abICS の使い方

Website: https://www.pasums.issp.u-tokyo.ac.jp/abics

GitHub: https://github.com/issp-center-dev/abICS

E-mail: abics-dev@issp.u-tokyo.ac.jp

東大物性研　**本山裕一**

2022-06-27 物性研CCMS 講習会

ablCS のインストール

- 最新版 (v2.0.0) は Python3.7 以上が必要
  - numpy
  - mpi4py
  - pymatgen >= 2019.12.3
- `pip install abics` でインストール可能
- インストール方法について詳しくは pip のマニュアルや文献を参照のこと
  - バージョン更新は `pip install -U abics`
  - バージョン指定は `abics==2.0.0`
  - インストール場所の変更は `install --user` や `--prefix=PATH`

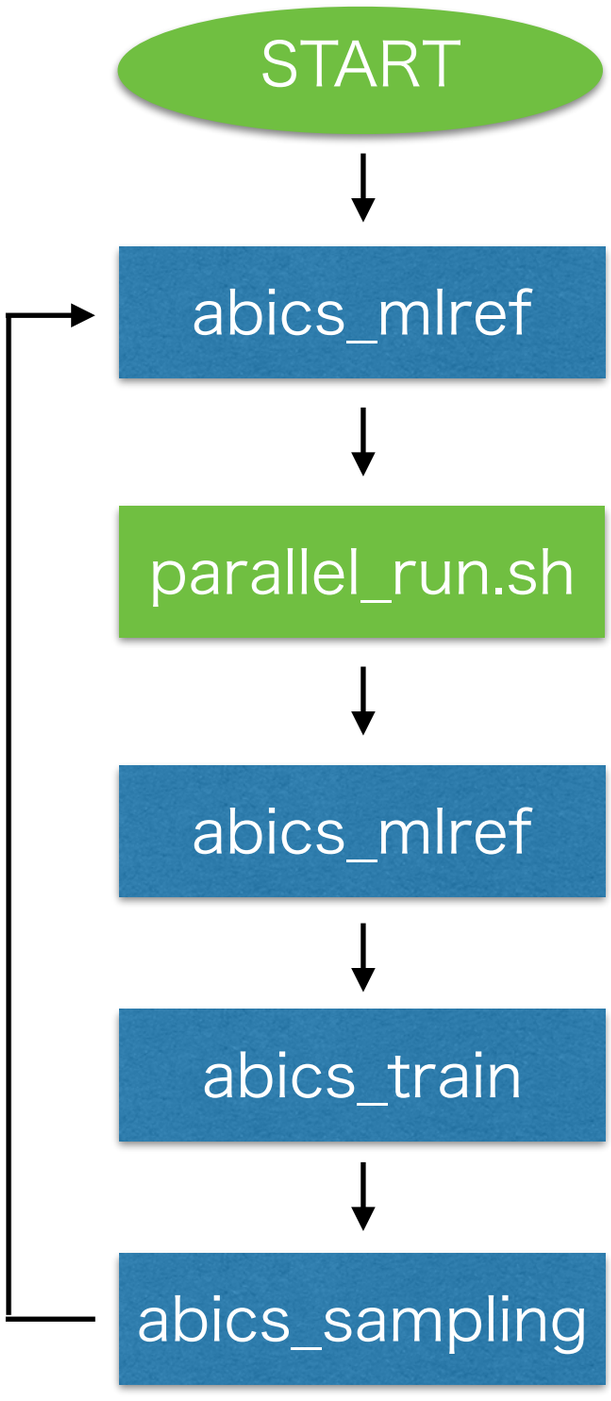
abICS のインストール（サンプル）

- サンプルの入力ファイルは pip install では入らない
  - https://github.com/issp-center-dev/abICS から持ってくる
    - `git clone https://github.com/issp-center-dev/abICS`
  - `examples` ディレクトリ以下にサンプルが存在
- マニュアル
  - https://issp-center-dev.github.io/abICS/docs/master/ja/index.html

abICS のインストール

- abICS のコマンドとして次のものがインストールされる
  - `abics_mlref`
    - 機械学習ソルバーの学習データセットを準備するスクリプト
      - MC で得た原子配置から第一原理計算の入力を作る
      - 第一原理計算の結果から学習データセットを作る
  - `abics_train`
    - `abics_mlref` で得られた学習データセットを用いて機械学習ソルバーを学習させるスクリプト
  - `abics_sampling`
    - 原子配置のモンテカルロ(MC) サンプリングを行うスクリプト
  - `st2abics`
    - 原子構造ファイル（POSCAR など）からabICS の入力ファイルを生成するヘルパースクリプト
  - `abicsRXsepT`
    - レプリカ交換モンテカルロ法の計算結果を整理するヘルパースクリプト

ablCS の流れ



abics_* ablCS のスクリプト

モンテカルロサンプリングから原子配位データを抽出
もしくはランダム生成（初回）

それぞれの配位に関して第一原理計算を（並列）実行
GNU parallel を利用した参考スクリプトがexamples にある

第一原理計算の結果から教師データ（エネルギー）を抽出

第一原理計算のエネルギーを再現するように
機械学習モデルを学習

機械学習モデルでエネルギーを計算しつつ
原子配位をカノニカル分布からサンプリング

ablCS の出力ファイル

- ALloop.progress
  - 現在ループのどこまで実行したのかを示すテキストファイル
- AL0, AL1, ...
  - 機械学習ネットワークのトレーニングデータ用のディレクトリ
  - abics_mlref が出力する
  - abics_train が読み込む
- MC0, MC1, ...
  - モンテカルロサンプリングの出力ディレクトリ
  - abics_sampling が出力する

- 実行中によくわからなくなったらこれらを消すと最初からやり直せる
  - バックアップ的に適当なディレクトリにmove するのがよい

ablCS の入力ファイル

- TOML 形式で書かれたテキストファイルを入力ファイルとしてとる
  - ミニマルな設定ファイルのフォーマット
  - TOML 仕様: https://toml.io/ja/v1.0.0-rc.2

```
[sampling] # セクションわけ
# key = value 形式
nreplicas = 8
kTstart = 600.0
kTend = 2000.0

[sampling.solver]
type = "aenet" # 文字列

[config]
# 配列・多重配列
unitcell = [[1.0, 0.0, 0.0],
            [0.0, 1.0, 0.0],
            [0.0, 0.0, 1.0]]
```

```
# 同じセクション名を複数並べるときは
# [[ ]] をつかう
[[config.base_structure]]
type = "O"
coords = [
    [0.0, 0.0, 0.0],
    [0.1, 0.0, 0.0],
    # ...(skipped)...
    ]

[[config.base_structure]]
type = "H"
coords = [[ ... ]]
```

[config] セクション

- 各スクリプトで共通して利用する
- 原子配置を設定する
  - unitcell, supercell
    - ユニットセルのサイズ、数
  - [[config.base_structure]]
    - サンプリングで動かさない原子の位置、種類
  - [[config.defect_structure]]
    - サンプリングで動かす原子の位置、種類

```
[config]
# ユニットセルの格子ベクトル
unitcell = [[8.1135997772, 0.0000000000, 0.0000000000],
            [0.0000000000, 8.1135997772, 0.0000000000],
            [0.0000000000, 0.0000000000, 8.1135997772]]

# ユニットセルの数
supercell = [1,1,1]
```

[config] セクション

- [[config.base_structure]]
  - サンプリングで動かさない原子の位置、種類
  - coords で位置を指定
    - N 行 3 列の行列で指定
  - type で原子種を指定
  - 複数原子を固定したいときは [[config.base_structure]] を複数設定

```
[[config.base_structure]]
type = "O"
coords = [
     [0.237399980, 0.237399980, 0.237399980],
     [0.762599945, 0.762599945, 0.762599945],
         ... skipped ...
     [0.262599975, 0.262599975, 0.762599945],
     ]
```

[config] セクション

- [[config.defect_structure]]
  - サンプリングで動かす原子の位置、種類
  - coords で位置を指定
    - N 行 3 列の行列で指定
  - [[config.defect_structure.group]] で原子種・数を設定
    - 原子団（e.g. OH ）を配置することも可能
      - species は原子団に含まれる原子種
      - coords は相対座標
        - 複数組指定することで回転・伸張も可能
- num が原子団の数

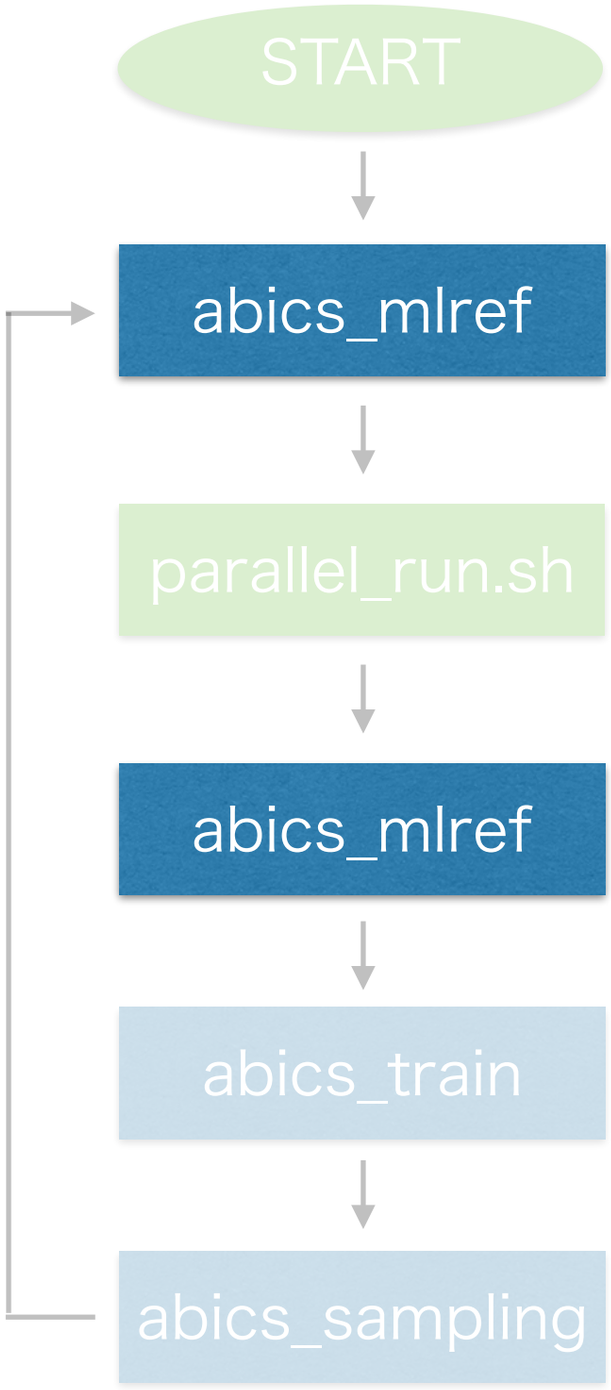
```
[[config.defect_structure.groups]]
name = 'Al'
# species = ['Al']    # default
# coords = [[[0,0,0]]]  # default
num = 16
```

st2abics

- [config] セクションを生成する補助スクリプト
  - VASPのPOSCAR や cif ファイルから生成できる
    - $ st2abics st2abics.toml mat.vasp > input.toml
  - [base_structure] で固定する原子種を設定
  - [[defect_structure]] で動かす原子種を設定

```
supercell = [2,2,2]

[base_structure]
species = ['O']
fix = false

[[defect_structure]]
site_center_species = ['Mg', 'Al']
```

ablCS の流れ



abics_* ablCS のスクリプト

モンテカルロサンプリングから原子配位データを抽出
もしくはランダム生成（初回）

それぞれの配位に関して第一原理計算を（並列）実行
GNU parallel を利用した参考スクリプトがexamples にある

第一原理計算の結果から教師データ（エネルギー）を抽出

第一原理計算のエネルギーを再現するように
機械学習モデルを学習

機械学習モデルでエネルギーを計算しつつ
原子配位をカノニカル分布からサンプリング

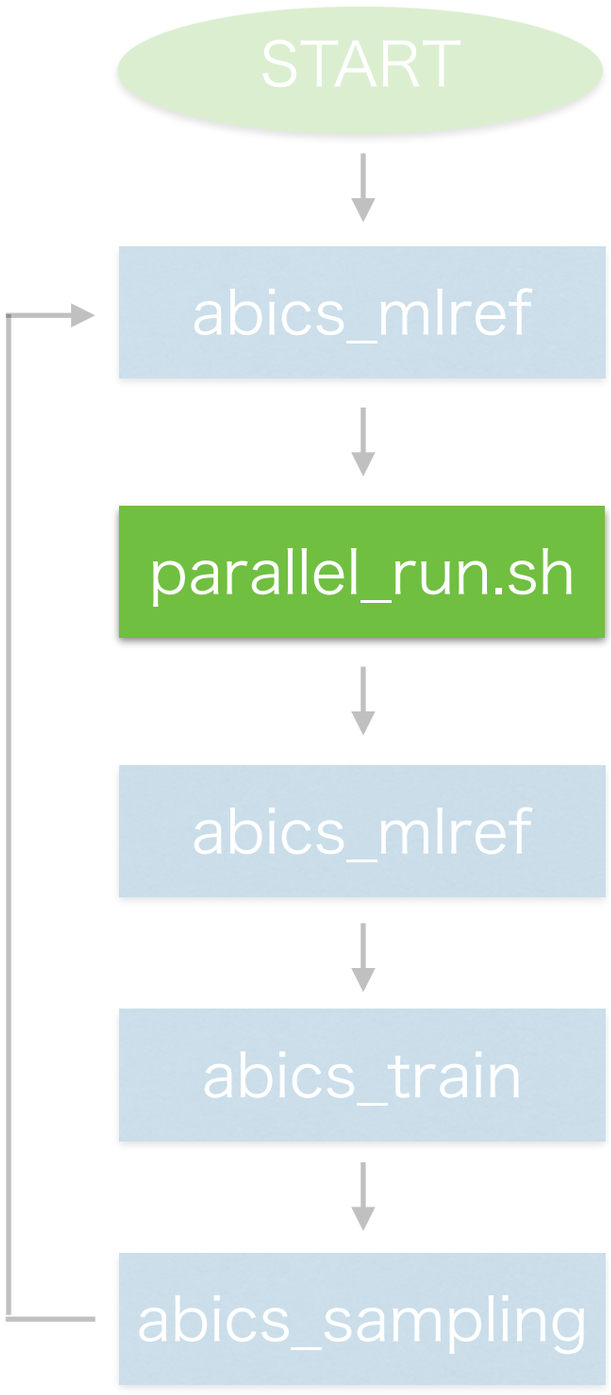
[mlref]

- abics_mlref スクリプトを制御するセクション
  - 機械学習ソルバーを学習するための入出力を制御するスクリプト
    - モンテカルロサンプリングから原子配位データを取り出して第一原理計算ソルバーの入力を作る
      - 初回はランダムサンプリング
    - 第一原理計算結果からエネルギーを取り出す
- nreplicas
  - レプリカ数
- ndata
  - 機械学習データとして取り出すモンテカルロサンプル数

[mlref.solver]

- abics_mlref スクリプトを制御するセクション
  - 第一原理計算ソルバーの入出力を読み書きするための情報
  - type
    - 第一原理計算ソルバーの種類
      - VASP, QE, OpenMX
  - base_input_solver
    - 第一原理計算ソルバーの入力ファイルを収めたディレクトリ
    - このディレクトリにあるファイルが自動的にコピーされる
    - ソルバーごとにファイル名に決まりがあるのでマニュアル参照
    - 複数定義可能（配列形式）
      - その回数だけ第一原理計算を実行する必要がある
      - 前の計算結果での原子座標を引き継ぐ
        - 第一原理計算における構造最適化の精度を徐々に上げる事が可能
  - perturb
    - 構造最適化のため、原子位置の初期値をランダムにずらすための変位（ガウスノイズの強さ）

ablCS の流れ



abics_* ablCS のスクリプト

モンテカルロサンプリングから原子配位データを抽出
もしくはランダム生成（初回）

それぞれの配位に関して第一原理計算を（並列）実行
GNU parallel を利用した参考スクリプトがexamples にある

第一原理計算の結果から教師データ（エネルギー）を抽出

第一原理計算のエネルギーを再現するように
機械学習モデルを学習

機械学習モデルでエネルギーを計算しつつ
原子配位をカノニカル分布からサンプリング

ハイスループット第一原理計算

- ALx/y/input_zzz/baseinput ディレクトリに第一原理計算ソルバの入力ファイルがある
  - x: 能動学習のループ数
  - y: レプリカ番号
  - zzz: モンテカルロサンプルの番号
- ディレクトリ一覧が rundirs.txt に保存される
- これらすべてに対して第一原理計算ソルバを実行し、エネルギーを計算する必要がある
  - 多くの計算機で使えそうな例として、 GNU Parallel を利用した例を examples で提供している
    - GNU Parallel: https://www.gnu.org/software/parallel/
    - parallel_run.sh および run_qe.sh / run_vasp.sh

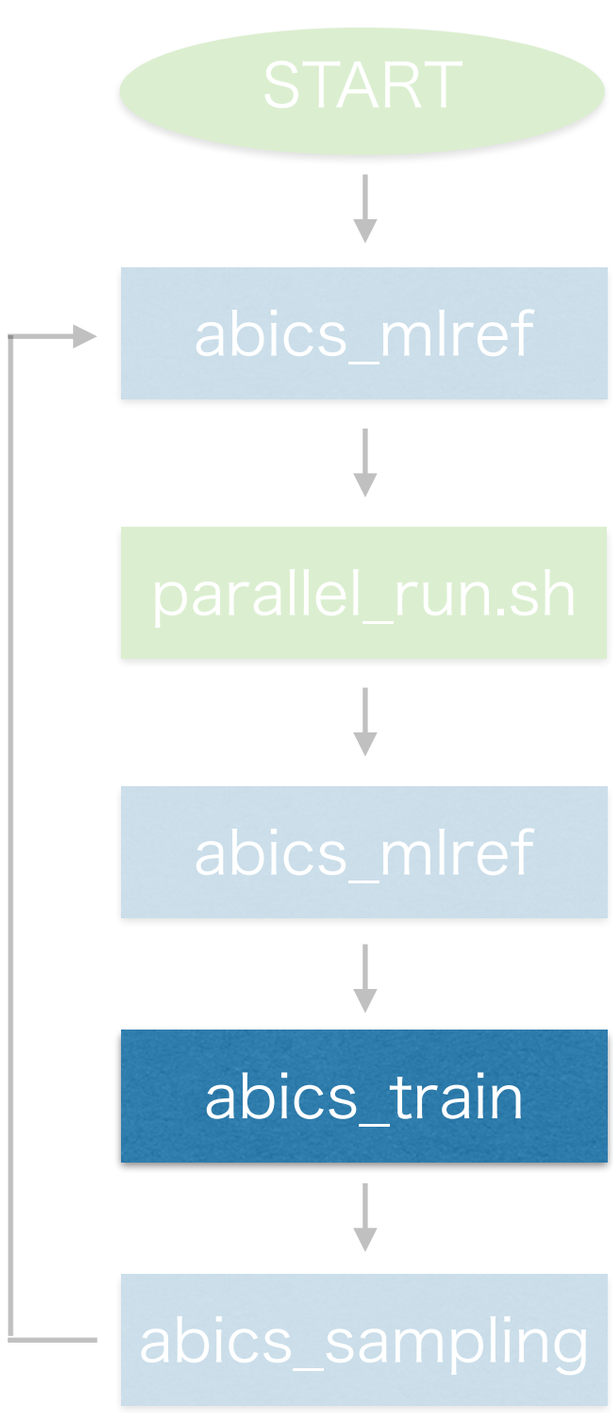
ハイスループット第一原理計算

- parallel
  - --delay 秒数
    - コマンドの最小実行間隔
  - -j 並列数
  - --joblog ログファイル
    - タスクの実行結果
  - --resume-failed
    - ログファイル中、失敗したタスクを実行
  - -a タスクリストファイル
    - 一行一タスク
    - 各行が「実行コマンド」の引数として実行される
  - 最終行
    - 実行コマンド
    - この例では run_pw.sh の引数としてQE の入力ファイルのディレクトリ名が渡される

```
# parallel_run.sh (抜粋)

RESTART=OFF # ON or OFF
if [ "_$RESTART" = "_ON" ]; then
    RESUME_OPT=--resume-failed
else
    RESUME_OPT=""
fi

parallel --delay 0.2 -j 4 \
     --joblog runtask.log \
     $RESUME_OPT -a rundirs.txt \
     "/bin/sh ./run_pw.sh"
sleep 30
```

```
# run_pw.sh (抜粋)
srun --exclusive --mem-per-cpu=1840 -n 32 -c 1 -N 1 \
    pw.x -in $1/scf.in > $1/stdout 2>&1

echo "$1" finished
```

ablCS の流れ



abics_* ablCS のスクリプト

モンテカルロサンプリングから原子配位データを抽出
もしくはランダム生成（初回）

それぞれの配位に関して第一原理計算を（並列）実行
GNU parallel を利用した参考スクリプトがexamples にある

第一原理計算の結果から教師データ（エネルギー）を抽出

第一原理計算のエネルギーを再現するように
機械学習モデルを学習

機械学習モデルでエネルギーを計算しつつ
原子配位をカノニカル分布からサンプリング

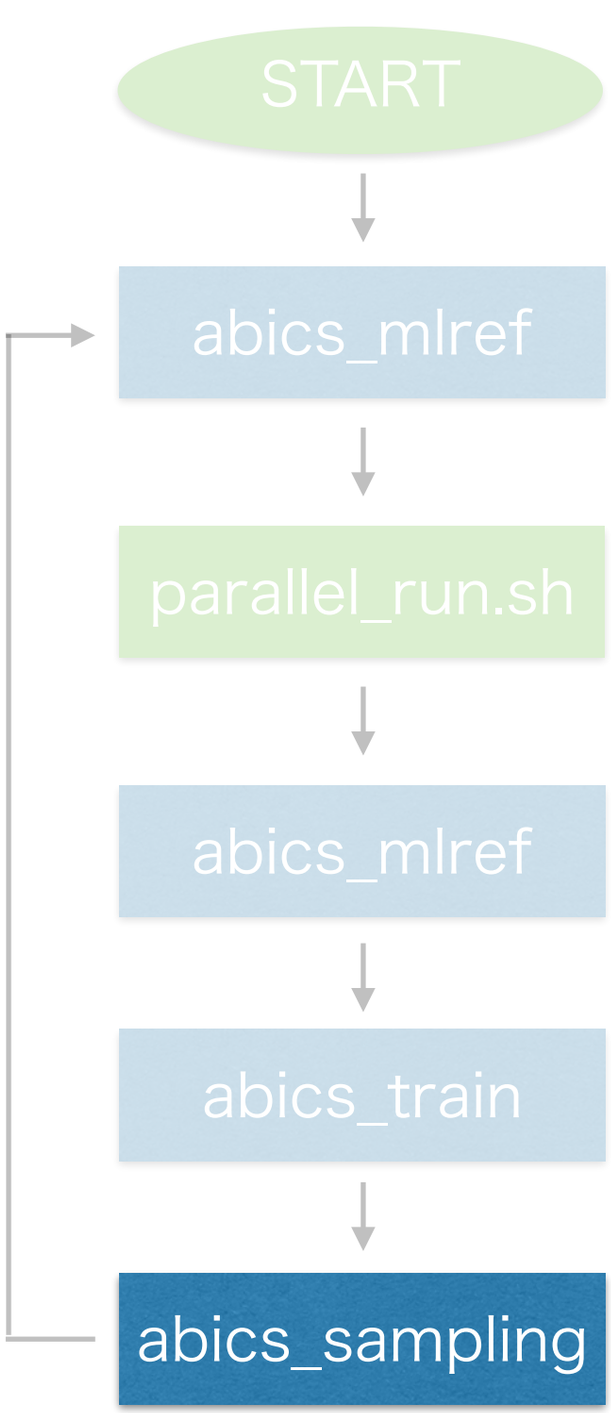
aenet

- 原子構造を入力としてエネルギーなどを得るニューラルネットワークモデルを扱うオープンソースソフトウェア
  - http://ann.atomistic.net/
  - 次の実行可能バイナリが生成される
    - `generate.x`
      - train.x で使えるように学習データセットを変換する
    - `train.x`
      - ニューラルネットモデルを学習する
    - `predict.x`
      - 原子構造から構造エネルギーを出力する
- MPI 並列化されている
  - 現在のabICS では generate.x と predict.x はシリアル版のみが利用可能

[train]

- abics_train スクリプトを制御するセクション
  - abics_mlref の結果を用いて機械学習モデルを学習するスクリプト
  - type
    - 機械学習モデルの種類
      - aenet が利用可能
  - exe_command
    - 機械学習モデルを学習するために実行するコマンド
    - aenet では generate.x と train.x を実行できるようにしてください
      - train.x はMPI 並列可能、 mpiexec コマンドなどを陽に指定してください
- base_input_dir
  - 学習機が用いる入力ファイル（設定など）
- ignore_species
  - 原子配置のうち、ソルバーの入力ファイルに出力しない原子種
  - 固定された原子などは機械学習モデルに与えないほうが効率が良くなることがある

ablCS の流れ



abics_* ablCS のスクリプト

モンテカルロサンプリングから原子配位データを抽出
もしくはランダム生成（初回）

それぞれの配位に関して第一原理計算を（並列）実行
GNU parallel を利用した参考スクリプトがexamples にある

第一原理計算の結果から教師データ（エネルギー）を抽出

第一原理計算のエネルギーを再現するように
機械学習モデルを学習

機械学習モデルでエネルギーを計算しつつ
原子配位をカノニカル分布からサンプリング

[sampling]

- abics_sampling スクリプトを制御するセクション
  - ・ レプリカ交換モンテカルロ法による有限温度サンプリングを行う
  - kTstart, kTend
    - ・ 温度の上限・下限
  - nreplicas
    - ・ レプリカ数、温度はこの数だけ分割される
  - nsteps
    - ・ モンテカルロサンプリング数
  - RXtrial_frequency
    - ・ レプリカ交換を何回の MC steps ごとに試みるか
    - ・ 1 のとき、毎回交換を試みる
  - sample_frequency
    - ・ 配位やエネルギーが記録される頻度
    - ・ 1 のとき、すべての配位が記録される
      - ・ ディスクI/O やファイル数に注意

[sampling.solver]

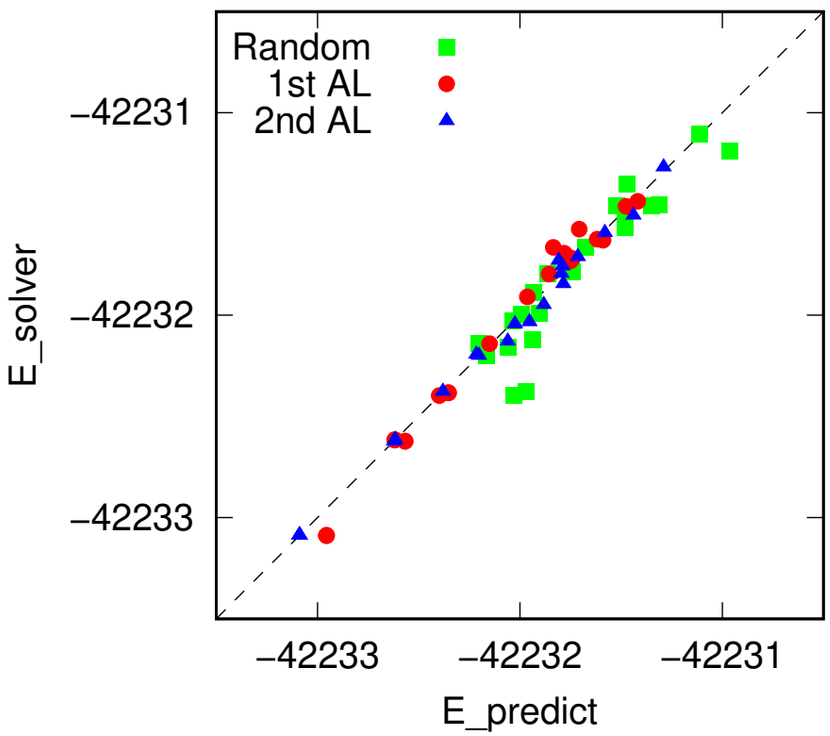
- abics_sampling スクリプトを制御するセクション
  - エネルギーソルバーの入出力を読み書きし、実行するための情報
  - type
    - エネルギーソルバーの種類
      - aenet
    - VASP などの第一原理計算ソルバーも指定可能
      - 並列計算実行機構において環境依存性が非常に強く、非推奨
  - run_scheme
    - 上記ソルバーを実行するための方法
    - aenet では "subprocess" を選択してください
  - base_input_dir
    - ソルバーが用いる入力ファイル
  - ignore_species
    - 原子配置のうち、ソルバーの入力ファイルに出力しない原子種
    - 固定された原子などは機械学習モデルに与えないほうが効率が良くなることがある

ablCS の流れ


START
abics_*
ablCS のスクリプト
abics_mlref
モンテカルロサンプリングから原子配位データを抽出
paralle
）実行
amples にある
abics
ー）を抽出
abics_train
第一原理計算のエネルギーを再現するように
機械学習モデルを学習
abics_sampling
機械学習モデルでエネルギーを計算しつつ
原子配位をカノニカル分布からサンプリング
計算が終わったら

計算の後処理

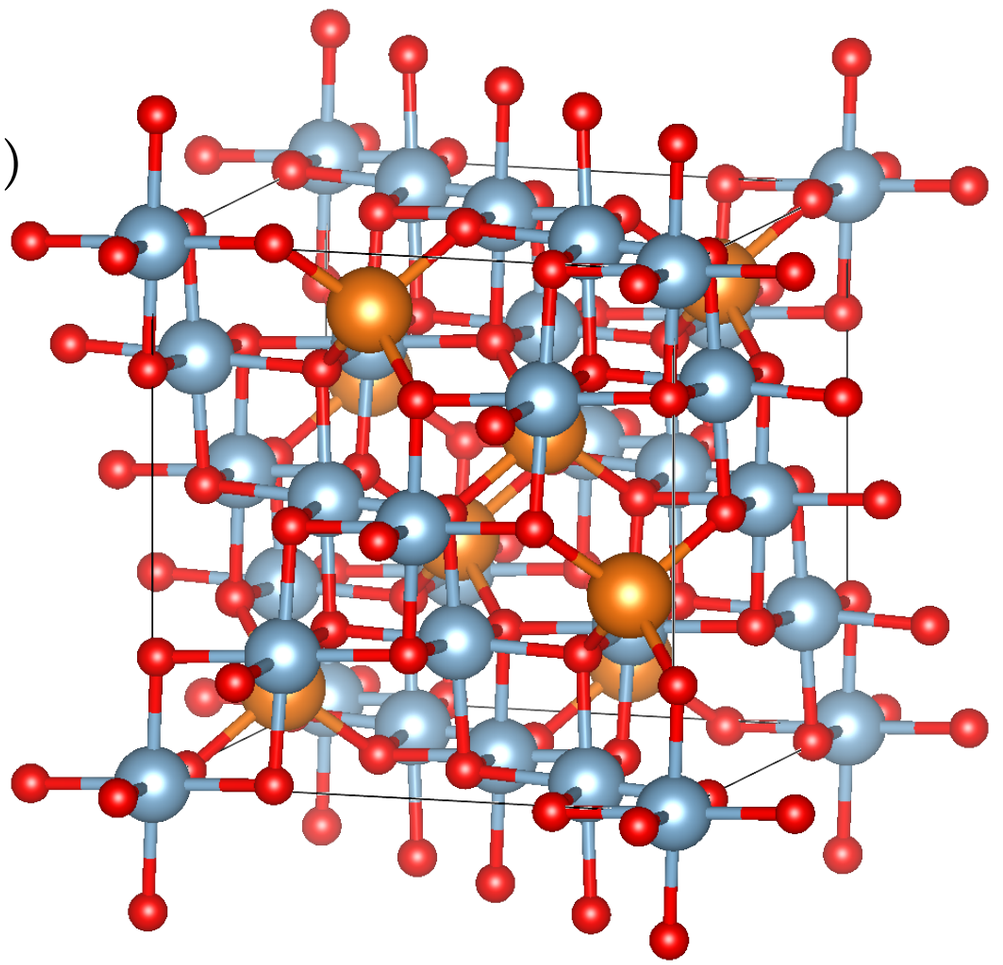
- ALx/y/energy_corr.dat
  - 機械学習モデルの出したエネルギー E_predict と第一原理計算ソルバーの出したエネルギー E_solver を並べたファイル
  - x: 能動学習のループ数
    - x=0 のときは機械学習モデルがないため対応するエネルギーは NaN が入っている
  - y: レプリカ番号
- 能動学習が進むと E_predict と E_solver の差がなくなっていく

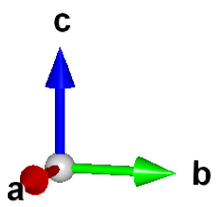


計算の後処理

- MCx/y/structure.zzz.vasp, structure_norel.zzz.vasp
  - モンテカルロサンプリングで生成した原子配置
    - VASP の POSCAR 形式
  - x: 能動学習のループ数
  - y: レプリカ番号
  - zzz: ステップ番号
  - norel: ignore_speciesで無視した原子も**含む**配置
- MCx/y/obs.dat
  - 各ステップにおける温度とエネルギー（機械学習ソルバー）
    - ステップ番号、温度、エネルギーの3列からなる

- 原子配置は各種ソフトウェアで可視化できる
  - 例：VESTA
    - https://jp-minerals.org/vesta/jp/





計算の後処理

- abicsRXsepT
  - レプリカ毎にディレクトリに入っている構造・エネルギーを温度毎に整理し直すスクリプト
  - MCx ディレクトリ内で用いる
    - mpiexec -np 8 abicsRXsepT ../input.toml
    - MPI 並列数は abics_sampling 実行時のものと同じにする
    - input.toml は abics_sampling と同じもの
    - 結果は Tseparate ディレクトリに保存される
      - 各温度点ごとのサブディレクトリに構造とエネルギーが保存される

計算の後処理

- calc_DOI.py
  - 最終的に何を計算するかは解きたい問題による
    - 例えば例題のMgAl2O4 ではMg/Al の反転度の温度依存性に興味がある
    - これを計算するスクリプトとして calc_DOI.py が examples に用意されている
  - MCx ディレクトリ内で用いる
  - 反転度の基準となる構造を示す MgAl2O4.vasp をカレントディレクトリに置く
- mpiexec -np 8 python3 ../calc_DOI.py ../input.toml
  - 結果が Tseparate/DOI_T.dat に出力される