Interatomic Potentials Repository Project

Lucas M. Hale

Zachary T. Trautt

Chandler A. Becker

National Institute of Standards and Technology

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

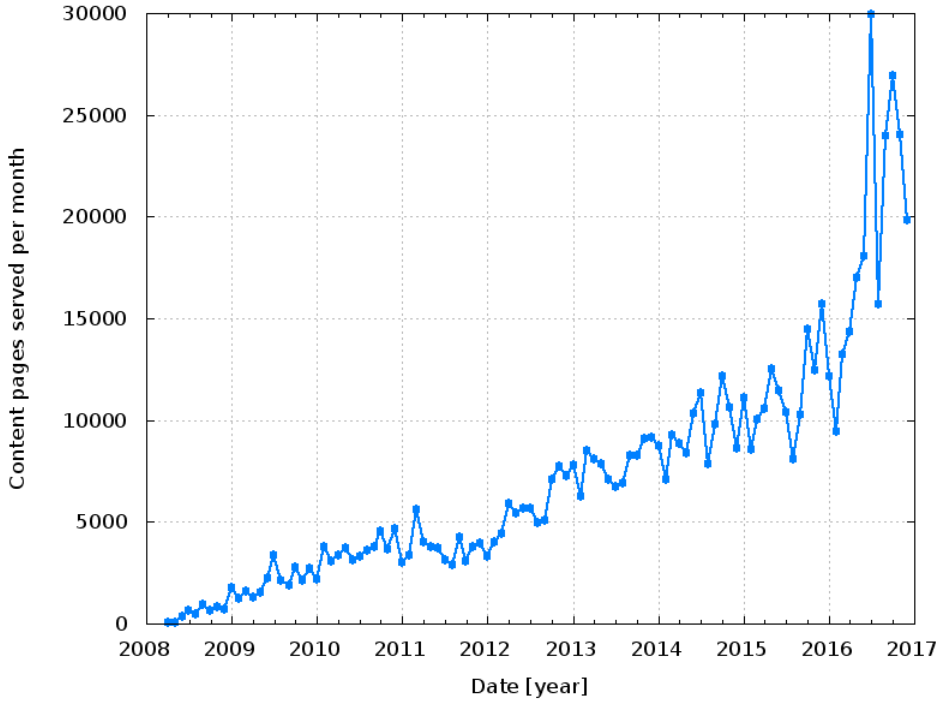
What is Our Motivation?

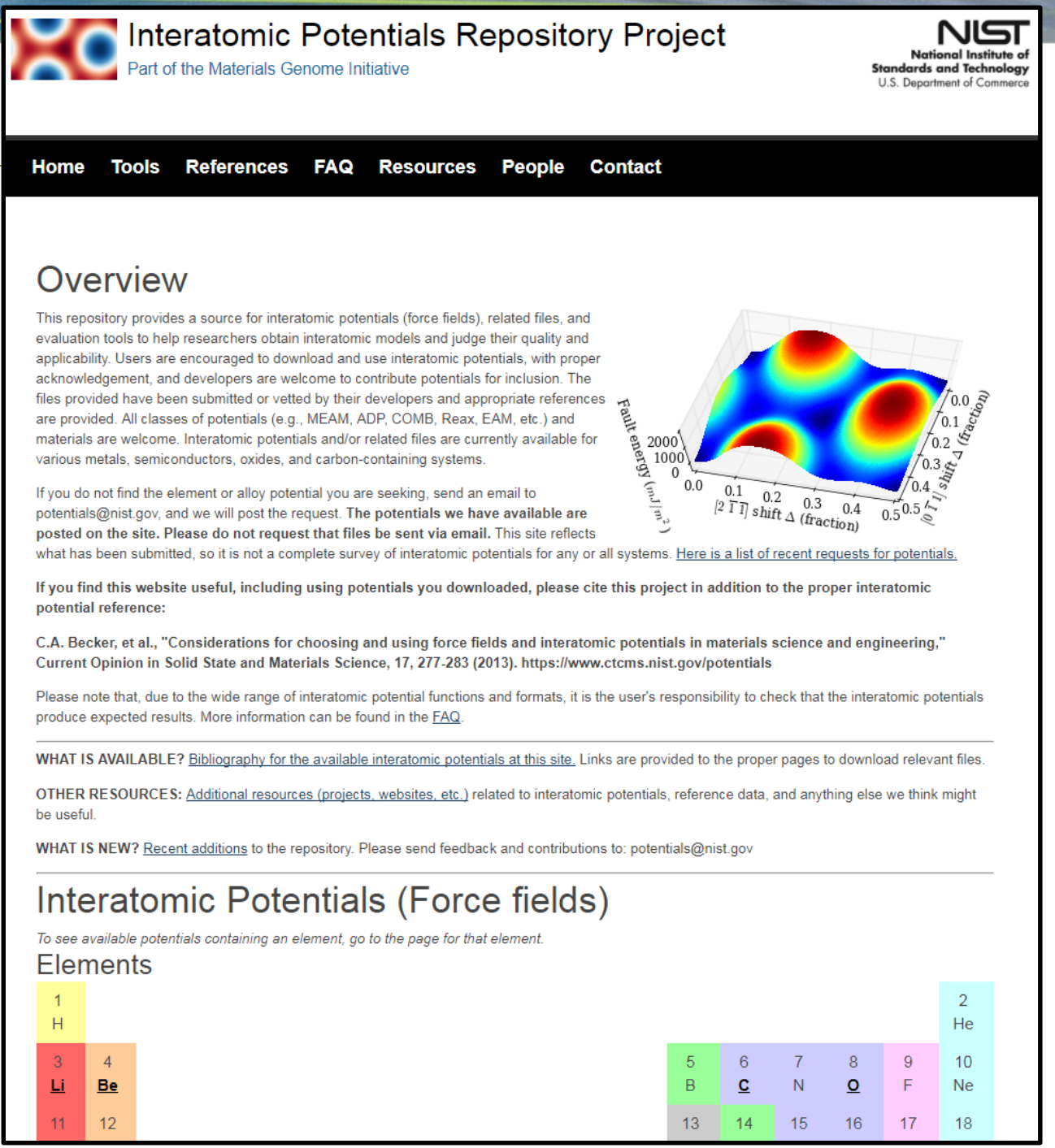
Provide tools and resources to make it easier for users to perform high-quality, meaningful molecular dynamics simulations.

Interatomic Potentials Repository

https://www.ctcms.nist.gov/potentials/

- All formats, material systems
- 168+ total potentials
- 130+ LAMMPS-compatible
- Email potentials@nist.gov to submit





Potential Information

Select element or system to view all associated potentials

Identified by primary citation or author

Version and usage notes

Multiple formats allowed

Computed properties

Gold (Au)

G. Grochola, S.P. Russo, and I.K. Snook, "On fitting a gold embedded atom method potential using the force matching method," J. Chem. Phys., 123 , 204719 (2005). DOI: 10.1063/1.2124667

Notes: This file was generated by C.A. Becker from the files below that were approved by G. Grochola (RMIT University) and posted with his permission on 21 Feb. 2011. This version is compatible with LAMMPS. Validation and usage information can be found in Au-Grochola-JCP05-conversion-notes_v2.pdf. If you use this eam.alloy file, please credit the website in addition to the original reference.

Format: EAM/alloy setfl
File(s): Au-Grochola-JCP05.eam.alloy

Notes: These files were approved by G. Grochola.

Format: EAM table
File(s):
F(ρ): Au-Grochola-JCP05-F.table
ρ(r): Au-Grochola-JCP05-rou.table
φ(r): Au-Grochola-JCP05-Phi.table

New! Computed Properties: 2005--Grochola-G--Au

X.W. Zhou, R.A. Johnson, and H.N.G. Wadley, "Misfit-energy-increasing dislocations in vapor-deposited CoFe/NiFe multilayers," Phys. Rev. B, 69 , 144113 (2004). DOI: 10.1103/PhysRevB.69.144113

Notes: This file was generated by C.A. Becker from the files sent by X.W. Zhou (Sandia National Laboratory) and posted with his permission. These files can be used to generate alloy potentials for Cu, Ag, Au, Ni, Pd, Pt, Al, Pb, Fe, Mo, Ta, W, Mg, Co, Ti, and Zr by editing EAM.input. However, as addressed in the reference, these potentials were not designed for use with metal compounds. See the Zhou04 page for more information.

Format: EAM/alloy setfl
File(s): Au.set

New! Computed Properties: 2004--Zhou-X-W--Au

J.B. Adams, S.M. Foiles, and W.G. Wolfer, "Self-diffusion and impurity diffusion of FCC metals using the 5-frequency model and the Embedded Atom Method," J. Mater. Res. 4 , 102-112 (1989). DOI: 10.1557/JMR.1989.0102

Notes: auu6.txt was obtained from http://enpub.fulton.asu.edu/cms/ potentials/main/main.htm and posted with the permission of J.B. Adams. The name of the file was retained, even though the header information lists the potential as 'universal 4.' This file is compatible with the "pair_style eam" format in LAMMPS (19Feb09 version).

Format: EAM
File(s): auu6.txt

New! Computed Properties: 1989--Adams-J-B--Au

G.J. Ackland, G.I. Tichy, V. Vitek, and M.W. Finnis, "Simple N-body potentials for the noble-metals and nickel," Phil. Mag. A 56 , 735-756 (1987). DOI: 10.1080/01418618708204485

Notes: The parameters in au.moldy were obtained from http://homepages.ed.ac.uk/graeme/moldy/moldy.html and posted with the permission of G.J.

Predicted Properties

Bulk properties

Defect structures

Focus on documenting methodology

0K Crystal Structure Properties for Au

prototype	E_{coh} (eV)	a_0 (Å)	b_0 (Å)	c_0 (Å)	C_{11} (GPa)	C_{22} (GPa)	C_{33} (GPa)	C_{12} (GPa)	C_{13} (GPa)	C_{23} (GPa)	C_{44} (GPa)	C_{55} (GPa)	C_{66} (GPa)
A1--Cu--fcc	-3.9242	4.0701			201.65			169.53			45.97		
A3'--alpha-La--double-hcp	-3.9194	2.8783		9.3909	229.78		245.94	158.11	142.81		27.30		
A3--Mg--hcp	-3.9147	2.8784		4.6919	227.81		252.83	156.58	135.53		28.73		
A2--W--bcc	-3.8634	3.2218			70.69			162.13			52.46		
A15--beta-W	-3.8236	5.1468			292.22			127.30			0.28		
A5--beta-Sn	-3.6657	5.2037		2.7696	193.04		186.51	5.37	28.26		8.14		-16.28
Ah--alpha-Po--sc	-3.5869	2.6799			210.13			29.73			-4.57		
A4--C--dc	-2.7489	5.6627			270.09			219.65			88.19		

Generalized Planar Fault Energy of pure Gold

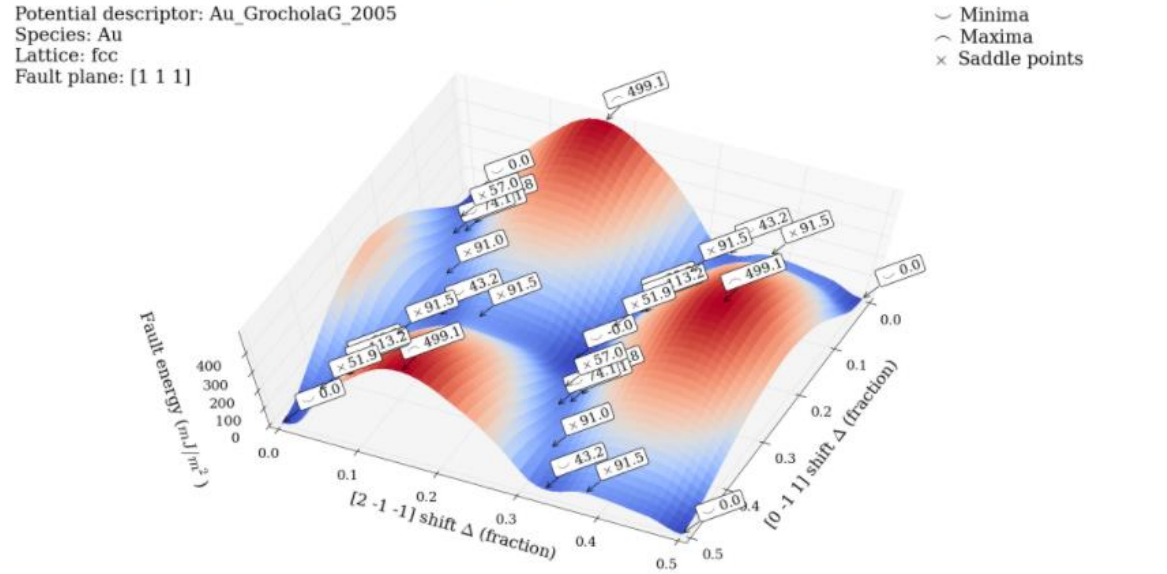
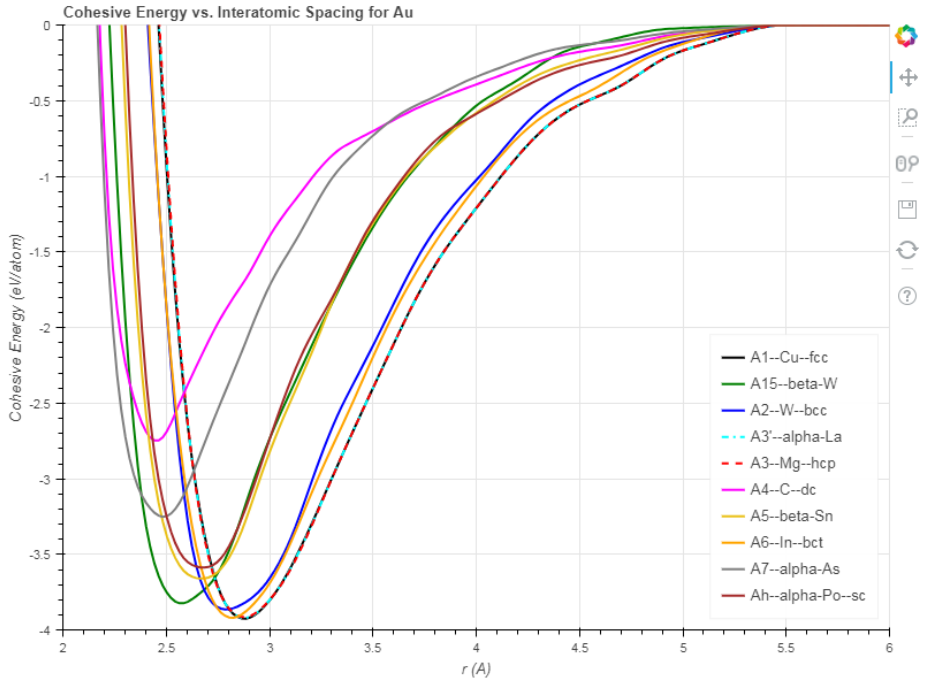


Illustration of a complete (1 1 1) fault energy surface with local minima, maxima, and saddle points labeled with their respective fault energy.

Cohesive Energy vs. Interatomic Spacing for Au

Download data



Different models for different uses

MEAM

EAM

EAM

Lennard-Jones

"Best" depends on what you want to do

ReaxFF

BOP

DFT

iprPy Computational Framework

https://github.com/usnistgov/iprPy

Design scientific calculations that

- are open source with minimum barriers for usage,
- have transparent methodologies supporting knowledge transfer and education,
- produce results that are both human and machine readable,
- allow investigations into method and parameter sensitivity,
- and can be integrated into workflows.

Python Calculation Scripts

Input parameter file → Calculation script → JSON/XML results

calc_surface_energy.in

```
#Run script for calc_surface_energy.py

#Command lines for LAMMPS (and MPI).
#For lammps_command, exclude passing in a script, i.e. no "-in term"
lammps_command              lmp_mpi
mpi_command

#Paths to the potential data model file, and directory containing potenti
potential_file              2017--Purja-Pun-G-P--Au--LAMMPS--test.xml
potential_dir               2017--Purja-Pun-G-P--Au--LAMMPS--test

#Initial system configuration to load
load_file                   48fe09d1-0aec-4e8e-8638-a22fae5df0d9.xml
load_style                  system_model
load_options                key relaxed-atomic-system
symbols
box_parameters

#System manipulations
x_axis
y_axis
z_axis
atomshift
sizemults                   5 5 10

#Defect parameters
surface_model               A1--Cu--fcc--110.xml
surface_cutboxvector

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

#Run parameters
energytolerance
forcetolerance
maxiterations
maxevaluations
maxatommotion
```

calc_surface_energy.py

```
def main(*args):
    """Main function called when script is executed directly."""

    # Read input file in as dictionary
    with open(args[0]) as f:
        input_dict = iprPy.tools.parseinput(f, allsingular=True)

    # Interpret and process input parameters
    process_input(input_dict, *args[1:])

    # Run surface_energy
    results_dict = surface_energy(input_dict['lammps_command'],
                                  input_dict['initialsystem'],
                                  input_dict['potential'],
                                  input_dict['symbols'],
                                  mpi_command = input_dict['mpi_command'],
                                  etol = input_dict['energytolerance'],
                                  ftol = input_dict['forcetolerance'],
                                  maxiter = input_dict['maxiterations'],
                                  maxeval = input_dict['maxevaluations'],
                                  dmax = input_dict['maxatommotion'],
                                  cutboxvector = input_dict['surface_cutboxvector'])

    # Save data model of results
    results = iprPy.buildmodel(record_style, 'calc_' + calc_style, input_dict,
                               results_dict)

    with open('results.json', 'w') as f:
        results.json(fp=f, indent=4)

def surface_energy(lammps_command, system, potential, symbols,
                   mpi_command=None, etol=0.0, ftol=0.0, maxiter=10000,
                   maxeval=100000, dmax=uc.set_in_units(0.01, 'angstrom'),
                   cutboxvector='c'):
    """
    Evaluates surface formation energies by slicing along one periodic
    boundary of a bulk system.

    Parameters
    ----------
    lammps_command : str
        Command for running LAMMPS.
    system : atomman.System
        The system to perform the calculation on.
    potential : atomman.lammps.Potential
        The LAMMPS implemented potential to use.
    symbols : list of str
```

results.json

```
{
    "calculation-surface-energy": {
        "key": "0549a9ee-5ffb-4397-b90c-113671fefdc4",
        "calculation": {
            "iprPy-version": "0.7.2",
            "LAMMPS-version": "26 Jan 2017-ICMS",
            "script": "calc_surface_energy",
            "run-parameter": {
                "size-multipliers": {
                    "a": [
                        0,
                        5
                    ],
                    "b": [
                        0,
                        5
                    ],
                    "c": [
                        0,
                        10
                    ]
                },
                "energytolerance": 0.0,
                "forcetolerance": {
                    "value": 1e-06,
                    "unit": "eV/angstrom"
                },
                "maxiterations": 10000,
                "maxevaluations": 100000,
                "maxatommotion": {
                    "value": 0.01,
                    "unit": "angstrom"
                }
            },
            "potential-LAMMPS": {
                "key": "8bfd0a48-8558-46f9-9d20-cd9e92cb83ae",
                "id": "2017--Purja-Pun-G-P--Au--LAMMPS--test",
                "potential": {
                    "key": "ef908258-25d1-439e-8223-3bf4df924ed0",
                    "id": "2017--Purja-Pun-G-P--Au"
                }
            },
            "system-info": {
                "family": "A1--Cu--fcc",
                "artifact": {
                    "file": "48fe09d1-0aec-4e8e-8638-a22fae5df0d9.xml",
                    "format": "system_model",
```

Why Design Calculations This Way?

Self-contained calculations

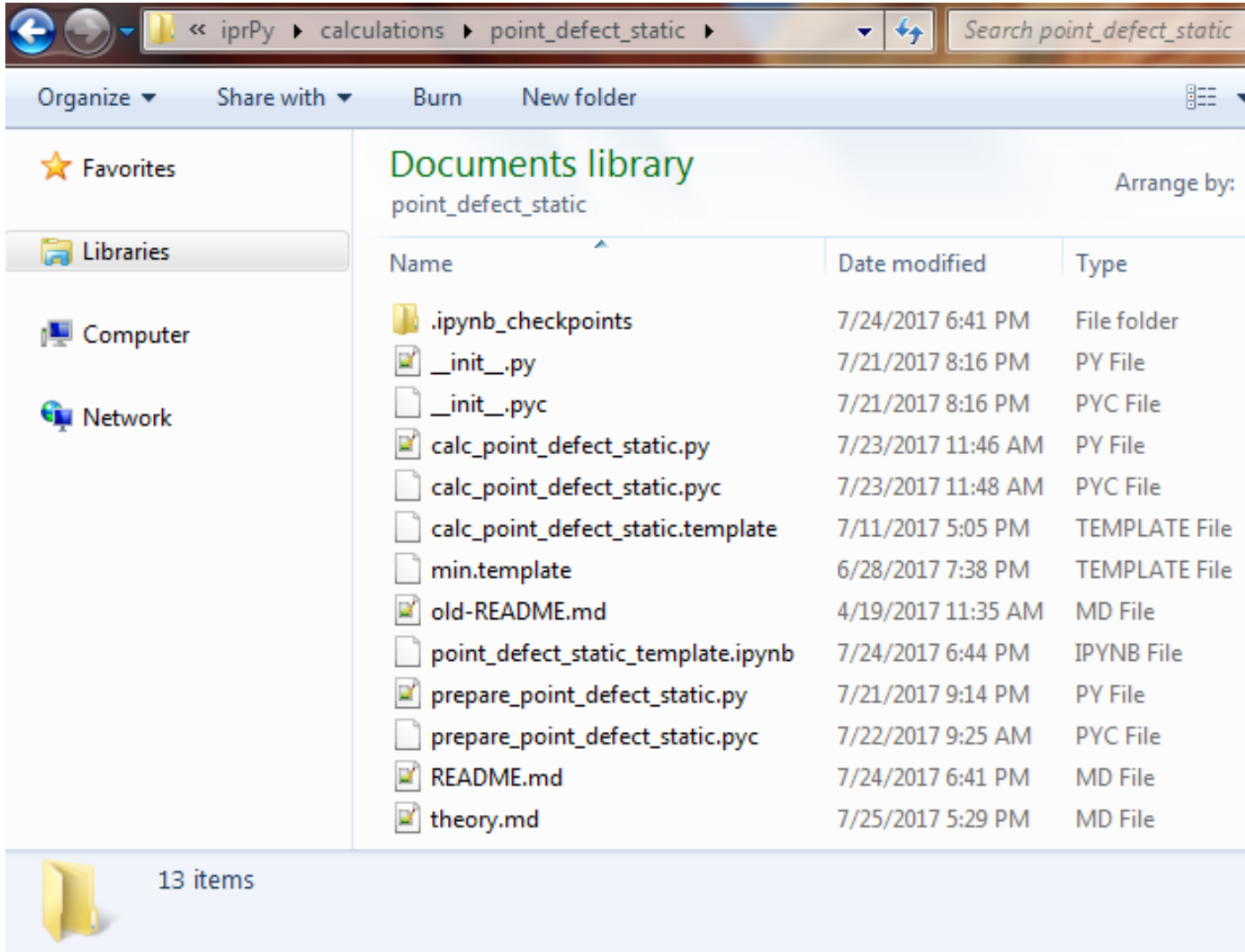
- Archive complete methodology
- Share capability with others
- Transfer knowledge to others

Clear input parameters

- Focus on terms relevant to calculation
- Integrate into workflows
- Investigate parameter sensitivities

Structured results

- Machine readable: store in database
- Human readable: understand contents



Jupyter Demonstration Notebooks

surface_energy 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: 2017-07-24

Disclaimers

Introduction

The **surface_energy** calculation evaluates the formation energy for a free surface by slicing an atomic system along a specific plane.

Disclaimer #1: Other atomic configurations at the free surface for certain planar cuts may have lower energies. The atomic relaxation will find a local minimum, which may not be the global minimum. Additionally, the material cut is planar perfect and therefore does not explore the effects of atomic roughness.

Disclaimer #2: Currently, the rotation capabilities of atomman limit this calculation such that only cubic prototypes can be rotated. Properties of non-cubic structures can still be explored, as long as the configuration being loaded has the plane of interest perpendicular to one of the three box vectors.

Method and Theory

An initial system is supplied, and a LAMMPS simulation performs an energy/force minimization on the system and estimates the total potential energy of the perfect bulk system, E_{bulk}^{total}. A corresponding defect system is constructed by changing one of the three boundary conditions from periodic to non-periodic. This effectively slices the system along the boundary plane creating two free surfaces, each with surface area $A_{surface}$. The defect system is then relaxed with an energy/force minimization and the total potential energy of the defect system, $E_{surface}^{total}$, is measured. The formation energy of the free surface, $E_{surface}^{f}$, is computed in units of energy over area as

$$E_{surface}^{f} = \frac{E_{surface}^{total} - E_{bulk}^{total}}{2A_{surface}}.$$

3.2 surface_energy()

```
def surface_energy(lammps_command, system, potential, symbols,
                   mpi_command=None, etol=0.0, ftol=0.0, maxiter=10000,
                   maxeval=100000, dmax=uc.set_in_units(0.01, 'angstrom'),
                   cutboxvector='c'):
    """
    Evaluates surface formation energies by slicing along one periodic
    boundary of a bulk system.

    Parameters
    ----------
    lammps_command :str
        Command for running LAMMPS.
    system : atomman.System
        The system to perform the calculation on.
    potential : atomman.lammps.Potential
        The LAMMPS implemented potential to use.
    symbols : list of str
        The list of element-model symbols for the Potential that correspond to
        system's atypes.
    mpi_command : str, optional
        The MPI command for running LAMMPS in parallel.  If not given, LAMMPS
        will run serially.
    etol : float, optional
        The energy tolerance for the structure minimization. This value is
        unitless. (Default is 0.0).
    ftol : float, optional
        The force tolerance for the structure minimization. This value is in
        units of force. (Default is 0.0).
    maxiter : int, optional
        The maximum number of minimization iterations to use (default is
        10000).
    maxeval : int, optional
        The maximum number of minimization evaluations to use (default is
        100000).
    dmax : float, optional
        The maximum distance in length units that any atom is allowed to relax
        in any direction during a single minimization iteration (default is
        0.01 Angstroms).
    cutboxvector : str, optional
        Indicates which of the three system box vectors, 'a', 'b', or 'c', to
        cut with a non-periodic boundary (default is 'c').

    Returns
    -------
    dict
        Dictionary of results consisting of keys:

        - **'dumpfile_base'** (*str*) - The filename of the LAMMPS dump file
          of the relaxed bulk system.
        - **'dumpfile_surf'** (*str*) - The filename of the LAMMPS dump file
          of the relaxed system containing the free surfaces.
        - **'E_total_base'** (*float*) - The total potential energy of the
```

Single file combining:

- Documentation
- Code identical to calculation script
- Working example

5. Report results

5.1 Define units for outputting values

- **length_unit** is the unit of length to display results in.
- **energy_unit** is the unit of energy to display cohesive energies in.
- **e_A_unit** is the energy per area to report the surface energy in.

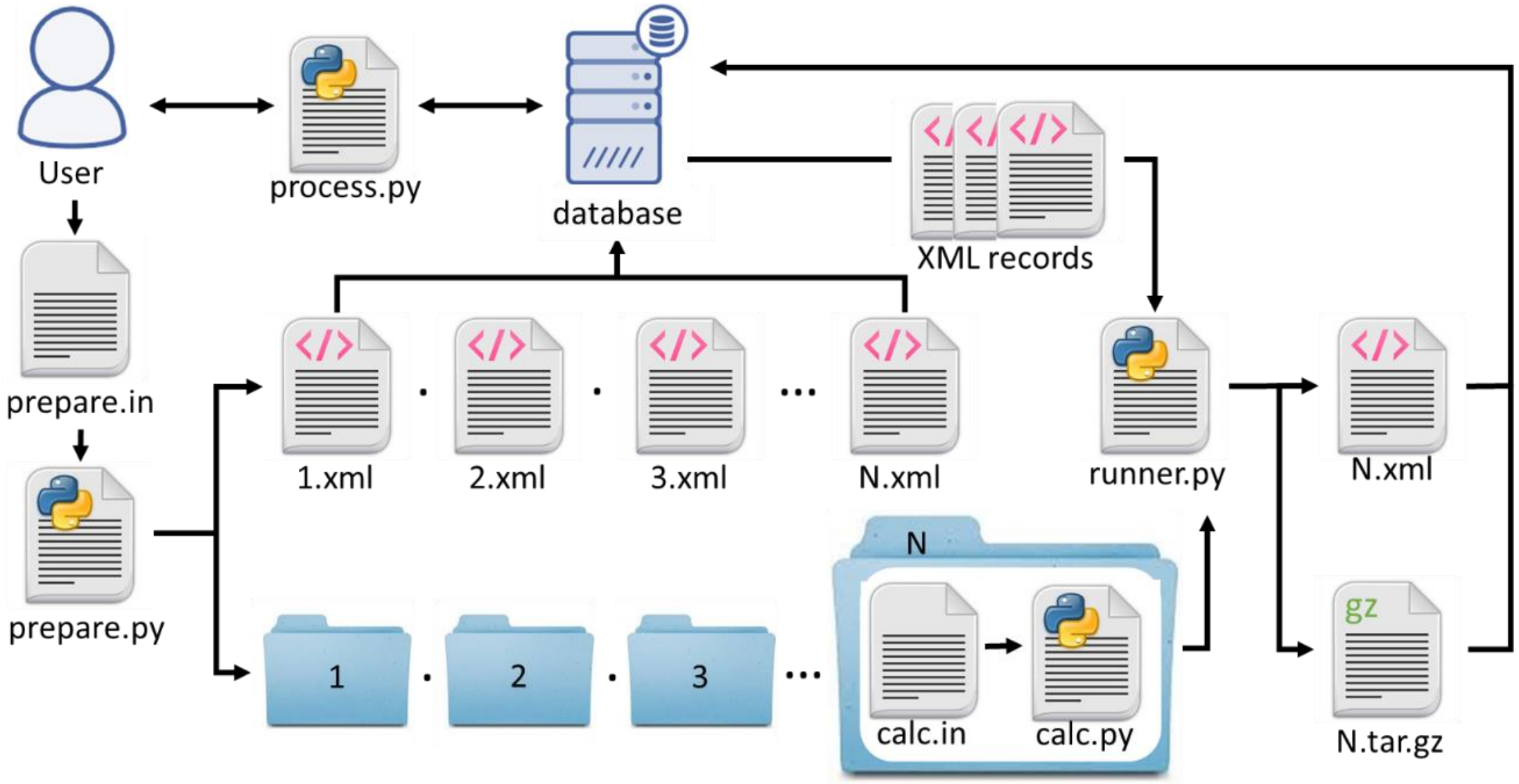
```
length_unit = 'angstrom'
energy_unit = 'eV'

#e_A_unit = energy_unit+'/'+length_unit+'^2'
e_A_unit = 'mJ/m^2'
```

5.2 Print E_{coh}, $A_{surface}$, and $E_{surface}^{f}$

```
print('E_coh =  ', uc.get_in_units(results_dict['E_coh'], energy_unit), energy_un
print('A_surface =', uc.get_in_units(results_dict['A_surf'], length_unit+'^2'), l
print('E_surface_f =', uc.get_in_units(results_dict['E_surf_f'], e_A_unit), e_A_u
```

```
E_coh =   -4.44999999835 eV
A_surface = 438.066793081 angstrom^2
E_surface_f = 2049.39393517 mJ/m^2
```

High-Throughput Tools

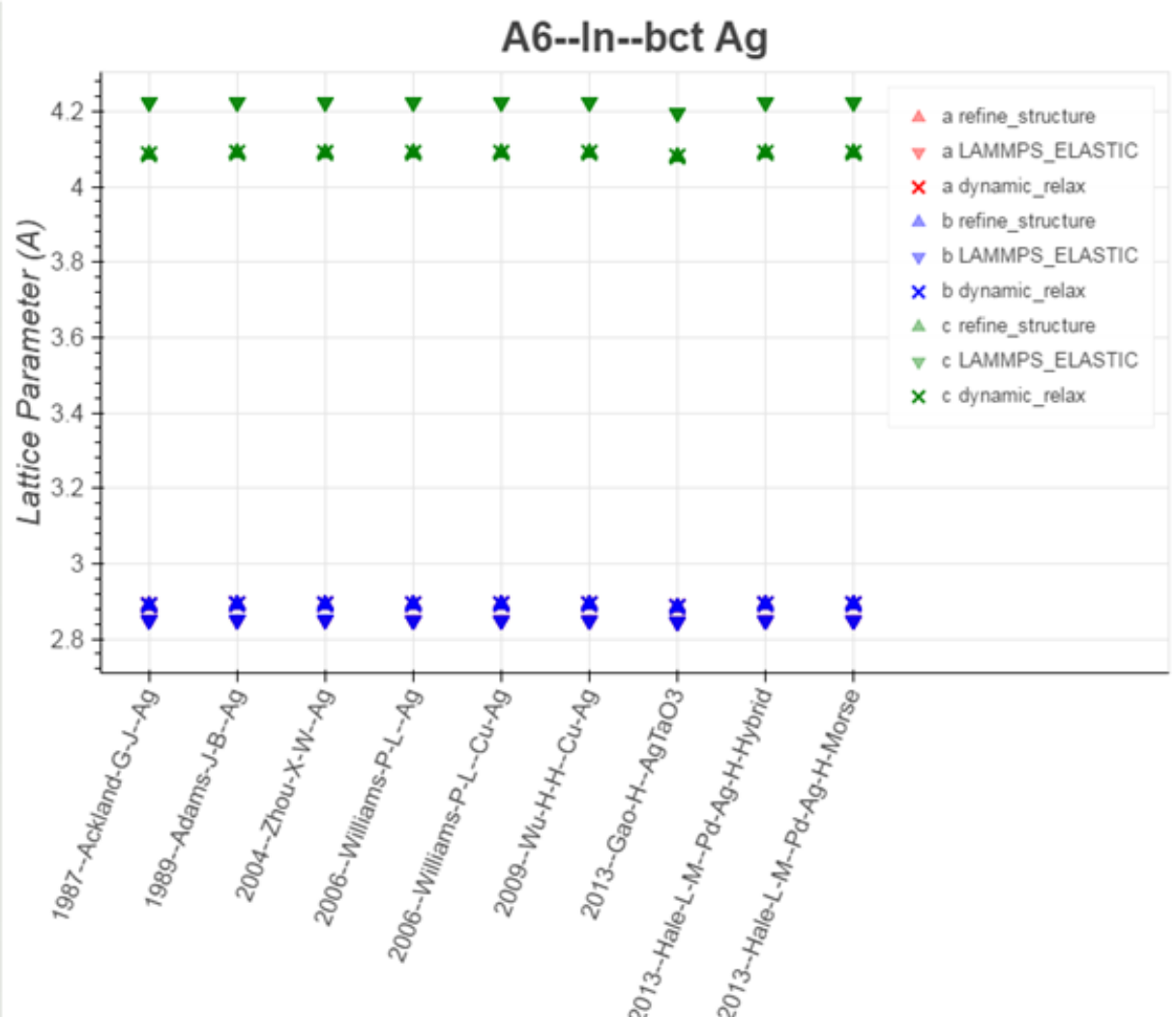


Implemented calculations

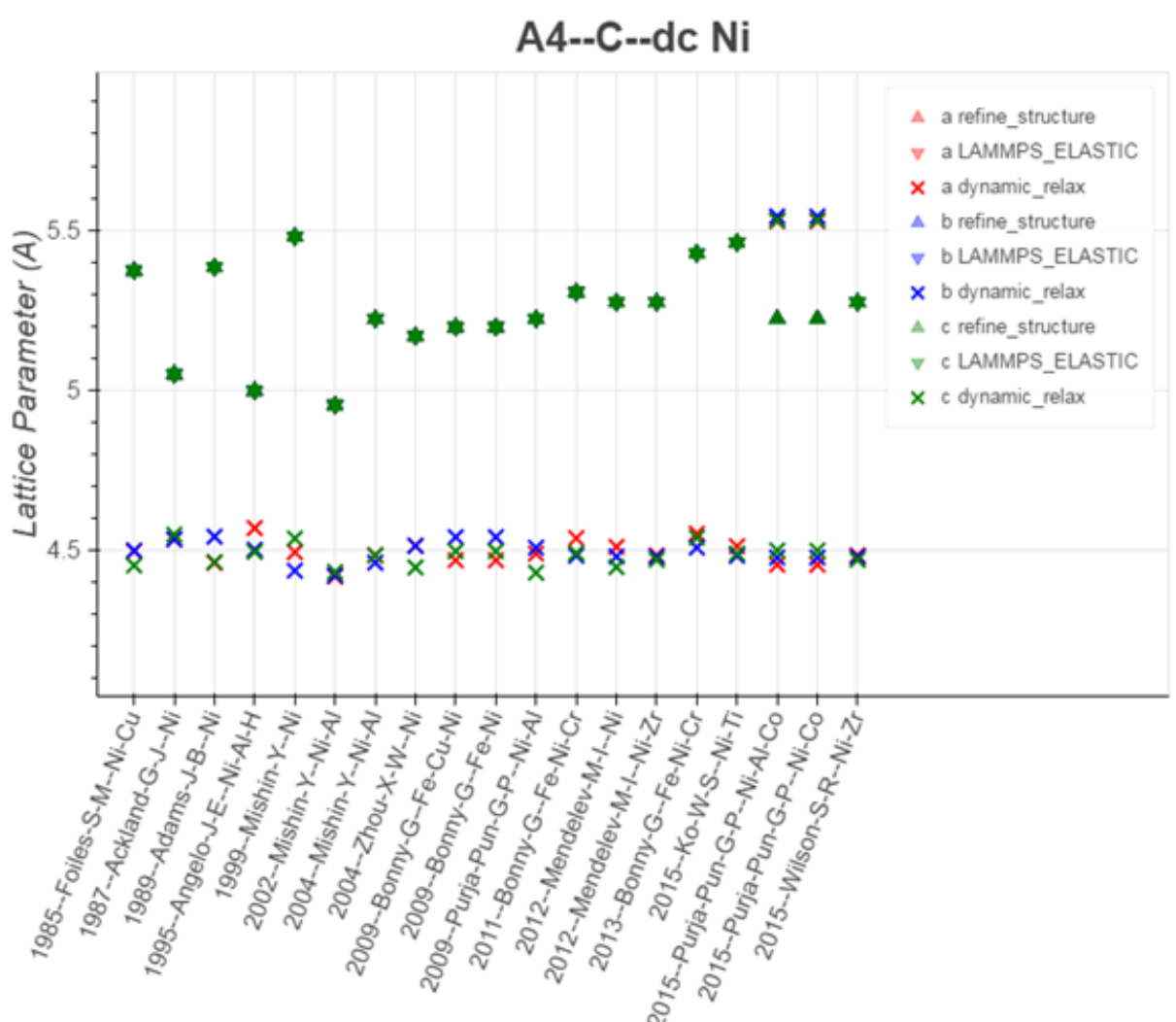
- Cohesive energy versus interatomic spacing
- Static relaxation of bulk systems and elastic constants (2 methods)
- Dynamic relaxation of bulk systems
- Free surface energies
- Generalized stacking faults (2 methods)
- Point defect formation energies
- Dislocation monopole generation

Method Comparison – Lattice Constants

minimize + box_relax doesn't fully relax bct → fcc

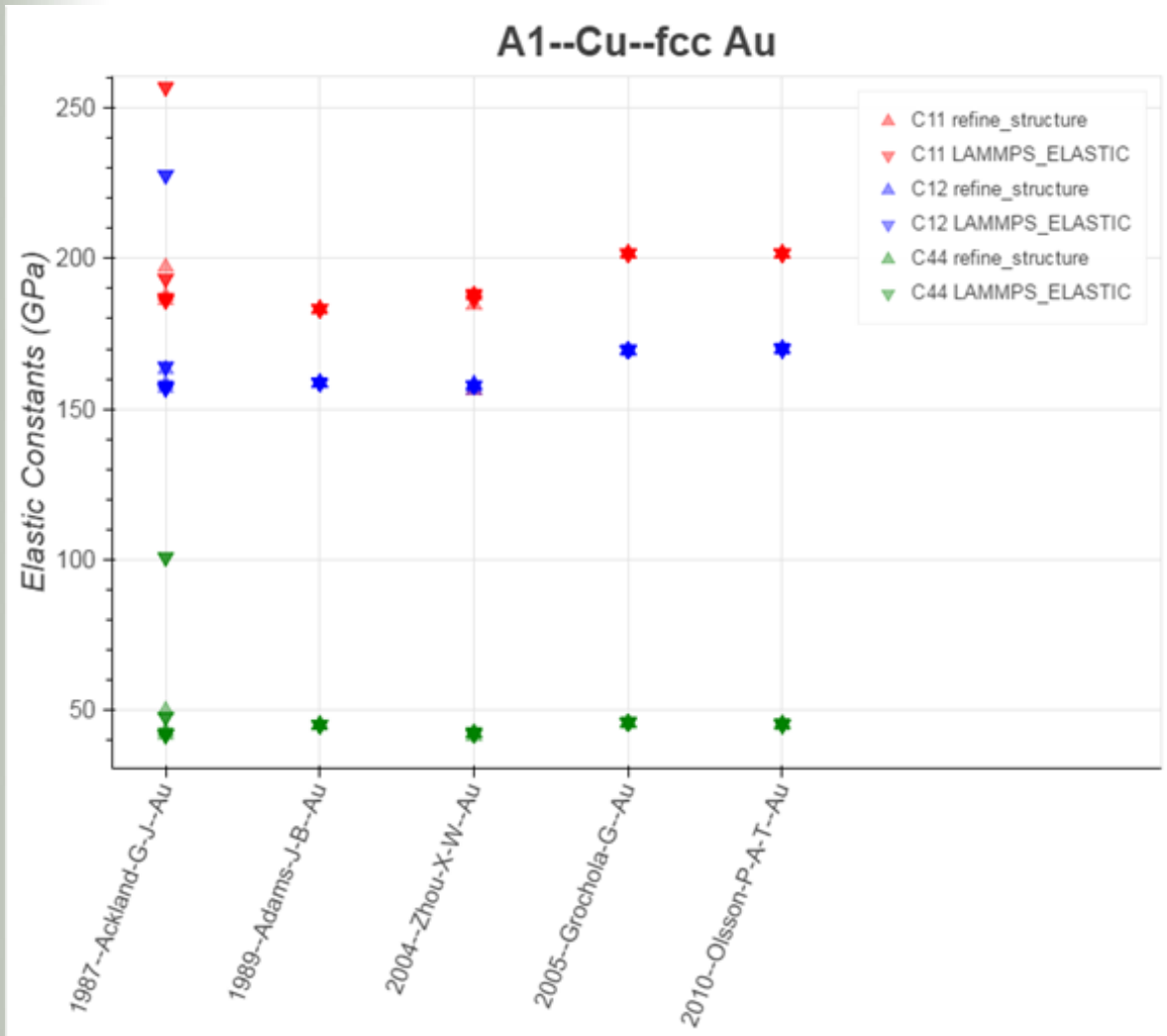


Static methods predict diamond cubic nickel as stable

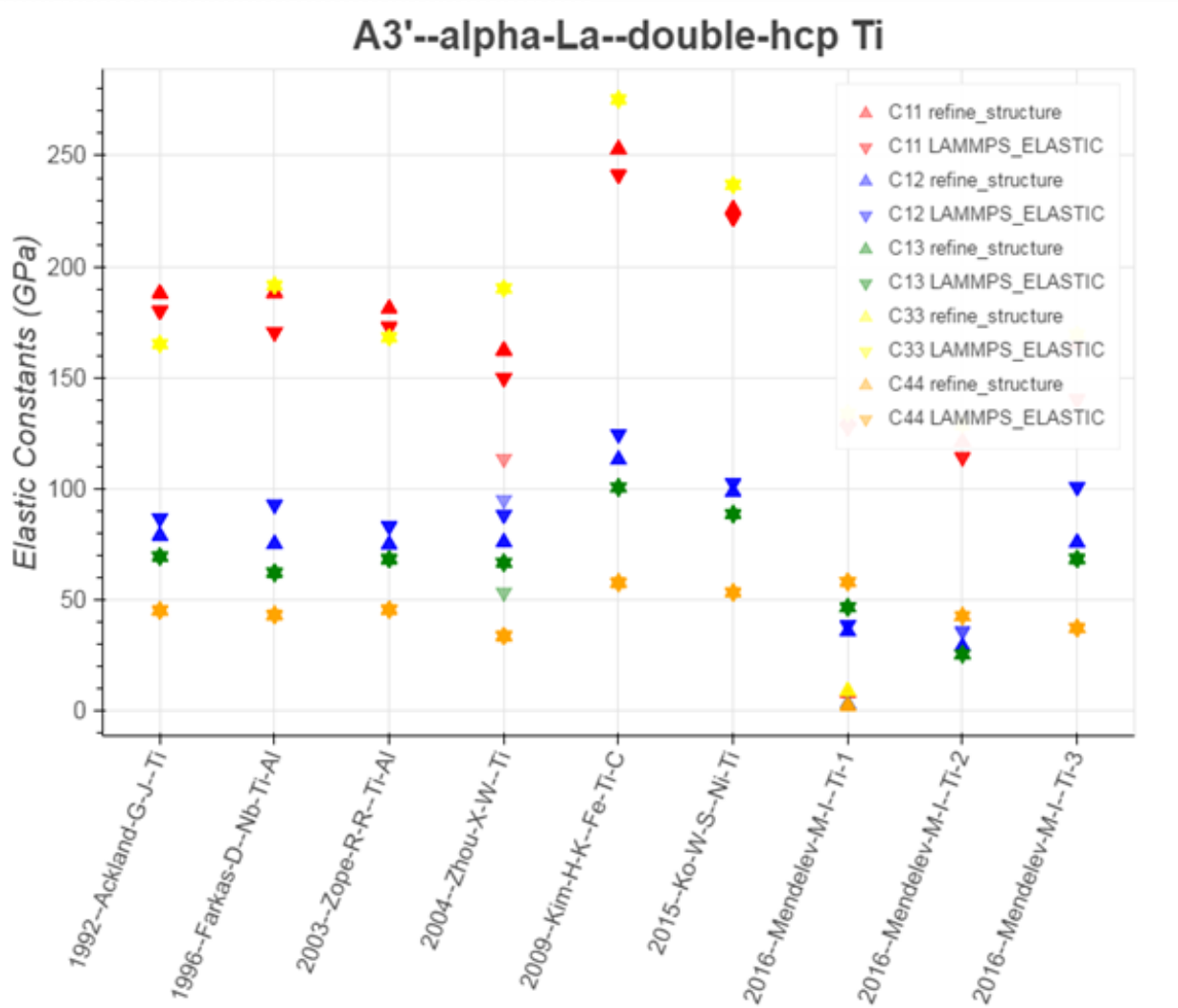


Method Comparison – Elastic Constants

Sensitivity to strain range



Sensitivity to method



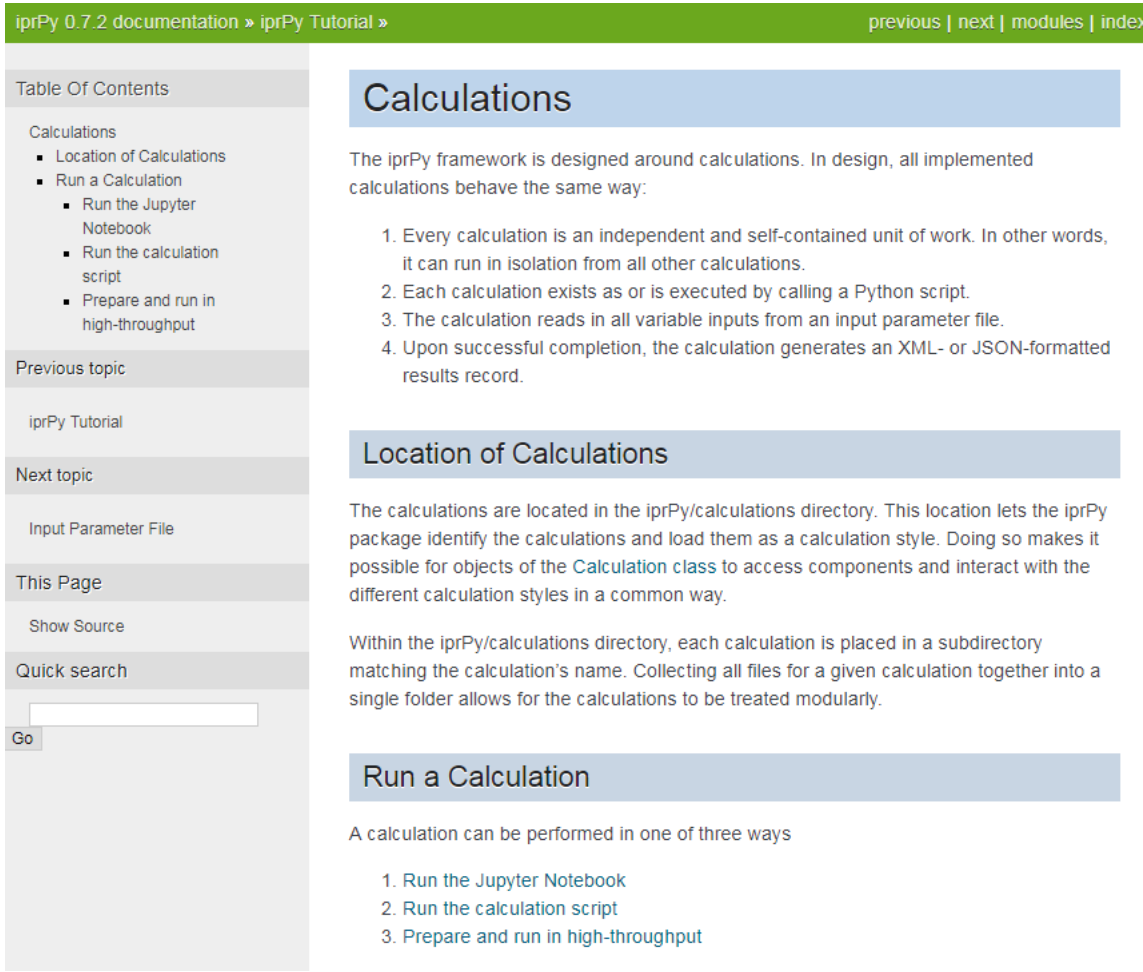
Minimize Barriers for Usage

Only requirements are Python 2.7 and system-independent packages

Treat calculations and databases modularly

Define common inputs for similar calculations

Full documentation (finishing up now)



Other Resources

Materials Resource Registry

https://materials.registry.nist.gov/

- Yellow pages for materials science
- All materials science, not just MD
- Search for resources
- FREE ADVERTISING!
- Add/edit your content by
  - Requesting an account or
  - Contact me

We Want to Work With You!

Atomistic Simulations for Industrial Needs Workshop will return next summer!

Interatomic Potential Repository: https://www.ctcms.nist.gov/potentials/

- Download potentials
- Add potentials: email potentials@nist.gov

iprPy Framework: https://github.com/usnistgov/iprPy

- Download and use the calculations
- Implement calculations into your workflows just cite us!
- Add new calculations
- Use design principles for research

Materials Resource Registry: https://materials.registry.nist.gov/

- Explore resources
- Add and advertise your work