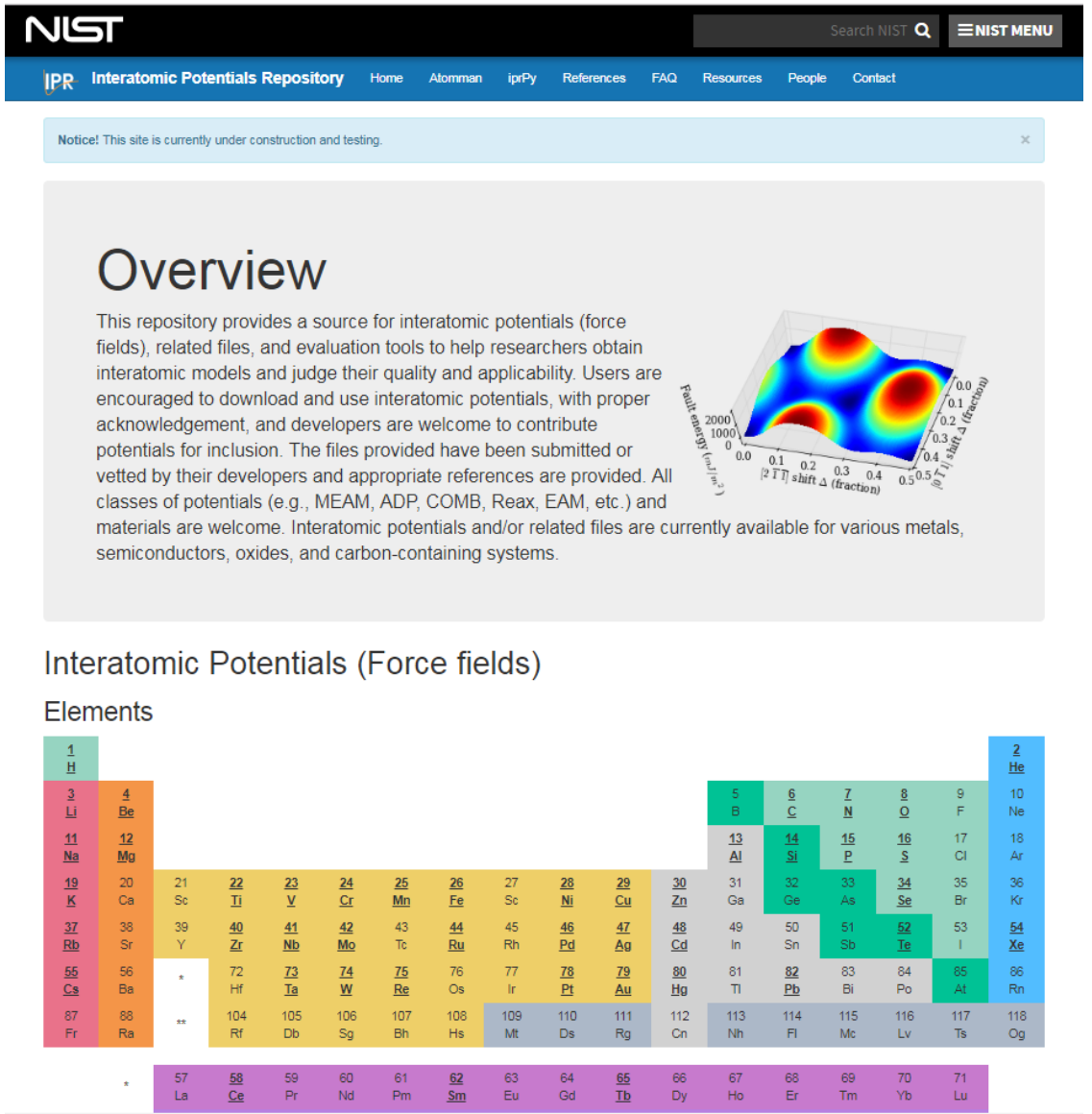
IPR

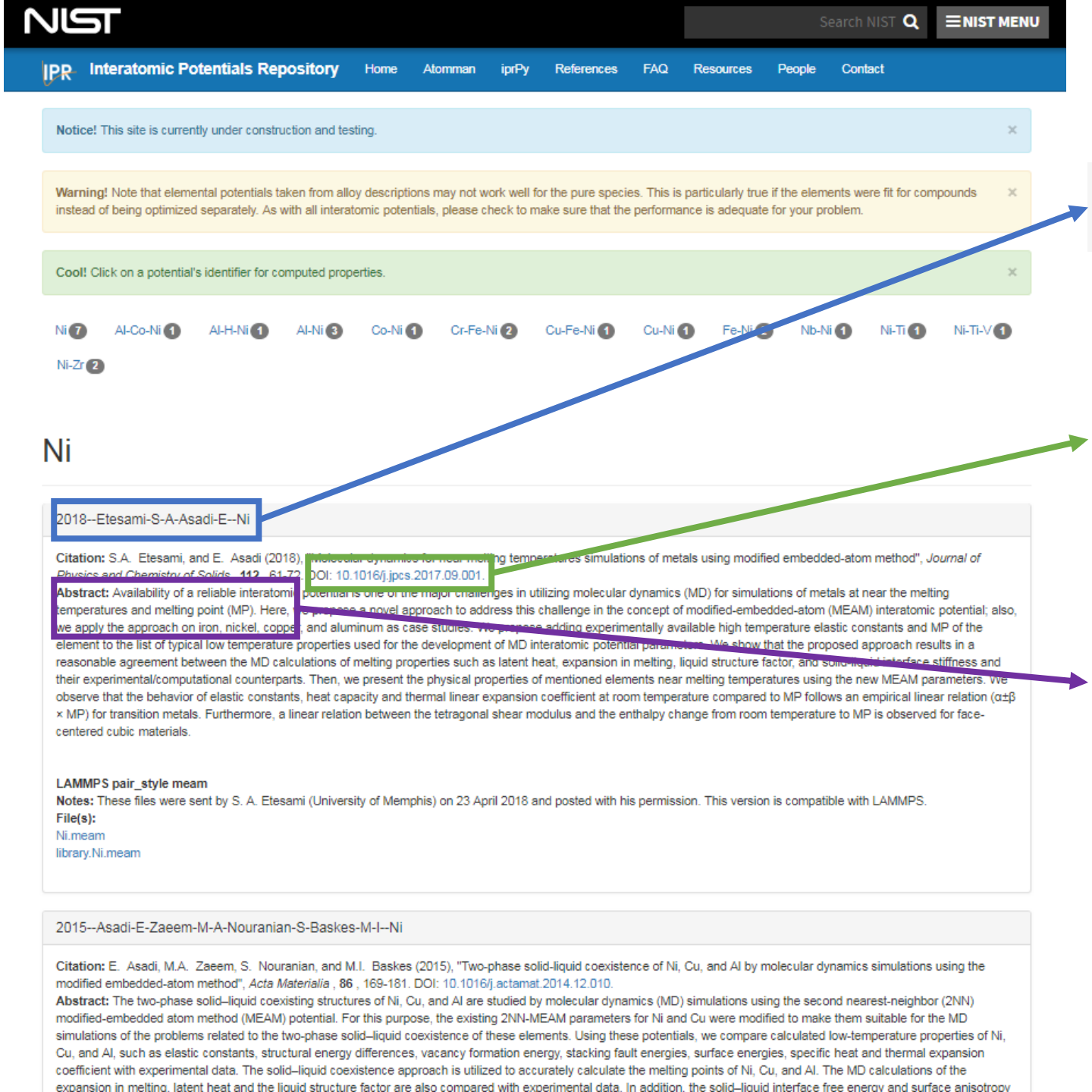
NIST
National Institute of Standards and Technology
U.S. Department of Commerce

See preview at https://www.ctcms.nist.gov/potentials/testing/

Interatomic Potentials Repository 2.0

- New appearance and backend!
- More consistent content representation
- Reaching 200 interatomic models
- Property evaluations updated
- Documentation for Python tools
- Working out final issues before release





Publication-derived unique identifiers

2018--Etesami-S-A-Asadi-E--Ni

DOI links to papers

DOI: 10.1016/j.jpcs.2017.09.001.

Abstracts for all associated citations

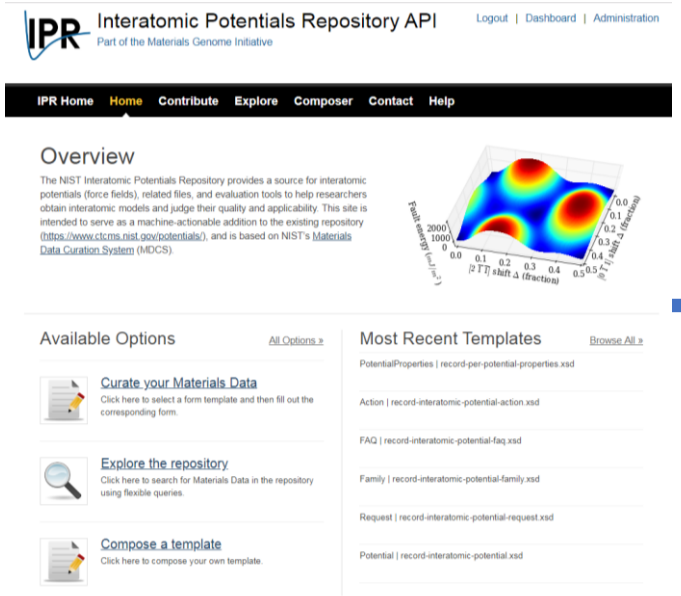
Abstract: Availability of a reliable interatomic
temperatures and melting point (MP). Here, w
we apply the approach on iron, nickel, copper

Consistent representations of:

- Citations
- Implementation formats
- Notes and descriptions

Content Now Stored on a CDCS Database

Curator at potentials.nist.gov



- "Potential" record for each entry
- Templates for other data types

XSL

HTML fragments

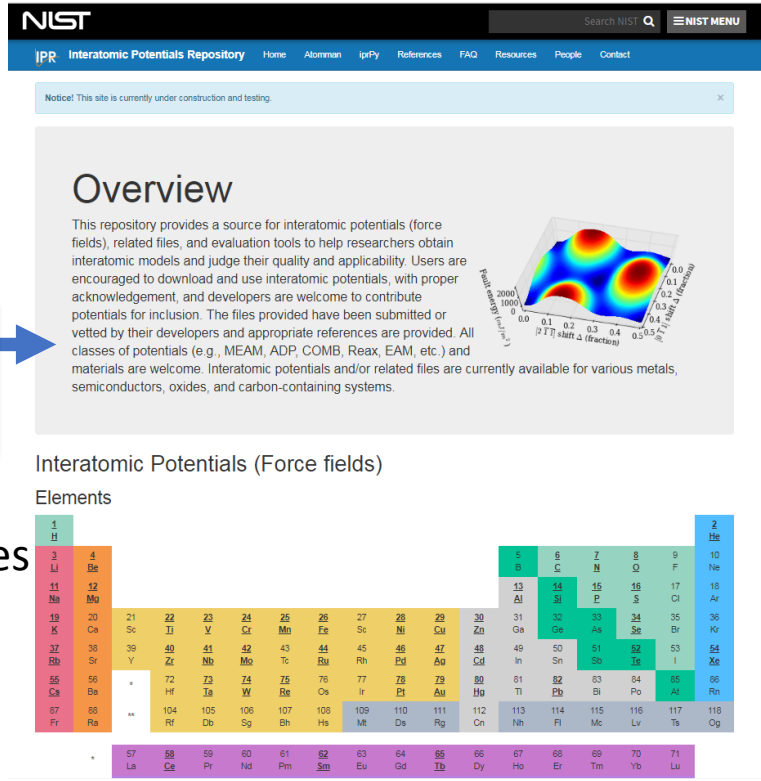
- Build formatted content

Python

XSL

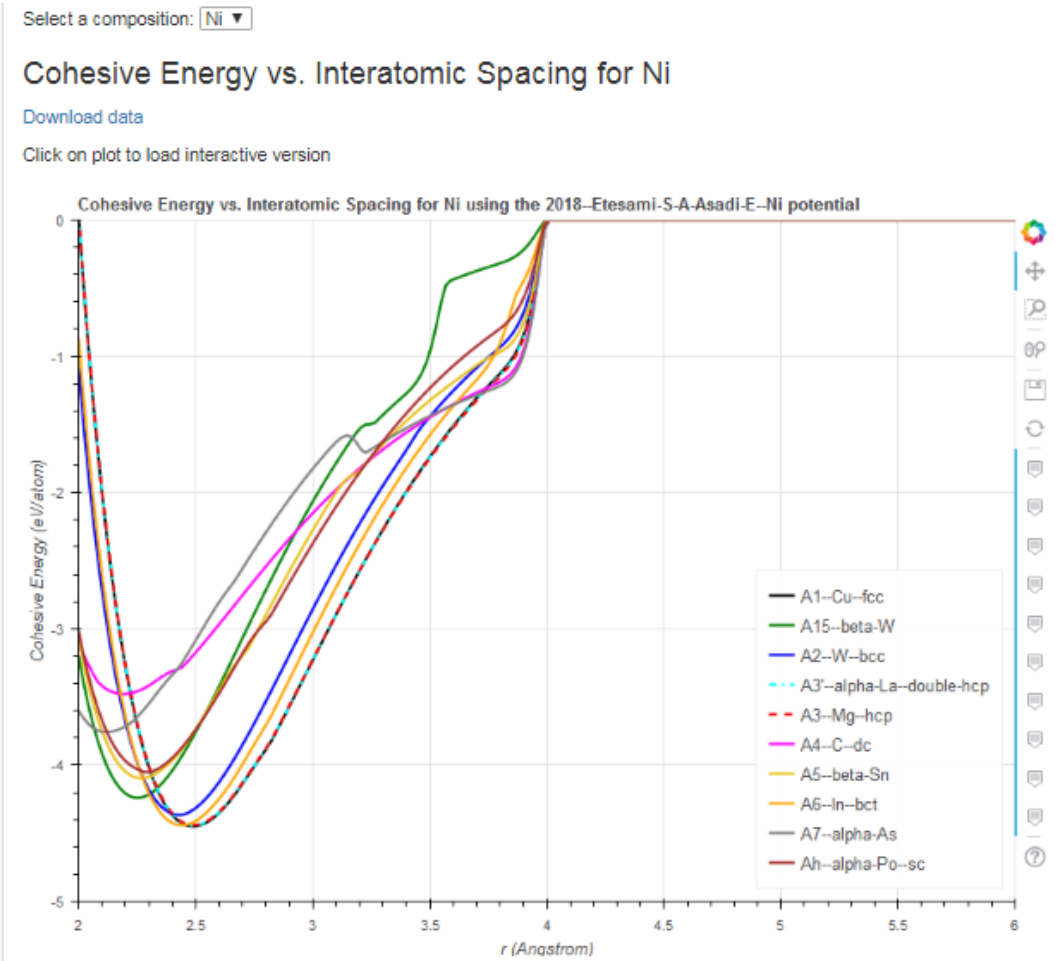
- Compile by species
- Build list of links

Repository at ctcms.nist.gov/potentials



Property Predictions

- Click on Potential ID to see computed properties
- Full methodology description
- Interactive plots
- Show by composition



prototype	method	E_{coh} (eV)	a_0 (Å)	b_0 (Å)	c_0 (Å)	α (degrees)	β (degrees)	γ (degrees)
A1--Cu--fcc	dynamic	-4.45	3.52	3.52	3.52	90.0	90.0	90.0
A3'--alpha-La--double-hcp	dynamic	-4.4454	2.4874	2.4874	8.1423	90.0	90.0	120.0
A3--Mg--hcp	dynamic	-4.441	2.4858	2.4858	4.0775	90.0	90.0	120.0
A2--W--bcc	dynamic	-4.3666	2.8073	2.8073	2.8073	90.0	90.0	90.0
A15--beta-W	static	-4.2399	4.5033	4.5033	4.5033	90.0	90.0	90.0
A5--beta-Sn	static	-4.1082	4.4211	4.4211	2.3532	90.0	90.0	90.0
Ah--alpha-Po--sc	static	-4.0531	2.2972	2.2972	2.2972	90.0	90.0	90.0
A4--C--dc	static	-3.4783	5.0624	5.0624	5.0624	90.0	90.0	90.0

JARVIS-FF

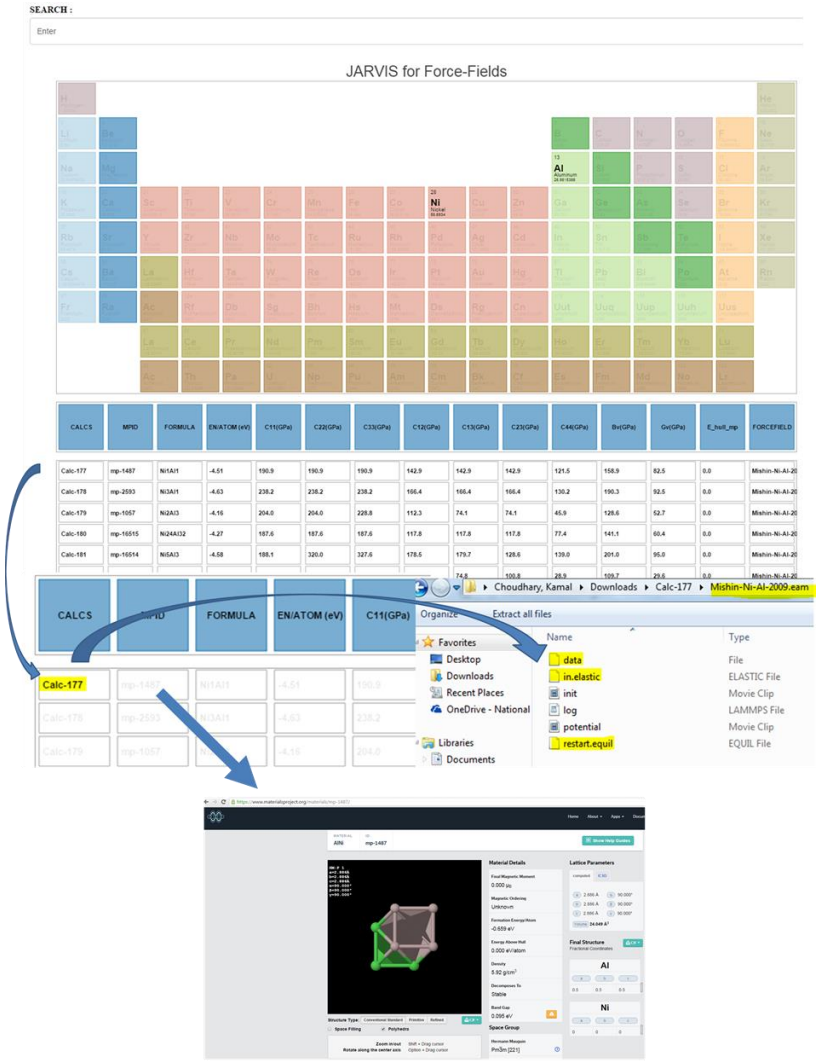
- Scope:
  to **facilitate the user in choosing** the right potential for their needs by providing comparison of material properties computed with as many force fields as possible

- more than **3000 materials** through more than **25000 classical force-field**

- Force-fields obtained from NIST's IPR and LAMMPS-potential folder. DFT relaxed structures obtained from Materials-Project
- High-throughput LAMMPS calculations

- **Relaxed structures, Elastic properties, Surface energies, Vacancy formation energies**
- Properties available for download, code available on github:

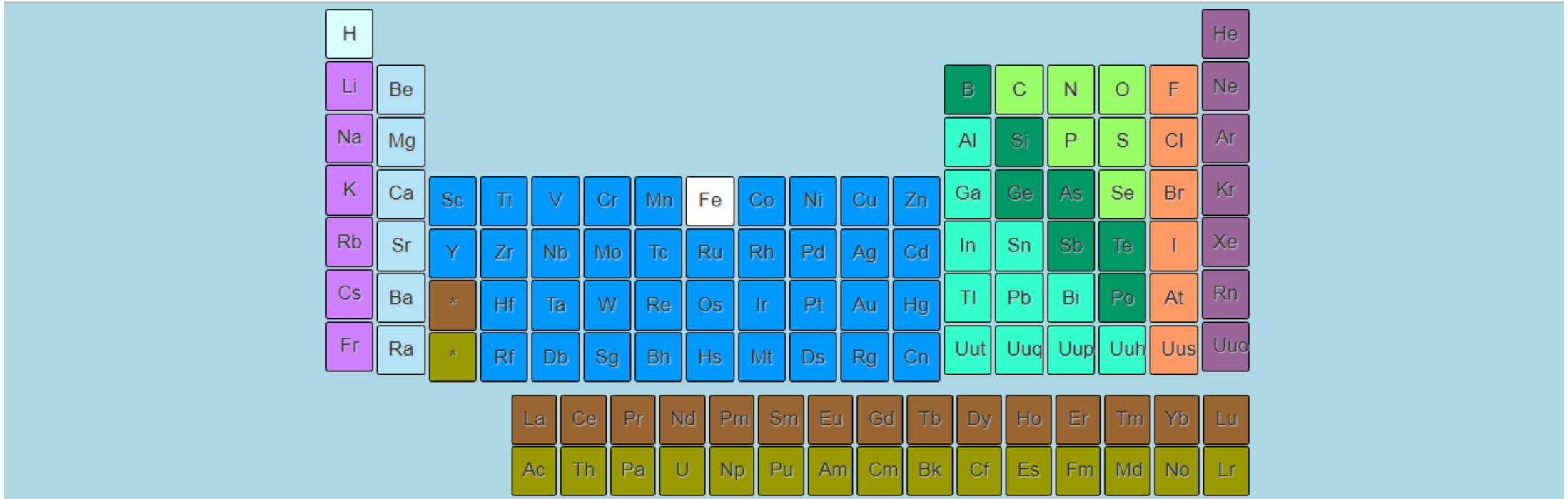
https://github.com/usnistgov/jarvis



https://github.com/usnistgov/jarvis

Jarvis-FF: Directly Compare Potentials to DFT

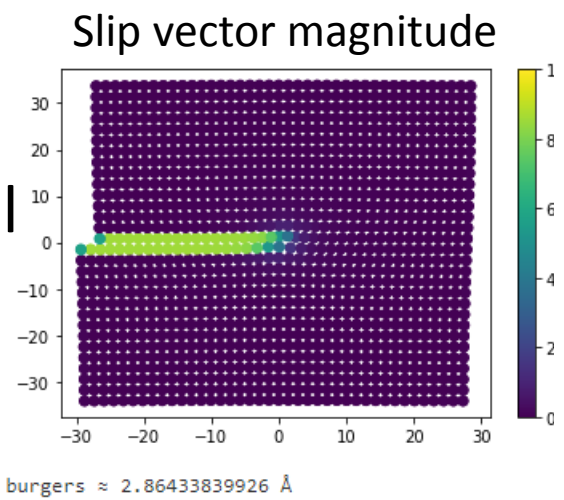
F. Tavazza, K. Choudhary

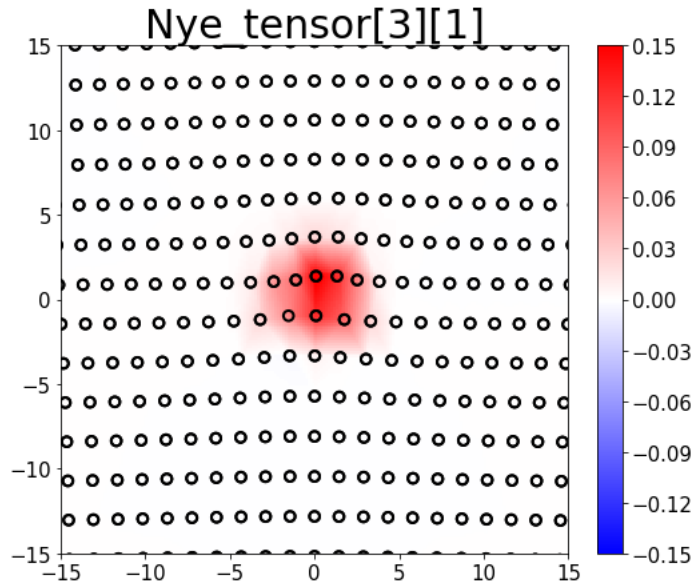


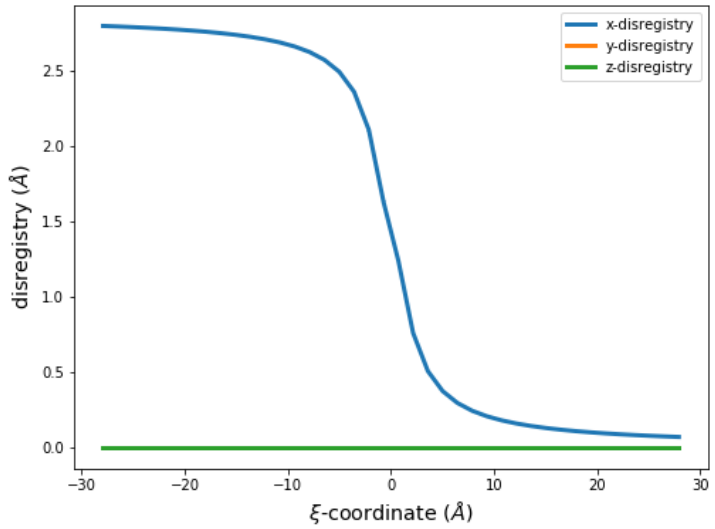
JARVIS-ID	Formula	Space-group	Force-field	B_v (GPa)	G_v (GPa)	Surface energy	Vacancy energy	MPID
JLMP-1514	Fe	Im-3m	Fe_5.eam.fs	178.2	89.4	Available	Available	mp-13
JLMP-1437	Fe	Im-3m	Al-Fe.eam.fs	177.9	89.4	Available	Available	mp-13
JLMP-1215	Fe	Im-3m	Fe-Ni.eam.alloy	177.8	89.3	Available	Available	mp-13
JLMP-1564	Fe	Im-3m	Fe.eam.fs	177.9	89.3	Available	Available	mp-13
JLMP-1510	Fe	Im-3m	Fe_2.eam.fs	177.8	89.3	Available	Available	mp-13
JLMP-1684	Fe	Im-3m	V-Fe.eam.fs	177.9	89.4	Available	Available	mp-13
JLMP-1569	Fe	Im-3m	Fe-P.eam.fs	177.9	89.4	Available	Available	mp-13
JLMP-1201	Fe	Im-3m	FeNiCr.eam.alloy	48.3	46.2	Available	Available	mp-13
JLMP-1520	Fe	Im-3m	Fe-C_Hepburn_Ackland.eam.fs	179.0	89.4	Available	Available	mp-13
JLMP-1187	Fe	Im-3m	FeNiCr_Bonny_2013_ptDef.eam.alloy	164.3	72.0	Available	Available	mp-13
JLMP-1725	Fe	Im-3m	Fe.set	168.2	89.6	Available	Available	mp-13
JLMP-1174	Fe	Im-3m	FeCuNi.eam.alloy	177.8	89.3	Available	Available	mp-13

atomman: Atomistic Manipulation Toolkit

- Represent atomic systems in Python
- Designed generically and for classical atomistics
- Dislocation generation and analysis functions
- Converters to/from ase, pymatgen, spglib
- Tools for setting up, running and analyzing LAMMPS calculations
- Supports the creation of Python calculation scripts around LAMMPS simulations







Generalized Stacking Faults of FCC Metals

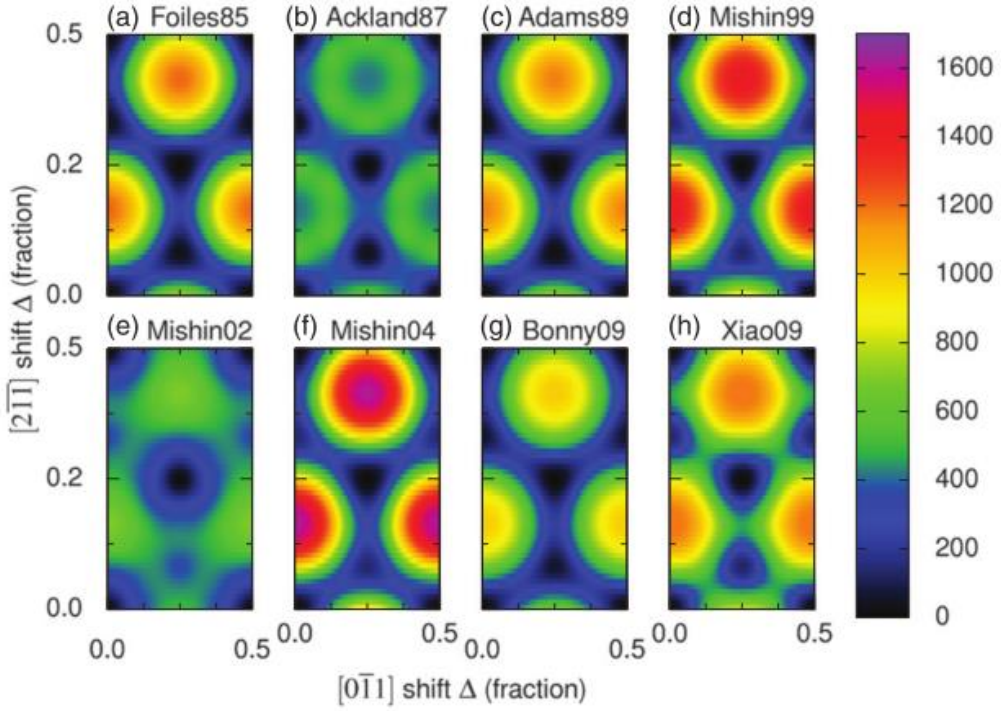
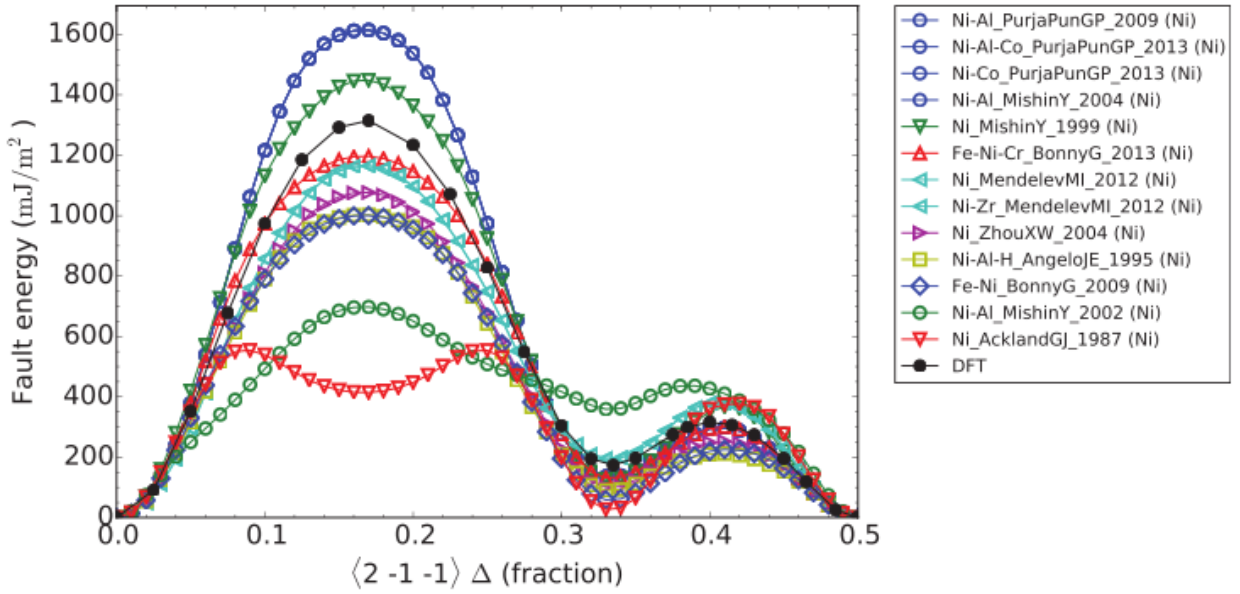


Figure 1. GPFE for the (111) orientation. Colors denote energy in mJ/m^2.



C.A. Becker, F. Tavazza, L.E. Levine, Philosophical Magazine 91(27) (2011) 3578-3597.
Z.T. Trautt, F. Tavazza, C.A. Becker, Model Simul Mater Sc 23(7) (2015).

iprPy Computational Framework

Iterate over

- Potentials
- Calculation methods
- Crystal structures
- Temperatures
- Configurations
- System dimensions
- Simulation parameters
- Other parameters

Which interatomic potential should I use?

How do I address quantitative and qualitative uncertainty?

Are my results sensitive to parameter choice?

Are my results sensitive to methodology choice?

Can others use, learn and verify my work?

iprPy Computational Framework

“Insanity is doing the same thing over and over again, but expecting different results.”

High-throughput is doing *almost* the same thing over and over again, and expecting different results.

High-throughput ≈ insanity?

“Insanity is repeating the same mistakes and expecting different results”

Simple Execution

- Independent unit of work
  - Not dependent on other calcs
  - No knowledge of framework
- Key-value input parameter files
- Can run calculation scripts from command line:

  `python calc_*.py calc_*.in`
- Outputs JSON results
- Python package knows results schema (automatic interpret)

```
# Input script for calc_E_vs_r_scan.py

# Command lines for LAMMPS and MPI
lammps_command             lmp_mpi
mpi_command

# Potential definition and directory containing associated files
potential_file             2006--Williams-P-L--Ag--LAMMPS--ipr1.json
potential_dir              2006--Williams-P-L--Ag--LAMMPS--ipr1

# Initial system configuration to load
load_file                  A1--Cu--fcc.json
load_style                 system_model
load_options
family                     A1--Cu--fcc
symbols                    Ag
box_parameters

# System manipulations
a_uvw
b_uvw
c_uvw
atomshift
sizemults                  3 3 3

# Units for input/output values
length_unit
pressure_unit
energy_unit
force_unit

# Run parameters
minimum_r                  2.0
maximum_r                  6.0
number_of_steps_r          200
```

Simple Execution

- Independent unit of work
  - Not dependent on other calcs
  - No knowledge of framework
- Key-value input parameter files
- Can run calculation scripts from command line:

  `python calc_*.py calc_*.in`
- Outputs JSON results
- Python package knows results schema (automatic interpret)


```
{    "calculation-E-vs-r-scan": {
        "key": "5aa662f4-17b4-4d22-955c-6d23e7d70e44",
        "calculation": {
            "iprPy-version": "0.8.1",
            "atomman-version": "1.2.2",
            "LAMMPS-version": "26 Jan 2017-ICMS",
            "script": "calc_E_vs_r_scan",
            "run-parameter": {
                "size-multipliers": {
                    "a": [0, 3],
                    "b": [0, 3],
                    "c": [0, 3]},
                "minimum_r": {
                    "value": 2.0,
                    "unit": "angstrom"},
                "maximum_r": {
                    "value": 6.0,
                    "unit": "angstrom"},
                "number_of_steps_r": 200}},
        "potential-LAMMPS": {
            "key": "19b06879-fe19-414e-a7a0-f0bbf62a94e6",
            "id": "2006--Williams-P-L--Ag--LAMMPS--ipr1",
            "potential": {
                "key": "ce32faaf-5d57-4861-a2ec-efbfc63f07e6",
                "id": "2006--Williams-P-L--Ag"}},
        "system-info": {
            "family": "A5--beta-Sn",
            "artifact": {
                "file": "A5--beta-Sn.json",
                "format": "system_model",
                "load_options": null},
            "symbol": "Ag"},
        "cohesive-energy-relation": {
            "r": {
                "value": [
                "unit": "angstrom"},
                "a": {
```

Jupyter Notebook versions

- Documentation, code and results in one sharable file

E_vs_r_scan Calculation

Lucas M. Hale, lucas.hale@nist.gov, *Materials Science and Engineering Division, NIST.*

Chandler A. Becker, chandler.becker@nist.gov, *Office of Data and Informatics, NIST.*

Zachary T. Trautt, zachary.trautt@nist.gov, *Materials Measurement Science Division, NIST.*

Version: 2018-06-24

Disclaimers

Introduction

The E_vs_r_scan calculation calculation creates a plot of the cohesive energy vs interatomic spacing, r, for a given atomic system. The system size is uniformly scaled (b/a and c/a ratios held fixed) and the energy is calculated at a number of sizes without relaxing the system. All box sizes corresponding to energy minima are identified.

This calculation was created as a quick method for scanning the phase space of a crystal structure with a given potential in order to identify starting guesses for further structure refinement calculations.

Disclaimer #1: the minima identified by this calculation do not guarantee that the associated crystal structure will be stable as no relaxation is performed by this calculation. Upon relaxation, the atomic positions and box dimensions may transform the system to a different structure

Disclaimer #2: it is possible that the calculation may miss an existing minima for a crystal structure if it is outside the range of values scanned, or has b/a, c/a values far from the ideal.

Method and Theory

An initial system (and corresponding unit cell system) is supplied. The r/a ratio is identified from the unit cell. The system is then uniformly scaled to all r_i values in the range to be explored and the energy for each is evaluated using LAMMPS and "ru 0" command, i.e. no relaxations are performed.

In identifying energy minima along the curve, only the explored values are used without interpolation. In this way, the possible energy minima structures are identified for r_i where $E(r_i) < E(r_{i-1})$ and $E(r_i) < E(r_{i+1})$.

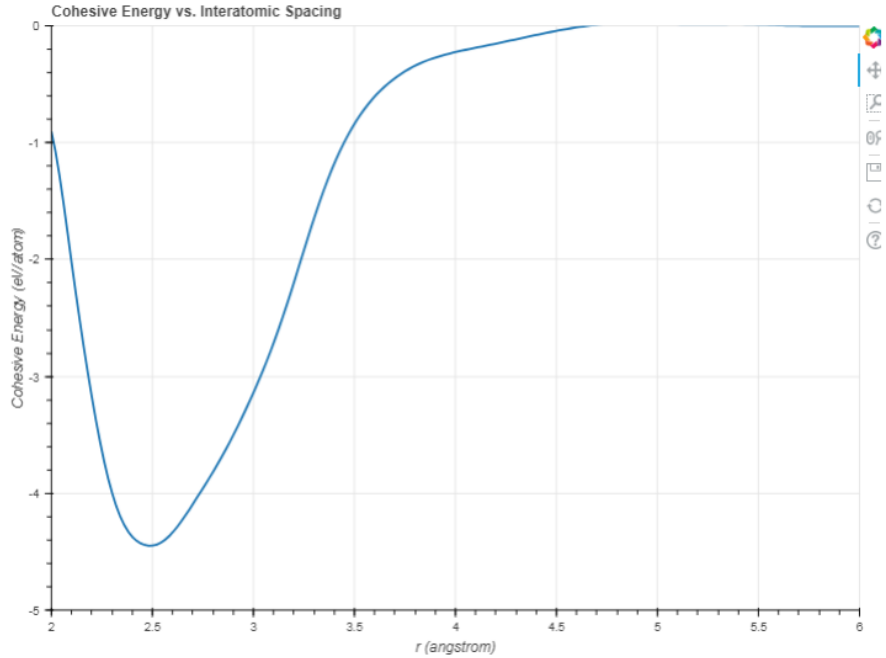
Demonstration

1. Setup

1.1 Library imports

```
# Make system a deepcopy of itself (protect original from changes)
system = deepcopy(system)

# Set ucell = system if ucell not given
if ucell is None:
    ucell = system

# Calculate the r/a ratio for the unit cell
r_a = r_a_ratio(ucell)

# Get ratios of lx, ly, and lz of system relative to a of ucell
lx_a = system.box.a / ucell.box.a
ly_a = system.box.b / ucell.box.a
lz_a = system.box.c / ucell.box.a
alpha = system.box.alpha
beta =  system.box.beta
gamma = system.box.gamma

# Build lists of values
r_values = np.linspace(rmin, rmax, rsteps)
a_values = r_values / r_a
Ecoh_values = np.empty(rsteps)

# Loop over values
for i in range(rsteps):

    # Rescale system's box
    a = a_values[i]
    system.box_set(a = a * lx_a,
                   b = a * ly_a,
                   c = a * lz_a,
                   alpha=alpha, beta=beta, gamma=gamma, scale=True

    # Get lammps units
    lammps_units = lmp.style.unit(potential.units)

    # Define lammps variables
    lammps_variables = {}
    system_info = system.dump('atom_data', f='atom.dat',
                              units=potential.units,
                              atom_style=potential.atom_style)
    lammps_variables['atomman_system_info'] = system_info
    lammps_variables['atomman_pair_info'] = potential.pair_info(sy

    # Write lammps input script
    template_file = 'run0.template'
    lammps_script = 'run0.in'
    with open(template_file) as f:
        template = f.read()
    with open(lammps_script, 'w') as f:
        f.write(iprPy.tools.filltemplate(template, lammps_variables,
                                         '<', '>'))

    # Run lammps and extract data
```

```
show(plot)
```

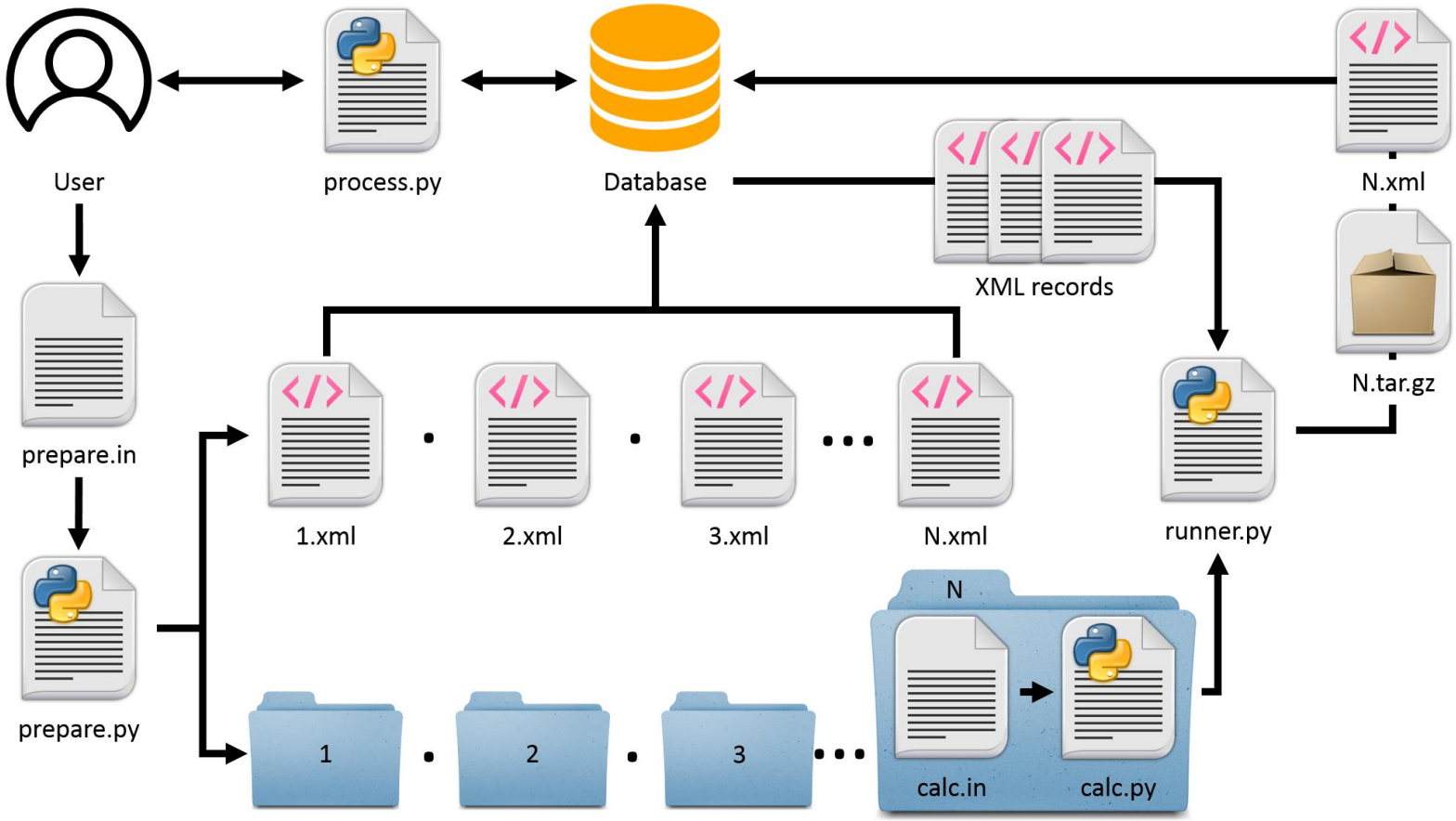


Basic execution

Prepare generates copies of a calculation each with a unique input file

Runner iteratively executes calculations

Database stores calculation records and archived calculation folders



L.M. Hale, Z.T. Trautt, C.A. Becker, MSMSE 26 (2018)

Making Prepare Easy and Powerful

- Call inline command with input file

```
./iprPy prepare <database> <directory> <calculation> param.in
```

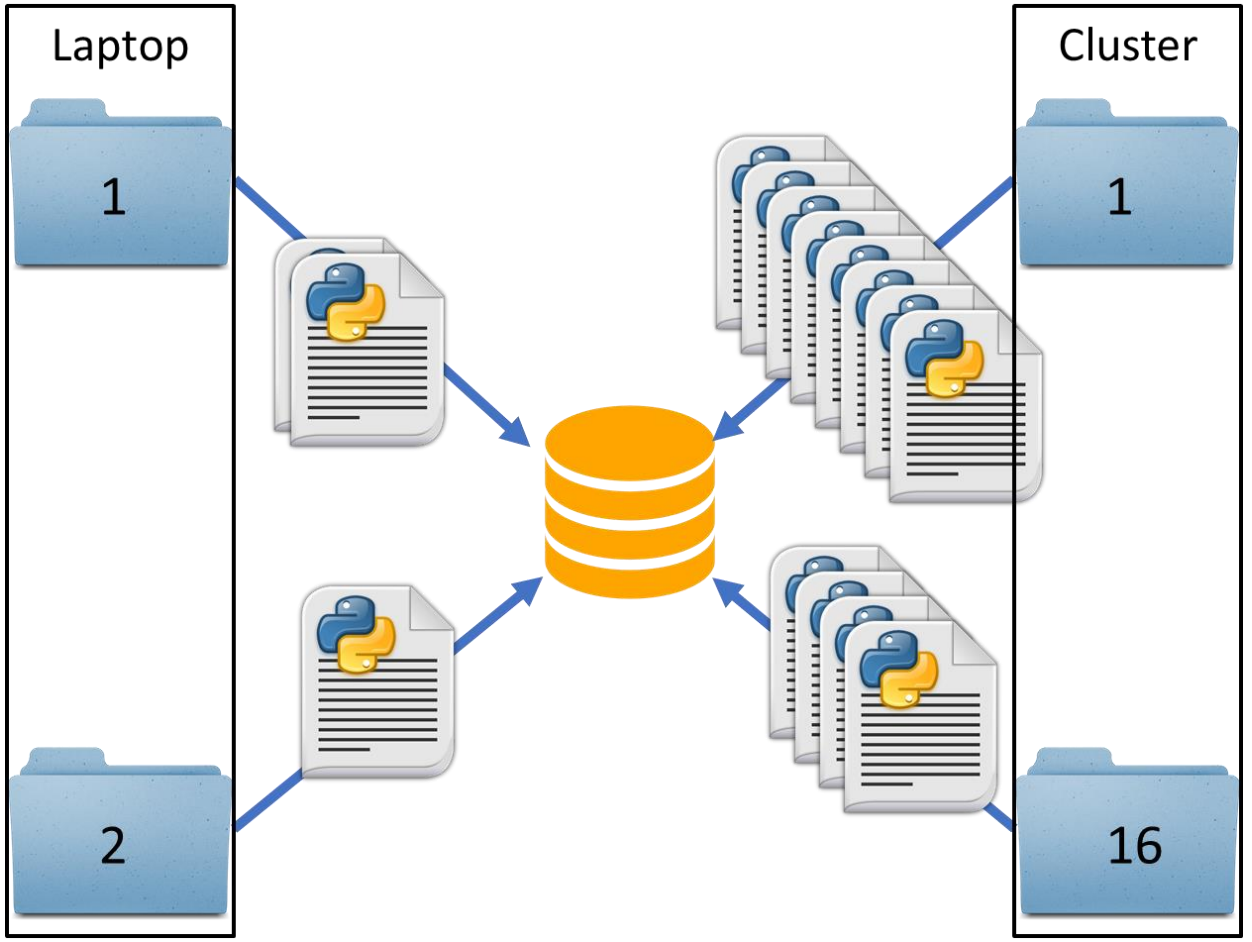
- Same keys as calculation!
  - But keys can have multiple values
  - Keys grouped in parameter sets

- Modularly defined generation functions
  - Example 1: prepare for all potentials with Al
  - Example 2: prepare using results

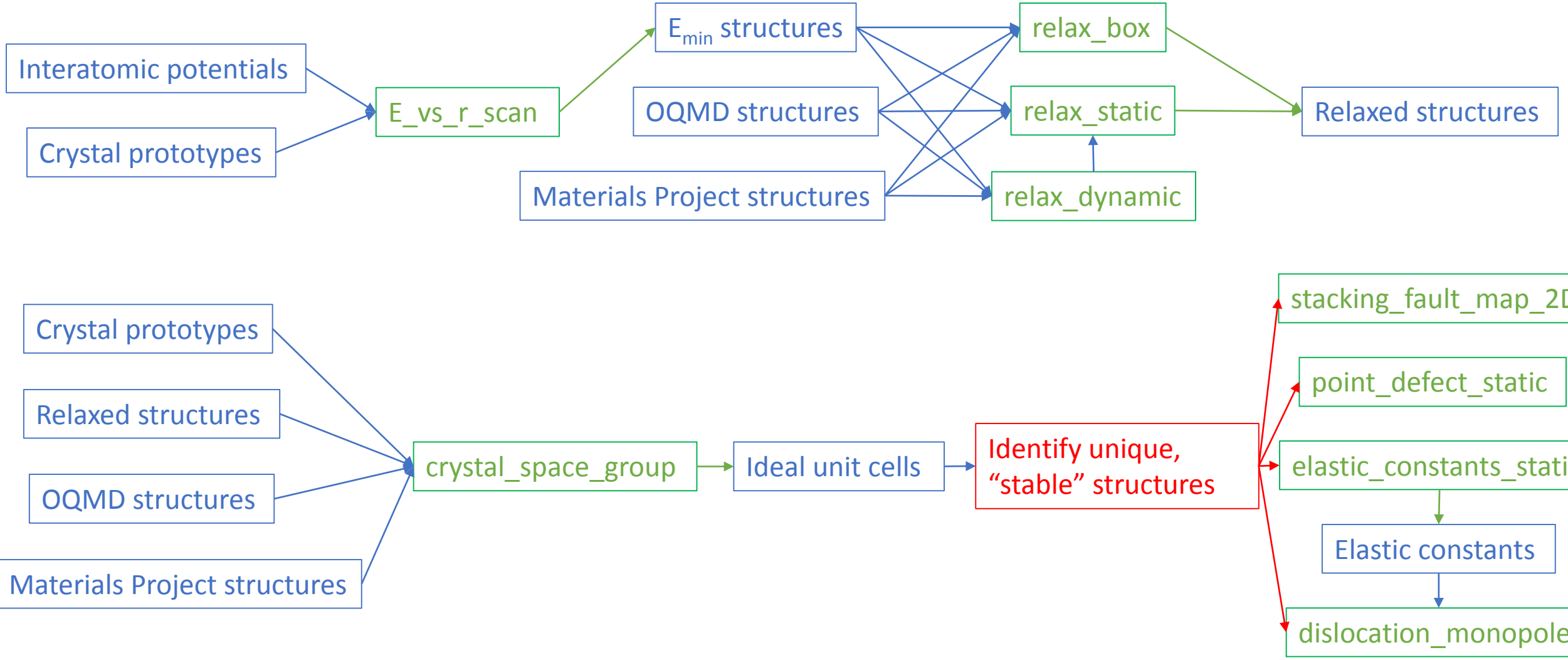
```
# Commands and executables
lammps_command              lmp_mpi
mpi_command

# Build load information based on prototype records
buildcombos                 crystalprototype load_file prototype
prototype_potential_name    2004--Zhou-X-W--Al--LAMMPS--ipr2

# System manipulations
a_uvw
b_uvw
c_uvw
atomshift
sizemults                   3 3 3

# Units that input/output values are in
length_unit
pressure_unit
energy_unit
force_unit

# Run parameters
minimum_r                   2.0
maximum_r                   6.0
number_of_steps_r           100
number_of_steps_r           200
number_of_steps_r           300
```

Using Runners to execute

- Each runner works in one directory
  - Separate by # cores needed
  - Run on any resource
- Multiple active runners in same/different folders
- Submit runner script to cluster queue



IPR Calculation Workflow



Select a composition: Cu

Reference structure matches:

A1--Cu--fcc = mp-30, oqmd-635950

A2--W--bcc = mp-998890, oqmd-637373

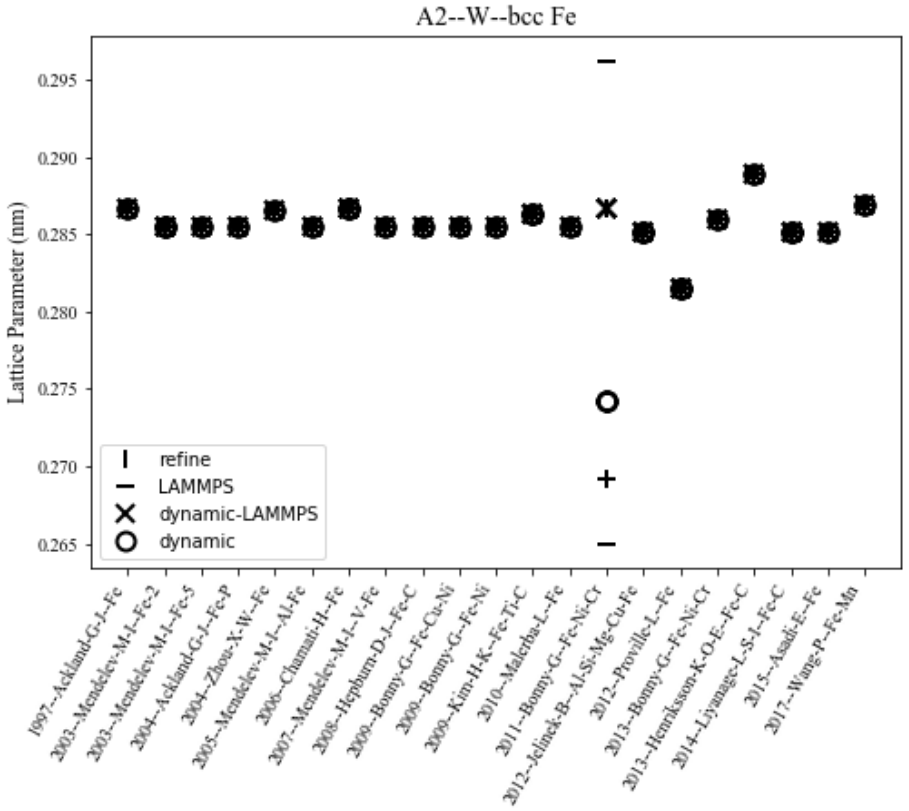
A3'--alpha-La--double-hcp = mp-989695

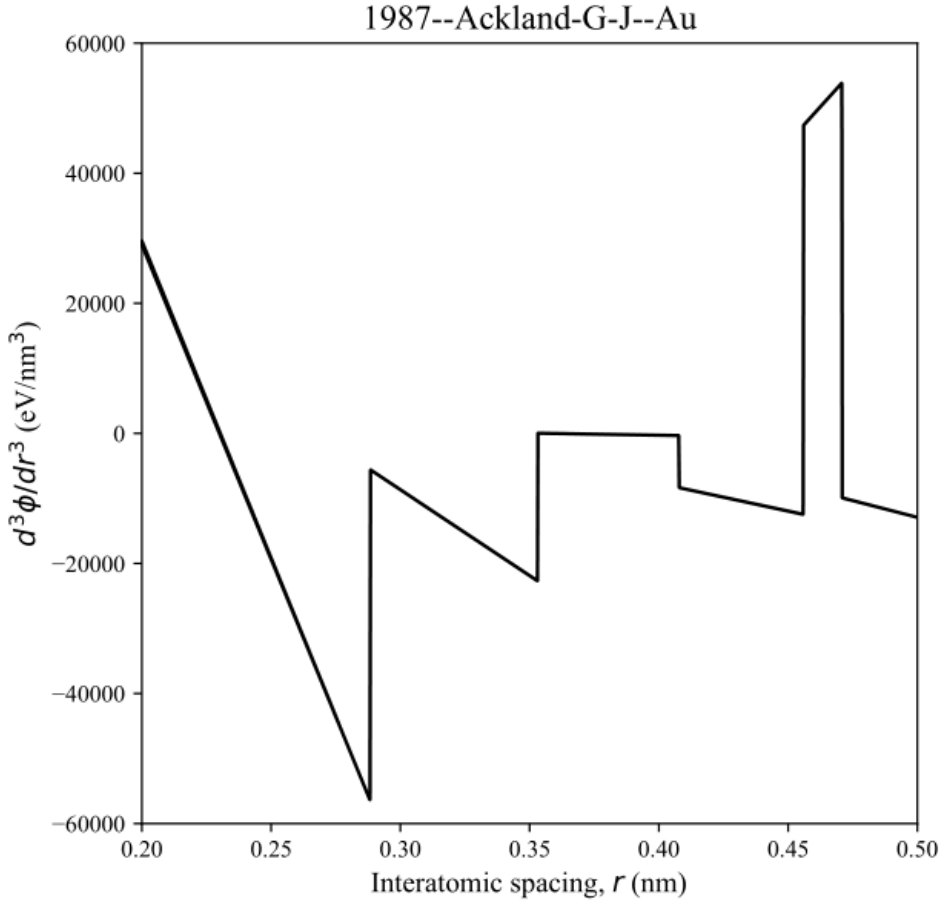
A3--Mg--hcp = mp-989782

A6--In--bct = mp-1010136

prototype	method	E_{coh} (eV)	a_0 (Å)	b_0 (Å)	c_0 (Å)	α (degrees)	β (degrees)	γ (degrees)
A1--Cu--fcc	dynamic	-3.3207	3.4921	3.4921	3.4921	90.0	90.0	90.0
A3'--alpha-La--double-hcp	dynamic	-3.3178	2.4697	2.4697	8.0474	90.0	90.0	120.0
A3--Mg--hcp	dynamic	-3.315	2.4703	2.4703	4.0137	90.0	90.0	120.0
A6--In--bct	static	-3.2913	2.8216	2.8216	2.6675	90.0	90.0	90.0
A2--W--bcc	static	-3.2911	2.768	2.768	2.768	90.0	90.0	90.0
A15--beta-W	dynamic	-3.2674	4.4406	4.4406	4.4406	90.0	90.0	90.0
A5--beta-Sn	static	-3.1196	4.4717	4.4717	2.3422	90.0	90.0	90.0
Ah--alpha-Po--sc	static	-3.0415	2.3038	2.3038	2.3038	90.0	90.0	90.0
A4--C--dc	static	-2.5365	5.0444	5.0444	5.0444	90.0	90.0	90.0

Potential & Bulk Property Method Analysis

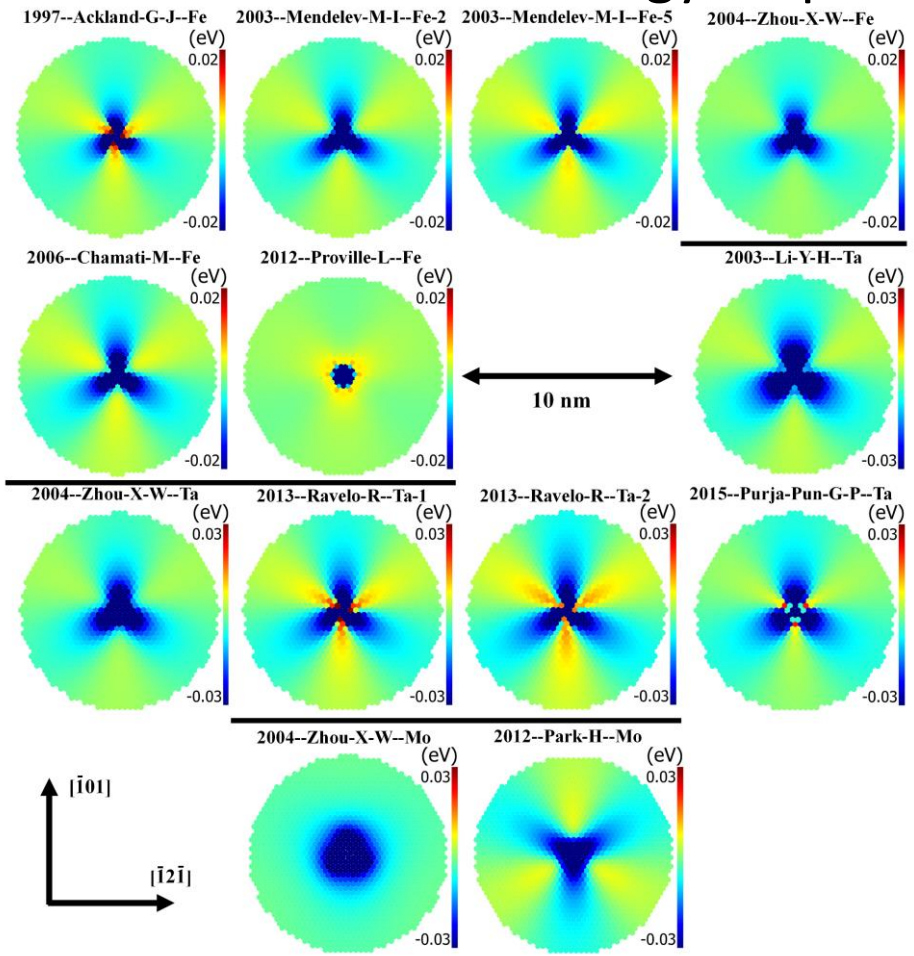




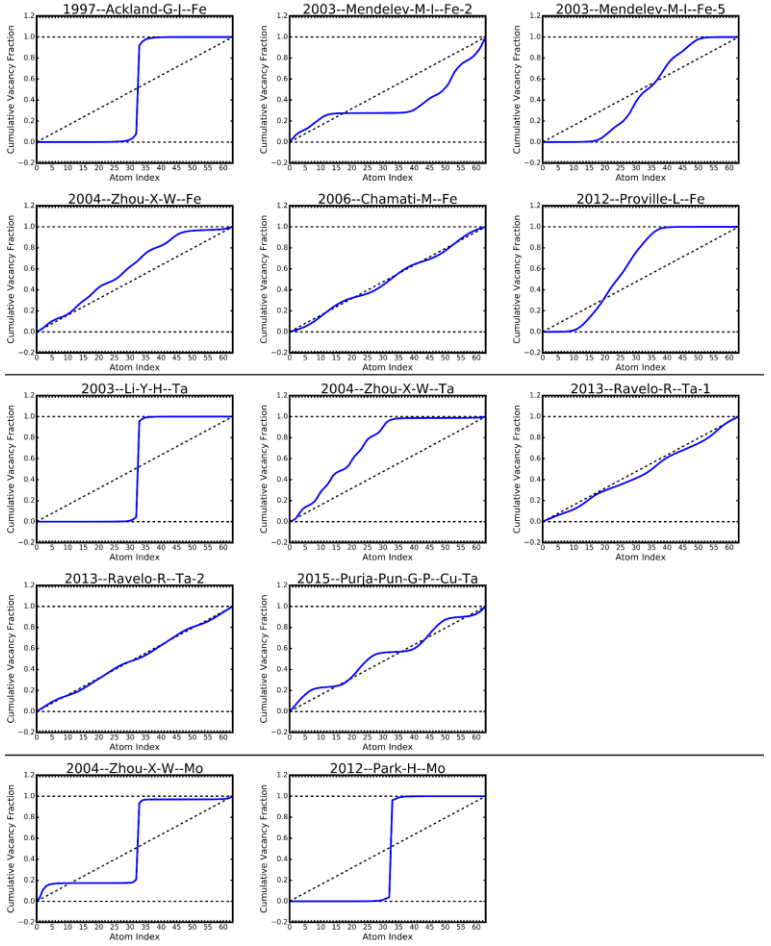
L.M. Hale, Z.T. Trautt, C.A. Becker, MSMSE 26 (2018)

Addressing Qualitative Uncertainty

Radial interaction energy map



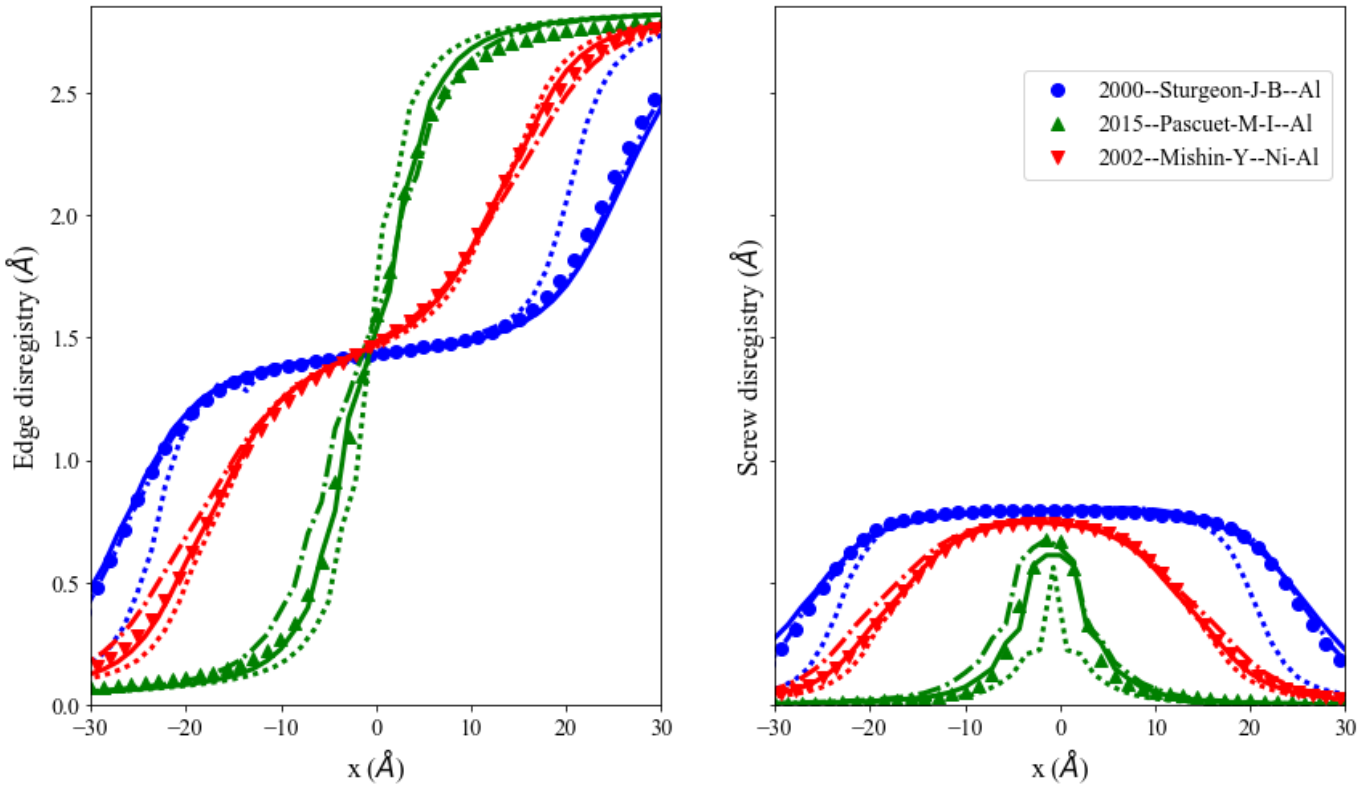
Dissociation along dislocation line



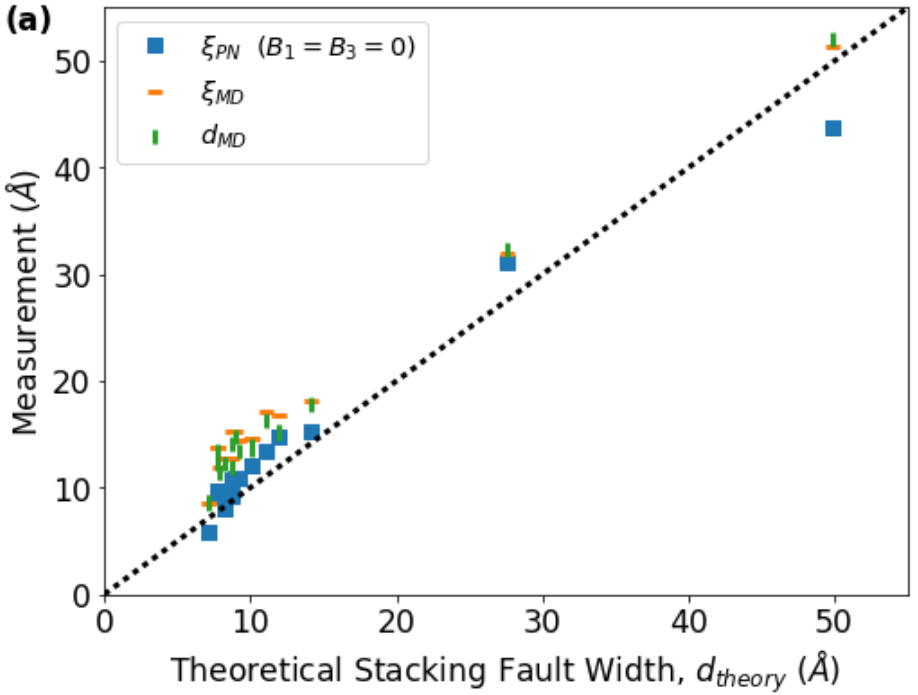
L.M. Hale, C.A. Becker, Computational Materials Science 135 (2017) 1-8.

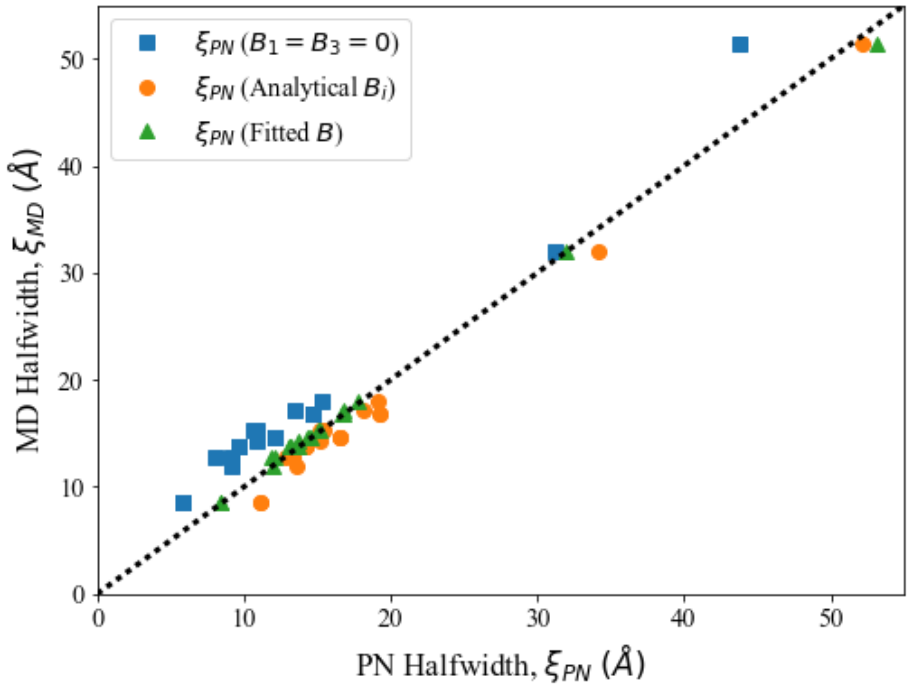
Method Comparison

Atomistic versus analytic dislocation models



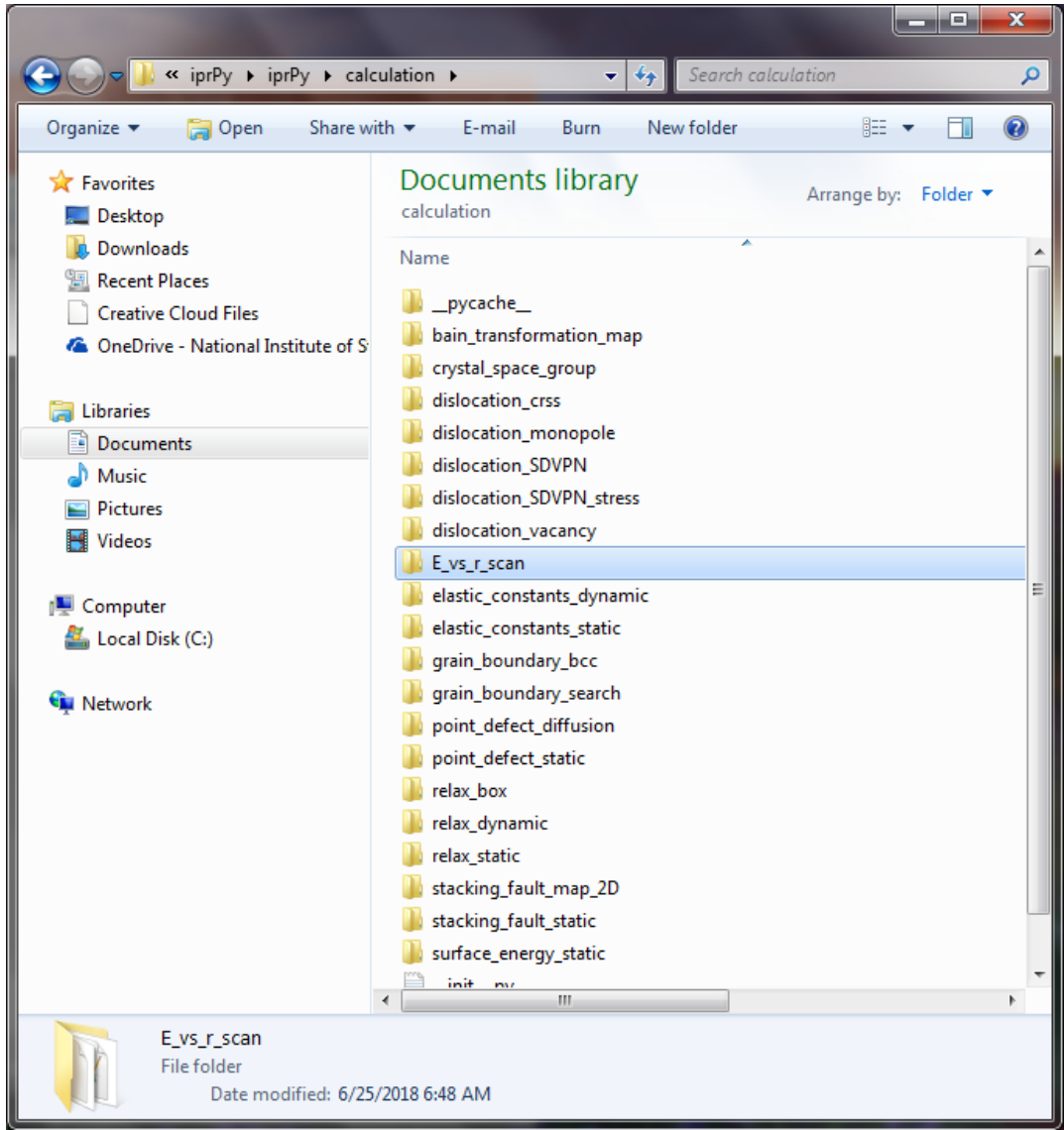
L. Hale JOM 70(7) (2018) 1100-1105





Repository of Calculations?

- “Verified” calculation methods
  - Alternates allowed
  - Repeatability across groups
- Accessible documentation
  - How to use, what parameters mean
  - Underlying theory
- Minimize implementation effort
  - Reuse parameters, modular code
  - What (time) cost would you spend on making code accessible to others?

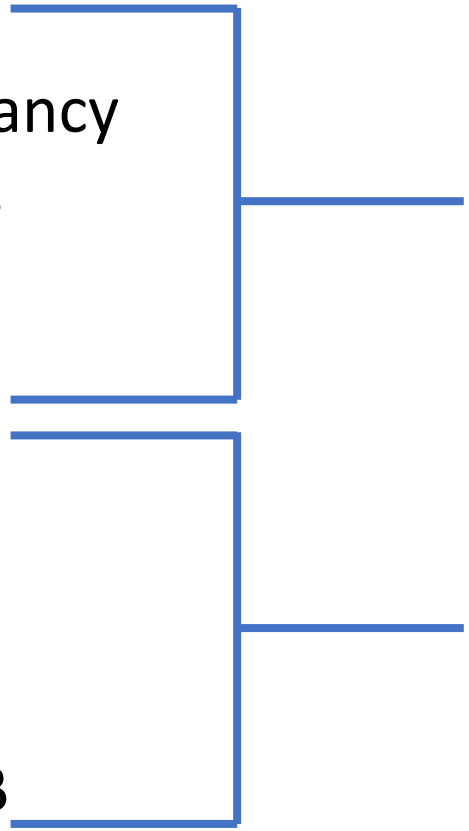


Currently Available Calculations

- **E_vs_r_scan**: cohesive energy versus interatomic spacing
- **relax_box**: relax box dimensions only
- **relax_static**: relax using energy/force minimization
- **relax_dynamic**: relax using MD iterations
- **crystal_space_group**: space group analysis of system
- **elastic_constants_static**: small strain static C_{ij} calculation
- **point_defect_static**: compute point defect formation energy
- **surface_energy_static**: free surface formation energies
- **stacking_fault_static**: compute stacking fault energy for one shift
- **stacking_fault_2D_map**: generalized 2D stacking fault energy map

Calculations Partially Implemented

- dislocation_monopole
- dislocation_monopole_vacancy
- dislocation_monopole_crss
- dislocation_SDVPN
- dislocation_SDVPN_stress
- bain_transformation_map
- grain_boundary_bcc_static
- grain_boundary_static
- phonon_static
- point_defect_mobility_NEB



Generalizing method to triclinic systems

Methods prototyped, need testing and support

Collaborate: How Can We Help Each Other?

- Cross-listing potentials from other locations
  - openKIM
  - LAMMPS potentials folder
  - …
- Integrating iprPy calculations into other infrastructures
- Uncertainty quantification of MD calculations
- Evaluating “reactive” potentials
- Incorporating your work to be open source and accessible?
  - Analysis methods in atomman
  - Property calculations in iprPy

Links

- Repository
  - https://www.ctcms.nist.gov/potentials/ (current)
  - https://www.ctcms.nist.gov/potentials/testing (preview of new)

- atomman
  - https://github.com/usnistgov/atomman (code)
  - https://www.ctcms.nist.gov/potentials/testing/atomman (preview of docs)

- iprPy
  - https://github.com/usnistgov/iprPy (code)
  - https://www.ctcms.nist.gov/potentials/testing/iprPy (preview of docs)