CaMMLS
Chemistry and Materials
Machine Learning School

Newcastle University

Machine Learning Interatomic Potentials I

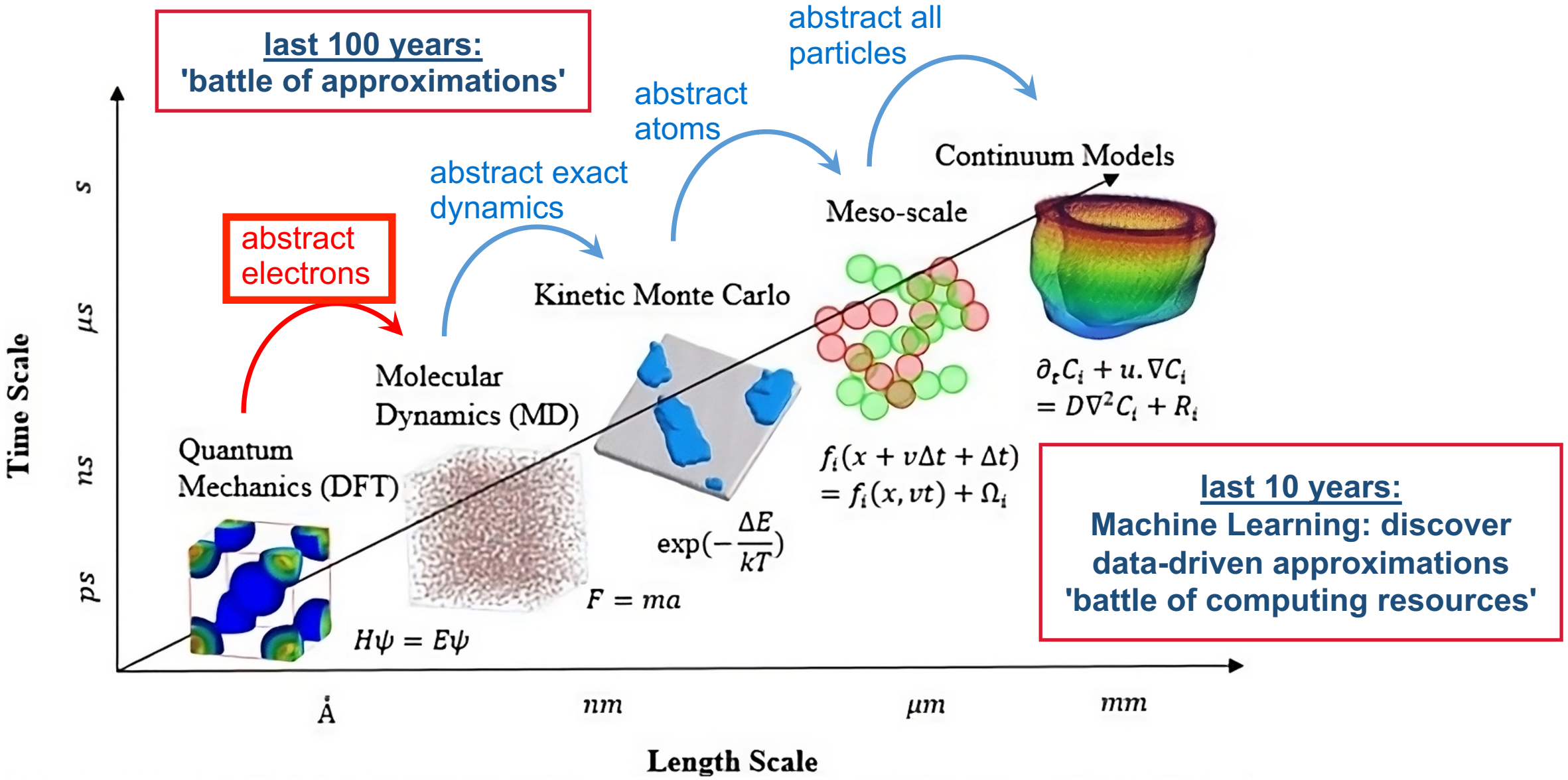
Ioan-Bogdan Magdău
ioan.magdau@ncl.ac.uk

CAMMLS Spring School 2026

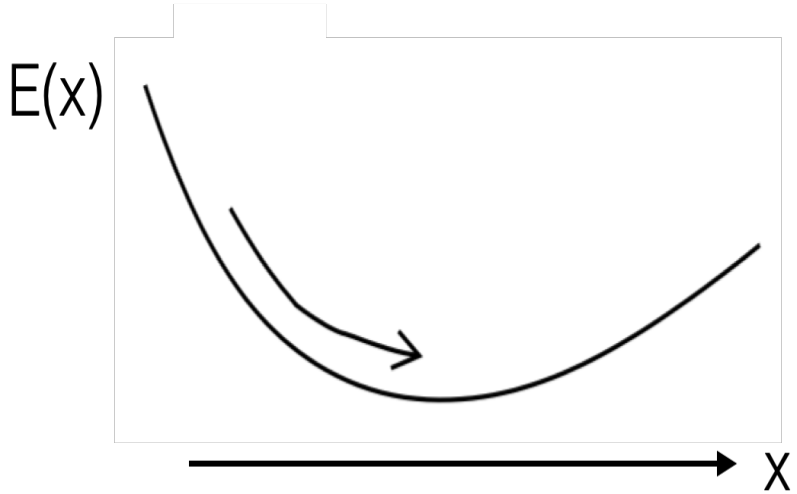
Lecture Outline

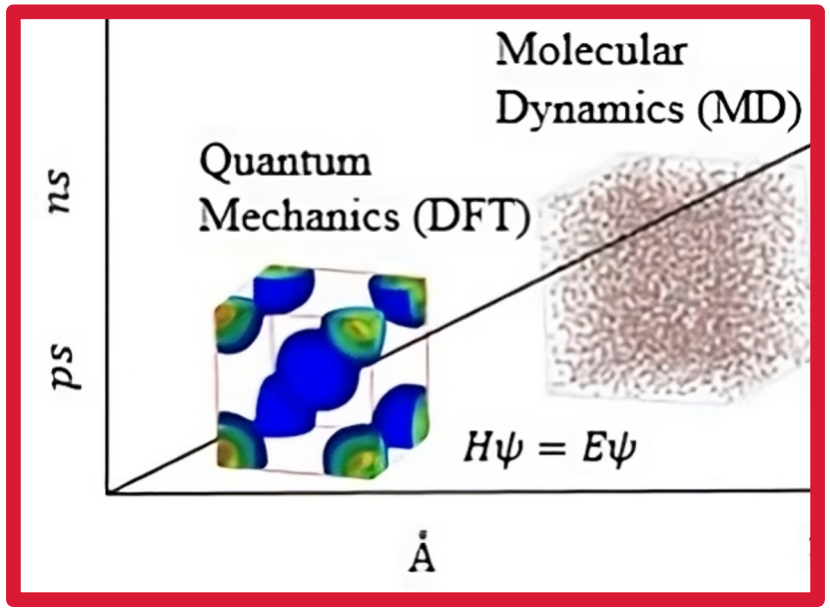
- Brief Intro to MLIPs: Context and Overview
- Anatomy of a potential: locality, E0s, forces
- Atomic Descriptors: Symmetry, Smoothness, Completeness
- Models Architectures: Linear, Kernel, MPNNs (MACE)
- Fitting and testing MLIPs

Multiscale Molecular Modelling - the Battle of Approximations



Force-field, Potential Energy, Molecular Dynamics





Molecular Dynamics is the method we use to explore the potential energy surface of the nuclei

We obtain forces as the gradient of the potential energy surface:

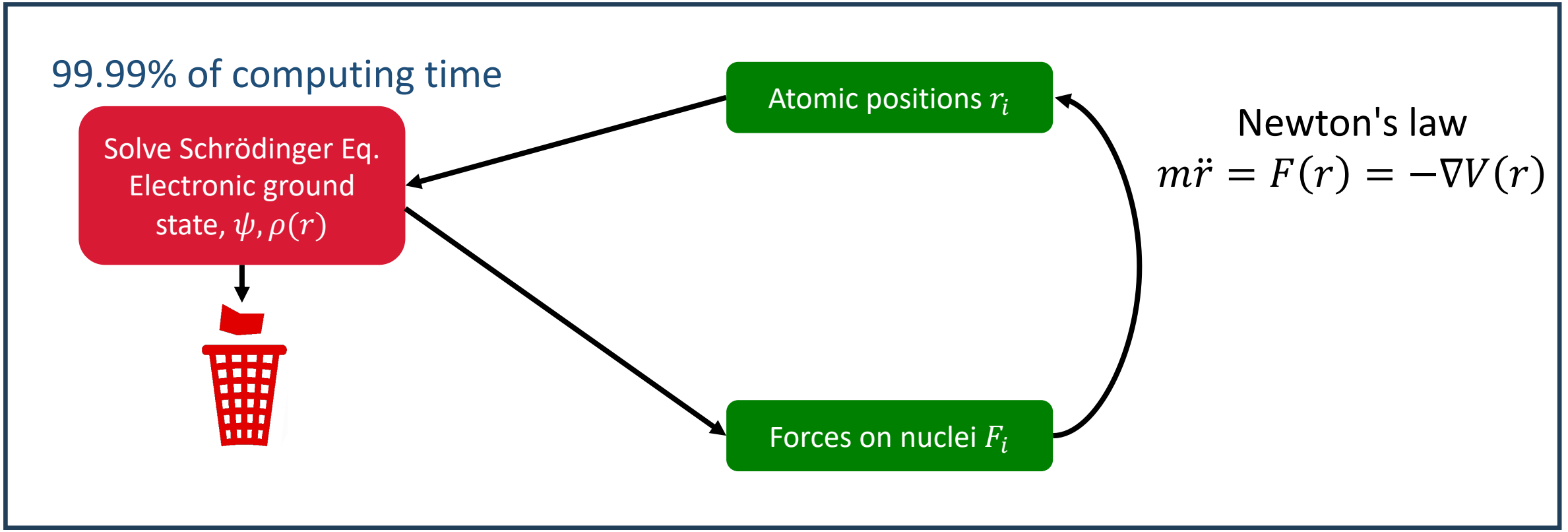
$$F = -\frac{dE(x)}{dx}$$

We integrate Newton's second low of motion to update positions

$$\mathbf{F}_i = m_i \mathbf{a}_i = m \frac{d^2 \mathbf{r}_i}{dt^2}$$

Ab Initio Molecular Dynamics

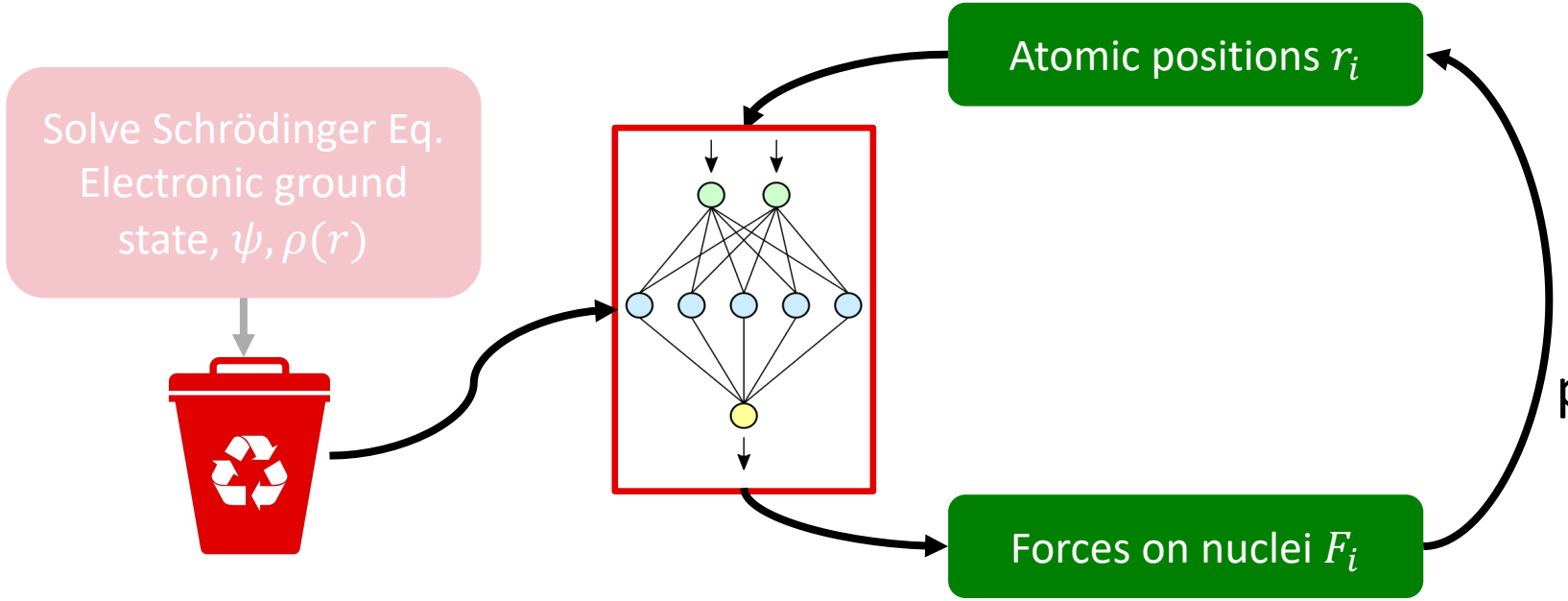
Solve very **expensive Schrodinger Equation** to obtain energies, forces
Update positions **very slightly** (1.0 femtosecond at a time)



Discard old Schrodinger solution, solve again for very similar geometry.

Machine Learning Interatomic Potentials (MLIPs)

MLIPs learn the mapping from **geometry** to **energy, forces**
Instead of using simple functions (MM), they use **universal approximators**



Newton's law

$$m\ddot{r} = F(r) = -\nabla V(r)$$

Map from atomic positions to total energy

$$E \equiv V(r): \ R^{3N} \rightarrow R$$

Can speed up the dynamics significantly! **1000-1M X**

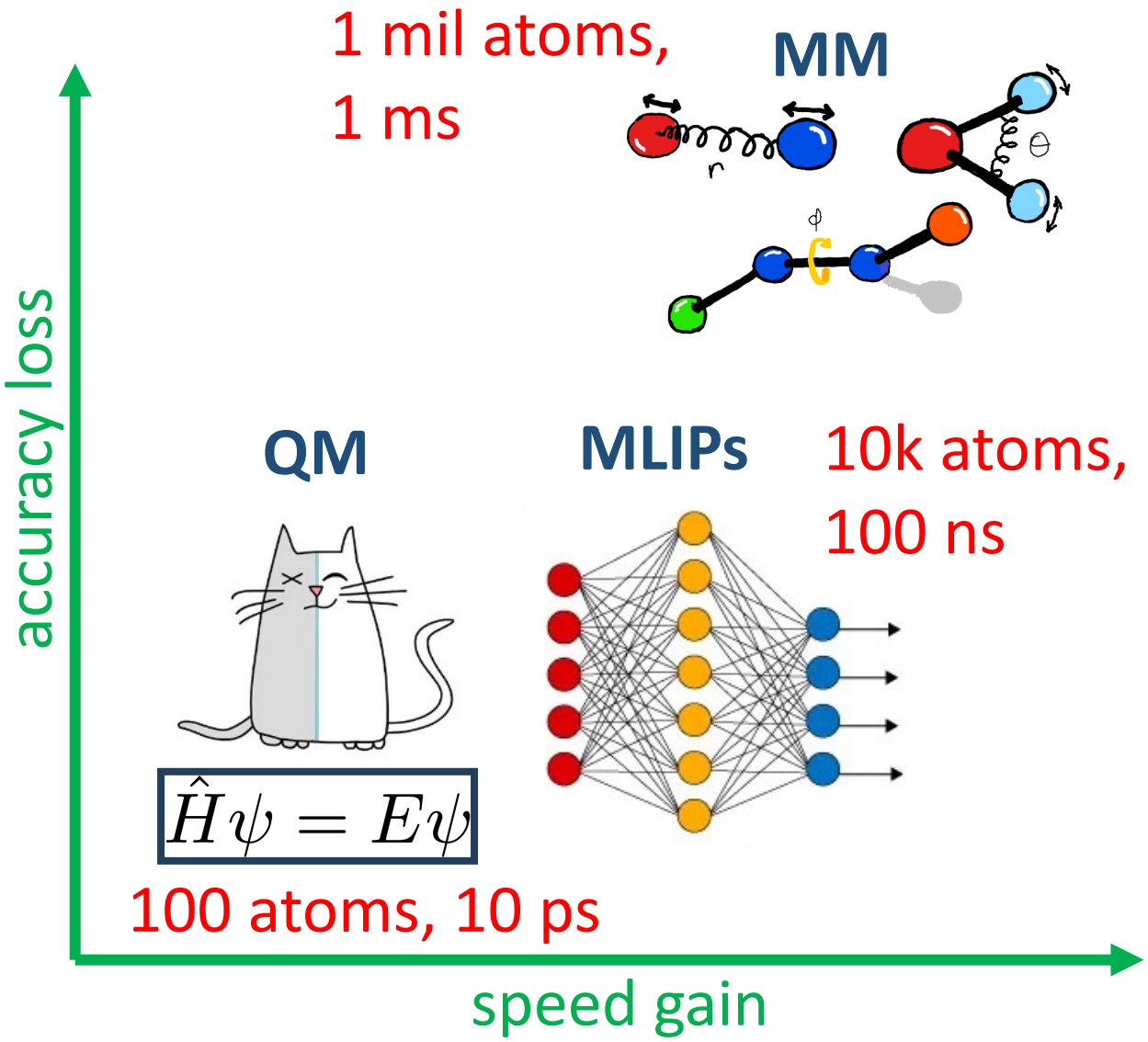
MLIPs generalize Molecular Mechanics (MM) Forcefields

The PES: $E = E(r_1, r_2, \dots)$

In QM, PES from solving the Schrodinger Equation (Born-Oppenheimer approximation)

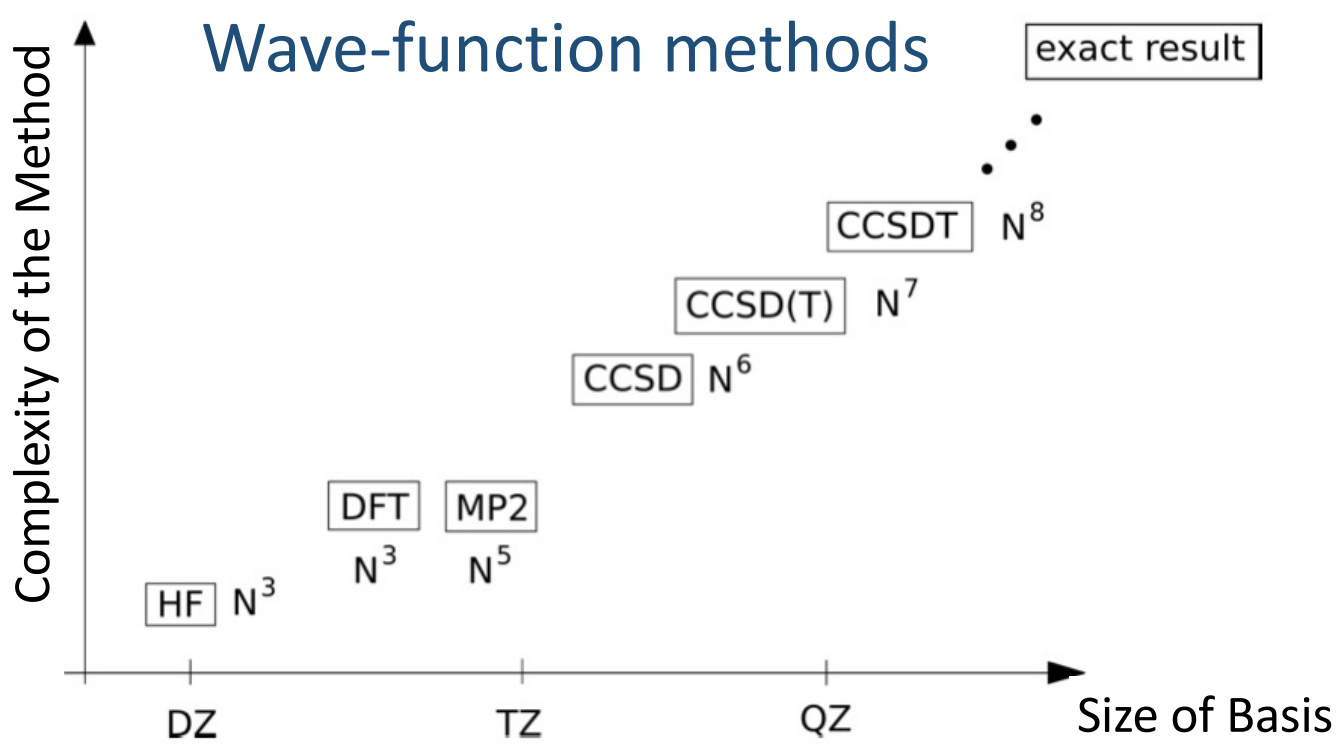
In MM, PES using empirical parameters: bond, angles, charges (**generally non-reactive**)

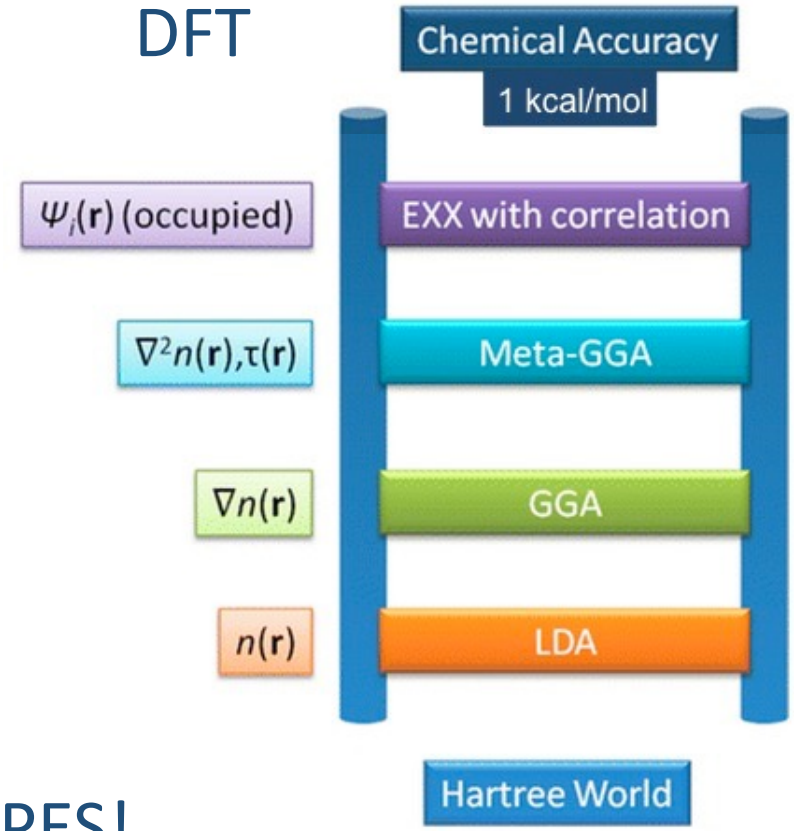
In MLIPS, PES interpolated from know geometries to new geometries (**capture chemical reactions**)



What's the speedup? It depends.

Not all QM methods are created equal:



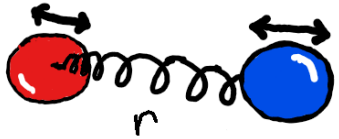


MLIP cost is constant regardless of accuracy of PES!

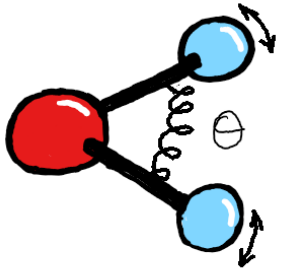
Training is more expensive of course!

Body-order in Molecular Mechanics

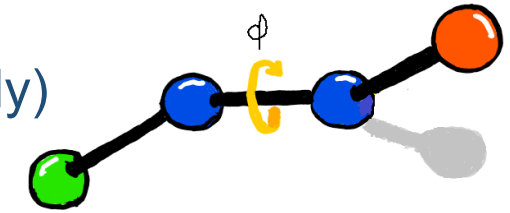
Bonds (2-body)



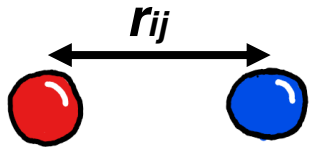
Angles (3-body)



Dihedrals (4-body)



Pairs (2-body)



$$U = \sum_{bonds} k_r(r - r_0)^2$$

$$+ \sum_{angles} k_\theta(\theta - \theta_0)^2$$

$$+ \sum_{dihedrals} k_\phi(1 + cos(n\phi + \phi_0))$$

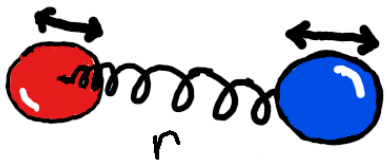
$$+ \sum_{i,j>i} 4\epsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right]$$

$$+ \sum_{i,j>i} \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}}$$

Short range
covalent bonding

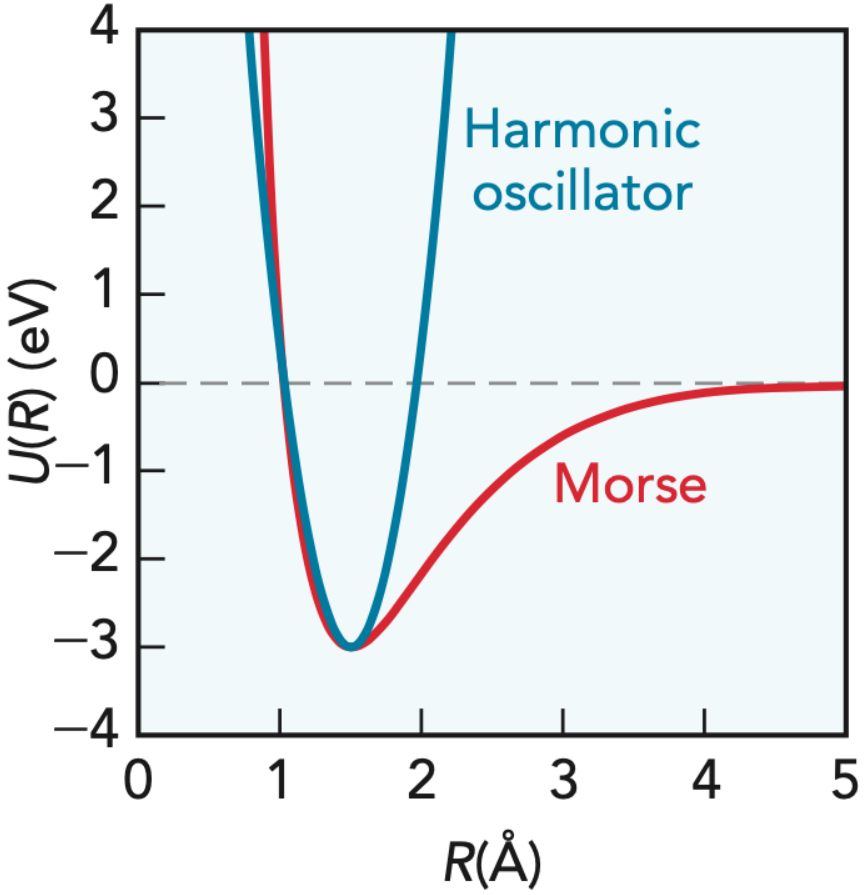
Long range
non-bonded interactions

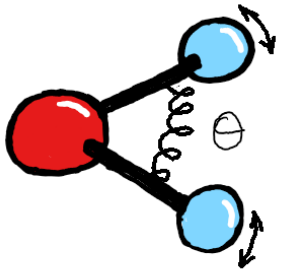
Is There Anything Missing?



$$E = k_1 r^2 + k_2 r^3 + k_3 r^4 + \cdots$$

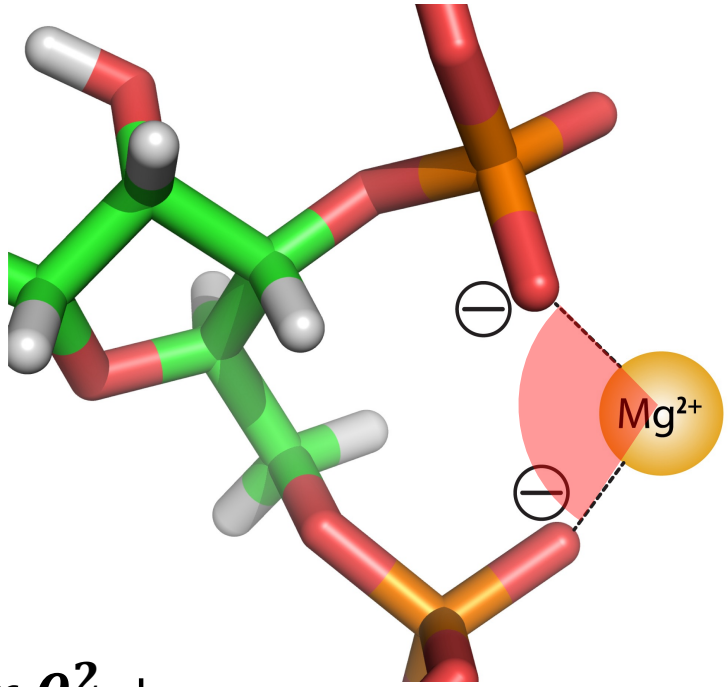
harmonic potential vs real potential





bonded angles

how about non-bonded angles?



higher-order nonlinear terms to improve the approximation

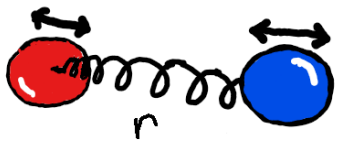
how about bond-angle interactions etc ...?

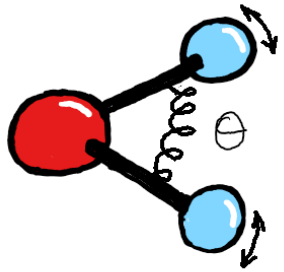
$$E = k_4 r\,\theta + k_5\, r\,\theta^2 + \cdots$$

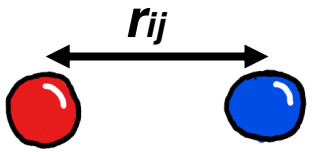
From Molecular Mechanics to Machine Learning

MM sums over bonds, angles, dihedrals ....

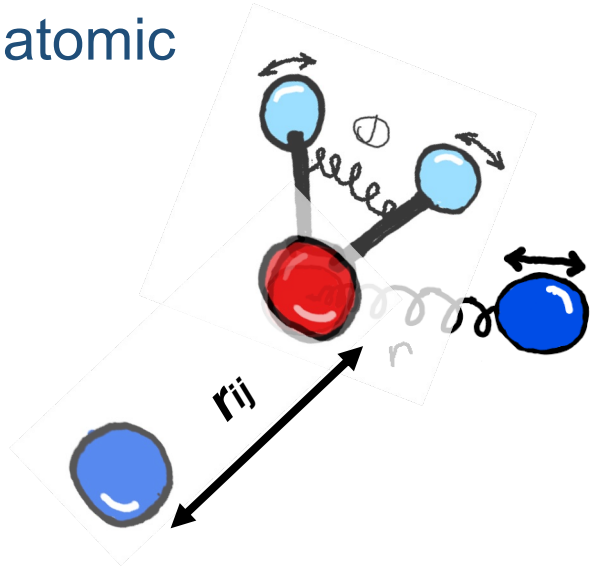
$$U(r) = \sum_{bonds} U_b + \sum_{angles} U_a + \ldots$$





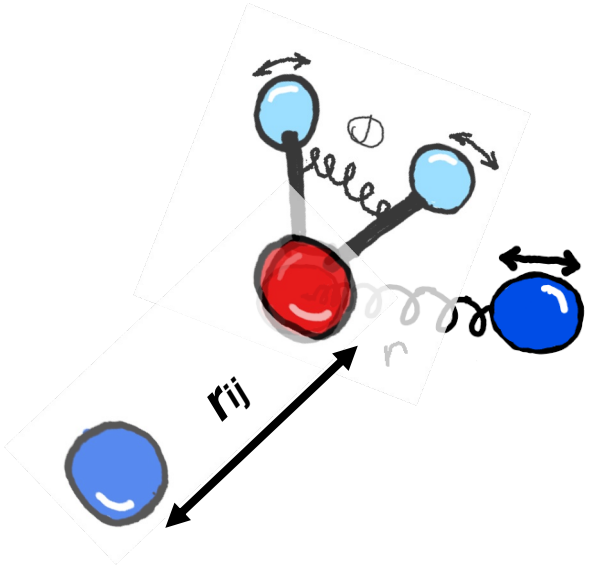


ML sums over atomic environments



$$U(r) = \sum_{atoms} U_i(bonds, angles, \ldots)$$

From Molecular Mechanics to Machine Learning



Let's take a bond term:

$$U_b = k(r-R)^2 = wk(r^2 - 2rR + R^2) + (1-w)k(r^2 - 2rR + R^2)$$

w- arbitrary decomposition, learned from data

One of the atoms:

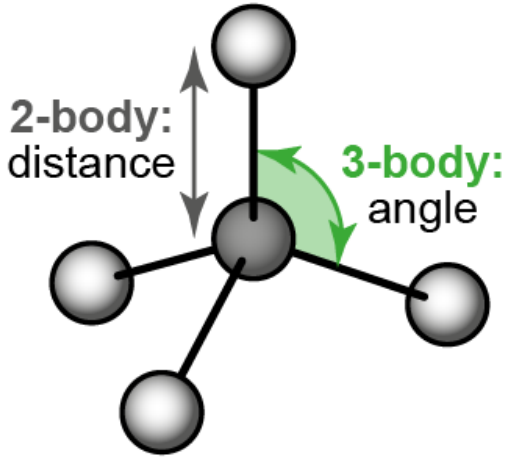
$$U_1(r) = wk(r^2 - 2rR + R^2)$$

Imagine a Radial Basis Set $\theta_0(r), \theta_1(r), \theta_2(r)$:

$$U_1(r) = wk\theta_2(r) - 2Rwk\theta_1(r) + R^2wk\theta_0(r)$$

MLIPs generalize MM Force Fields

a Conventional *N*-body terms



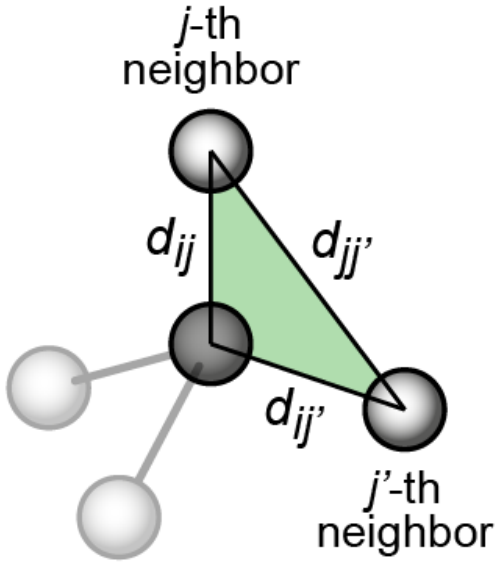
Bonds (2-body)

Angles (3-body)

Dihedrals / Impropers (4-body)

enumerated from the outset

b General 3-body descriptor



all Pairs within cutoff (2-body)

all Triangles (3-body)

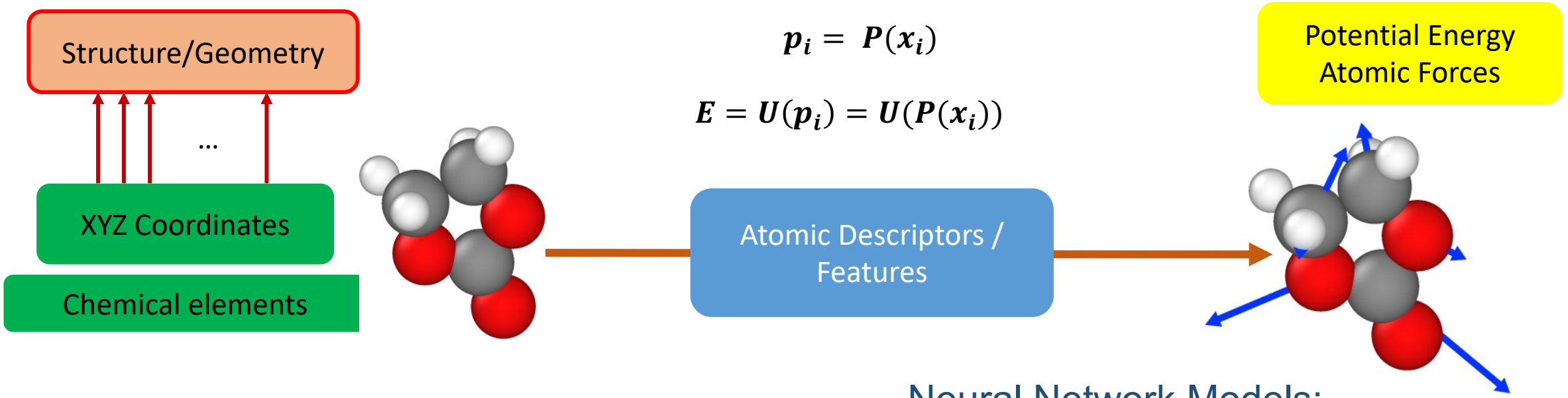
all Tetrahedra (4-body) ...

determined on the fly: inferred from data

Maximum Body-order of MLIP:

$$\frac{\partial^N P}{\partial x_{i_1}, \ldots, \partial x_{i_N}} \neq 0$$

General Anatomy of the MLIP



Linear Models:

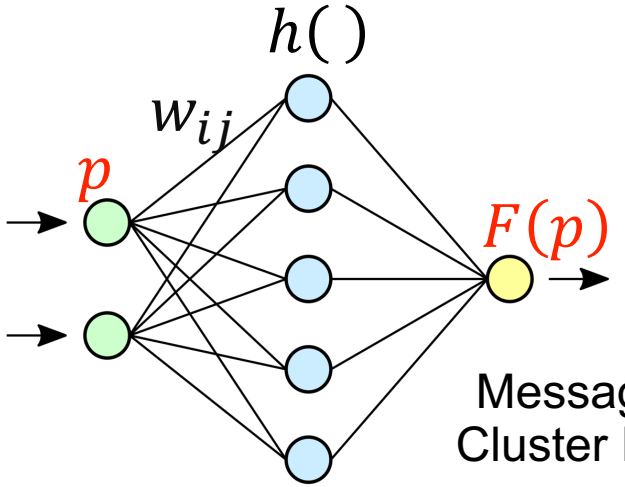
$$F(p) = \sum_i w_i \, p^i$$

Kernel Models:

$$F(p) = \sum_i w_i \, K(p, p_i)$$

Gaussian Approximation Potentials (GAP)

Neural Network Models:



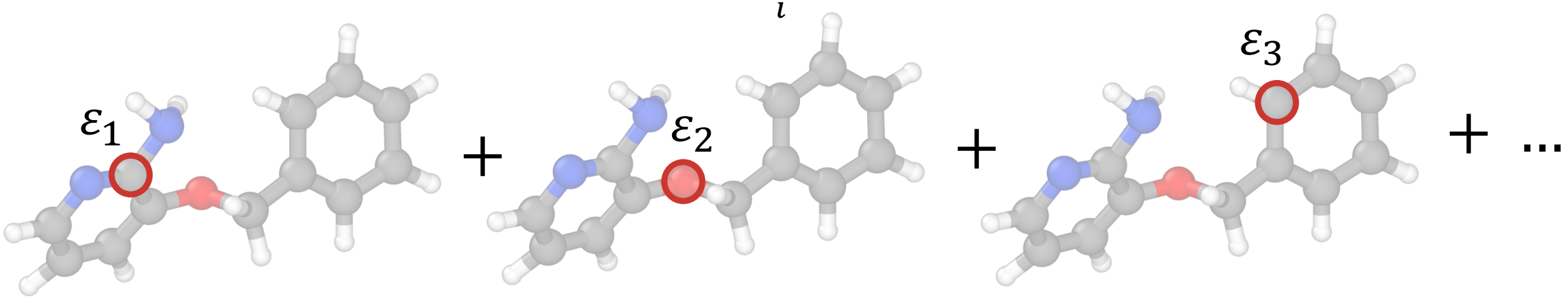
Message-passing Atomic Cluster Expansion (MACE)

Central Concepts in MLIPs: Locality of Quantum Mechanics

Sort-range ML models rely on energy decomposition and locality:

- total energy can be written as a sum of atomic energies:

$$E_{tot} = \sum_i \varepsilon_i$$



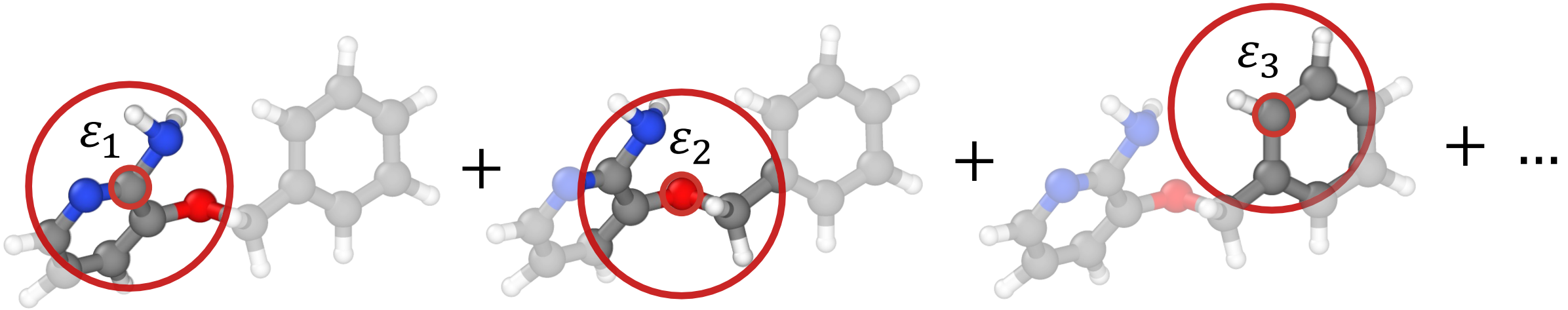
Central Concepts in MLIPs: Locality of Quantum Mechanics

Sort-range ML models rely on energy decomposition and locality:

- total energy can be written as a sum of atomic energies:

$$E_{tot} = \sum_i \varepsilon_i$$

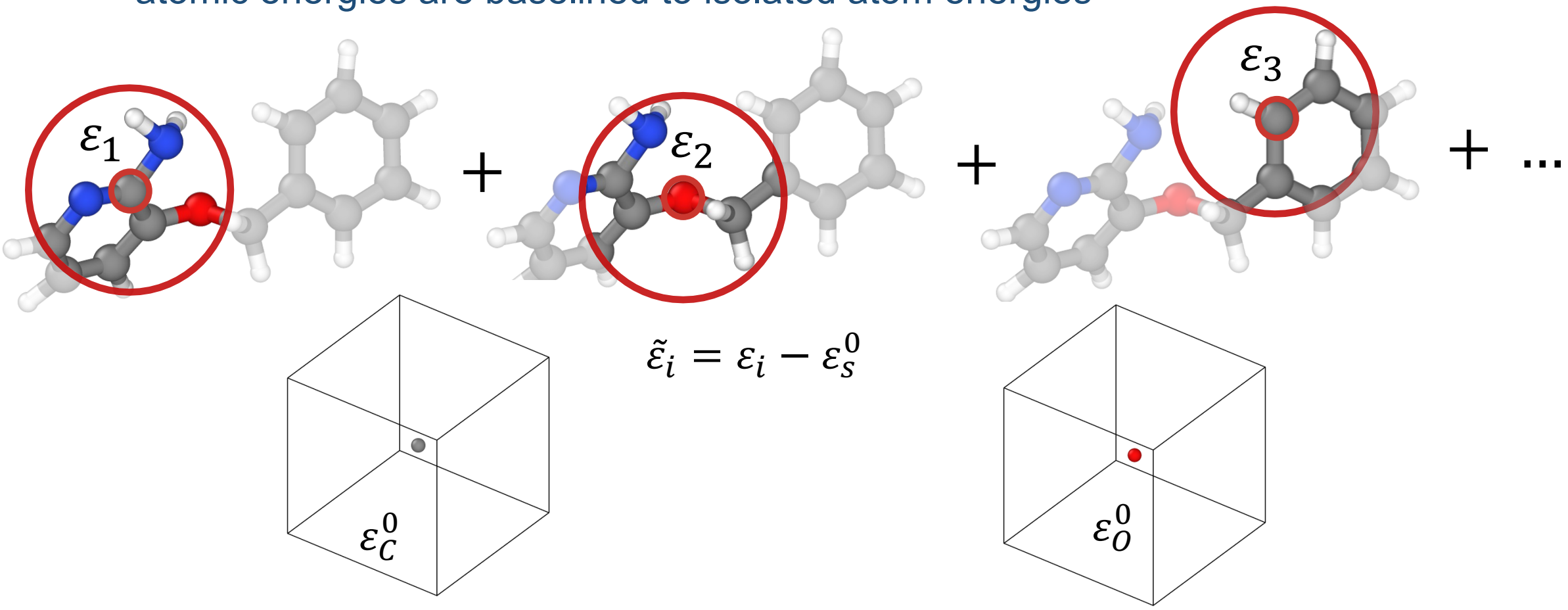
- atomic energies are a function of local environment (within a cutoff)



$$\varepsilon_i = \varepsilon_i(r < R_{cut})$$

Central Concepts in MLIPs: Isolated energies

- atomic energies are baselined to isolated atom energies



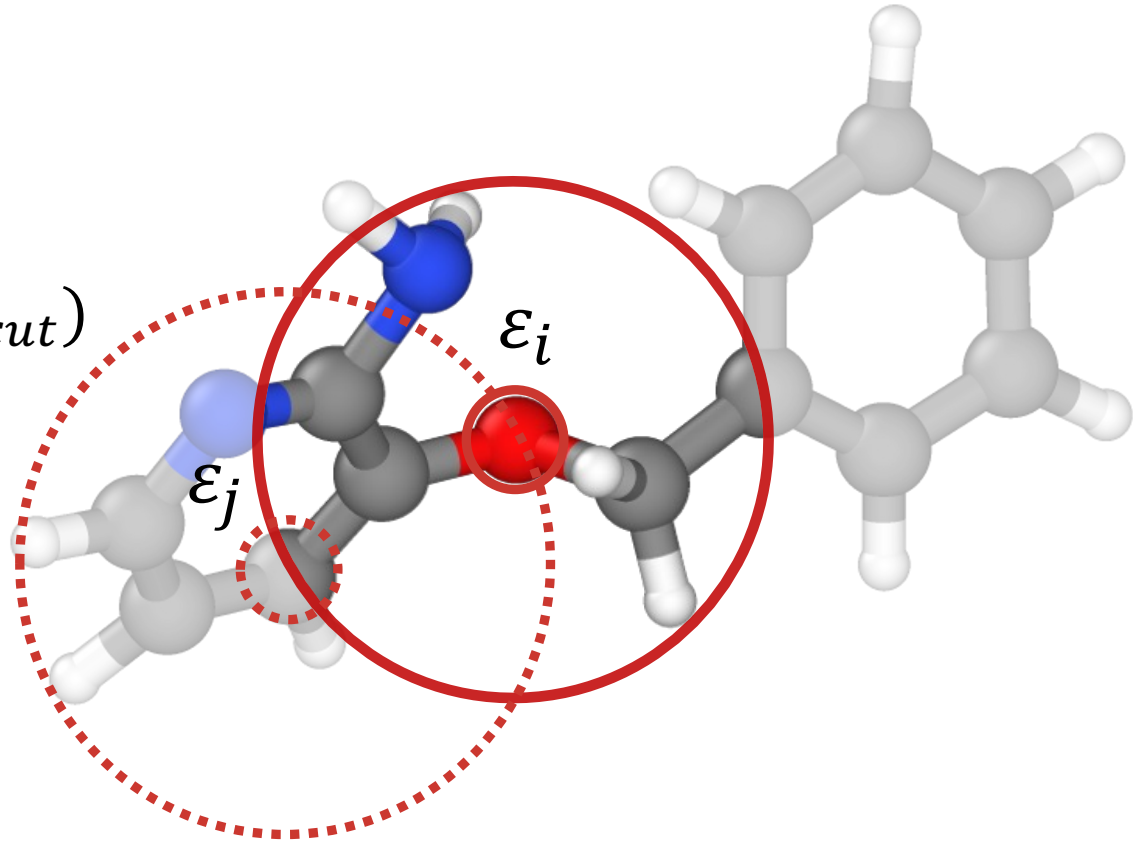
This ensures transferability across chemical compounds and a physically meaningful baseline in the absence of data for chemical reactions.

Central Concept in MLIPs: Calculating forces

- forces are derivatives of **total** energy:

$$F_i = \frac{\partial E_{tot}}{\partial r_i} = \sum_i \frac{\partial \varepsilon_i}{\partial r_i} + \sum_{j \neq i} \frac{\partial \varepsilon_j}{\partial r_i}$$

$$F_i = \sum_i \frac{\partial \varepsilon_i}{\partial r_i} (r < R_{cut}) + \sum_{j \neq i} \frac{\partial \varepsilon_j}{\partial r_i} (r < 2R_{cut})$$

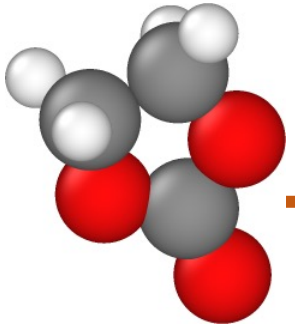


This ensures:

- energy conservation
- force equivariance
- extends effective interaction

Representation and Atomic Descriptors

Use Atomic Descriptors to represent Structure/Geometry



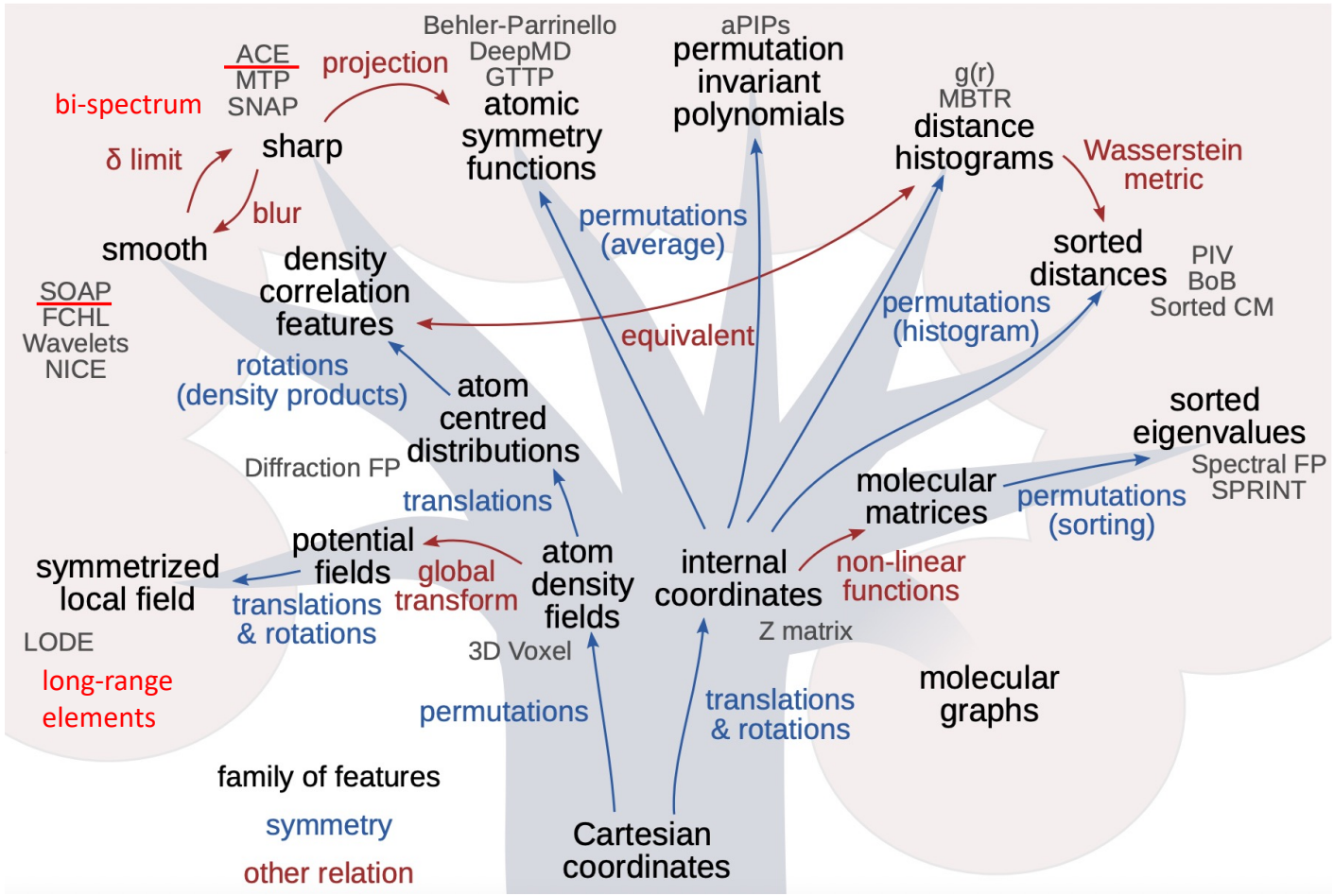
$$\boldsymbol{p_i} = \boldsymbol{P}(\boldsymbol{x_i})$$

Atomic Descriptors

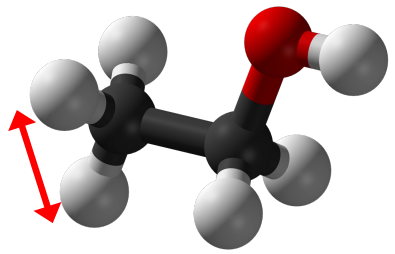
Properties

- symmetry: translational, rotational, permutational
- smoothness
- completeness
- correlation order (2, 3, 4-body)

Large and growing family tree of atomic descriptors!



Descriptor Symmetries: Permutation, Rotation, Translation

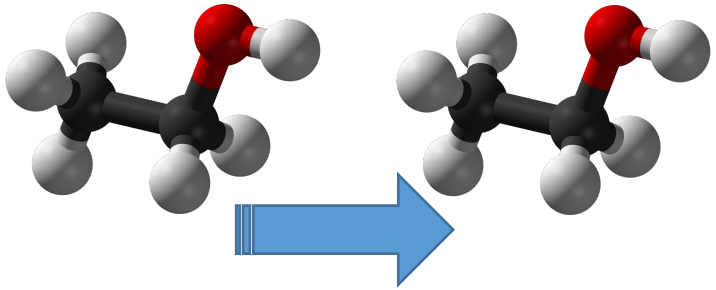


PERMUTATION

Permutation is a <u>discrete symmetry</u> of the energy!

Descriptors should be **Invariant** to **Permutations:**

$$P(x_i) = P(x_{\sigma(i)})$$



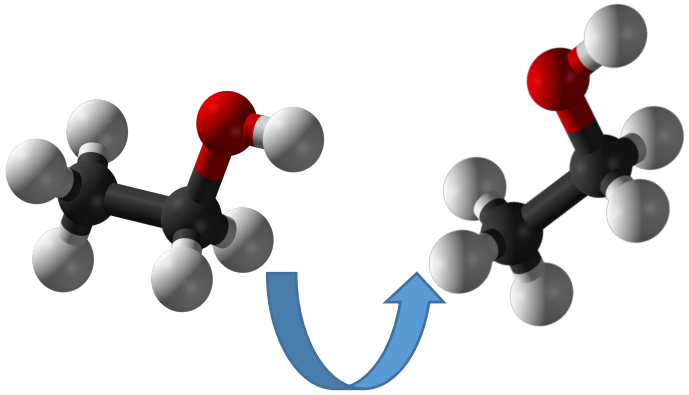
TRANSLATION

Translation is a <u>continuous symmetry</u> of the energy!

Infinitely many environments map to the same energy!

Difficult to `learn`, our descriptors require **Translational Symmetry**:

$$P(x_i) = P(x_i + T)$$



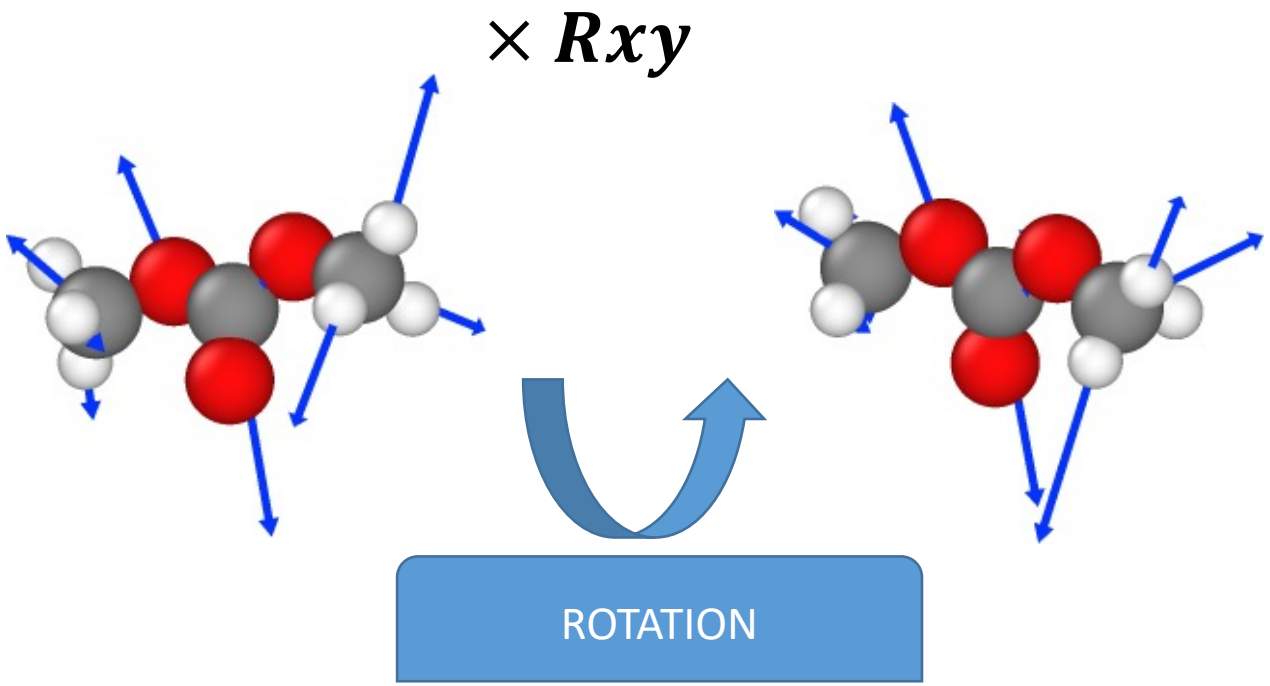
ROTATION

Rotation is also a <u>continuous symmetry</u> of the energy!

Descriptors should be **Invariant** to **Rotations:**

$$P(x_i) = P(x_i \times R)$$

Rotational Equivariance



What about properties that do change with rotations?

E.g. **Forces, Dipole vectors rotate with the coordinate frame!**

We want to avoid learning the transformation, instead we can make atomic descriptors **Equivariant with Rotations**:

$$\boldsymbol{P(x_i \times R) = P(x_i) \times R}$$

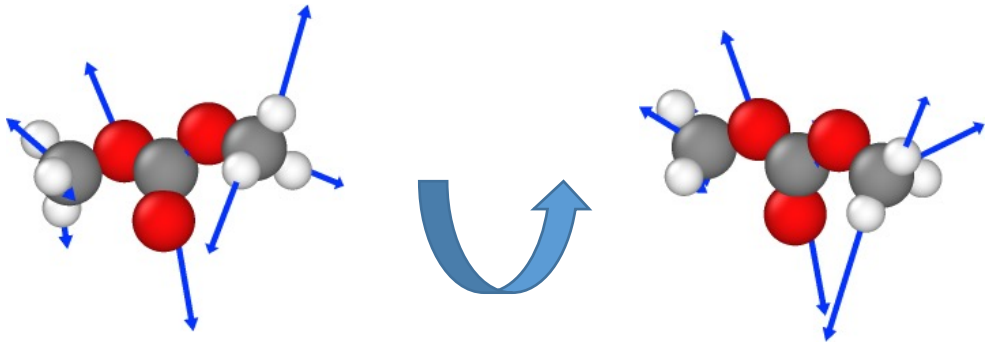
Question: can we fit equivariant forces with an invariant MLIP?

Let's define a rotated coordinate system (i.e. Rotate molecule in Ovito):

$$x' = Rx$$

Energy is rotationally invariant:

$$E(x) = E(x')$$



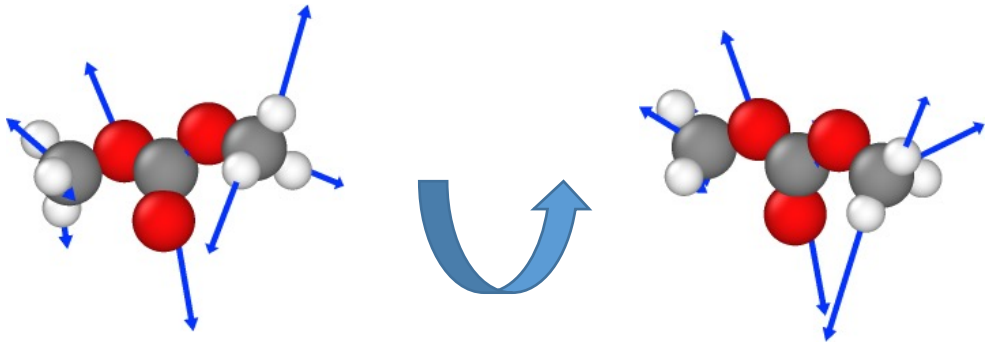
Question: can we fit equivariant forces with an invariant MLIP?

Let's define a rotated coordinate system (i.e. Rotate molecule in Ovito):

$$x' = Rx$$

Energy is rotationally invariant:

$$E(x) = E(x')$$



Forces in the original and rotated frame:

The relation between the forces:

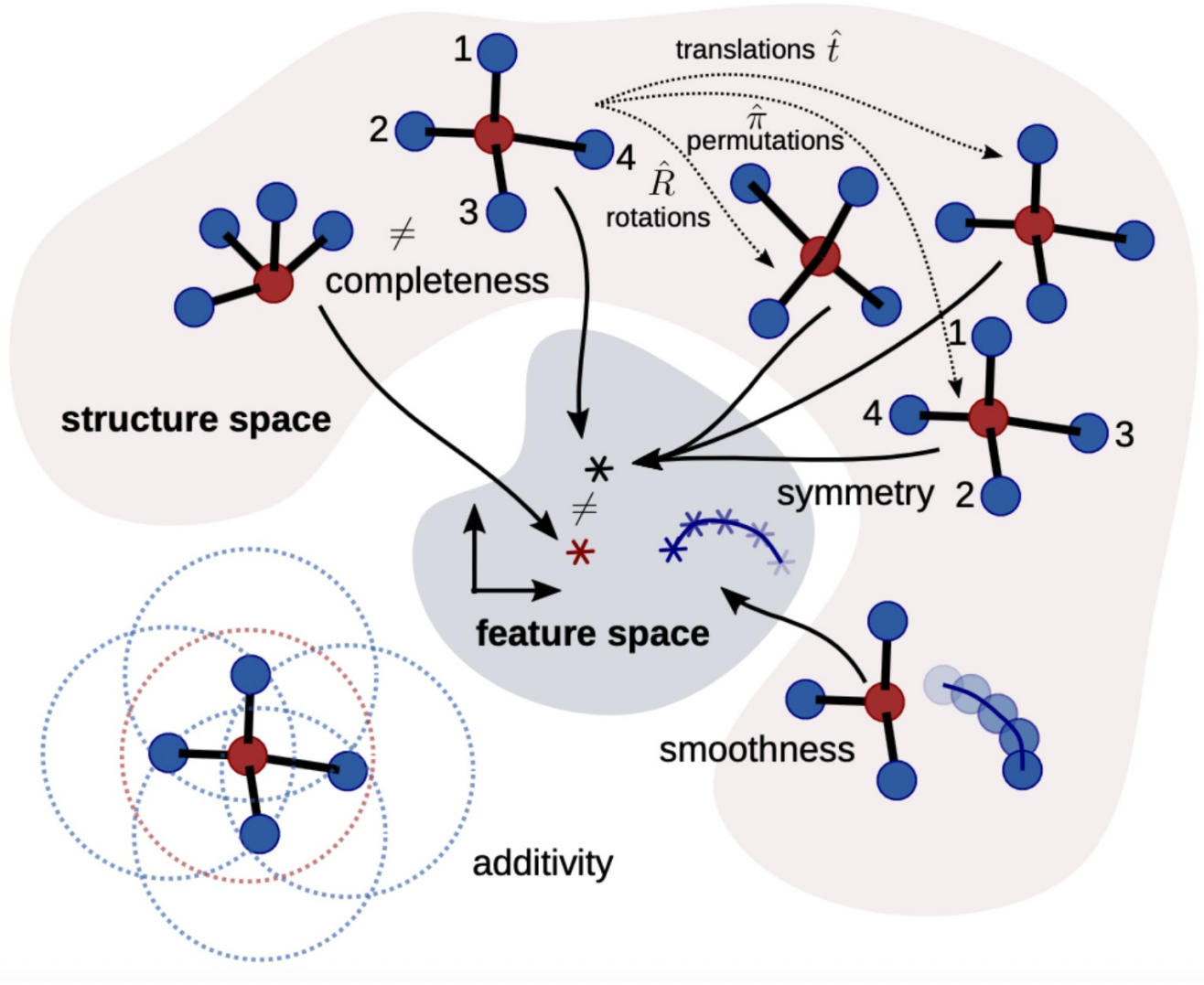
$$F(x) = -\frac{\partial E(x)}{\partial x} \qquad F(x') = -\frac{\partial E(x')}{\partial x'}$$

$$\boxed{\frac{\partial E(x)}{\partial x} = \frac{\partial E(x)}{\partial x'}\frac{\partial x'}{\partial x} = \frac{\partial E(x')}{\partial x'}R}$$

$$F(x) = RF(x')$$

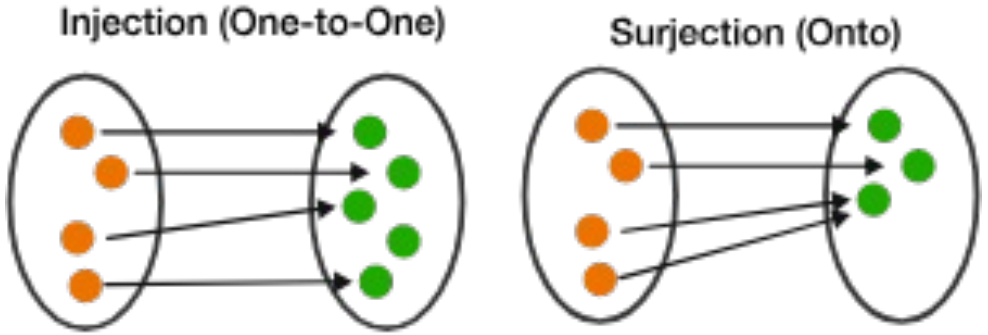
YES! If we obtain forces as the **gradient of an invariant energy**, we ensure they are equivariant! This will work for all vectors that can be expressed as gradients of a scalar field.

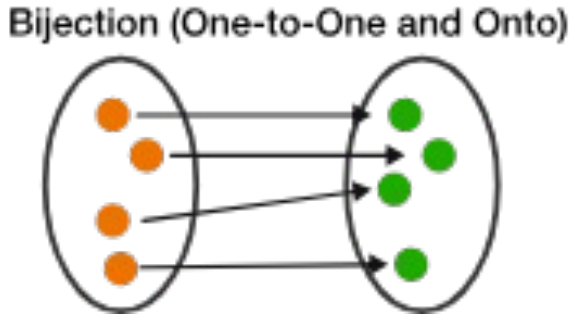
Other Properties of a 'Good' Descriptor: Completeness



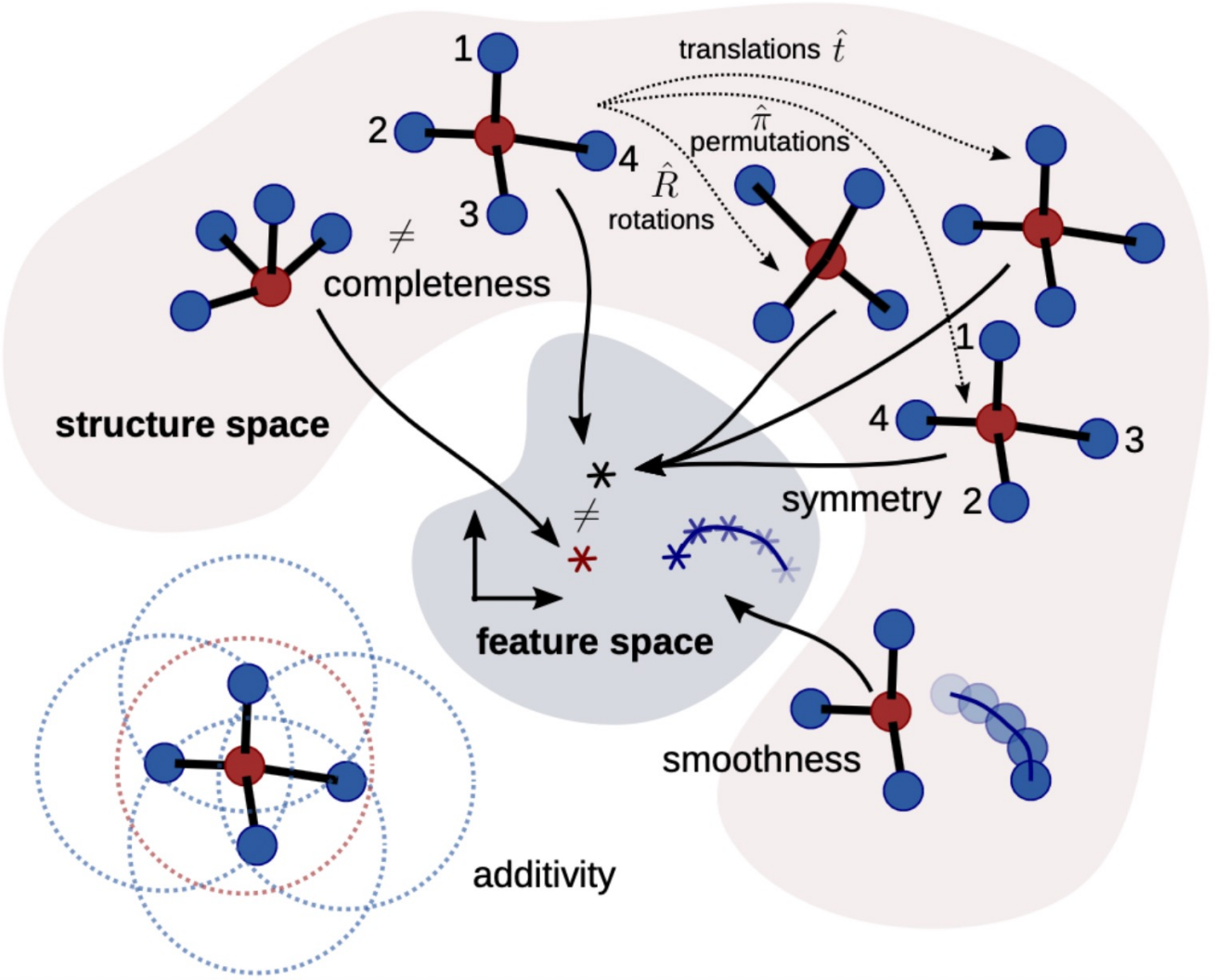
Descriptor Space ≫ Geometry Space

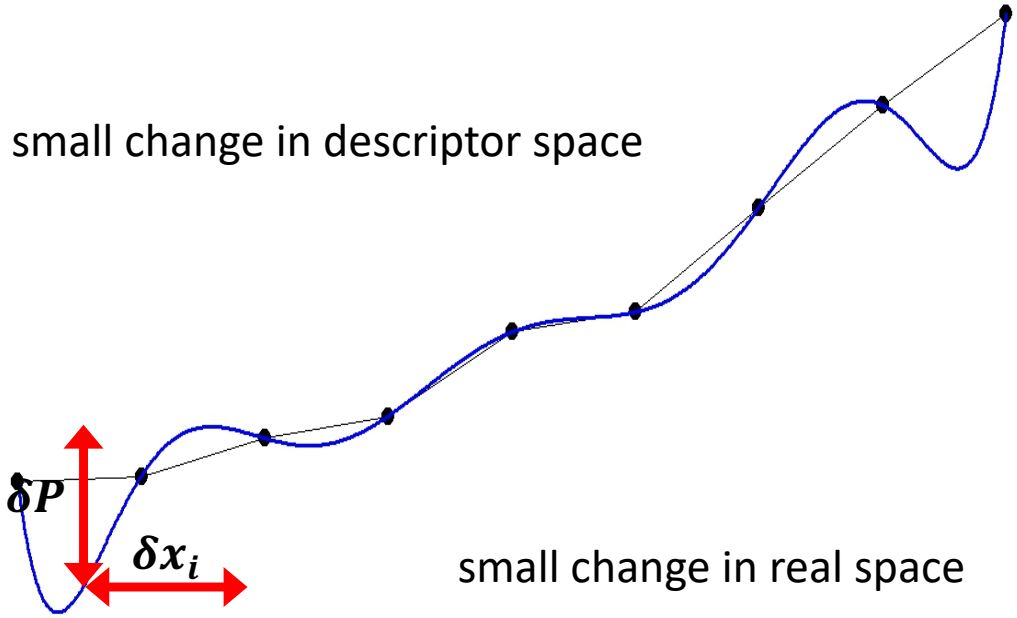
- **many** P **do not** map onto real x_i
- **all** x_i have to map onto a P
- **different** x_i must map onto different P





Other Properties of a 'Good' Descriptor: Smoothness





descriptors need to change smoothly with geometry:

energy is smooth!

so should their arbitrary derivatives:

forces, Hessians are smooth!

Examples: SOAP

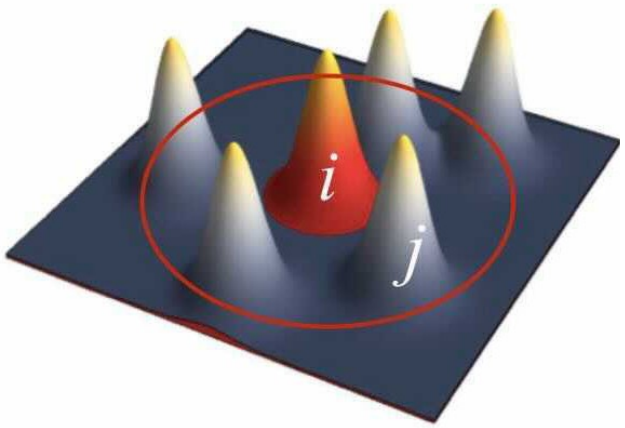
Atomic density **centered** on atom i:

$$\rho^i(\boldsymbol{r}) = \sum_j \exp\left(\frac{-|\boldsymbol{r} - \boldsymbol{r}_{ij}|^2}{2\sigma_{atom}^2}\right) f_{cut}(|r_{ij}| < R_{cut})$$

- **permutational invariance**
- **translational invariance**

Expand into **radial / angular** basis:

$$\rho^{i\alpha}(\boldsymbol{r}) = \sum_{nlm} c_{nlm}^{i\alpha} R_n(r) Y_{lm}(\hat{r})$$



Power spectrum (the 'feature' vector):

$$P = p_{nn'l}^i = \frac{1}{\sqrt{2l+1}} \sum_m c_{nlm}^i \, c_{n'lm}^i$$

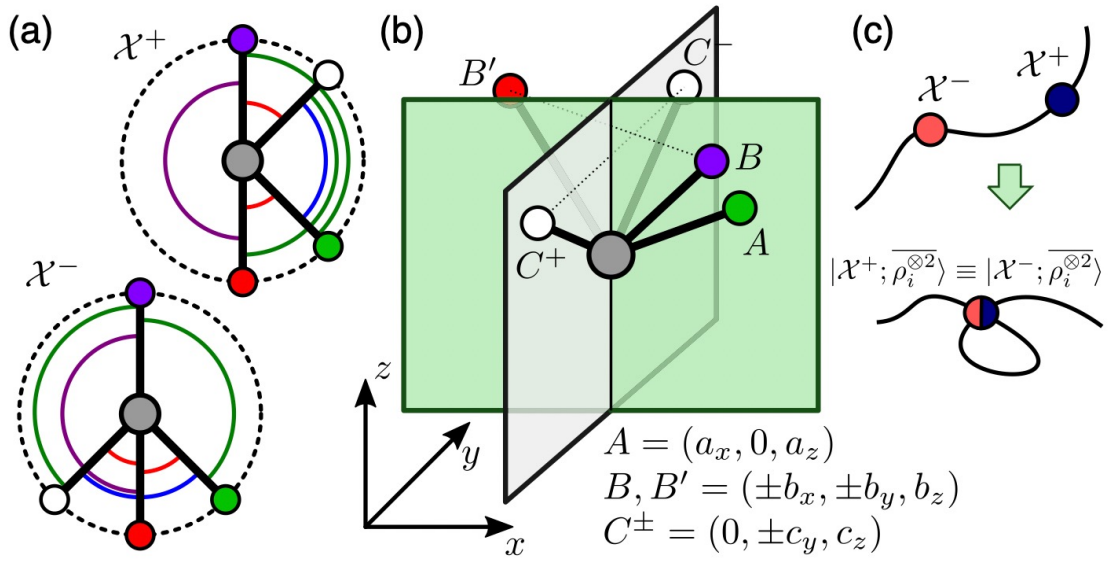
- **rotational invariance**

SOAP is a 3-body Descriptor

3-body Degeneracy:

Two different environments (4 neighbors): same SOAP

SOAP is incomplete!



SN Pozdnyakov, MJ Willatt, AP Bartók, C Ortner, G Csányi, and M Ceriotti PRL **125**, 166001, 2020

body order ↔ minimum neighborhood degeneracy!

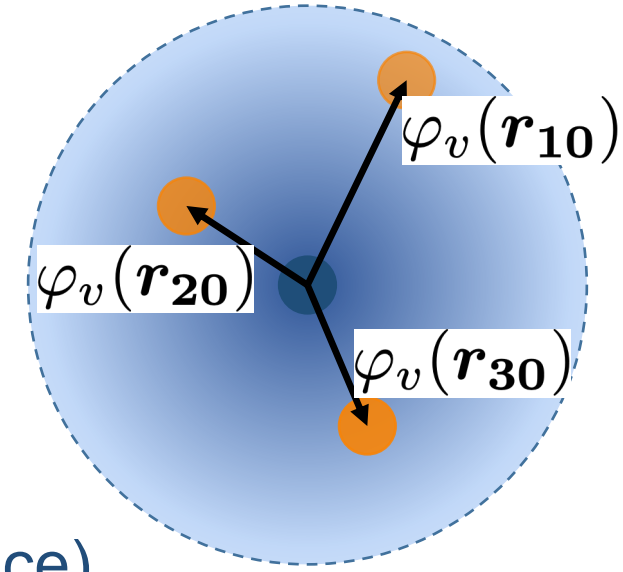
4-body: minimum 7 neighbors proven!

ACE: arbitrary body-order expansion!

Example: ACE - Body-ordered Descriptors

Atomic Cluster Expansion (ACE) as a systematic approach:

One–particle basis, atom-centred (translational invariance)



Radial Angular

$$\varphi_{nlm,z_i z_j}(\boldsymbol{r}) = R_{nl,z_i z_j}(r_{ji}) Y_l^m(\hat{\boldsymbol{r}})$$

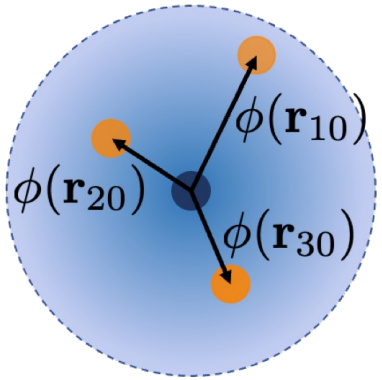
Evaluate for each neighbour and sum (permutation invariance)

$$A_{iv} = \sum_{j \in \mathcal{N}(i)} \varphi_v(\sigma_j, \sigma_i) \quad \text{(2-body)}$$

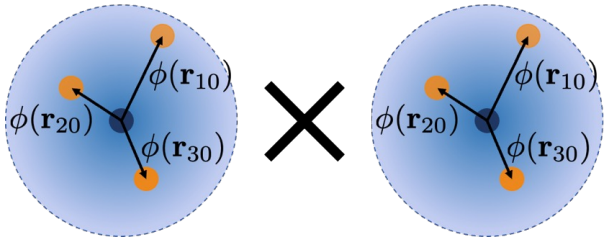
Phys. Rev. B **100**, 249901 (2019)

ACE: Systematic Body-ordered Expansion

Construct the product basis ($\nu + 1$ body terms):



$$\mathbf{A}_{i,\mathbf{v}} = \prod_{\xi=1}^{\nu} A_{i,v_\xi} \qquad (\nu + 1 \text{ body})$$



Symmetrize the product atomic-basis using the Clebsch–Gordan coefficients (rotational invariance)

$$\boldsymbol{B}_{i,\boldsymbol{v}} = \sum_{\boldsymbol{w}} \mathcal{C}_{\boldsymbol{v}\boldsymbol{w}} \boldsymbol{A}_{i,\boldsymbol{v}}$$

Phys. Rev. B **100**, 249901 (2019)

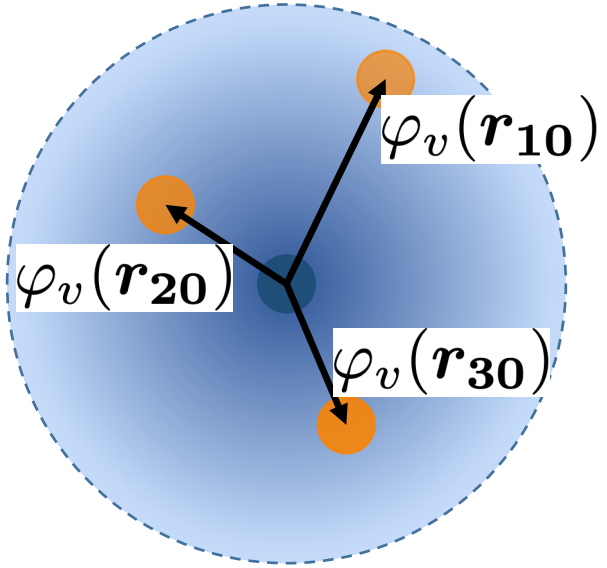
Linear ACE

ACE inspired from cluster-expansions of the energy function:

$$E_i = E_i^{(0)} + \sum_v \tilde{c}_v^{(1)} A_{iv} + \sum_{v_1 v_2}^{v_1 \geq v_2} \tilde{c}_{v_1 v_2}^{(2)} A_{iv_1} A_{iv_2} + \sum_{v_1 v_2 v_3}^{v_1 \geq v_2 \geq v_3} \tilde{c}_{v_1 v_2 v_3}^{(3)} A_{iv_1} A_{iv_2} A_{iv_3} + \ldots$$

The problem becomes linear:

$$E = \sum_{i,\boldsymbol{v}} c_{i,\boldsymbol{v}} \boldsymbol{B}_{i,\boldsymbol{v}}$$



... and we know how to solve it:

$$L = \sum_j \| E(X_j) - \mathbf{B}(X_j)^T \mathbf{w} \|^2$$

ACE is a precursor to MACE

Dealing with chemical species: embedding

From categorical space to continuous space

One-Hot Encoding

- categorical data as binary vector
- each word is represented by a vector where:
  - One position is "1" (indicating the word).
  - All others are "0."

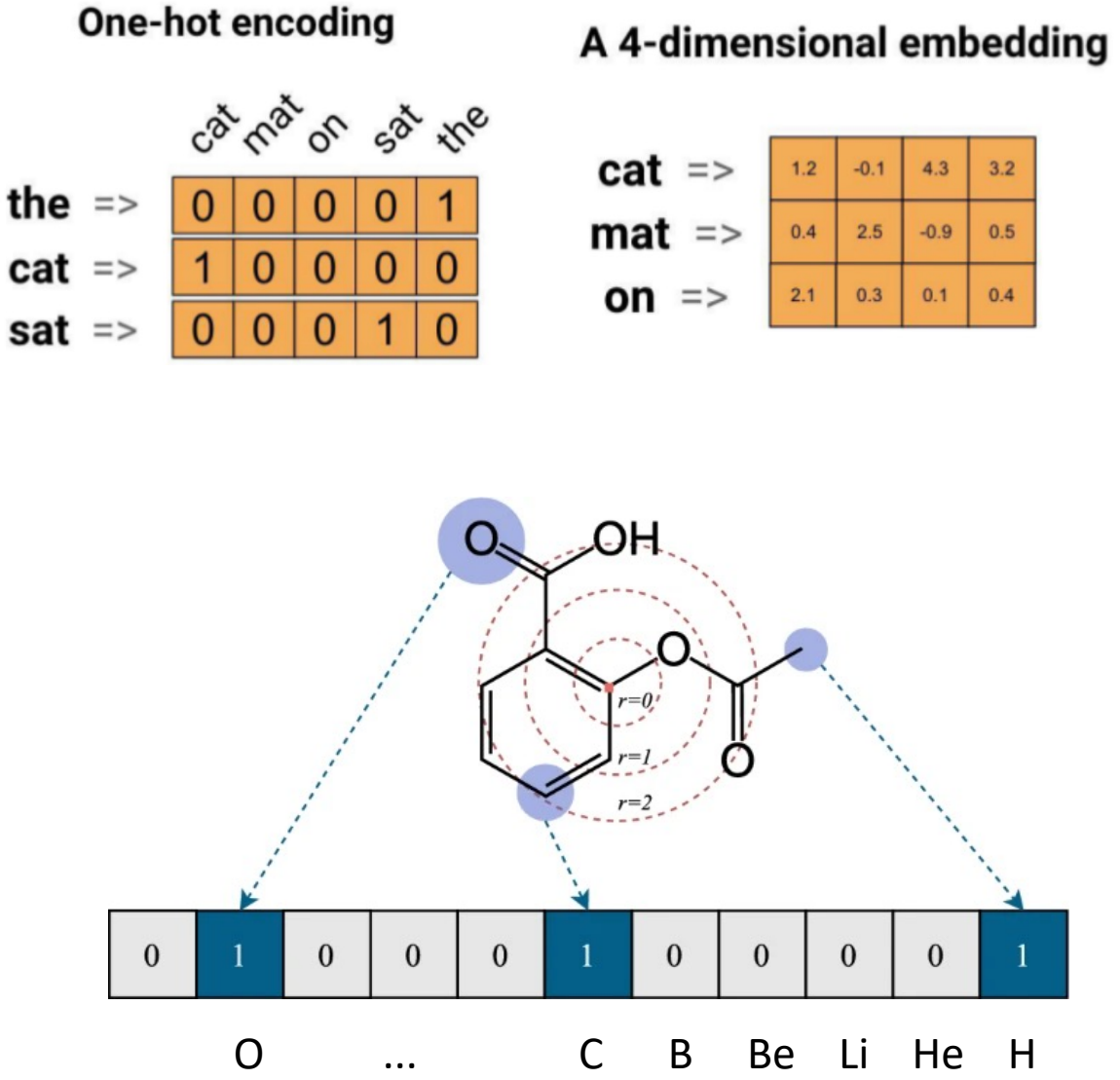
  "the" → [0, 0, 0, 0, 1]

  "cat" → [1, 0, 0, 0, 0]
- high dimensionality: one vector per word
- no notion of similarity between "cat" and "dog."

4-Dimensional Embedding

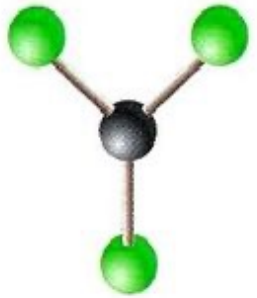
- dense, low-dimensional representations of data
- each word (e.g., "cat," "mat," "on") is represented by a 4-dimensional vector with learned values

MACE starts with **one-hot encoding** and learns an **embedding**

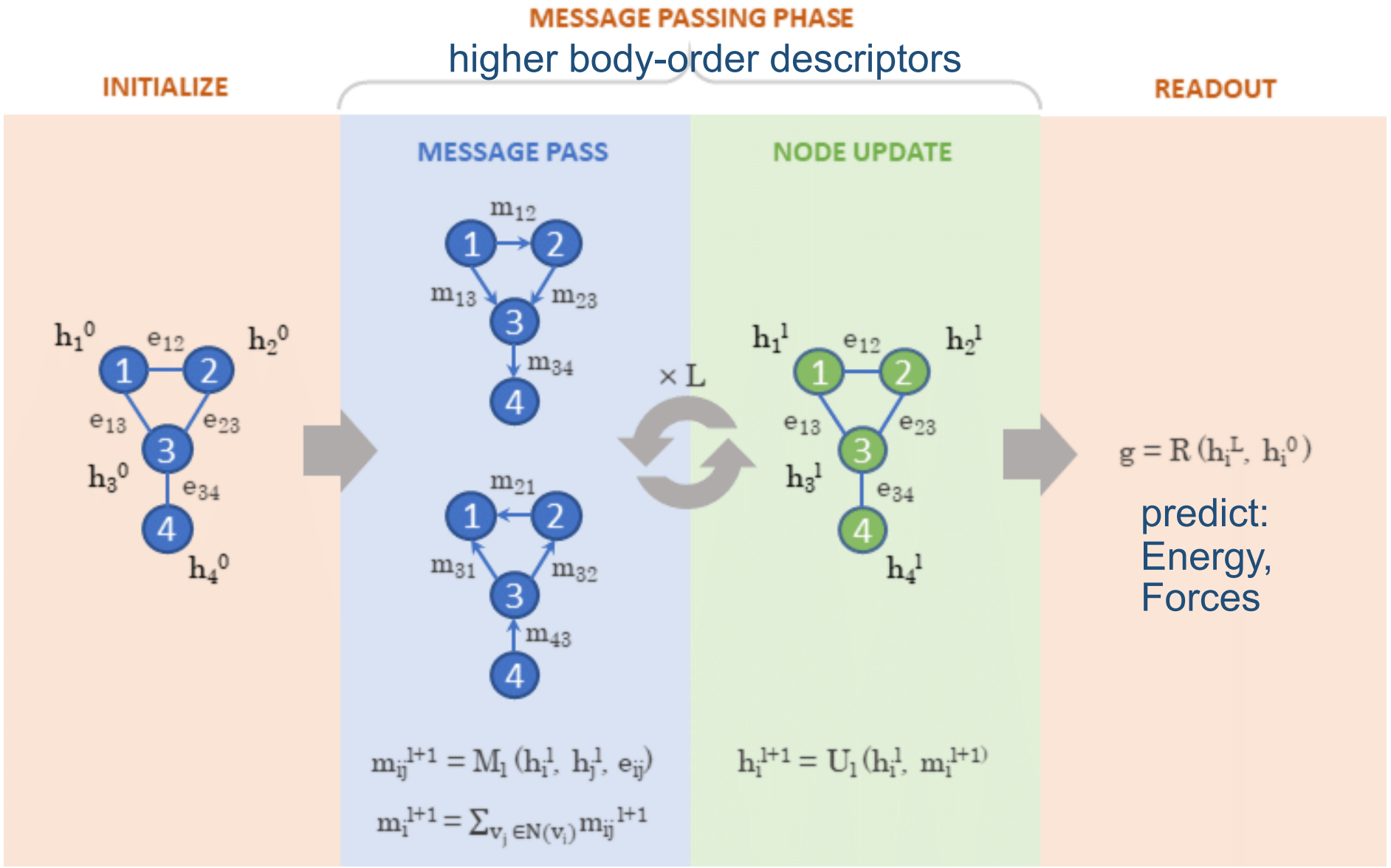


MPNN+ACE: MACE

Molecules are graphs



- each node is an atom
- messages are interactions between neighbor atoms
- **MACE** uses **ACE** basis as initial features



‘Embedding’
How the model sees the data

‘Interaction’
Pooling information from neighbours

‘Product’
Form powers of features, for higher body order

Atom’s chemical species

graph ‘edge’

Species one-hot

Node embedding

$$h^{(0)}_{i,k00} = \sum_z W_{kz}\delta_{zzi}$$

radial embedding

$$j^n_0(r_{ij}) = \sqrt{\frac{2}{r_{\text{cut}}}}\frac{\sin\left(\pi n \frac{r_{ij}}{r_{\text{cut}}}\right)}{r_{ij}} f_{\text{cut}}(r_{ij})$$

Ylm’s (angular embedding)

$$Y^{m_1}_{l_1}(\hat{r}_{ij})$$

Node features 0 (n_node, k)

edge features (n_edge, 8)

edge attributes (n_edge, (ell+1)^2)

linear

$$\bar{h}^{(s)}_{i,kl_2m_2} = \sum_{\tilde{k}} W^{(s)}_{k\tilde{k}l_2} h^{(s)}_{i,\tilde{k}l_2m_2}$$

radial MLP

$$R^{(s)}_{k\eta_1l_1l_2l_3}(r_{ij}) = \text{MLP}\left(\{j^n_0(r_{ij})\}_n\right)$$

One-particle basis ‘conv_tp’

$$\phi^{(s)}_{ij,k\eta_1l_3m_3} = \sum_{l_1l_2m_1m_2} c^{\eta_1m_1l_1m_2}_{l_3m_3} R^{(s)}_{k\eta_1l_1l_2l_3}(r_{ij}) \times Y^{m_1}_{l_1}(\hat{r}_{ij}) \tilde{h}^{(s)}_{j,k_2l_2m_2}$$

Neighbour sum + linear

$$A^{(s)}_{i,kl_3m_3} = \sum_{\tilde{k},\eta_1} W^{(s)}_{k\tilde{k}\eta_1l_3} \sum_{j\in\mathcal{N}(i)} \phi^{(s)}_{ij,\tilde{k}\eta_1l_3m_3}.$$

Product

$$B^{(t)}_{i,\eta_\nu kLM} = \sum_{lm} \mathcal{C}^{LM}_{\eta_\nu,lm} \prod_{\xi=1}^{\nu} A^{(t)}_{i,kl_\xi m_\xi}, \qquad m^{(t)}_{i,kLM} = \sum_\nu \sum_{\eta_\nu} W^{(t)}_{z_ikL,\eta_\nu} B^{(t)}_{i,\eta_\nu kLM}$$

update

$$h^{(t+1)}_{i,kLM} = \sum_{\tilde{k}} W^{(t)}_{kL,\tilde{k}} m^{(t)}_{i,\tilde{k}LM} + \sum_{\tilde{k}} W^{(t)}_{kz_iL,\tilde{k}} h^{(t)}_{i,\tilde{k}LM}$$

readout

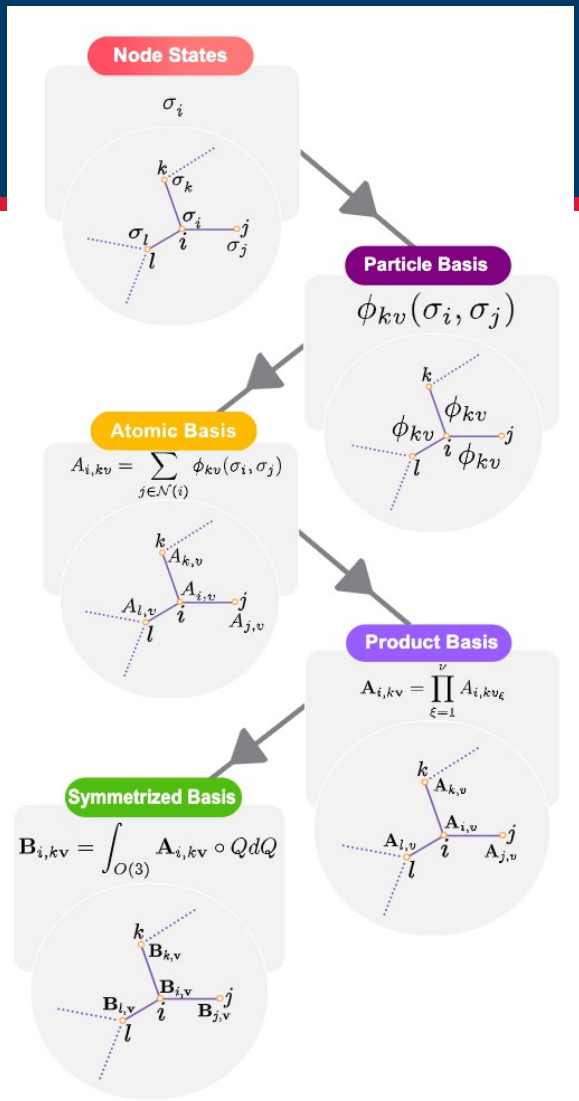
Node energies

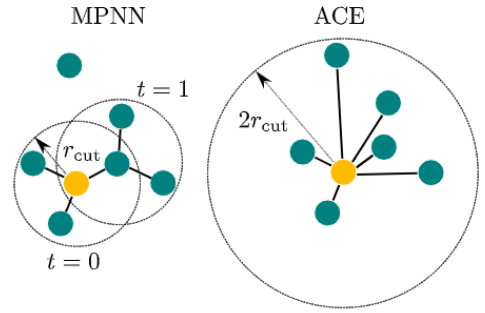
Node features 1 (n_node, k)

edge features (n_edge, 8)

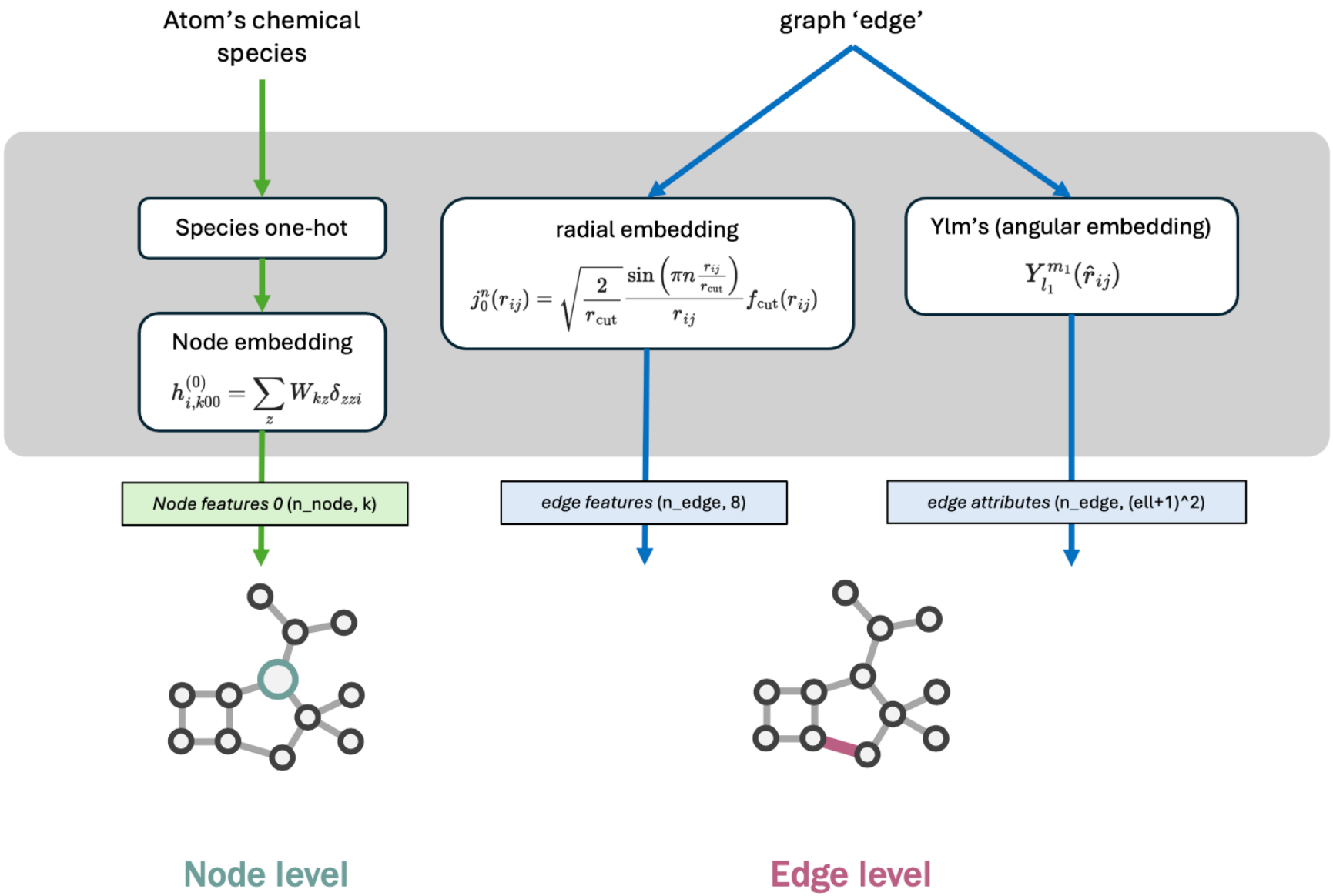
edge attributes (n_edge, (ell+1)^2)

repeat Interaction + product





1. Embeddings: initializing the graph



Computed once in the beginning.

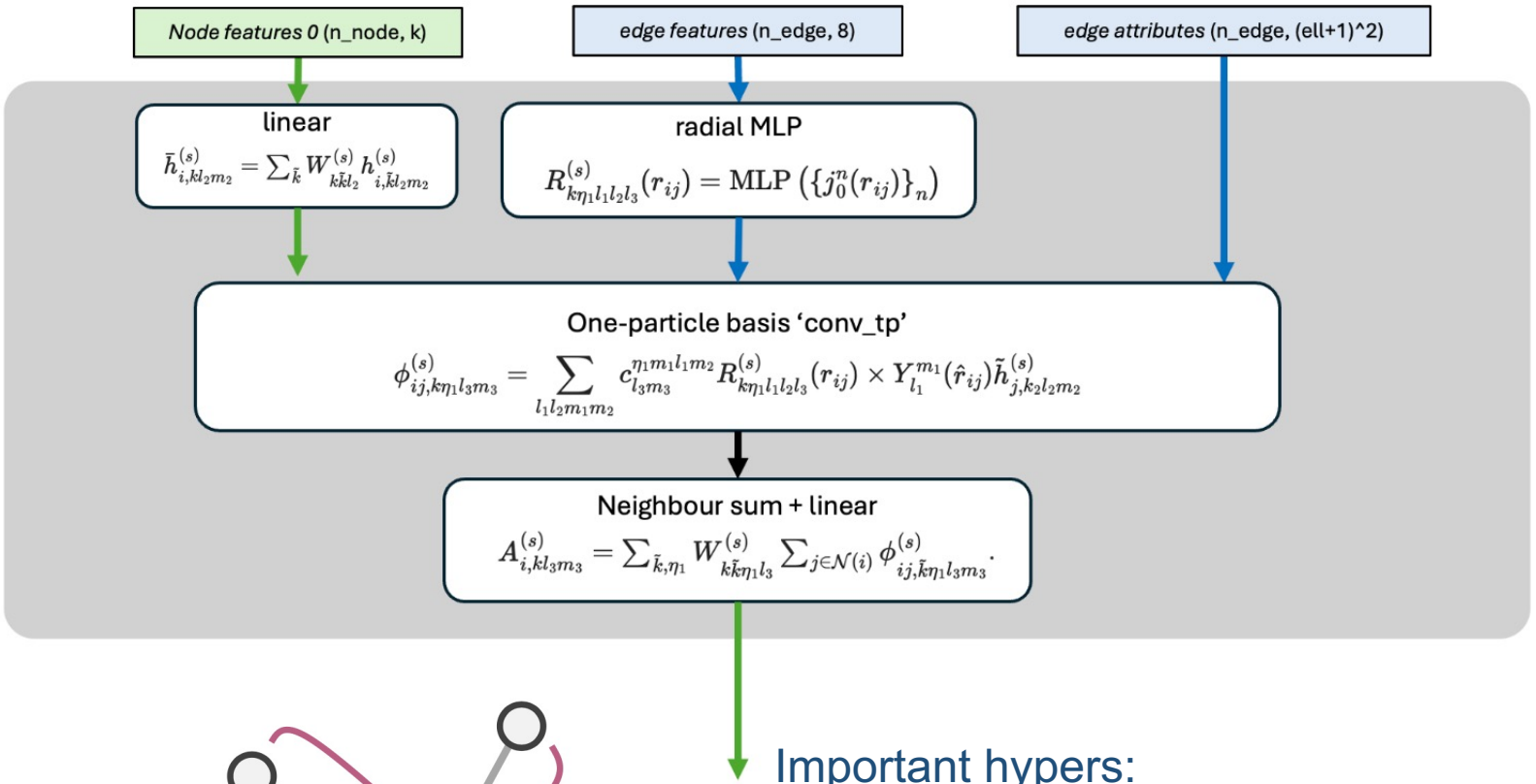
For each atom/node expand distances/ angles into radial/angular basis.

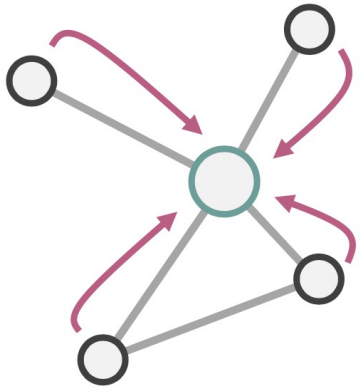
Encode species one-hot and project into a learnable embedding.

Important hypers:

- R_{cut} (r_max)
- L_{max} (max_ell)
- N_{max} (num_radial_basis)

2. Interaction: pooling across neighbours





Important hypers:

- number of interactions S
- k (number of channels)

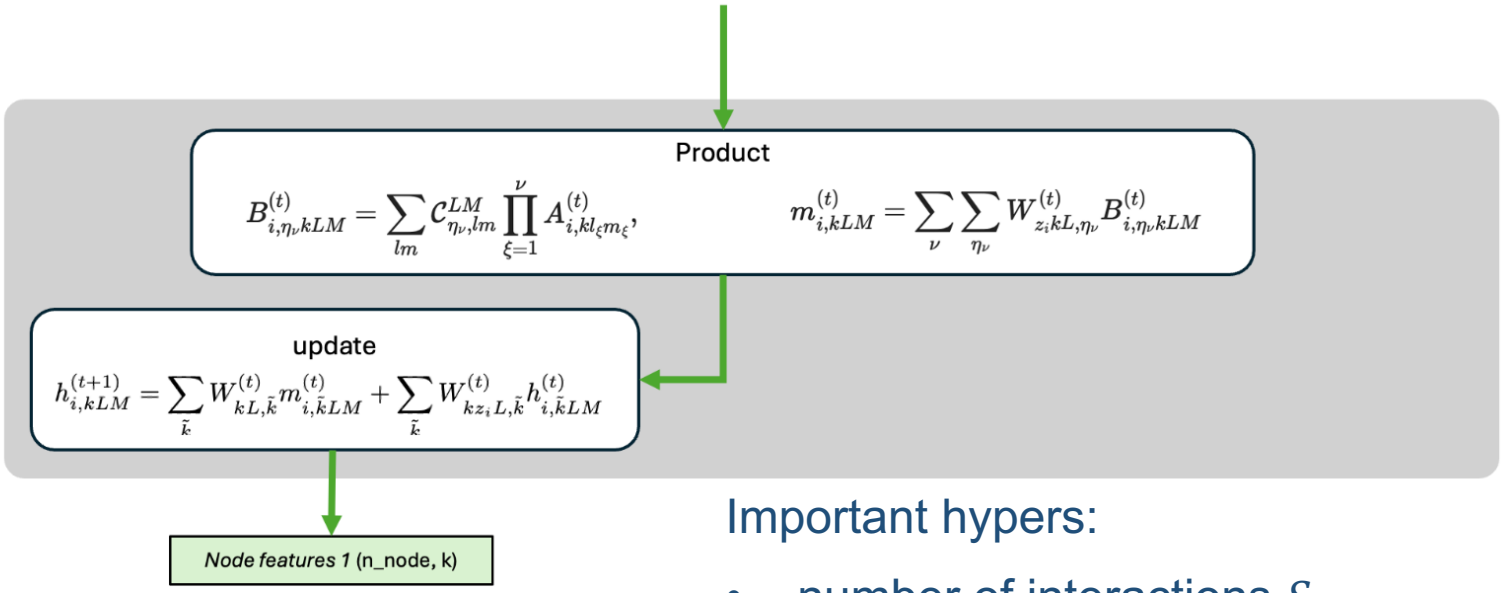
Updated every time the model evaluates the interaction layer (S times)

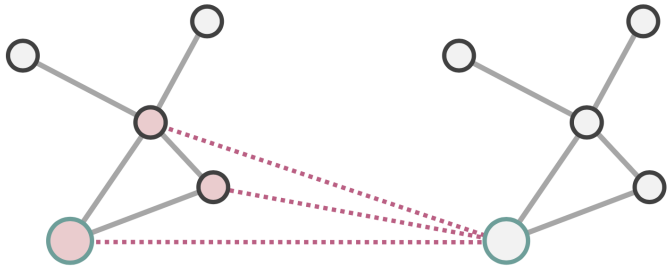
The one-particle basis (ϕ) is constructed using **e3nn**, and unlike in ACE, it is equivariant (there are η_1 ways to construct equivariances)

The atomic basis (A) is obtained by pooling over neighbors (permutational invariance)

Everything is multiplied by additional learnable weights

3. Product: updating the node features





Important hypers:

- number of interactions S
- k (number of channels)
- L_{max} (maximum equivariance):
  32x1e+32x1o:
  32 invariant and
  32 equivariant channels
  $k \times (Y_{00}, Y_{1,-1}, Y_{1,0}, Y_{1,+1})$
- ν correlation order

Updated every time the model evaluates the product layer (S times)

The atomic basis is tensor-multiplied with itself to form the product basis (main difference to NequIP: short-cut to obtaining higher body orders with fewer layers)

The fully symmetric basis (B) is constructed by contracting the atomic basis (A) using **e3nn**. There are η_ν ways to achieve this.

Finally, the message is obtained by multiplying with learnable weights and the node feature is updated.

4. Readout from each layer

In an S-layer MACE, the readout from the features at layer s is:

$$\mathcal{R}^{(s)}\left(\boldsymbol{h}_i^{(s)}\right) = \begin{cases} \sum_k W_k^{(s)} h_{i,k00}^{(s)} & \text{if } 1 < s < S \\ \mathrm{MLP}\left(\left\{h_{i,k00}^{(s)}\right\}_k\right) & \text{if } s = S \end{cases}$$

Forces obtained via autodiff. Loss in each batch:

$$\mathcal{L} = \frac{\lambda_E}{B}\sum_{b=1}^{B}\left(\frac{E_b - \hat{E}_b}{N_b}\right)^2 + \frac{\lambda_F}{3B}\sum_{b=1}^{B}\sum_{i_b,\alpha=1}^{N_b,3}\left(-\frac{\partial E_b}{\partial r_{i_b,\alpha}} - \hat{F}_{i_b,\alpha}\right)^2$$

Total field-of view:

$$R_{cut} \times S$$

Total body-order (within the first)

$$(\nu + 1) \times S + 1$$

Training the Model: Minimizing the Loss

Input: atomic environments p

Output: observables $F(p)$

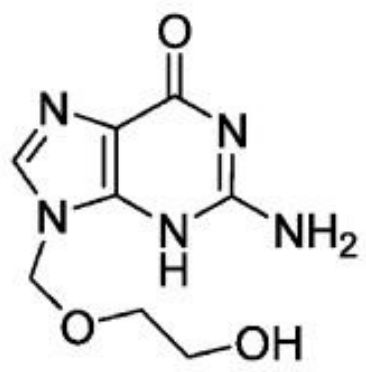
Learning:

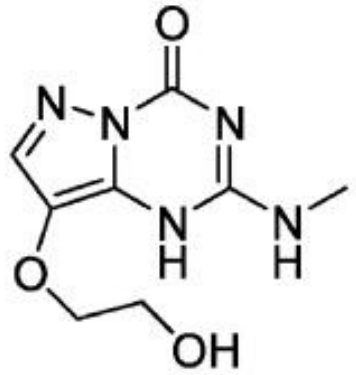
minimize loss on train set $\{p, F(p)\}$

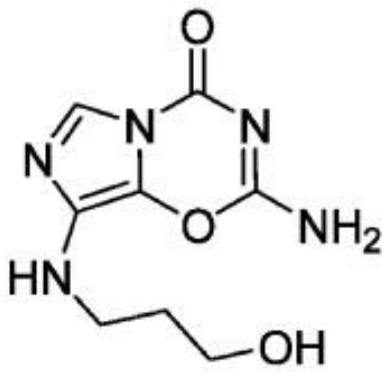
$$\mathcal{L} = \sum_{n=1}^{N_{data}} [F_n - F(p_n)]^2$$

to determine w_{ij} etc

Data can be different molecules, or different geometries or both ...



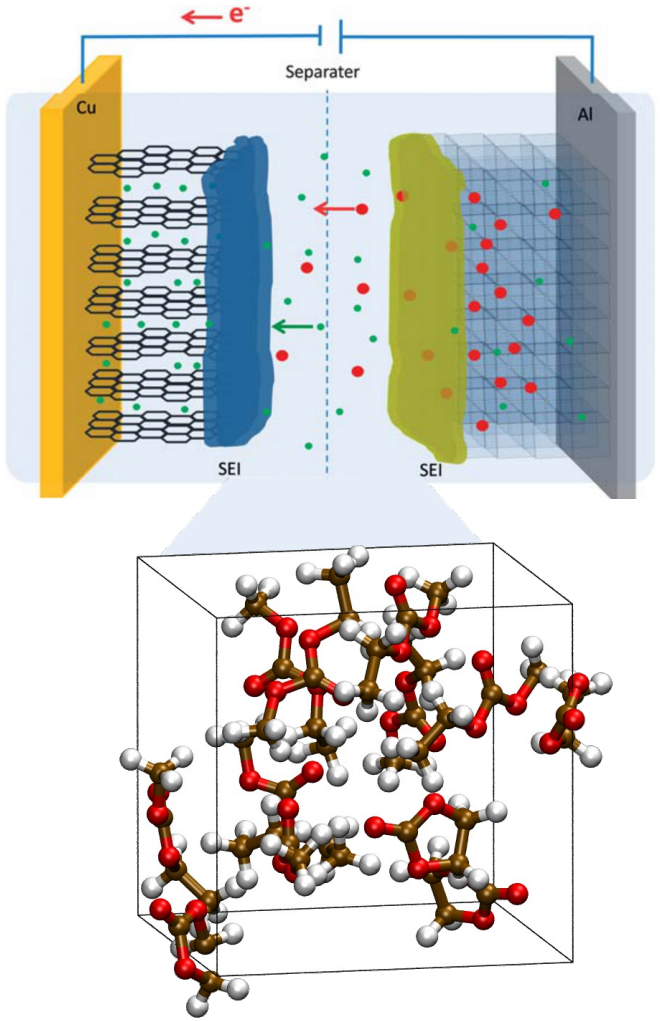


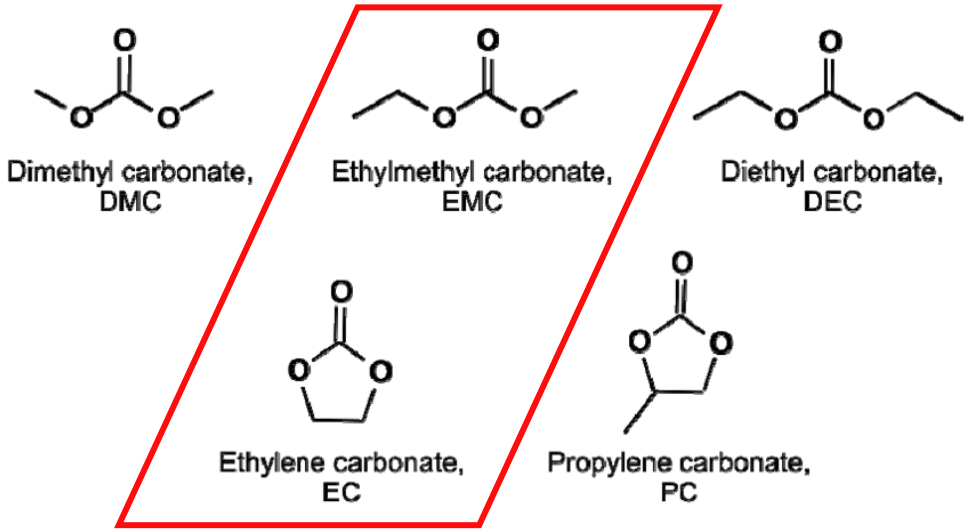


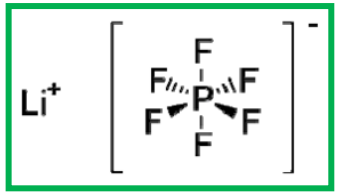
p_1	p_2	p_3
F_1	F_2	F_3
$F(p_1)$	$F(p_2)$	$F(p_3)$
F_1-$F(p_1)$	F_2-$F(p_2)$	F_3-$F(p_3)$

Case study in the notebooks: MLIP for an organic solvent

Li-ion battery







Liquid Solvent:

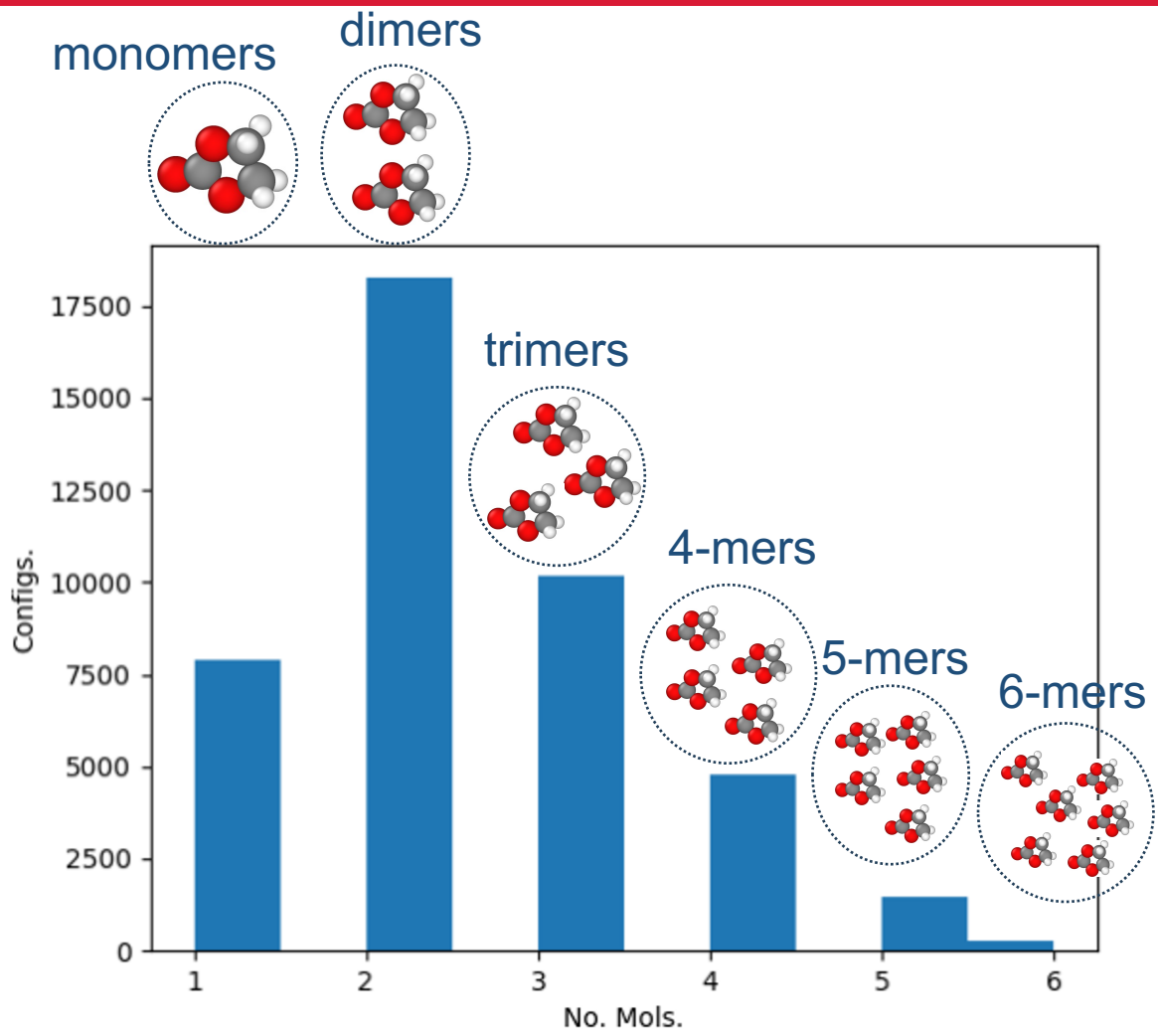
- 3 atomic species: H, C, O
- neutral molecules $(\sim 1/r^3)$

+Salt:

- +3 atomic species: add Li, P, F (quadratic scaling)
- charged long-range effects $(\sim 1/r)$

Case study: MLIP for an organic solvent

- training data type: molecular clusters
- Neural Network + long-range
- we will fit a MACE model on a subset of this data



THE JOURNAL OF PHYSICAL CHEMISTRY B

Schrödinger

High-Dimensional Neural Network Potential for Liquid Electrolyte Simulations

Steven Dajnowicz*, Garvit Agarwal*, James M. Stevenson, Leif D. Jacobson, Farhad Ramezanghorbani, Karl Leswing, Richard A. Friesner, Mathew D. Halls, and Robert Abel

Journal of Chemical Theory and Computation > Vol 22/Issue 7 > Article

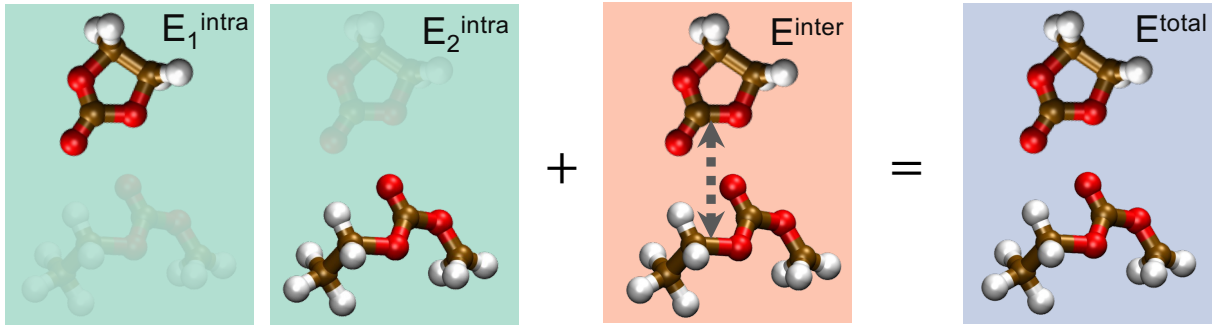
Open Access

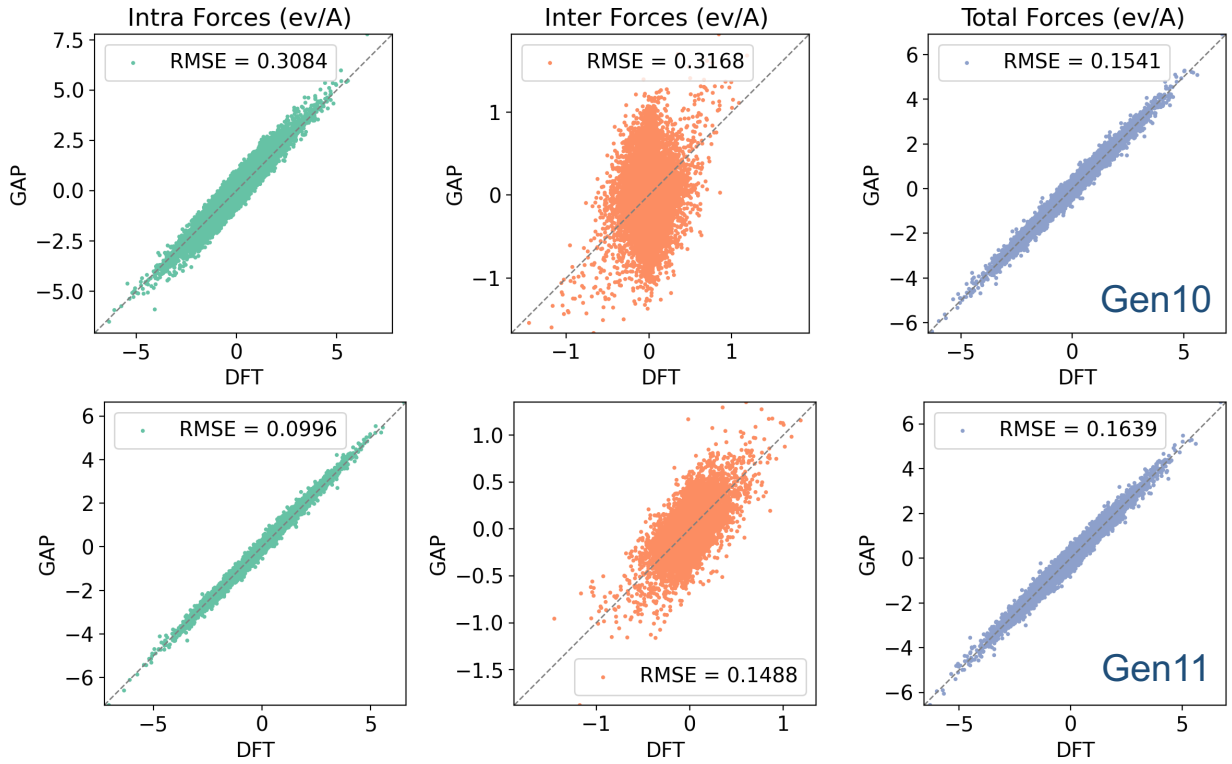
CONDENSED MATTER, INTERFACES, AND MATERIALS | March 31, 2026

Limitations of Cluster-Trained MLIPs for Liquid Density and Diffusivity

Viktor Svahn, Ioan-Bogdan Magdău, Samuel P. Niblett, Gábor Csányi, Kersti Hermansson, and Jolla Kullgren*

Intra-/Inter- Decomposition as a Test





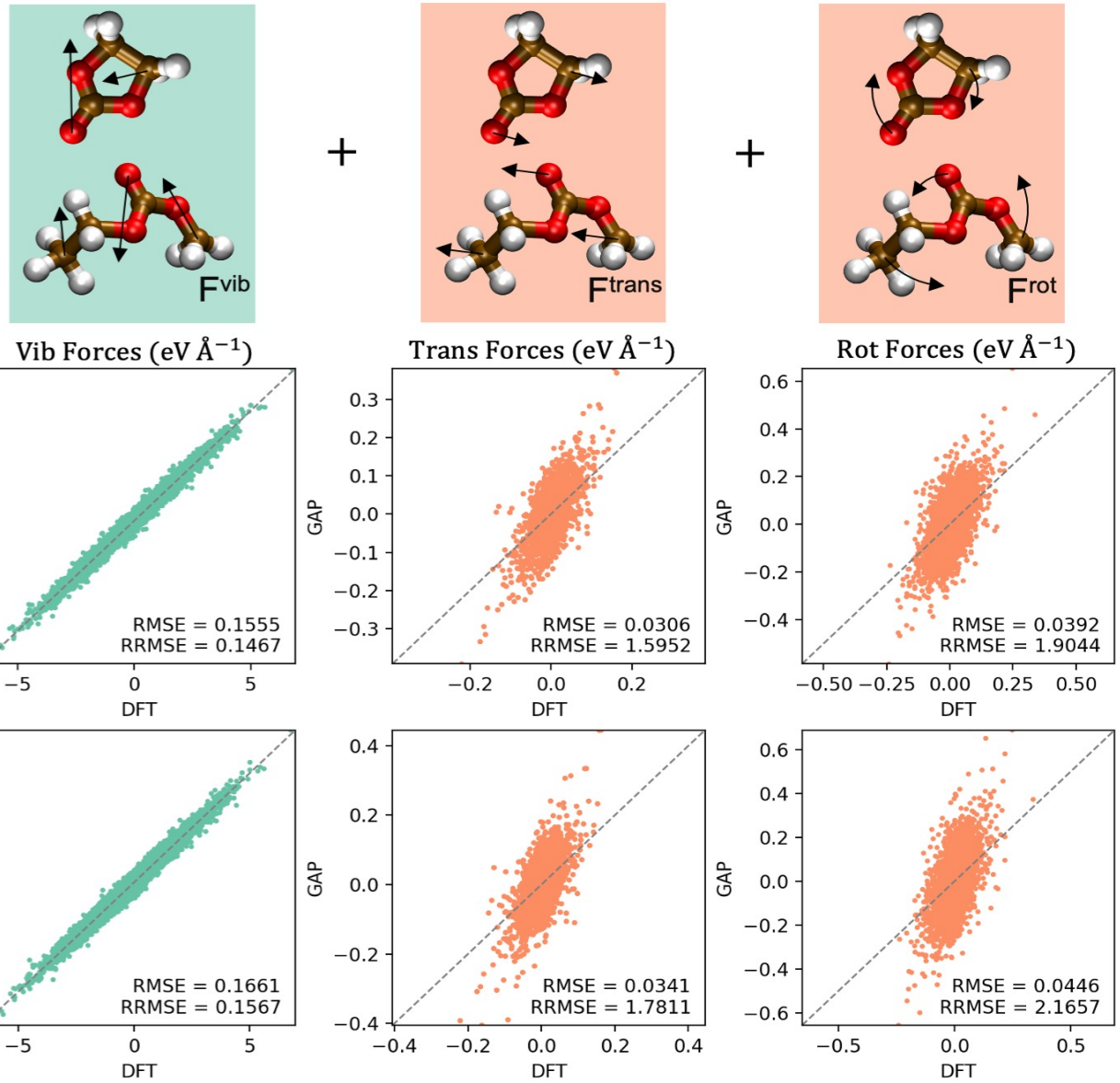
$$L = \left\| F_{total}^{DFT} - F_{total}^{ML} \right\|^2$$

$$L = \left\| \underbrace{\left(F_{intra}^{DFT} - F_{intra}^{ML}\right)}_{\text{dominates loss}} + \underbrace{\left(F_{inter}^{DFT} - F_{inter}^{ML}\right)}_{\text{drives dynamics}} \right\|^2$$

Intra/Inter test:

- **split liquid** configs into molecular
- recompute **molecules** (same geometry) **in vacuum**: intra contributions
- compute inter- as **difference between total and intra-**

Trans-/Rot-/Vib- Decomposition of Forces



Translational forces:

$$F_j^{\text{trans}} = \sum_{k \in j} f_k \qquad \text{(per molecule)}$$

$$f_i^{\text{trans}} = \frac{m_i}{M_j} F_j^{\text{trans}} \qquad \text{(per atom)}$$

Rotational forces:

$$T_j = \sum_{k \in j} f_k \times r_k \qquad \text{(per molecule)}$$

$$f_i^{\text{rot}} = m_i r_i \times (I_j^{\alpha\beta})^{-1} T_j \qquad \text{(per atom)}$$

Vibrational forces:

$$f_i^{\text{vib}} = f_i - f_i^{\text{trans}} - f_i^{\text{rot}} \qquad \text{(per atom)}$$