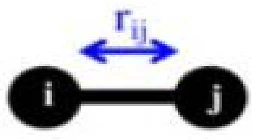
Machine-learned interatomic potentials

Plan

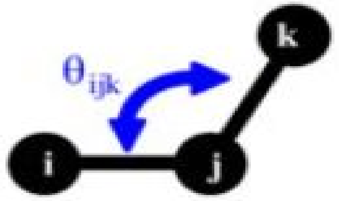
- (Semi-)empirical interatomic potentials: LJ, EAM, Tersoff, Coulomb, OPLS-AA, ReaxFF
- Advantages and disadvantages of (semi-)empirical interatomic potentials
- Machine-learning interatomic potentials: NNP, GAP, SNAP, MTP, ACE, DeepMD, PINN, NeQUIP
- Functional form of MTP
- Passive learning, training and validation errors, overfitting, uncertainty of estimation
- Comparison of different MLIPs
- Active learning
- Examples
- MLIP-2 package

Interatomic potentials

Interatomic potential is a mathematical function to calculate the potential energy of a system of N atoms. The general form is:



$$V = \sum_{i,j}^{N} V_2(r_{ij}) + \sum_{i,j,k}^{N} V_3(r_{ij}, r_{jk}, \theta_{ijk}) + \cdots \qquad (1)$$



Here $r_{ij} = |\vec{r}_i - \vec{r}_j|$ is the distance between atoms and $\theta_{ijk} = \mathrm{acos}\,\frac{(\vec{r}_{ij}, \vec{r}_{jk})}{r_{ij} \cdot r_{jk}}$ is the angle between the vectors $\vec{r}_{ij}$ and $\vec{r}_{jk}$.

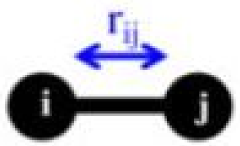
V_2 term describes two-body interactions, V_3 term and the terms of the higher order describe many-body interactions.

Bonded and non-bonded interactions

Bonded interactions

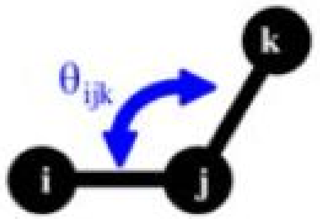
Bond potential

$$V_{bond} = \sum_{bonds} \frac{1}{2} k_{ij} (r_{ij} - r_0)^2$$

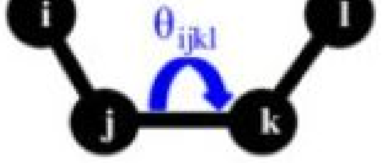


Angle bending potential

$$V_{angle} = \sum_{angles} \frac{1}{2} k_\theta (\theta_{ijk} - \theta_0)^2$$



Torsion potential



$$V_{dihedral} = \sum_{dihedrals} \frac{1}{2} V_n [1 + \cos(n\theta_{ijkl} - \delta)]$$

Non bonded interactions

Coulomb potential

$$V_{coulomb} = \sum_{ij} \frac{q_i q_j}{4\Pi \varepsilon_0 r_{ij}}$$

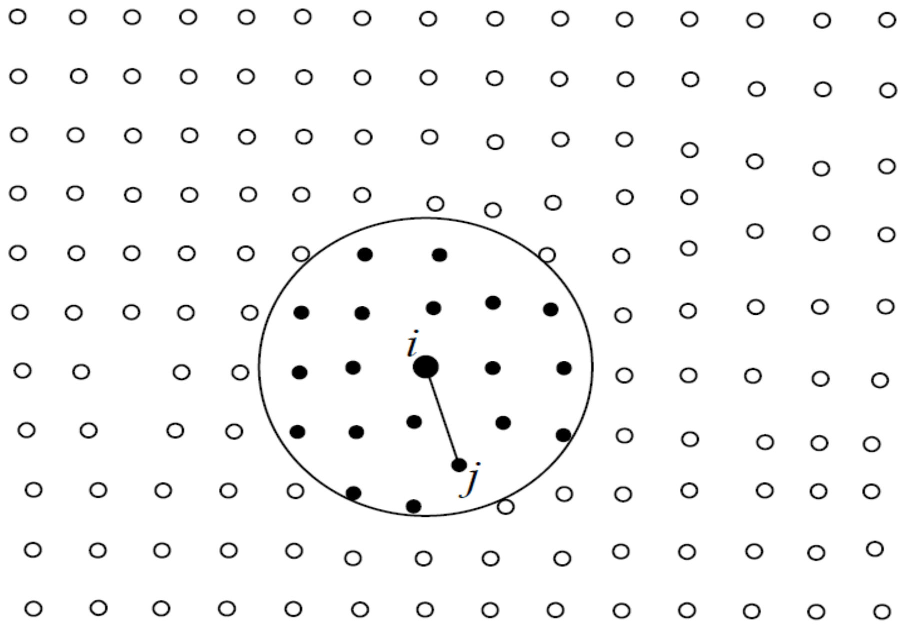
Lennard -Jones potential

$$V_{vdw} = \sum_{ij} 4\varepsilon_{ij} \left[\left(\frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}} \right)^{6} \right]$$

Local interatomic potentials

The sums in (1) run over all N atoms. If the range of interatomic potential is finite, we introduce the so-called cut-off radius R_{cut} and consider the atomic environments:

$$\mathfrak{n}_i = \{r_{ij} : j = 1, \dots, N_{nb}^i\},$$



where the number of neighbors N_{nb}^i is determined by R_{cut}, i.e. $r_{ij} < R_{cut}$. The potential energy $V = \sum_{i=1}^{N} V_i = \sum_{i=1}^{N} V(\mathfrak{n}_i)$. It should be smooth with respect to the atoms leaving and entering the environment and $V\left(r_{ij}\right) = 0$ if $r_{ij} \geq R_{cut}$.

(Semi-)empirical interatomic potentials

- Lennard-Jones potential (for noble gases and van der Waals interactions)
- Embedded atom model (for metals and alloys)
- Tersoff potential (for semiconductors and insulators)
- Coulomb potential (for long-range interactions)
- OPLS-AA (for organic molecules; is not applicable to chemical reactions)
- ReaxFF (for molecular systems; applicable to chemical reactions)

Advantages and disadvantages of (semi-)empirical potentials

Advantages:

- fast (can be applied to the systems of billion atoms)
- have a physical meaning

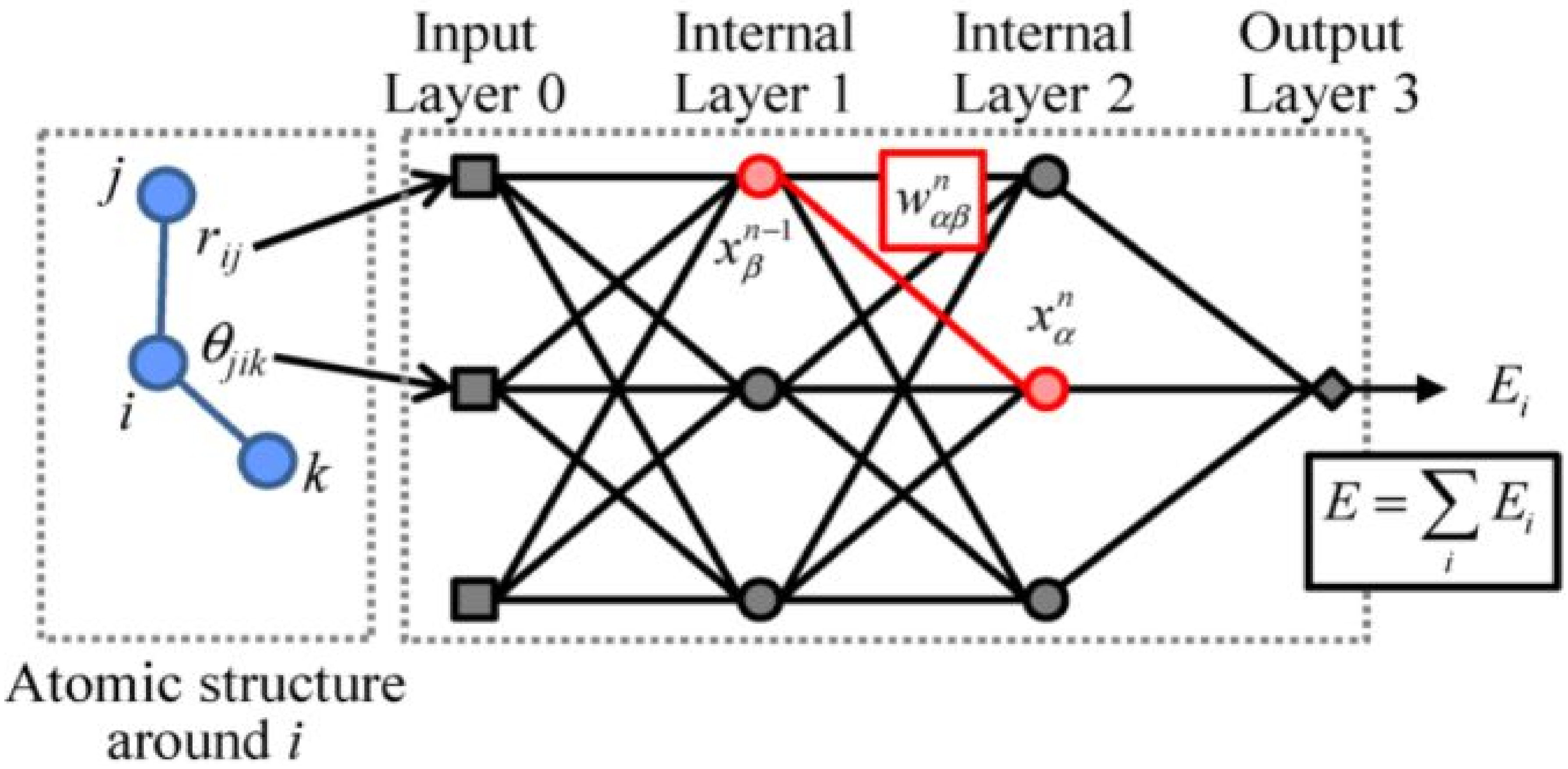
Disadvantages:

- not accurate enough (typically parametrized to DFT data, but are not able to predict properties of materials with DFT accuracy)
- applicable only to specific materials under specific conditions (problems with transferability)
- mainly used only for materials with small number of components

Alternative: machine-learned interatomic potentials!

Machine-learning interatomic potentials (MLIPs): general idea

Atomic positions are fed into the input layer, the output layer delivers the energy, and the hidden layers inserted in between provide additional adjustable parameters and enhance the flexibility of the model.

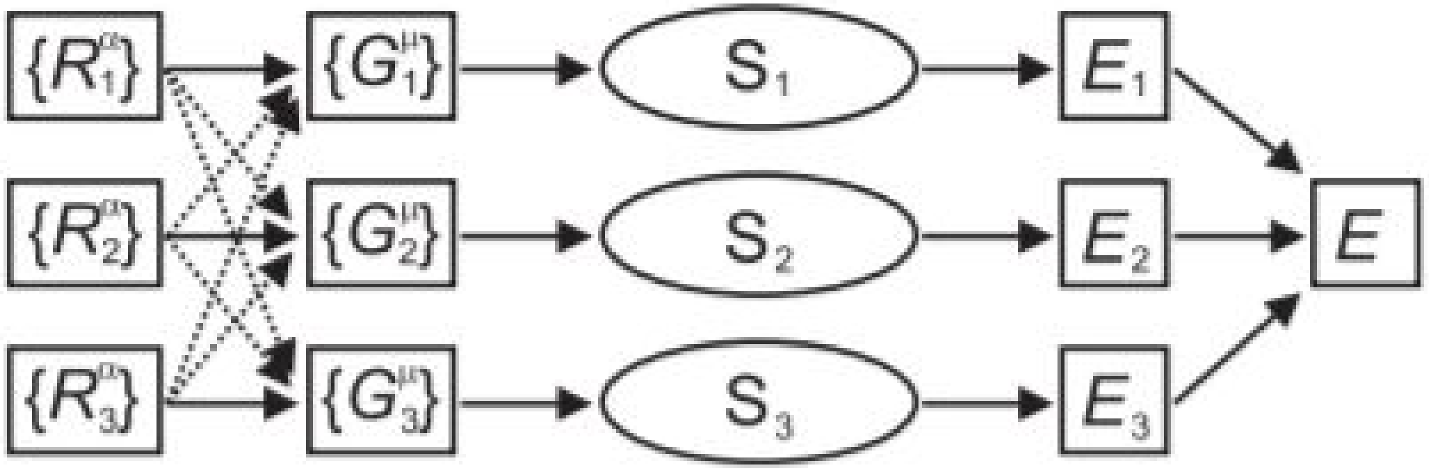


Neural Network Potential – the first MLIP

First machine-learning interatomic potential was proposed by Behler and Parrinello in 2007 [PRL 98, 146401 (2007)]. It is based on neural networks.

$$E_i^{NNP} = f_a^2\left[w_{01}^2 + \sum_{k=1}^{3} w_{k1}^2 f_a^1\left(w_{0k}^1 + \sum_{\mu=1}^{2} w_{\mu k}^1 G_i^\mu\right)\right]$$

Here w_{kl}^m is the weight parameter connecting node *l* in layer *m* with node *k* in layer *m*-1, f_a^m is the activation function, G_i^1 describes two-body interactions (radial part), and G_i^2 describes three-body interactions (angular part).



Machine-learned interatomic potentials

- Neural Network Potential (NNP) – based on neural networks
- Gaussian Approximation Potential (GAP) – based on Gaussian process regression
- Spectral Neighbor Analysis Potential (SNAP) – based on bispectrum components from GAP, polynomial potential
- Moment Tensor Potential (MTP) – based on moment tensor descriptors, polynomial potential

Recently developed MLIPs: polynomial ACE, and neural network-based DeepMD, PINN, and NeQUIP

Moment Tensor Potential

MTP was proposed in [Multiscale Model. Simul., Vol. 14, No. 3, pp. 1153-1173]. This potential is local, i.e. its energy E^{MTP} is the sum of contributions $V(\mathfrak{n}_i)$ of individual atomic neighborhoods $\mathfrak{n}_i$, $i = \overline{1, n}$:

$$E^{MTP} = \sum_{i=1}^{n} V(\mathfrak{n}_i) .$$

The function V is invariant to atomic permutations, rotations, and reflections, it is smooth with respect to atoms leaving and entering the interaction neighborhood. This function is linearly expanded through a set of basis functions B_α:

$$V(\mathfrak{n}_i) = \sum_{\alpha} \xi_\alpha B_\alpha(\mathfrak{n}_i) ,$$

where $\xi = \{\xi_\alpha\}$ is the set of linear parameters.

Moment Tensor Descriptor

Denote $\mathfrak{n}_i = (\{r_{i1}, z_i, z_1\}, \dots, \{r_{ij}, z_i, z_j\}, \dots, \{r_{iN_{Nb}}, z_i, z_{N_{Nb}}\})$, where $\vec{r}_{ij}$ are relative atomic positions, z_i, z_j are the types of central and neighboring atoms, Nb is the number of neighbors. Moment Tensor Descriptor:

$$M_{\mu,\nu}(\mathfrak{n}_i) = \sum_j f_\mu(|r_{ij}|, z_i, z_j) \underbrace{\vec{r}_{ij} \otimes \cdots \otimes \vec{r}_{ij}}_{\nu \text{ times}},$$

which consists of the radial part:

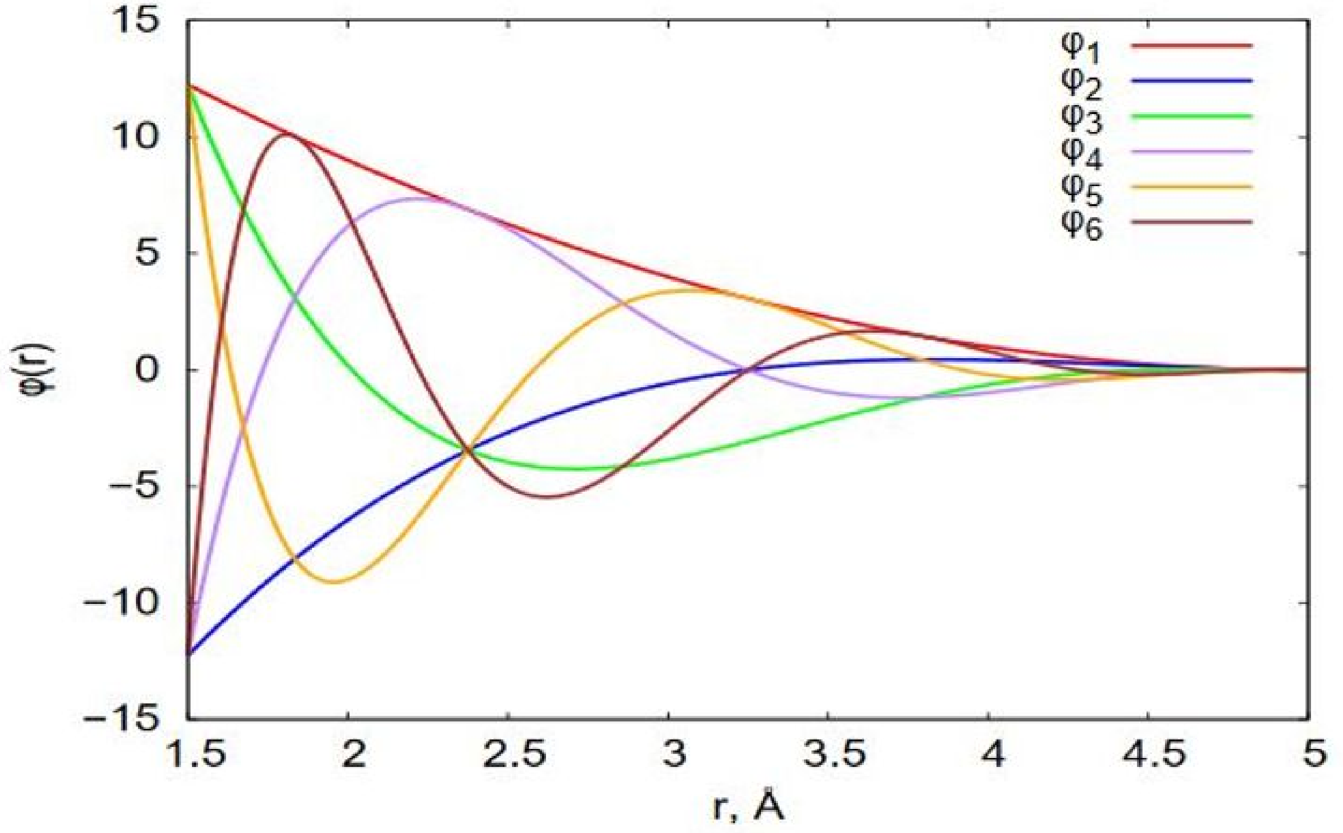
$$f_\mu(|r_{ij}|, z_i, z_j) = \sum_\beta c^{(\beta)}_{\mu, z_i, z_j} T_\beta(|r_{ij}|)(R_{cut} - |r_{ij}|)^2,$$

where $\boldsymbol{c} = \{c^{(\beta)}_{\mu, z_i, z_j}\}$ is the set of radial parameters, T_β are Chebyshev polynomials, and the angular part $\vec{r}_{ij} \otimes \cdots \otimes \vec{r}_{ij}$, where "$\otimes$" is the outer product, e.g., if $\nu = 2$:

$$\vec{r}_{ij} \otimes \vec{r}_{ij} = \begin{pmatrix} x_{ij}^2 & x_{ij}y_{ij} & x_{ij}z_{ij} \\ y_{ij}x_{ij} & y_{ij}^2 & y_{ij}z_{ij} \\ z_{ij}x_{ij} & z_{ij}y_{ij} & z_{ij}^2 \end{pmatrix}.$$

Radial basis example

$$\varphi_\beta(r) = \begin{cases} T_\beta(r)(R_{cut} - r)^2, & r < R_{cut} = 5\text{ Å} \\ 0, & r \geq R_{cut} = 5\text{ Å} \end{cases}$$



Level of Moment Tensor Descriptor

Level of Moment Tensor Descriptor:

$$\operatorname{lev} M_{\mu,\nu} = 2 + 4\mu + \nu$$

Level of Moment Tensor Descriptor Product:

$$\operatorname{lev} \prod_{p=1}^{P} M_{\mu_p,\nu_p} = \sum_{p=1}^{P} (2 + 4\mu_p + \nu_p)$$

Examples:

$$\operatorname{lev} M_{0,0} = 2, \operatorname{lev} M_{0,1} = 3, \operatorname{lev} M_{1,1} = 7, \operatorname{lev} M_{0,2} = 4$$

$$\operatorname{lev} M_{1,0}^{2} = 12, \operatorname{lev} M_{0,0}^{4} = 8, \operatorname{lev} M_{2,0}^{3} = 30$$

$$\operatorname{lev} (M_{1,1} \cdot M_{0,1}) = 10, \operatorname{lev} (M_{1,2} : M_{0,2}) = 12, \operatorname{lev} ((M_{0,3} M_{0,2}) \cdot M_{0,1}) = 12$$

Basis functions

Basis function B_α is a contraction of any number of Moment Tensor Descriptors, yielding a scalar, e.g.:

$$M_{\mu,0};\, M_{\mu,1} \cdot M_{\mu,1};\, M_{\mu,2} : M_{\mu,2},;\, \left(M_{\mu,2} \cdot M_{\mu,1}\right) \cdot M_{\mu,1},;\, \dots$$

Thus, level of basis function B_α:

$$\mathrm{lev} B_\alpha = \mathrm{lev} \prod_{p=1}^{P} M_{\mu_p,\nu_p} = \sum_{p=1}^{P} (2 + 4\mu_p + \nu_p) \ .$$

To define an MTP we choose some $\mathrm{lev}_{\max}$ and include in basis any function B_α with $\mathrm{lev} B_\alpha \leq \mathrm{lev}_{\max}$.

Example: MTP of level 8

Let $\mathrm{lev}_{\max} = 8$. Then we have 9 basis functions B_α:

$$B_1 = M_{0,0}; \mathrm{lev} M_{0,0} = 2 \le \mathrm{lev}_{\max}$$

$$B_2 = M_{1,0}; \mathrm{lev} M_{1,0} = 6 \le \mathrm{lev}_{\max}$$

$$B_3 = M_{0,0}^2; \mathrm{lev} M_{0,0}^2 = 4 \le \mathrm{lev}_{\max}$$

$$B_4 = M_{0,1} \cdot M_{0,1}; \mathrm{lev}(M_{0,1} \cdot M_{0,1}) = 6 \le \mathrm{lev}_{\max}$$

$$B_5 = M_{0,2} : M_{0,2}; \mathrm{lev}(M_{0,2} : M_{0,2}) = 8 \le \mathrm{lev}_{\max}$$

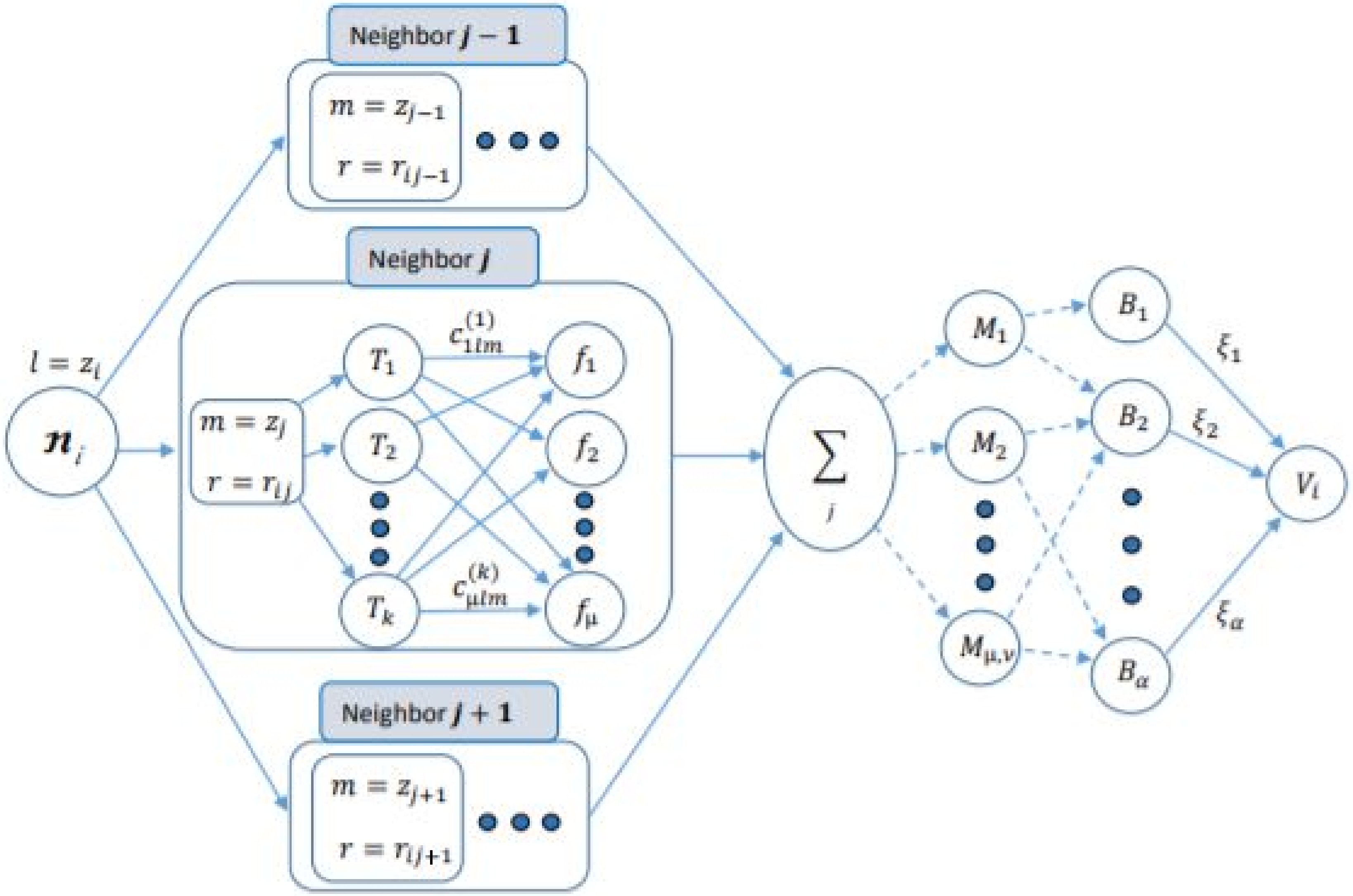
$$B_6 = M_{0,0} M_{1,0}; \mathrm{lev}(M_{0,0} M_{1,0}) = 8 \le \mathrm{lev}_{\max}$$

$$B_7 = M_{0,0}^3; \mathrm{lev} M_{0,0}^3 = 6 \le \mathrm{lev}_{\max}$$

$$B_8 = M_{0,0}(M_{0,1} \cdot M_{0,1}); \mathrm{lev}(M_{0,0}(M_{0,1} \cdot M_{0,1})) = 8 \le \mathrm{lev}_{\max}$$

$$B_9 = M_{0,0}^4; \mathrm{lev} M_{0,0}^4 = 8 \le \mathrm{lev}_{\max}$$

Computation scheme of MTP



Passive training (fitting) of MLIP

Let K be a number of configurations in a training set with DFT energies, forces, and stresses. Denote a set of MLIP parameters to be found by $\boldsymbol{\theta}$. The fitting consists of finding the parameters $\boldsymbol{\theta}$ that minimize the following loss function:

$$\sum_{k=1}^{K}\Bigg[w_e\left(E_k^{\mathrm{DFT}}-E_k^{\mathrm{MLIP}}(\boldsymbol{\theta})\right)^2 + w_f\sum_{i=1}^{n}\left|f_{i,k}^{\mathrm{DFT}}-f_{i,k}^{\mathrm{MLIP}}(\boldsymbol{\theta})\right|^2 + w_s\sum_{i=1}^{6}(\sigma_{i,k}^{\mathrm{DFT}}-\sigma_{i,k}^{\mathrm{MLIP}}(\boldsymbol{\theta}))^2\Bigg] \to \min,$$

where w_e, w_f, and w_s are non-negative weights.

Training errors

After fitting of MLIP we compute training errors:

$$\mathrm{RMSE}_{\mathrm{tr}}(E)^2 = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{E_k^{\mathrm{DFT}}}{n} - \frac{E_k^{\mathrm{MLIP}}(\boldsymbol{\theta})}{n}\right)^2,$$

$$\mathrm{RMSE}_{\mathrm{tr}}(f)^2 = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{3n}\sum_{i=1}^{n}\left|f_{i,k}^{\mathrm{DFT}} - f_{i,k}^{\mathrm{MLIP}}(\boldsymbol{\theta})\right|^2,$$

$$\mathrm{RMSE}_{\mathrm{tr}}(\sigma)^2 = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{6}(\sigma_{i,k}^{\mathrm{DFT}} - \sigma_{i,k}^{\mathrm{MLIP}}(\boldsymbol{\theta}))^2.$$

Validation of MLIP and overfitting

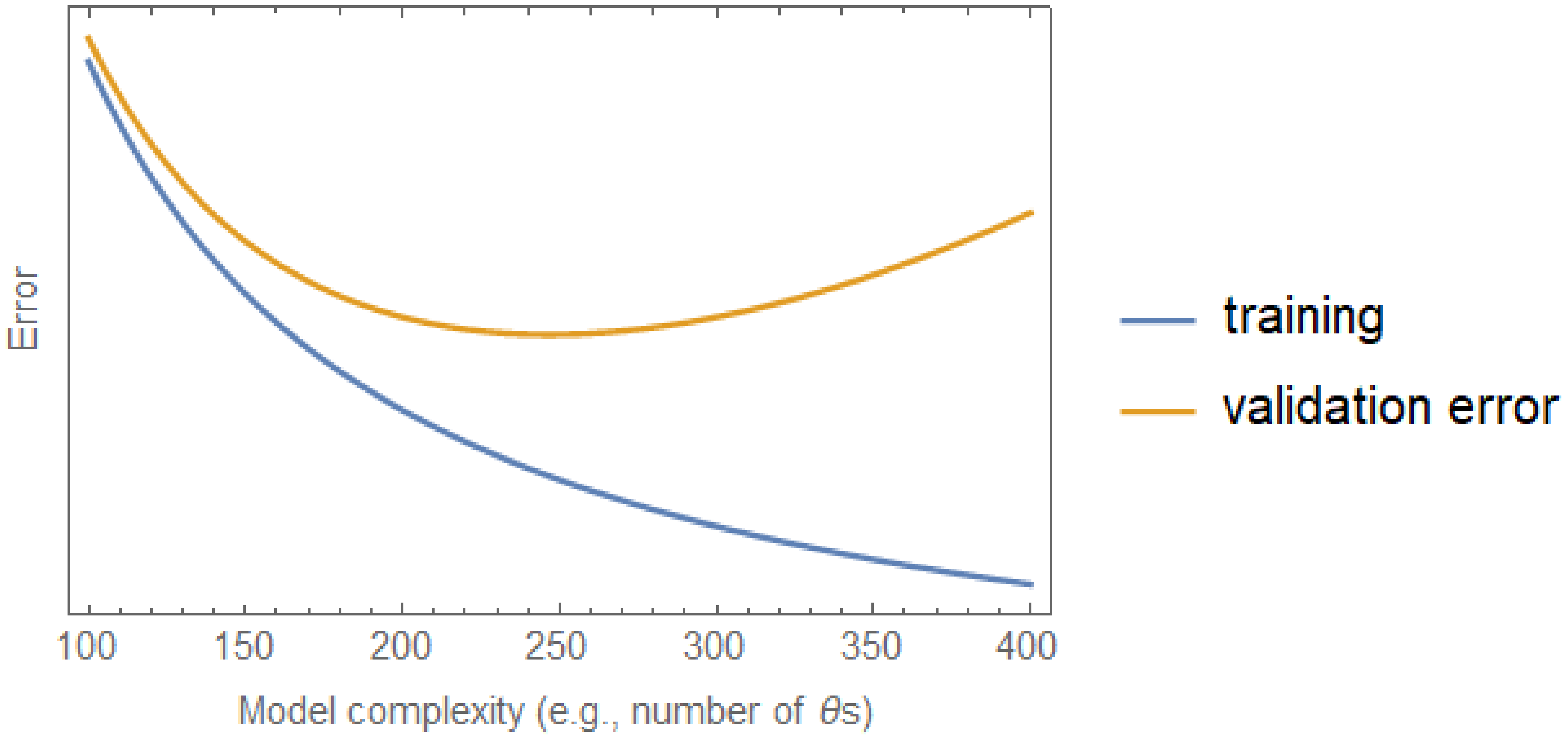
In order to estimate the quality of the trained MLIP we compute validation errors. For this aim we construct a validation set of *M* configurations that are not included in the training set and are not correlated with the ones in the training set. Typically $M/K \approx 0.2$. Next we calculate the validation errors.

Overfitting: $\mathrm{RMSE}_{\mathrm{tr}}(E)^2 \ll \mathrm{RMSE}_{\mathrm{vld}}(E)^2$ (e.g., the number of parameters in MLIP is greater than the number of configurations in the training set)

Underfitting: both $\mathrm{RMSE}_{\mathrm{tr}}(E)^2$ and $\mathrm{RMSE}_{\mathrm{vld}}(E)^2$ are too big (e.g., not enough parameters in MLIP)

Good fitting: $\mathrm{RMSE}_{\mathrm{tr}}(E)^2 \approx \mathrm{RMSE}_{\mathrm{vld}}(E)^2$ and these errors are small enough

Validation of MLIP: illustration



Ensemble of MLIPs and uncertainty estimation

As we typically start from random initial guess while MLIP training then any potential trained is something like a random value. Each potential can give different target values (predictions), i.e., energies, forces, stresses. Thus, we train an ensemble of 5-10 MLIPs and estimate an uncertainty of their predictions. In order to estimate an uncertainty of MLIP predictions we calculate average predictable value and standard deviation from this average value, e.g.:

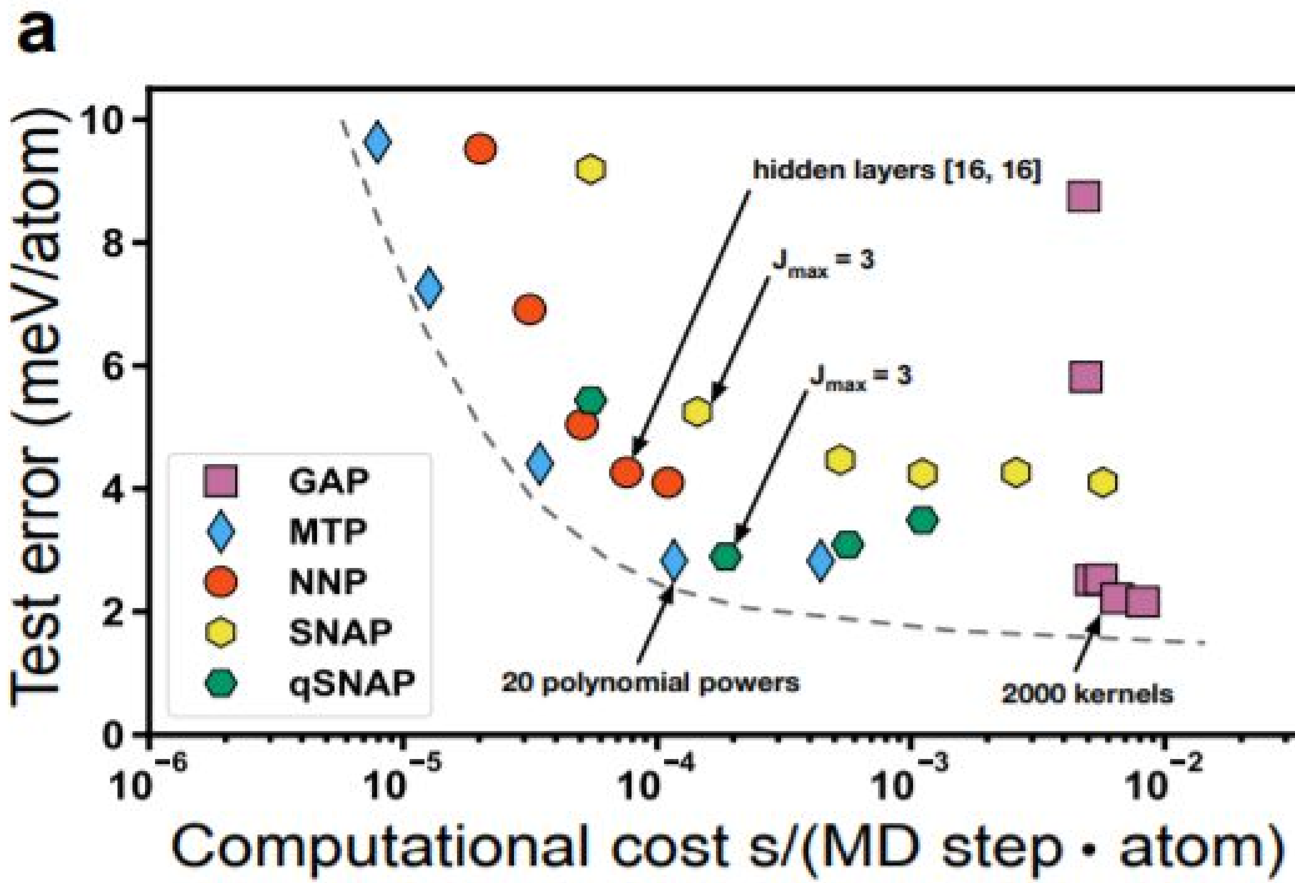
$$\mathrm{aRMSE}_{\mathrm{tr}}^{i}(E)^2 = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{E_k^{\mathrm{DFT}}}{n} - \frac{E_k^{\mathrm{MLIP}_i}(\boldsymbol{\theta})}{n}\right)^2,$$

$$\mathrm{aveRMSE}_{\mathrm{tr}}(E) = \frac{1}{N}\sum_{i=1}^{N}\mathrm{aRMSE}_{\mathrm{tr}}^{i}(E),$$

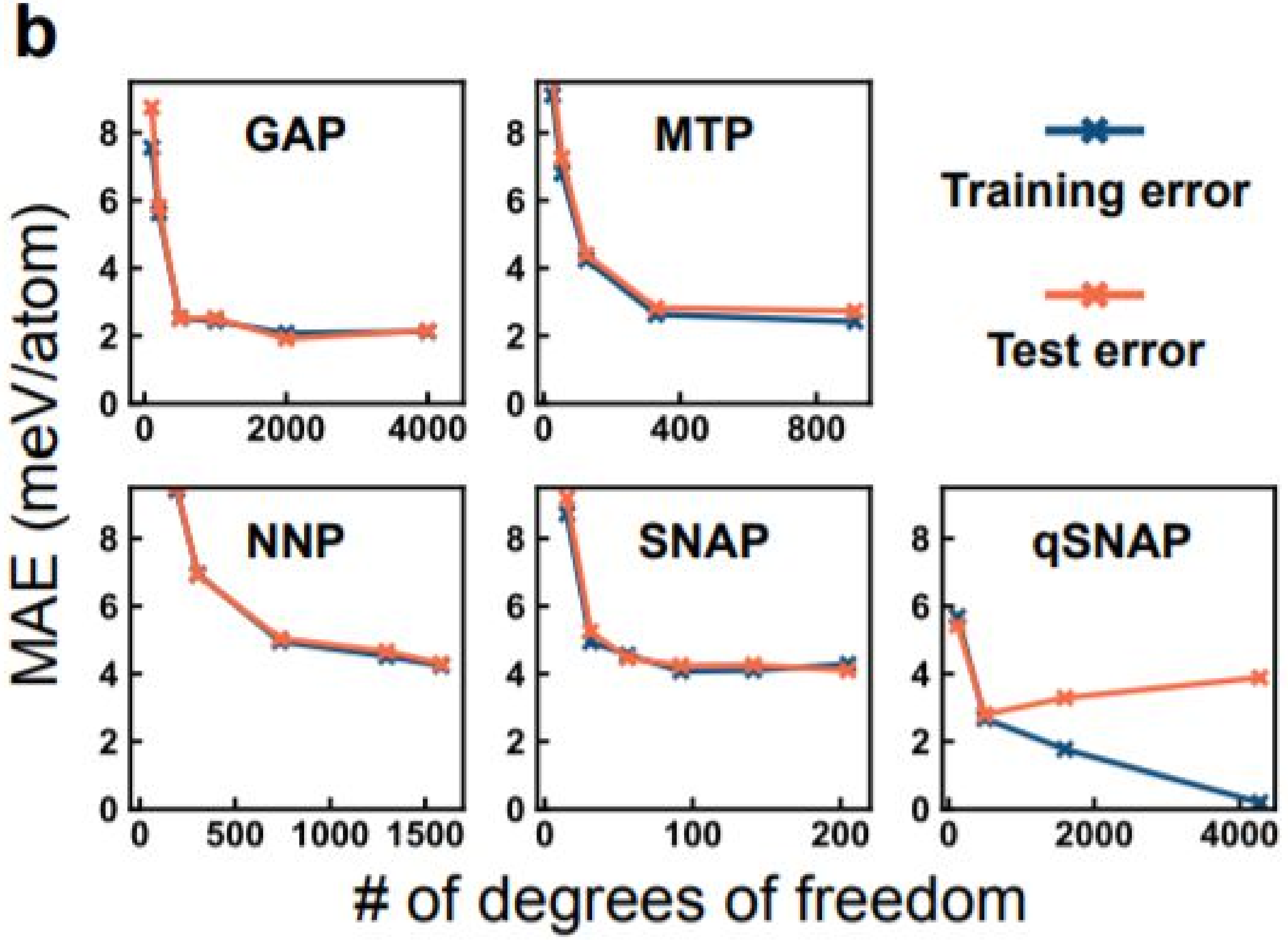
$$\mathrm{stdRMSE}_{\mathrm{tr}}(E) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(\mathrm{aveRMSE}_{\mathrm{tr}}(E) - \mathrm{aRMSE}_{\mathrm{tr}}^{i}(E)\right)^2}.$$

Comparison of different MLIPs: performance

In [J. Phys. Chem. A 2020, 124, 4, 731–745] NNP, GAP, SNAP, qSNAP, and MTP were trained on the same datasets and compared to each other in terms of performance and accuracy.



Comparison of different MLIPs: accuracy

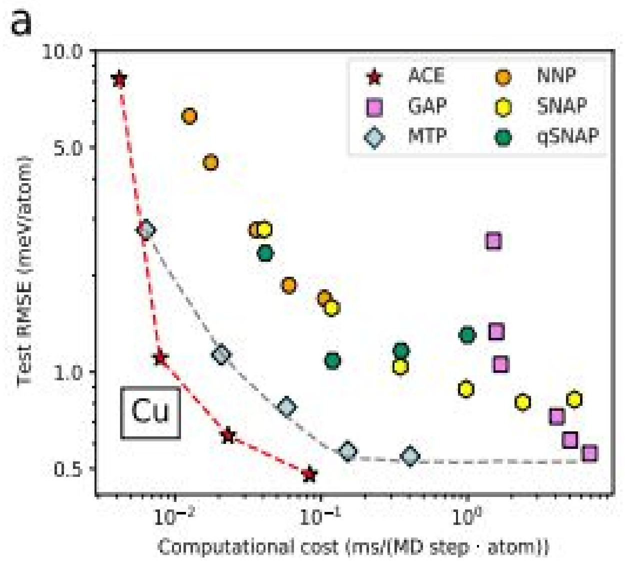


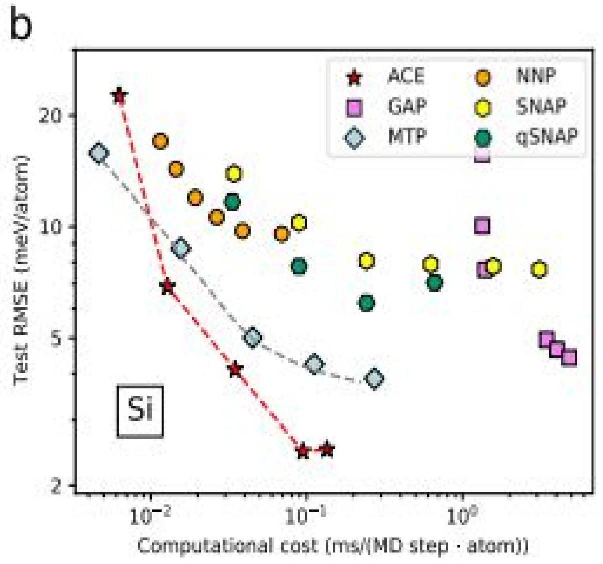
Atomic Cluster Expansion

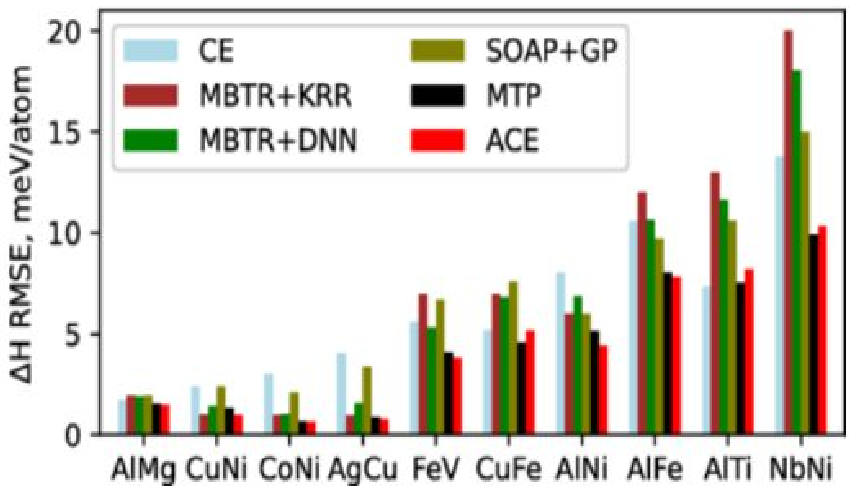
Recently, a new MLIP which is called ACE was developed [Phys. Rev. B 100, 249901 (2019)]. Descriptors of this MLIP are close to Moment Tensor Descriptors

$$M_{\mu,\nu}(\mathfrak{n}_i) = \sum_j f_\mu\left(|r_{ij}|, z_i, z_j\right) \underbrace{\vec{r}_{ij} \otimes \cdots \otimes \vec{r}_{ij}}_{\nu \text{ times}} ,$$

but spherical harmonics $Y_{lm}(\boldsymbol{r}/|r|)$ are used in ACE instead of $\boldsymbol{r}^{\otimes\nu}$ in MTP ($|l| \le m \le \nu$). Recently, ACE was implemented, tested for some materials and compared to the other MLIPs ([npj Computational Materials (2021) 7:97], [Phys.Rev.Mat. **6**, 013804 (2022)]).

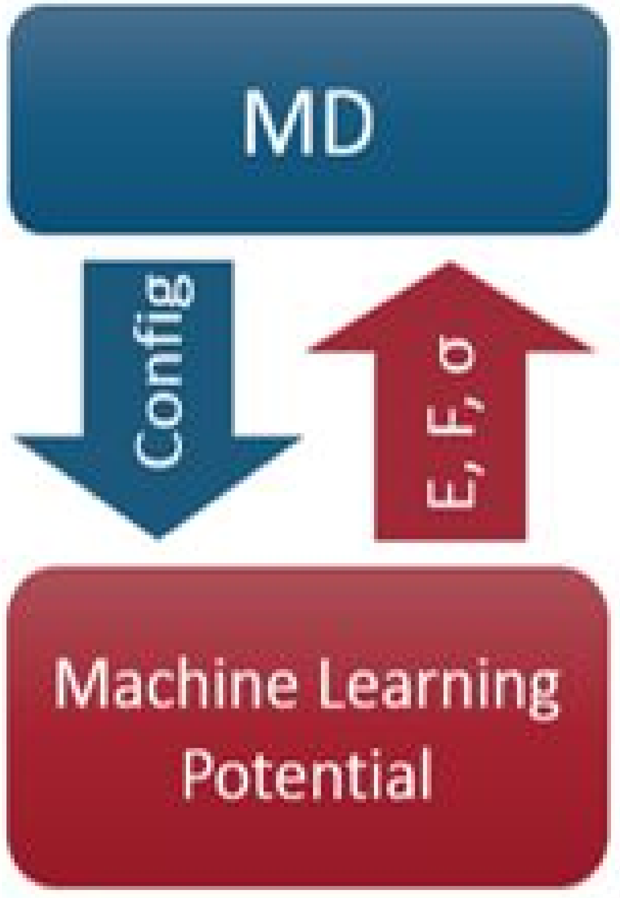






Passive learning

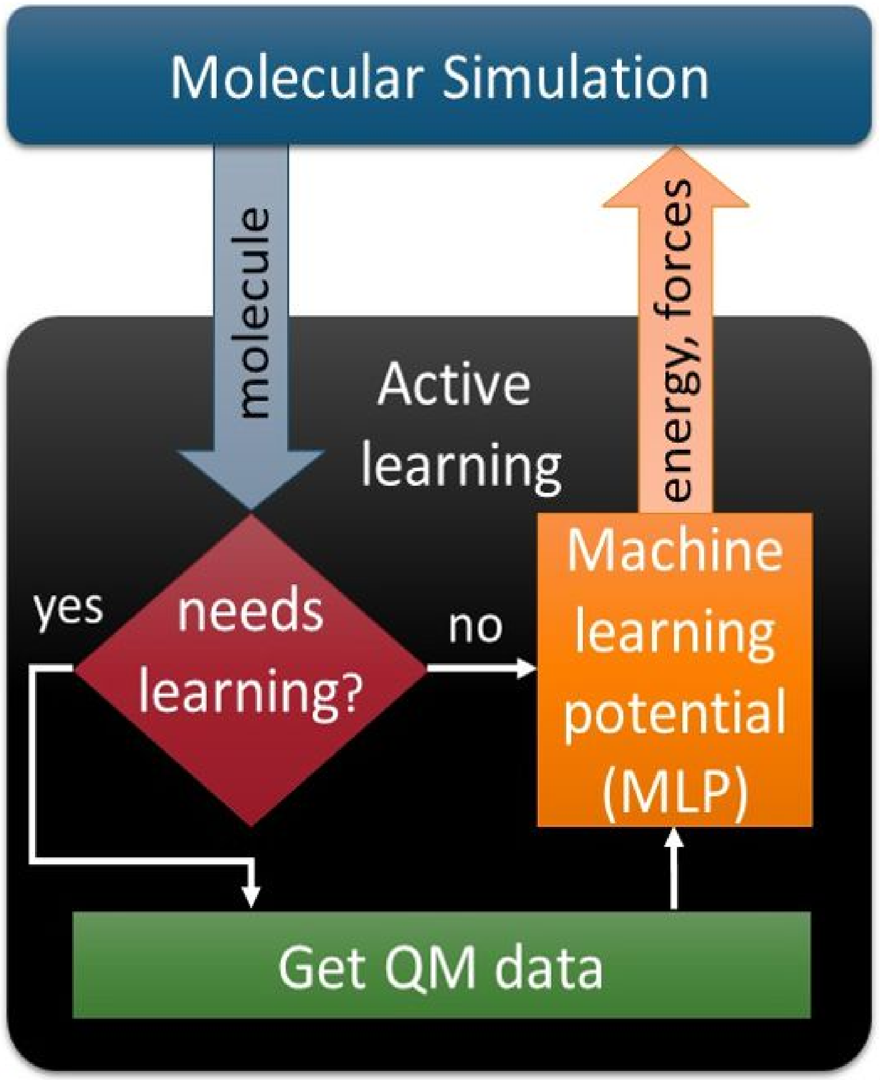
- Training set is being created off-line (e.g., random perturbations of atoms, non-correlated configurations from MD, etc.)
- Training set can cover only a small part of the configuration space
- MLIP trained on such the training set can have problems with transferability and often has “artifacts”
- Training set can contain “unnecessary” configurations and, thus, “unnecessary” ab initio calculations could be conducted
- Simulation (e.g., MD) runs with the same MLIP



How to optimally create a training set? Active learning!

Active learning

- Training set is being created on-line (i.e., during any simulation like MD, relaxation, etc.)
- MLIP "participates" in constructing the training set
- Training set covers almost the whole configuration space
- MLIP created during the active learning is transferable and does not have "artifacts"
- Active learning allows reducing a number of expensive ab initio calculations
- MLIP is being updated after each active learning step

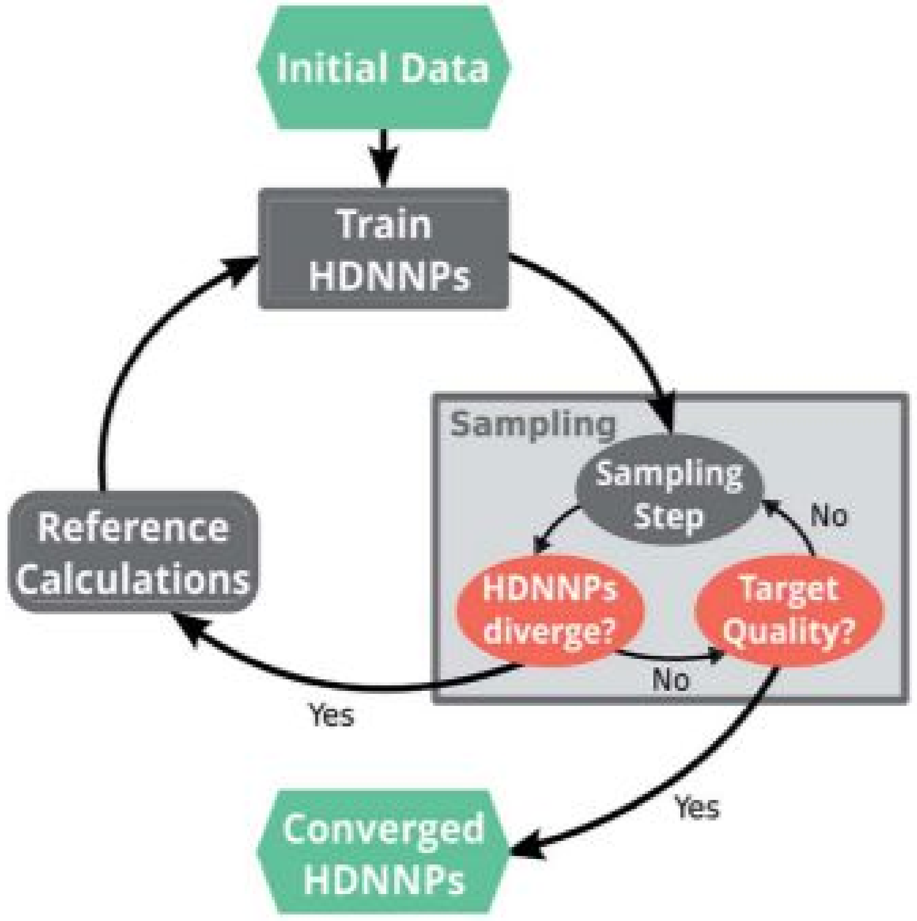


Example: active learning with an ensemble of MLIPs

In the paper [Chem. Sci., 2017, 8, 6924] the active learning algorithm with the ensemble of NNPs was proposed.

Algorithm

1. Create initial training set and train an ensemble of NNPs.
2. Run sampling step: calculate energy and forces for the current configuration. If the prediction uncertainty (standard deviation) of NNPs ensemble is too high, then we stop sampling step.
3. Run reference (ab initio) calculations for the selected configurations, update training set, re-train the ensemble of NNPs.
4. Run steps 2-3 until no problem configuration is found (i.e., the standard deviation is small enough).



Active learning (AL) with D-optimality criterion: active set

Assume we have a training set of K configurations $(\mathrm{cfg}_1, \ldots, \mathrm{cfg}_K)$ and the MTP with m optimized parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_m)$. Denote the MTP energy by $E = E(\boldsymbol{\theta}, \mathrm{cfg})$. By active set we define a subset of size m configurations from the training set that maximizes the determinant (volume) of the matrix:

$$A = \begin{bmatrix} \frac{\partial E}{\partial \theta_1}(\boldsymbol{\theta}, \mathrm{cfg}_1) & \cdots & \frac{\partial E}{\partial \theta_m}(\boldsymbol{\theta}, \mathrm{cfg}_1) \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial \theta_1}(\boldsymbol{\theta}, \mathrm{cfg}_m) & \cdots & \frac{\partial E}{\partial \theta_m}(\boldsymbol{\theta}, \mathrm{cfg}_m) \end{bmatrix}$$

For the case $K > m$ in order to find the active set we use the so-called MaxVol algorithm. If $K \leq m$ we include some of (or, even, all) K configurations in the active set and fill in the rest of the lines with zeros outside the diagonal and with ones in the diagonal.

AL with D-optimality criterion: extrapolation grade

Assume we have any atomistic simulation: MD, relaxation, etc. In order to decide whether to consider a given configuration cfg^* as a candidate for inclusion in the training set we introduce the so-called extrapolation grade:

$$\gamma(\mathrm{cfg}^*) = \max_{1 \le j \le m} \left(|c_j| \right), \text{ where}$$

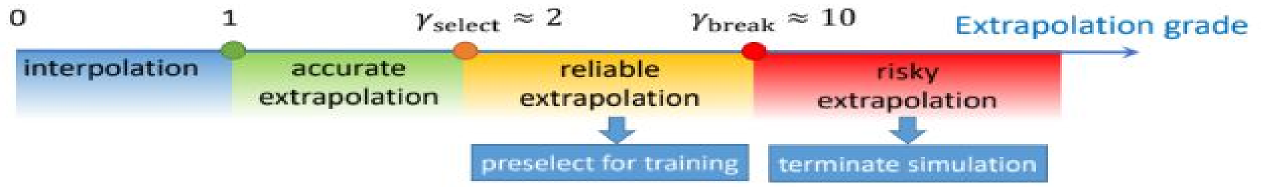
$$\boldsymbol{c} = \left(\frac{\partial E}{\partial \theta_1}(\boldsymbol{\theta}, \mathrm{cfg}^*) \ldots \frac{\partial E}{\partial \theta_m}(\boldsymbol{\theta}, \mathrm{cfg}^*) \right) A^{-1}.$$

This grade defines the maximal factor by which the determinant $\det(A)$ can increase if cfg^* is added to the active set, i.e., if $\gamma(\mathrm{cfg}^*) > 1$ then $\det(A)$ could be increased. We say that the MTP extrapolates if $\gamma(\mathrm{cfg}^*) > 1$, and interpolates otherwise.

AL with D-optimality criterion: two-threshold scheme

For creating the so-called preselected set we choose two thresholds:

$\gamma_{\mathrm{select}} \approx 2$ and $\gamma_{\mathrm{break}} \approx 10$ [Mach. Learn.: Sci. Technol. 2 (2021) 025002].



If $\gamma_{\mathrm{select}} \leq \gamma(\mathrm{cfg}^*) \leq \gamma_{\mathrm{break}}$ then we add the configuration to the preselected set and continue an atomistic simulation. If $\gamma(\mathrm{cfg}^*) > \gamma_{\mathrm{break}}$ the we break an atomistic simulation and form a matrix $P \times m$, where P is the number of preselected configurations:

$$B = \begin{bmatrix} \frac{\partial E}{\partial \theta_1}(\boldsymbol{\theta}, \mathrm{cfg}_1) & \cdots & \frac{\partial E}{\partial \theta_m}(\boldsymbol{\theta}, \mathrm{cfg}_1) \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial \theta_1}(\boldsymbol{\theta}, \mathrm{cfg}_P) & \cdots & \frac{\partial E}{\partial \theta_m}(\boldsymbol{\theta}, \mathrm{cfg}_P) \end{bmatrix}$$

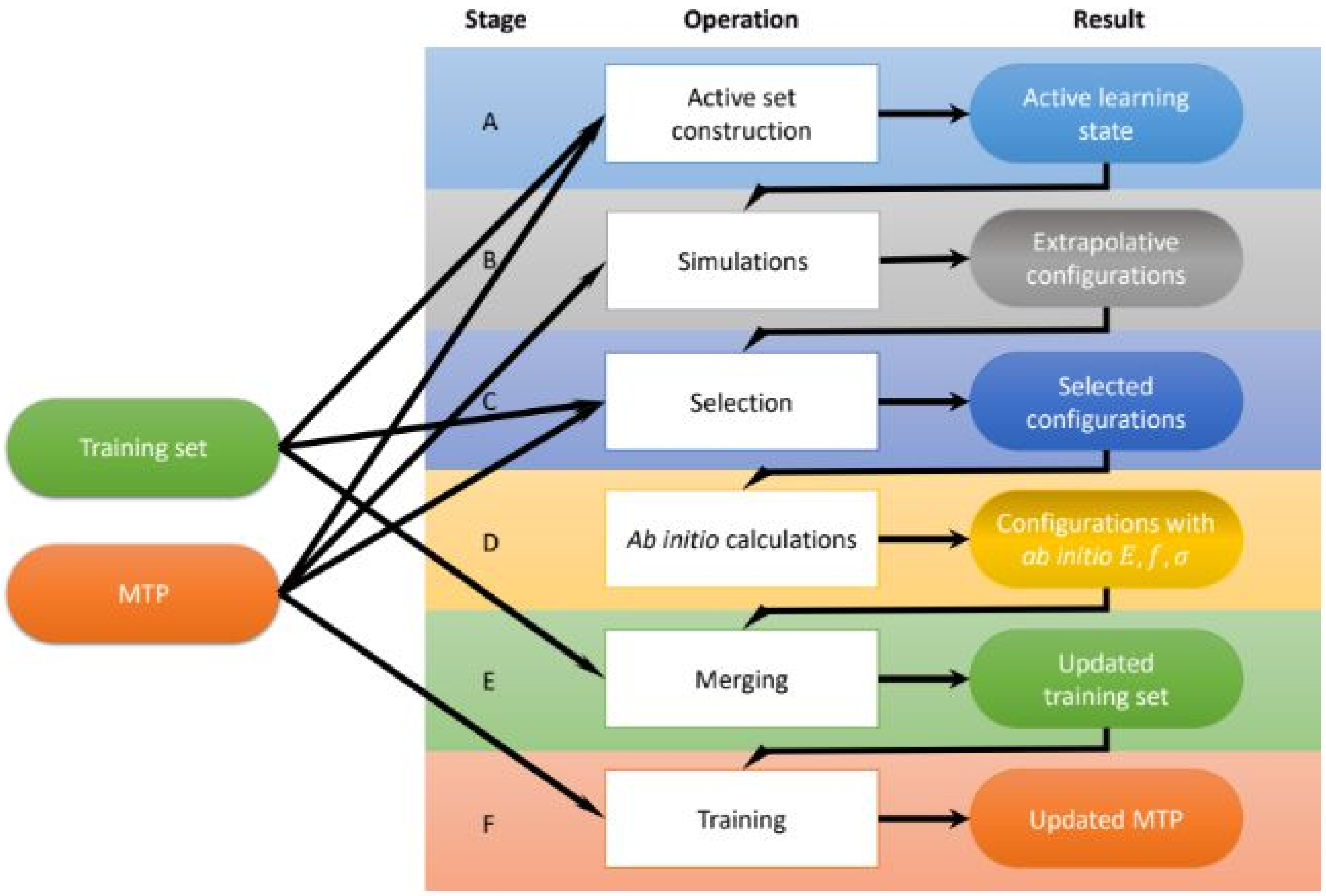
AL: update the active set and the training set

With the MaxVol algorithm we select the configurations that maximize the volume of the matrix A (or, $\det(A)$). Thus, we select $S \leq m$ configurations from the preselected set, or, in other words, we find a submatrix of maximum volume from a matrix:

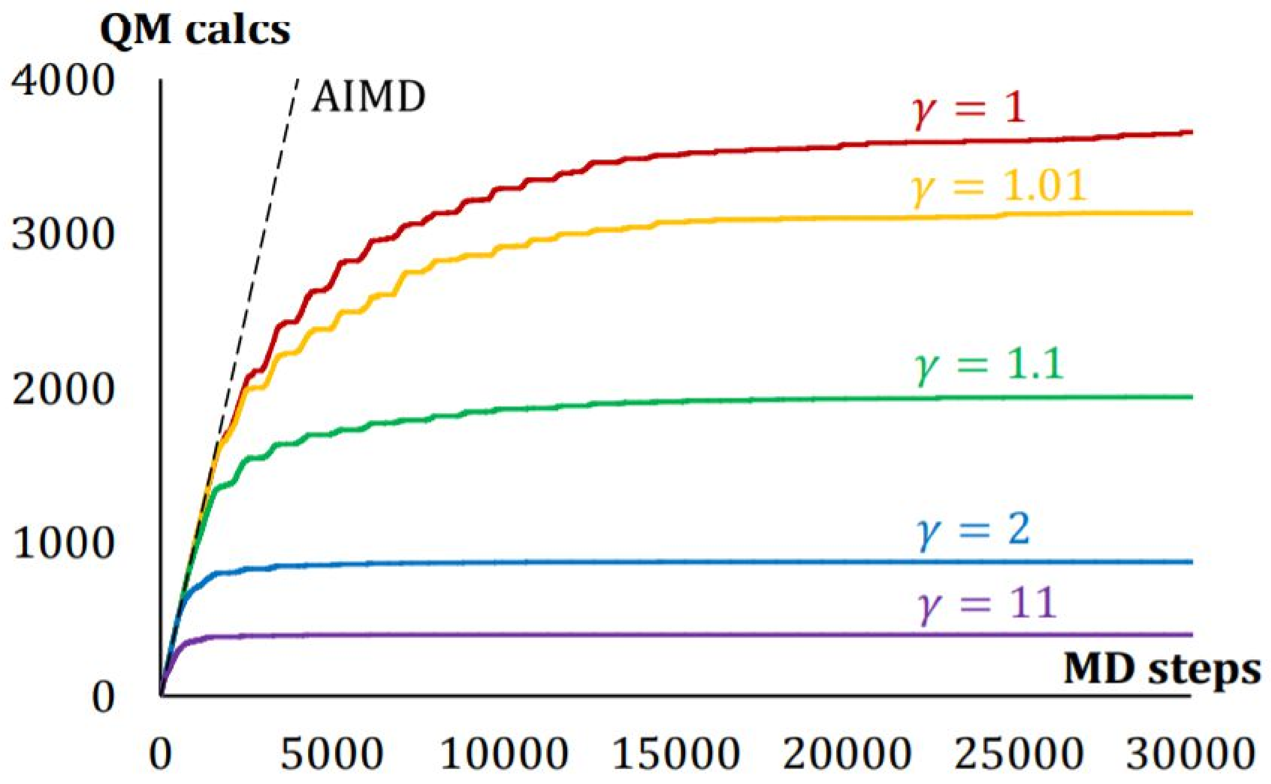
$$\tilde{B} = \begin{bmatrix} \frac{\partial E}{\partial \theta_1}(\boldsymbol{\theta}, \mathrm{cfg}_1) & \cdots & \frac{\partial E}{\partial \theta_m}(\boldsymbol{\theta}, \mathrm{cfg}_1) \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial \theta_1}(\boldsymbol{\theta}, \mathrm{cfg}_{P+m}) & \cdots & \frac{\partial E}{\partial \theta_m}(\boldsymbol{\theta}, \mathrm{cfg}_{P+m}) \end{bmatrix}$$

where the first m lines are from the matrix A. Next, we use DFT calculations for obtaining energies, forces, and stresses of these configurations, append them to the training set. We also update the active set by substituting S lines in the matrix A. Finally, we re-train MTP on the updated training set.

Scheme of active learning iterations

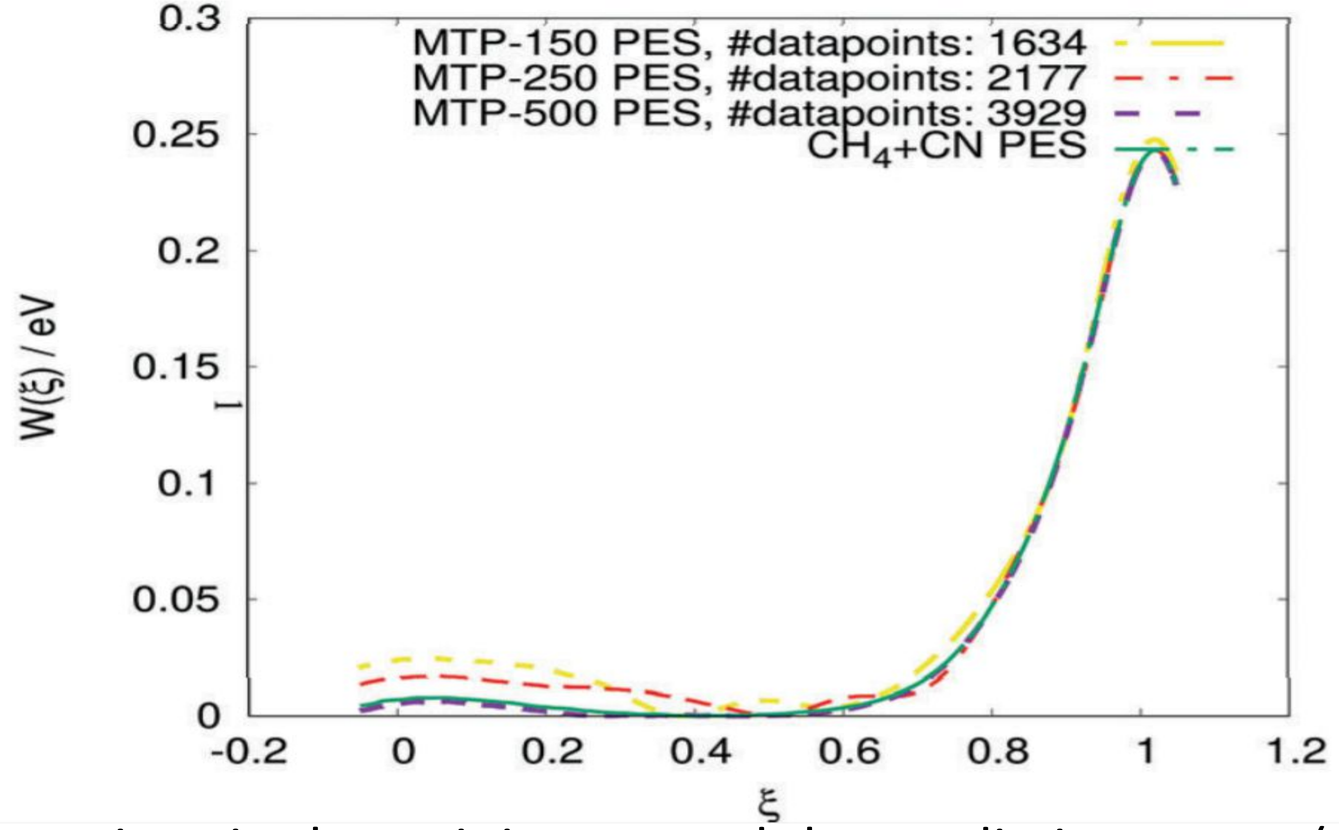


Example: MD in NVT-ensemble for bcc-Li at T=300 K



Number of DFT (QM) calculations decreases with the increase of γ_{select} [Computational Materials Science 140 (2017) 171–180]. Most of the QM calculations are performed within the few picoseconds at $\gamma_{\text{select}} \geq 1.1$.

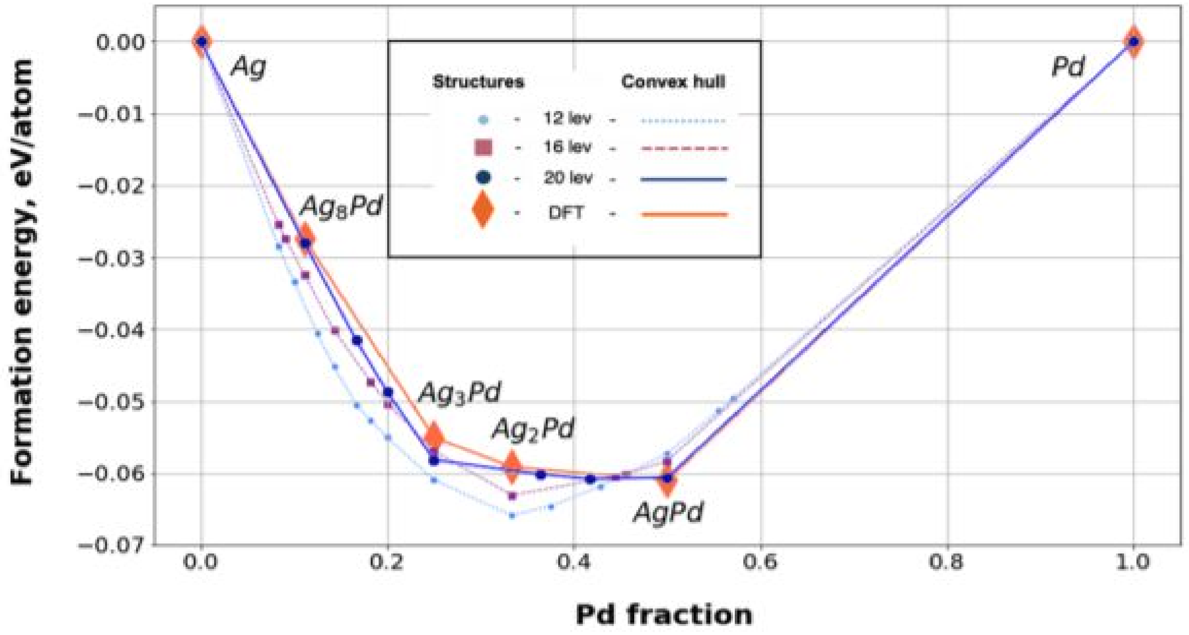
Example: MD with Andersen thermostat for the chemical reaction $CH_4+CN \rightarrow CH_3+HCN$ at T=300 K



Number of configurations in the training set and the predictive power (or, accuracy) of MTP increases with the increase of the number of MTP parameters [Phys. Chem. Chem. Phys., 2018, 20, 29503-29512].

Example: Convex hull of AgPd

lev_{max} (# parameters)	Train set size	Energy RMSE (meV/atom)
12 (29+2+4*3*12=175)	226	3.8
16 (92+2+4*4*12=286)	442	2.4
20 (288+2+4*5*12=530)	920	1.7



Convex hulls obtained by MTPs of different levels. The same conclusion as in the previous slide [Mach. Learn.: Sci. Technol. 2 (2021) 025002].

MLIP-2 package

Developers: Alexander V. Shapeev, Evgeny V. Podryabinkin, Konstantin Gubaev, and Ivan S. Novikov

Moment Tensor Potentials and algorithms for their training are implemented in the MLIP-2 package

MLIP-2 is an open-source code available at: https://gitlab.com/ashapeev/mlip-2

An interface between LAMMPS and MLIP-2 available at: https://gitlab.com/ashapeev/interface-lammps-mlip-2