

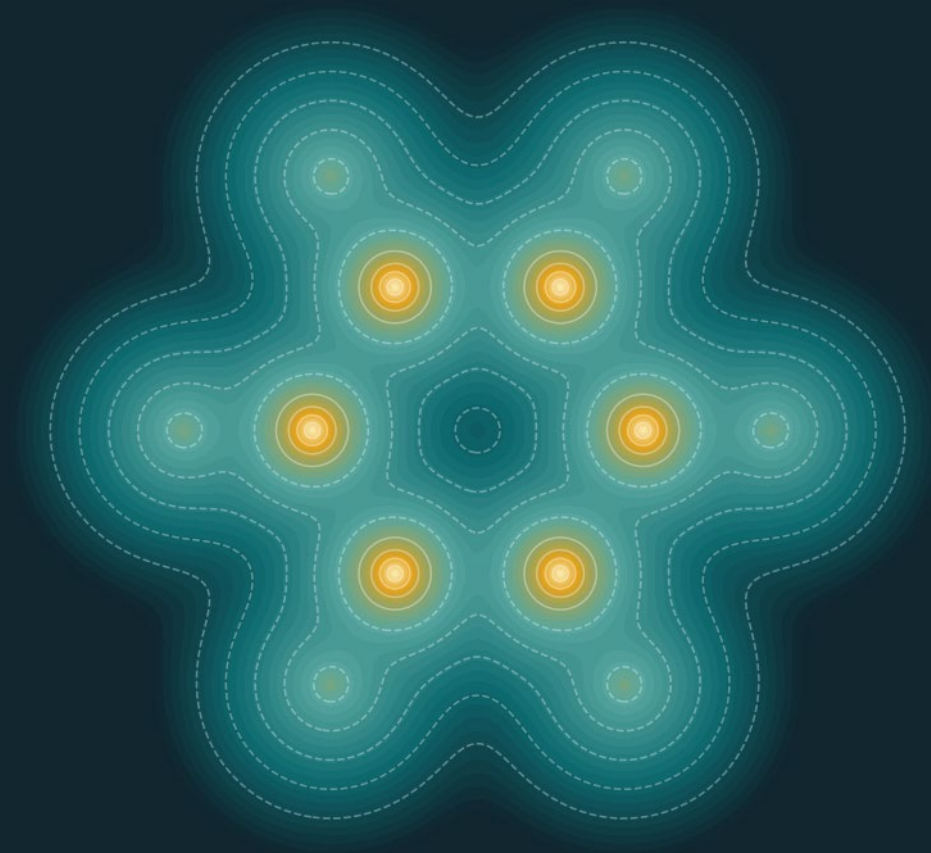
# From atoms to graphs

Representing molecules and materials for graph  
neural networks

Geometric GNNs and Generative AI for Materials Discovery

Arghya Bhowmik · DTU

KIT Karlsruhe · 25 September 2026



Electron density of benzene (schematic)

# Plan for this session

1

## What we're modelling

- Molecules and crystals
- When representations fail
- Tools and datasets

2

## What we predict and why

- DFT simulations
- Energy and forces
- Choosing a representation

3

## Building representations

- Fingerprints and descriptors
- Cutoff and periodic graphs
- Symmetry and equivariant layers

4

## Applications

- What GNNs are used for
- Biomolecular simulation
- What is new since 2023

**Hands on part in 3<sup>rd</sup> slot: Notebook 1**, with six short exercises that are ready to run

# What is a molecule?

A molecule is a group of atoms. Each atom has a nuclear charge  $Z_i$  and a position  $\mathbf{r}_i$  in 3D. The electrons hold the atoms together, and chemists describe this with **bonds**. Each element usually forms a fixed number of bonds:



1 bond



4 bonds

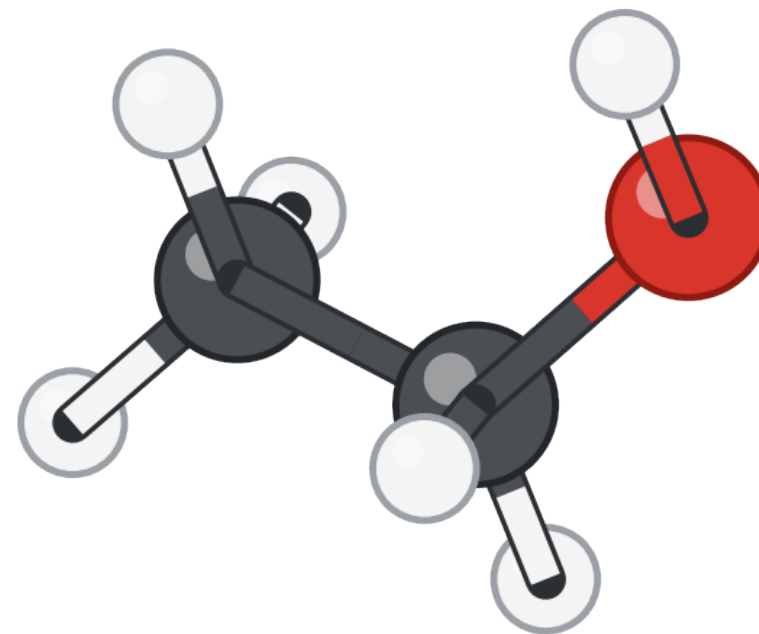


3 bonds



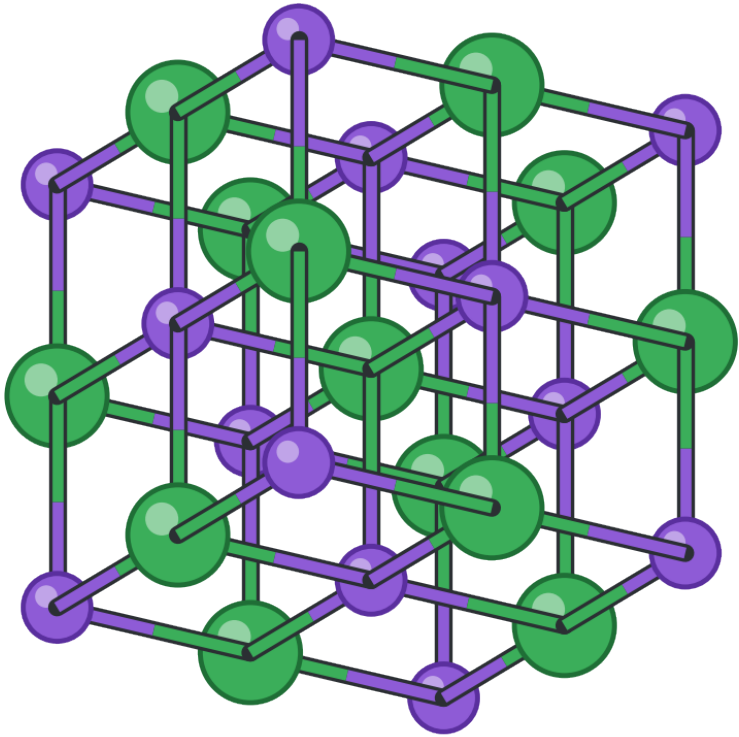
2 bonds

**Key point:** the atoms and their positions are the real data. Bonds are a simplified picture of where the electrons are, and they are less clear-cut than they look.

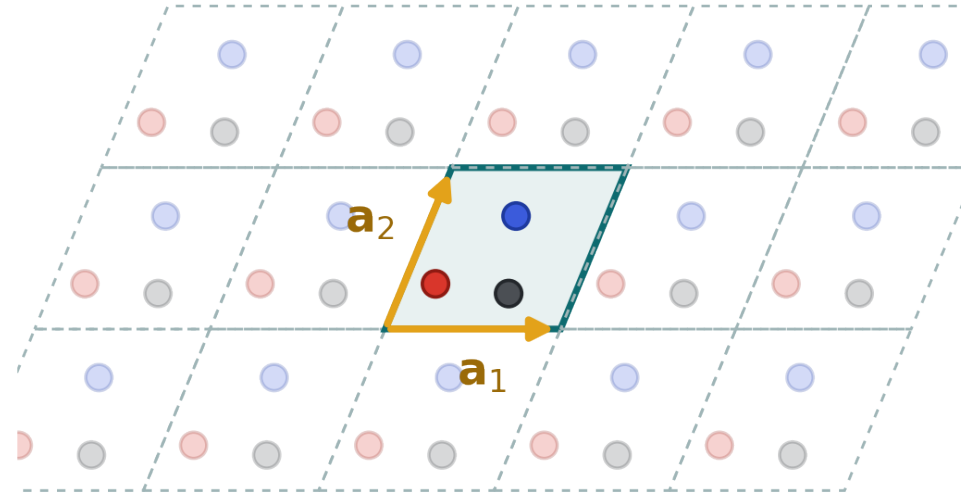


Ethanol,  $\text{C}_2\text{H}_5\text{OH}$ : 9 atoms, 8 bonds

# What is a crystal?



Rock salt, NaCl: Na<sup>+</sup> purple, Cl<sup>-</sup> green



The highlighted unit cell repeats in every direction

## Unit cell

Three lattice vectors  $\mathbf{a}_1$ ,  $\mathbf{a}_2$ ,  $\mathbf{a}_3$  and the atoms inside the cell.

## Periodicity

The cell repeats forever. Fractional coordinates go from 0 to 1 and wrap around, like on a torus.

## Symmetry

Translations, rotations and reflections. Every crystal belongs to one of 230 space groups.

# One molecule, written four ways

## Formula



**Keeps:** which atoms, and how many

**Loses:** how they are connected

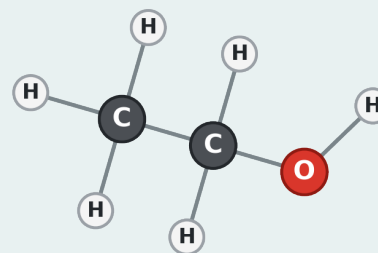
## SMILES string



**Keeps:** the connectivity, as a string

**Loses:** the 3D shape. One molecule can also have many valid strings

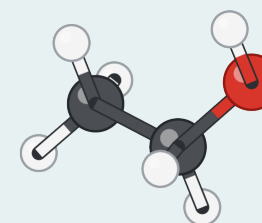
## Bond graph



**Keeps:** the connectivity, as a graph  $G = (V, E)$

**Loses:** the geometry

## 3D coordinates



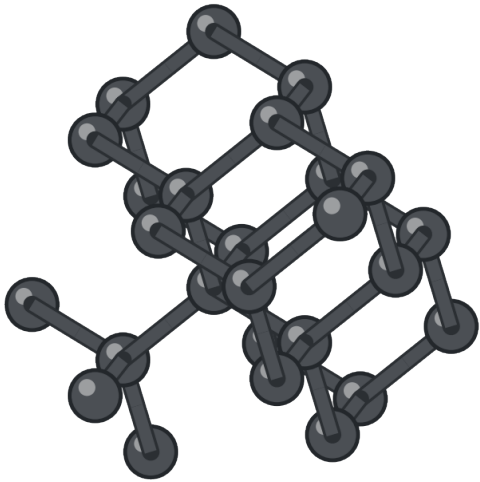
**Keeps:** everything the energy depends on

**Loses:** nothing, but the model must ignore rotation, translation and atom order

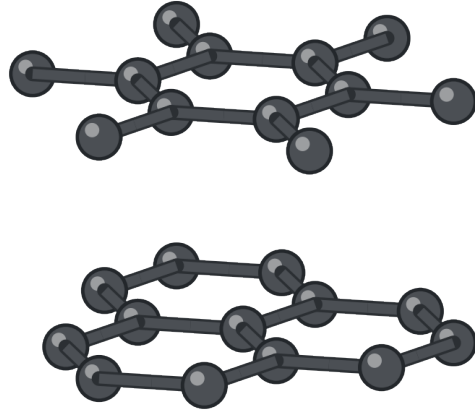
left to right: more information, but also more symmetry to deal with

The energy needs the 3D positions, but many properties only need the graph. We come back to this in Part 2.

# Crystals have the same issues, and one more

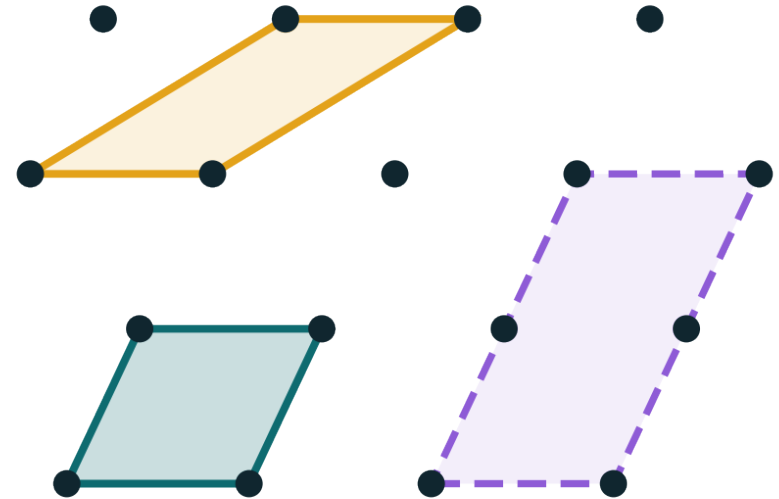


**Diamond**



**Graphite**

Both are pure carbon, but with different structures (polymorphs)



The same lattice, described by three different unit cells

**A good representation should not change with:**

rotation

translation

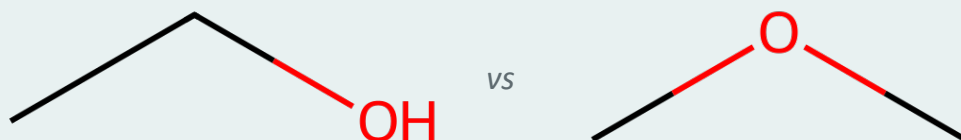
atom order

choice of unit cell

**but it must still tell  
different structures apart**

# When different structures look the same

## Constitutional isomers



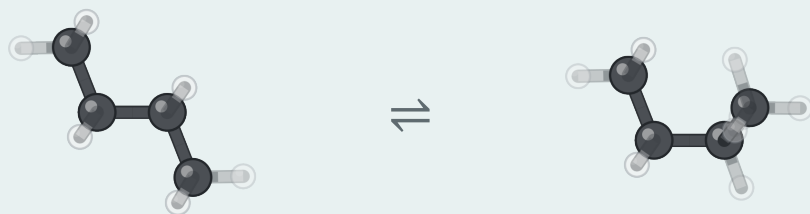
Ethanol and dimethyl ether are both  $C_2H_6O$ . The formula can't tell them apart, but the bond graph can.

## Stereoisomers



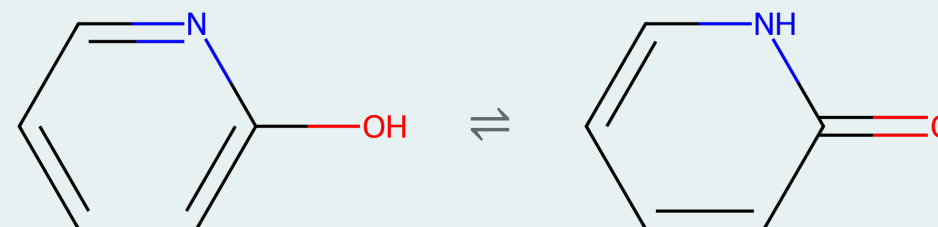
(R)- and (S)-thalidomide have the same bond graph but are mirror images, with different effects in the body. You need stereo labels or 3D.

## Conformers



Butane, anti and gauche: the same molecule, twisted around one bond. Only 3D shows it, and the energies differ by about 0.04 eV.

## Tautomers



2-Hydroxypyridine and 2-pyridone: a hydrogen moves and the bond graph changes. Both exist in solution, so which one do you store?

# Useful Python tools

## RDKit

For molecules: reads SMILES, knows bonds and stereochemistry, makes 3D conformers, and computes descriptors and fingerprints.

```
mol = Chem.MolFromSmiles("CCO")  
mol = Chem.AddHs(mol)  
AllChem.EmbedMolecule(mol)
```

**XYZ** · element and x, y, z, for molecules

## ASE

Stores atoms (element, position, cell) for both molecules and crystals. Reads and writes most file formats, finds neighbours, and runs relaxations and MD.

```
atoms = read("NaCl.cif")  
atoms.calc = calculator  
BFGS(atoms).run(fmax=0.05)
```

**CIF** · cell, fractional coordinates and symmetry, for crystals

## pymatgen

For crystals, from the Materials Project team: symmetry, comparing structures, phase diagrams and stability.

```
s = Structure.from_file("NaCl.cif")  
sga = SpacegroupAnalyzer(s)  
sga.get_space_group_symbol()
```

**SMILES / SDF** · molecular graphs, with or without 3D

# Datasets to get started

## 1 Molecular properties

**QM9:** ~134k small organic molecules (up to 9 heavy atoms) with DFT properties

**MoleculeNet:** a collection of benchmarks, including solubility, toxicity and binding

## 2 Materials databases

**Materials Project:** ~150k inorganic crystals with energies, band gaps and phase diagrams, plus a Python API

**OQMD, AFLOW, NOMAD, Alexandria:** large DFT databases · **COD:** measured structures

## 3 Forces and trajectories

Every step of relaxations or MD runs is saved, so each structure has **energy and forces**. This is the training data for ML potentials.

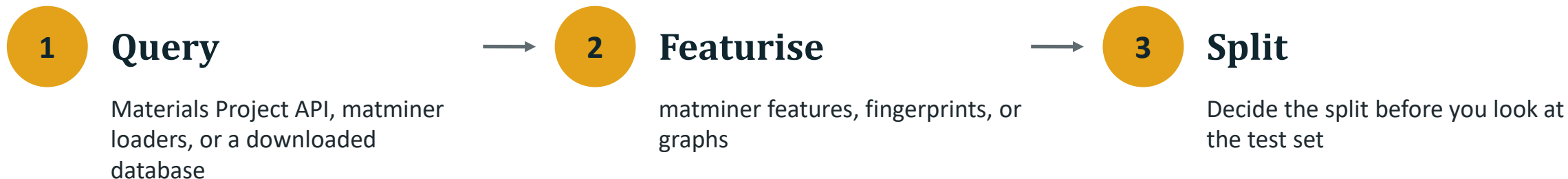
**MD17 / rMD17** (single molecules) · **OC20** (catalyst surfaces) · **MPtrj, Alexandria** (crystals)

## 4 Biomolecules and drug-like molecules

**SPICE:** over a million structures of drug-like molecules, peptides and fragments, with energies and forces · **ANI-1x, QMugs**

**PDBbind:** measured protein–ligand binding strengths, not DFT

# From a database to a training set



**Matbench:** standard materials tasks with fixed splits, so you can compare your results with published ones.

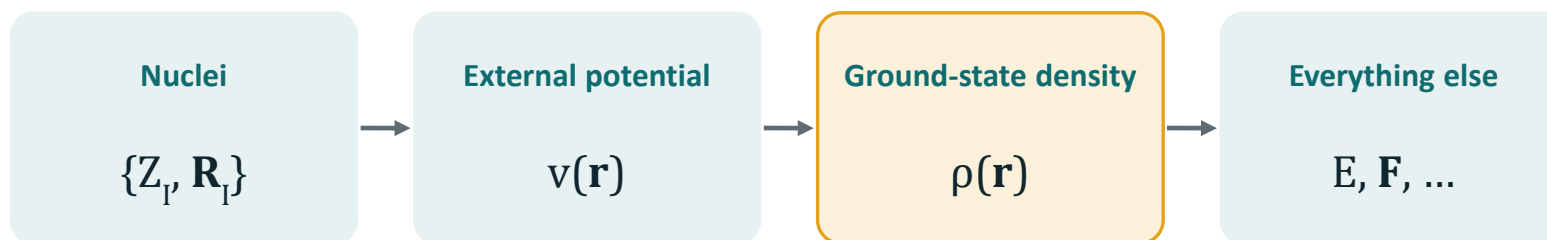
## ! Consistency

Energies from different DFT methods or settings can't be compared directly (PBE and  $r^2$ SCAN, for example). Don't mix sources in one training set without correcting for this.

## ! Data leakage

Databases have many near-duplicates, so a random split makes your test error look better than it is. Split by scaffold for molecules, and by composition or structure for materials.

# Density functional theory (DFT)



1

## Hohenberg–Kohn

For the ground state, the electron density  $\rho(\mathbf{r})$  determines everything, including the energy. So we can work with a function in 3D instead of a wavefunction in 3N dimensions.

2

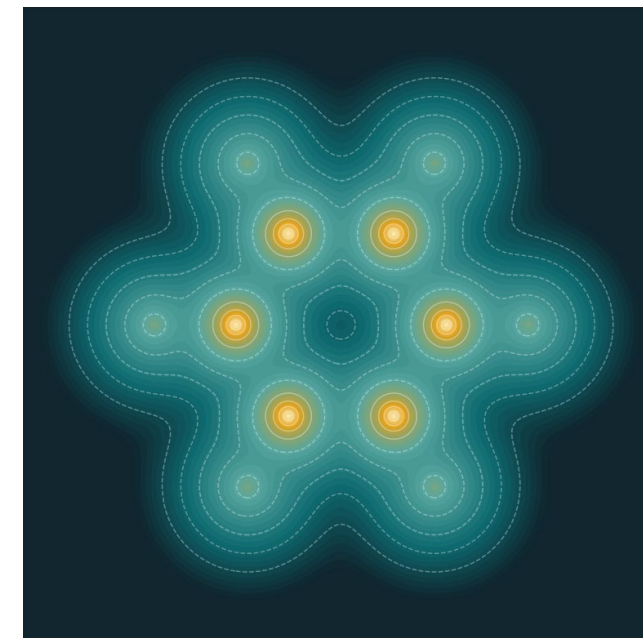
## Kohn–Sham

Makes this practical: solve for non-interacting electrons in an effective potential, and repeat until the density stops changing.

3

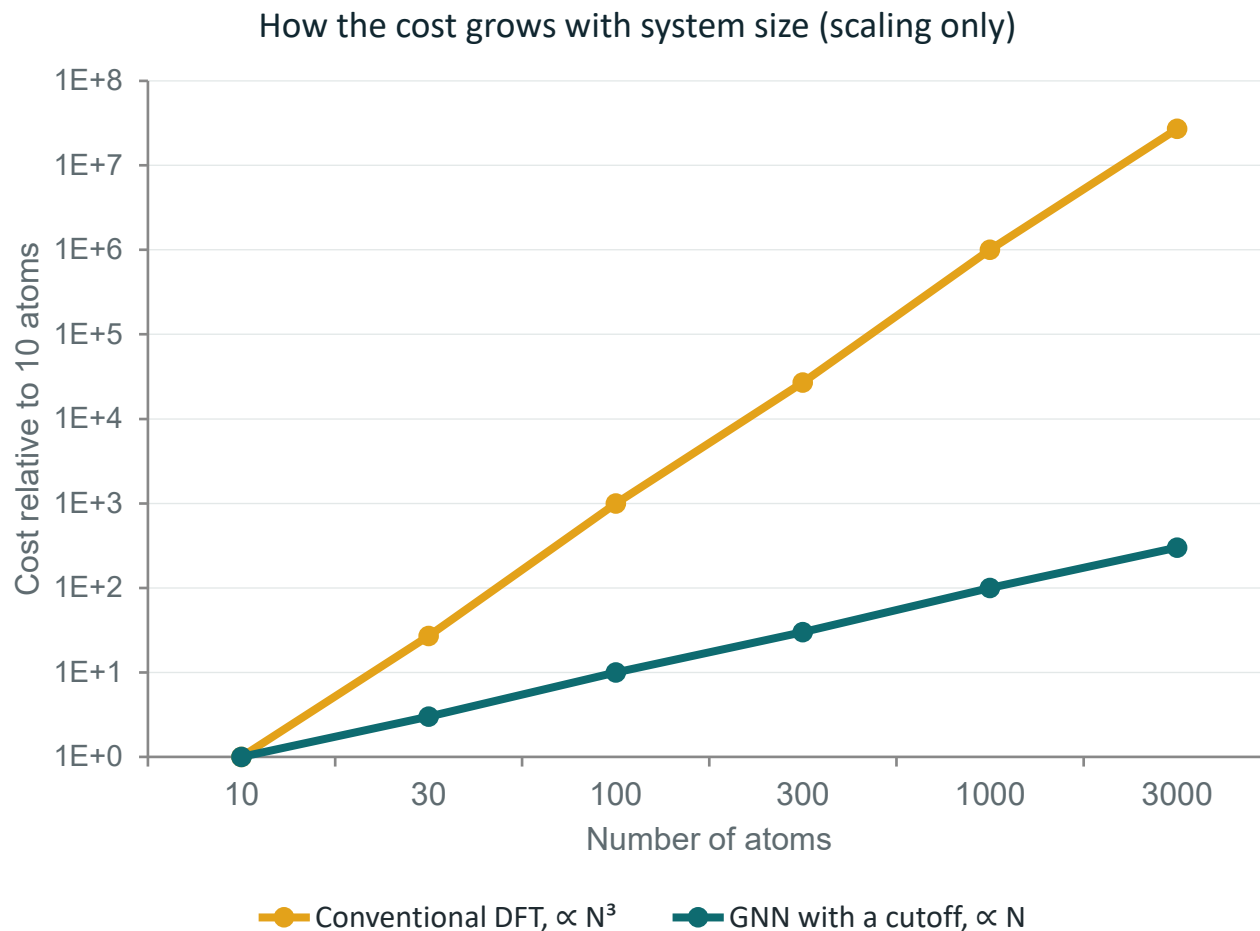
## Limitation

The exchange–correlation part is unknown and has to be approximated (LDA, GGA such as PBE, meta-GGA, hybrids). This choice decides the accuracy.



Electron density of benzene (schematic)

# DFT is accurate, but expensive



## Conventional DFT

$N^3$

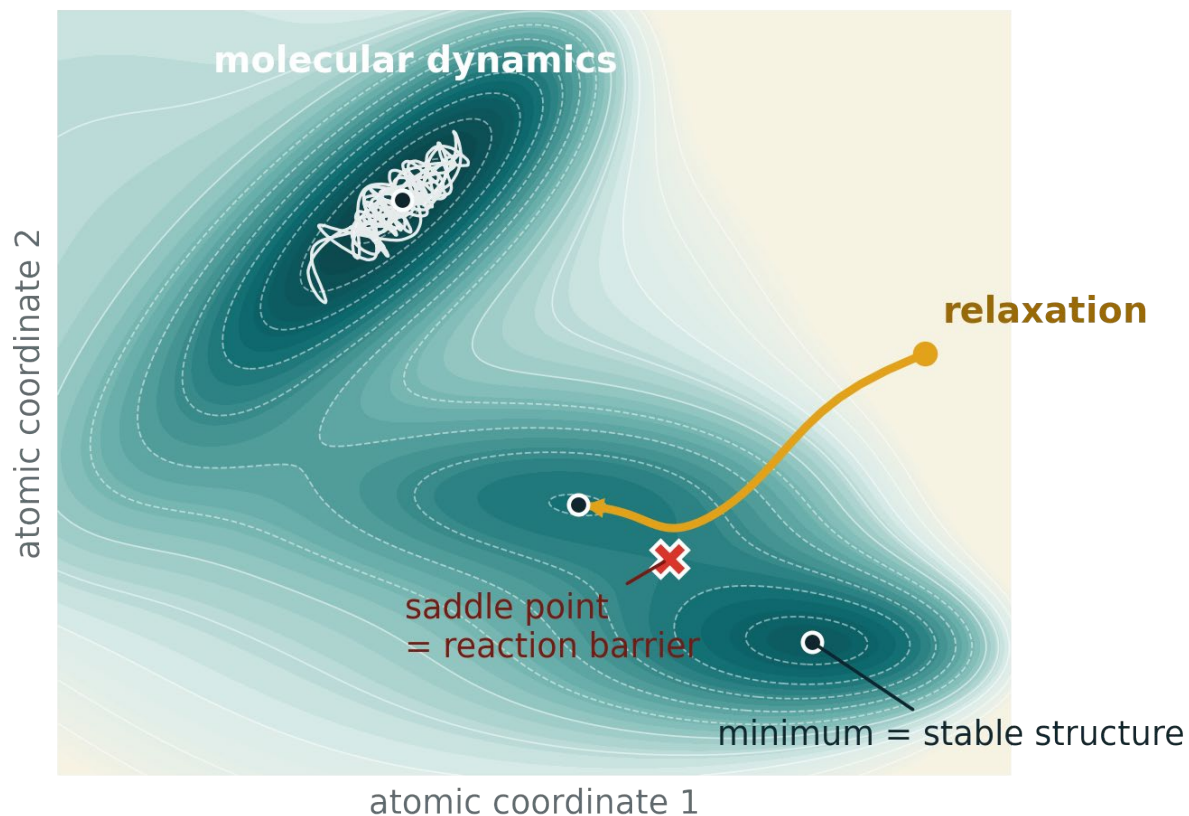
Double the number of atoms and the cost goes up about eight times. Large systems quickly become impossible, so in practice DFT is limited to small systems.

## GNN potentials

$N$

With a cutoff, the cost grows linearly with the number of atoms. That makes much bigger systems possible, and each calculation is much faster.

# Energy and forces are the main target



$$E : \mathbb{R}^{3N} \rightarrow \mathbb{R}$$

$$\mathbf{F}_i = -\partial E / \partial \mathbf{r}_i$$

- **Relaxation:** follow the forces downhill to a minimum, which is the stable structure
- **Molecular dynamics:** solve Newton's equations,  $m_i d^2 \mathbf{r}_i / dt^2 = \mathbf{F}_i$ , to follow the atoms over time
- **Phonons:** second derivatives at a minimum give vibrations and heat capacity
- **Barriers:** saddle points between minima give reaction and diffusion rates
- **Stability:** comparing the energies of minima gives phase diagrams (session 3)

# Which representation does your target need?

more information, but more cost

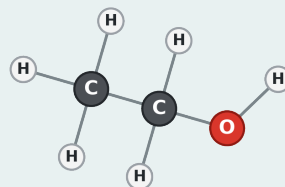
## Composition only



**Enough for:** bulk trends when the structure isn't known yet, e.g. screening new compositions (Roost)

**Misses:** polymorphs, isomers, geometry

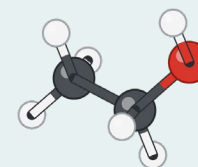
## 2D bond graph or SMILES



**Enough for:** solubility, toxicity and most property models for organic molecules

**Misses:** conformers, and stereochemistry unless it is labelled

## 3D geometry

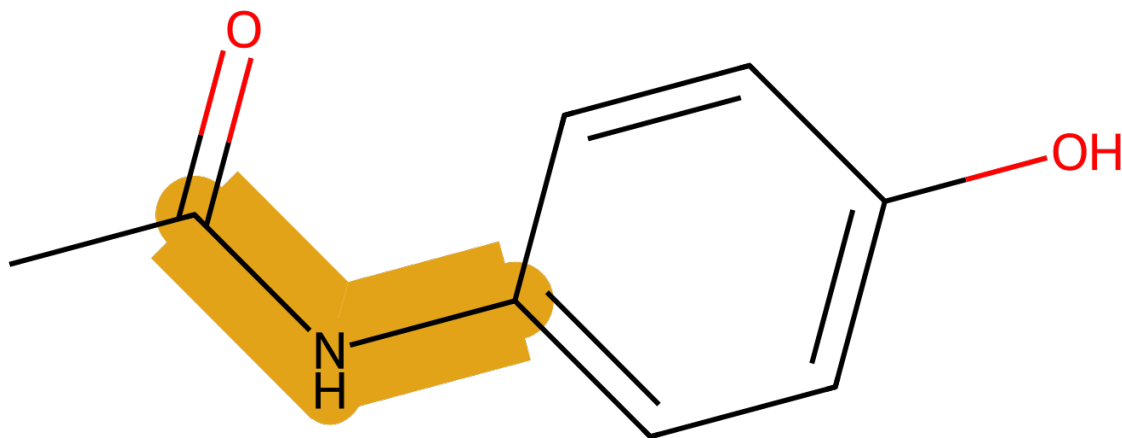


**Needed for:** energy and forces, binding, spectra, conformational effects

**Cost:** you need 3D structures, and the model has to handle symmetry

**Rule of thumb:** use the simplest representation that can still tell apart the structures your property depends on.

# Hand-made fingerprints from chemistry knowledge



Morgan fingerprint of paracetamol: each atom's neighbourhood, like the highlighted one, switches on a bit

| Descriptor                | Paracetamol         |
|---------------------------|---------------------|
| Molecular weight          | 151.2 g/mol         |
| logP (lipophilicity)      | 1.35                |
| Polar surface area        | 49.3 Å <sup>2</sup> |
| H-bond donors / acceptors | 2 / 2               |

These **work well** when you know what controls the property. Lipinski's rule of five for oral drugs uses exactly these numbers.

**Limitation:** you need to understand the chemistry behind the property, and the features are fixed before the model sees any data.

# ACSF: atom-centred symmetry functions

## 1 Cutoff function

$$f_c(R) = \frac{1}{2} \left[ \cos \left( \frac{\pi R}{R_c} \right) + 1 \right] \quad \text{for } R \leq R_c, \quad 0 \text{ otherwise}$$

Each neighbour's contribution fades smoothly to zero at  $R_c$ , so nothing jumps when an atom crosses the cutoff.

## 2 Radial functions

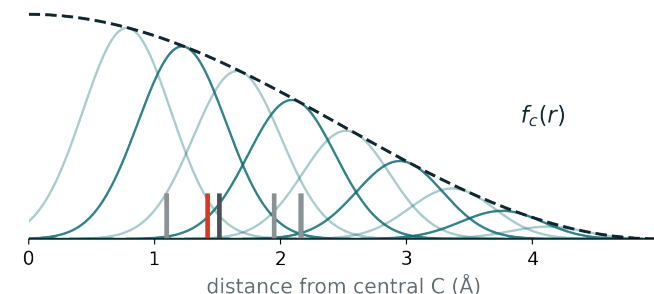
$$G_i^{\text{rad}} = \sum_{j \neq i} e^{-\eta(R_{ij} - R_s)^2} f_c(R_{ij})$$

A Gaussian in the distance, centred at  $R_s$  with width  $\eta$ . Each  $(\eta, R_s)$  pair gives one number: roughly, how many neighbours sit at that distance.

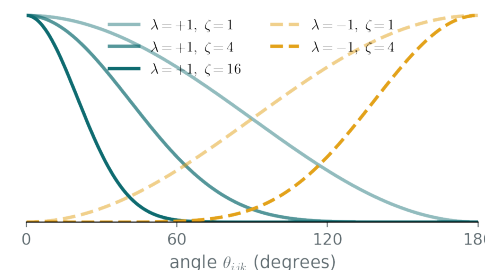
## 3 Angular functions

$$G_i^{\text{ang}} = 2^{1-\zeta} \sum_{j, k \neq i} (1 + \lambda \cos \theta_{ijk})^\zeta e^{-\eta(R_{ij}^2 + R_{ik}^2 + R_{jk}^2)} \\ \times f_c(R_{ij}) f_c(R_{ik}) f_c(R_{jk})$$

Adds the angles between pairs of neighbours.  $\lambda = \pm 1$  decides whether small or large angles count, and  $\zeta$  sets how sharp the function is.



Radial functions (lines) and cutoff (dashed) for the carbon in ethanol. Ticks mark its neighbours; the bars are the resulting values.



The angular term  $(1 + \lambda \cos \theta)^\zeta$  for a few choices of  $\lambda$  and  $\zeta$

Behler–Parrinello:  $\mathbf{E} = \sum_i \mathbf{NN}(\mathbf{G}_i)$ , one small network per element

# SOAP: smooth overlap of atomic positions

## 1 Blur the neighbours into a density

$$\rho_i(\mathbf{r}) = \sum_j \exp\left(-\frac{|\mathbf{r} - \mathbf{r}_{ij}|^2}{2\sigma^2}\right) f_c(r_{ij})$$

Put a Gaussian of width  $\sigma$  on each neighbour inside the cutoff. With several elements, make one density per element.

## 2 Expand it in basis functions

$$\rho_i(\mathbf{r}) = \sum_{n,l,m} c_{nlm} g_n(r) Y_{lm}(\hat{\mathbf{r}})$$

Radial functions  $g_n$  and spherical harmonics  $Y_{lm}$ . The coefficients  $c_{nlm}$  describe the environment, but they change when it rotates.

## 3 Keep the part that rotation can't change

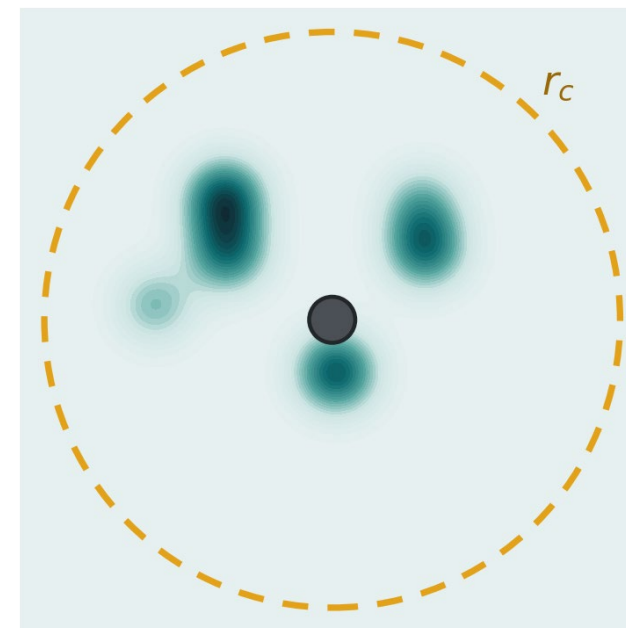
$$p_{nn'l} = \sum_{m=-l}^l c_{nlm}^* c_{n'lm}$$

A rotation only mixes the  $m$  values within each  $l$ , so this sum stays the same. This is the power spectrum, the SOAP vector.

## 4 Compare two environments

$$K(i, k) = \left( \frac{\mathbf{p}_i \cdot \mathbf{p}_k}{|\mathbf{p}_i| |\mathbf{p}_k|} \right)^\zeta$$

The normalised dot product says how similar two environments are. This kernel is the basis of GAP potentials.

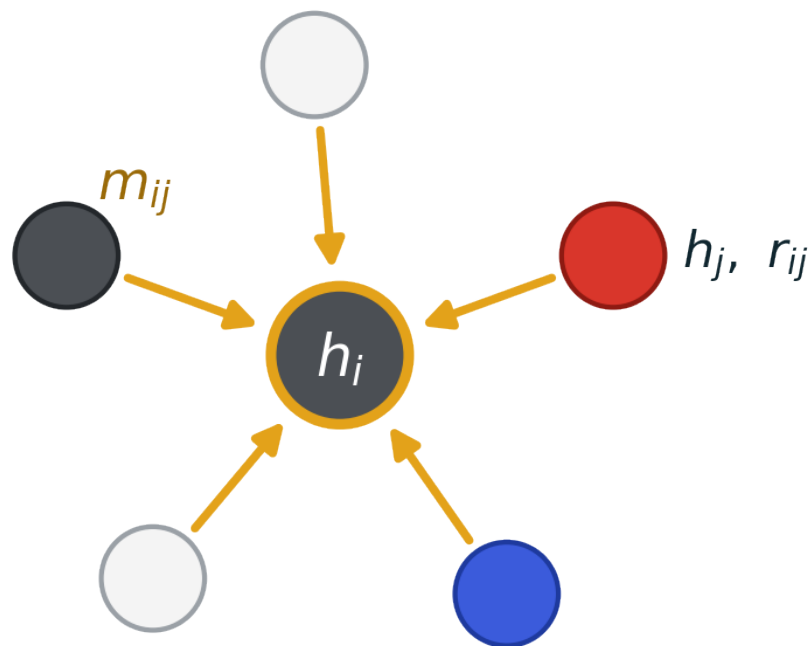


Neighbour density around the carbon in ethanol

**Settings:** cutoff  $r_c$ , width  $\sigma$ , and how many radial ( $n_{\max}$ ) and angular ( $l_{\max}$ ) functions to keep

Both ACSF and SOAP are local, smooth and invariant, so writing  $\mathbf{E} = \sum_i \epsilon(\mathbf{G}_i)$  works well. But they are fixed and general, not tuned to your property, and they get large when there are many elements.

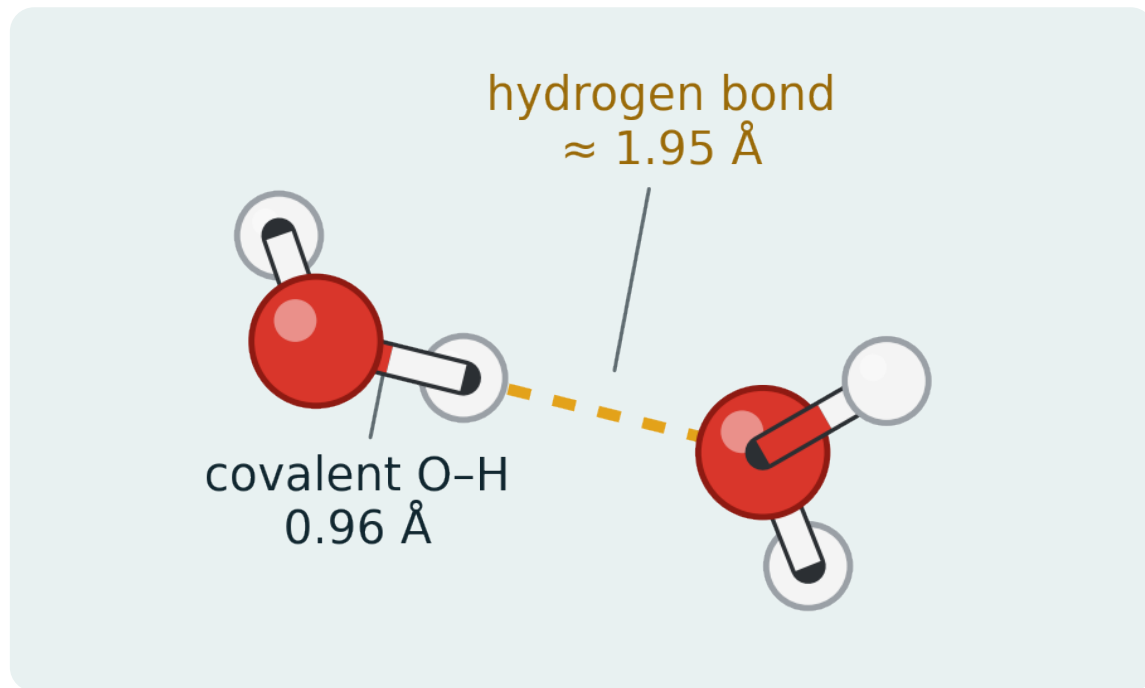
# GNNs learn the fingerprint from data



$$\begin{aligned}m_{ij} &= \varphi(h_i, h_j, r_{ij}) \\h'_i &= \psi(h_i, \sum_j m_{ij}) \\E &= \sum_i \varepsilon(h_i)\end{aligned}$$

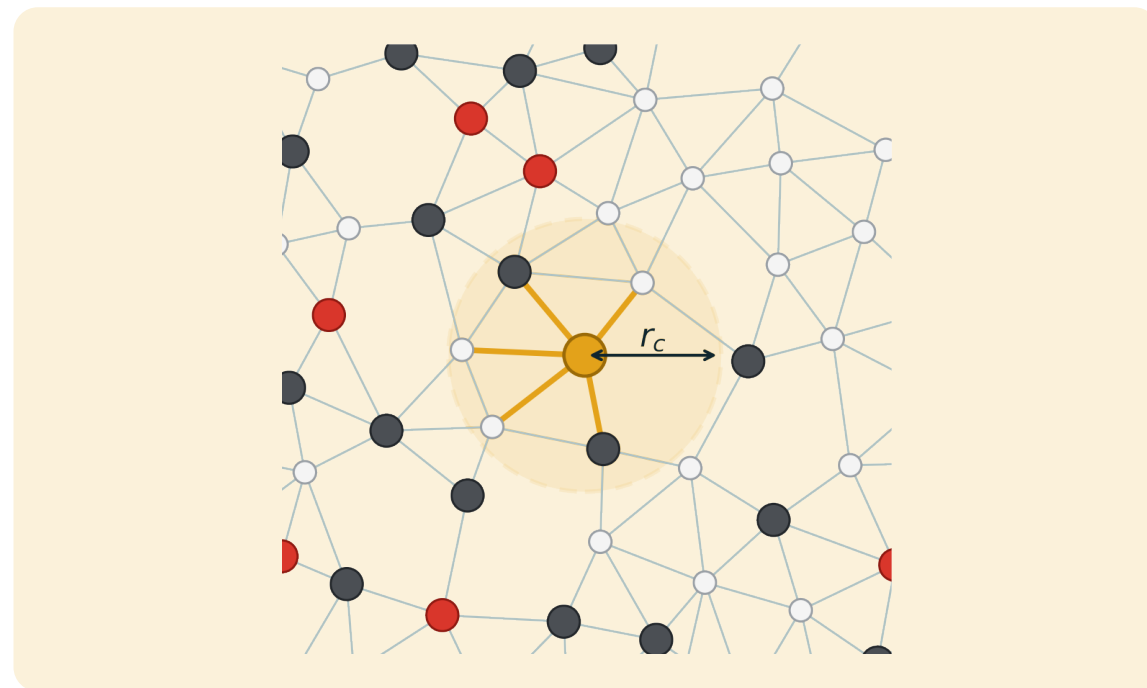
- The atom vector  $h_i$  is **trained on the target itself**, so it keeps what matters for that property.
- **2D graph GNN**: edges are bonds. Enough when the connectivity decides the property.
- **Geometric GNN**: edges carry distances, and later directions. Needed when 3D matters.
- **Covered earlier this week**: message passing, equivariance, attention and graph transformers.

# Which atoms should we connect?



## Bonds come in all strengths

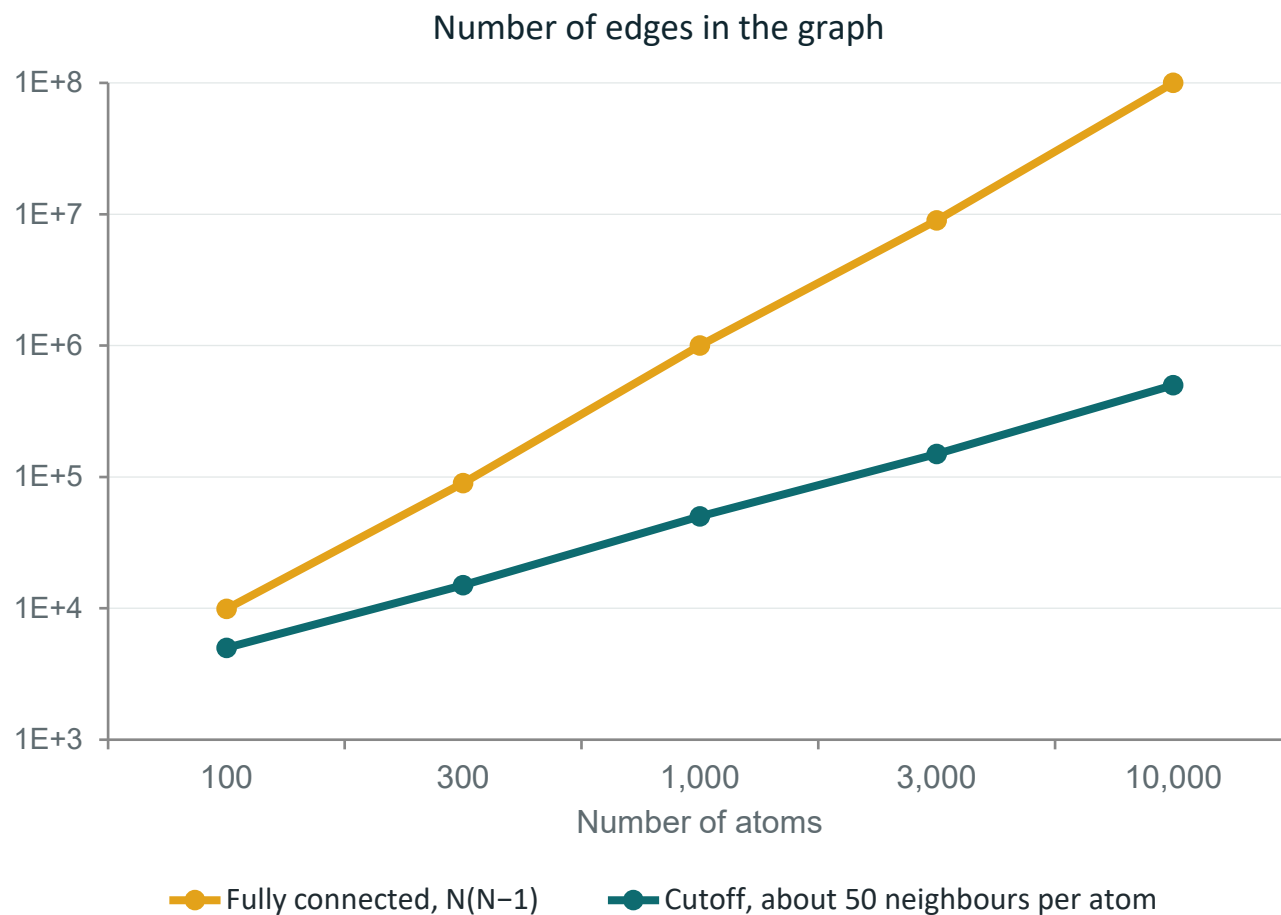
Covalent, ionic and metallic bonds, hydrogen bonds, van der Waals contacts. There is no clear line between bonded and not bonded.



## What we do instead

Connect each atom to every atom within a cutoff radius  $r_c$ , usually 4–6 Å. Each edge stores the distance  $r_{ij}$ , expanded in radial basis functions.

# A cutoff makes the cost grow linearly



## Fully connected

every pair of atoms is an edge, so there are  $N(N-1)$  edges:  $O(N^2)$



## With a cutoff

atoms can't be packed arbitrarily close, so each one has a limited number of neighbours:  $O(N)$



## It also makes physical sense

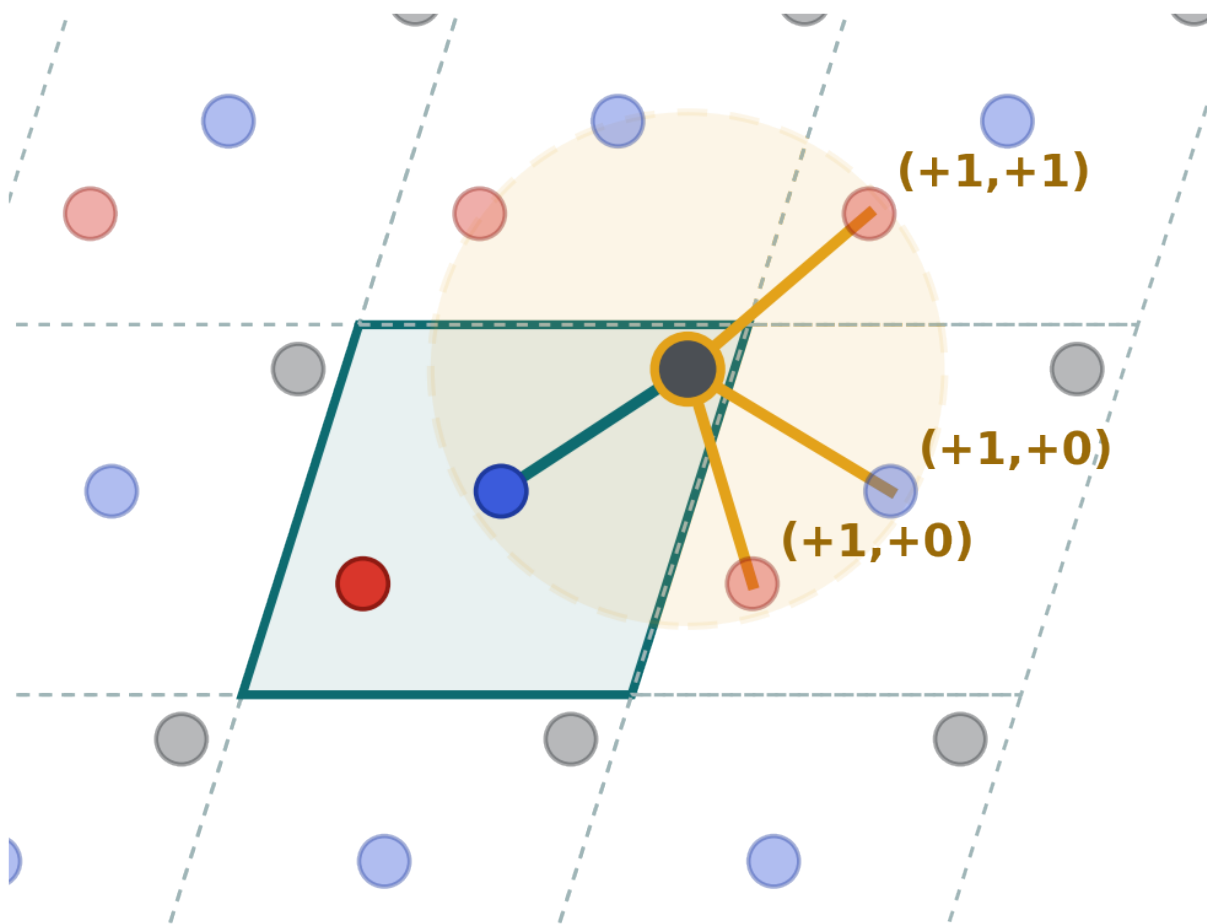
electrons are “nearsighted” (Kohn), so an atom’s energy depends mostly on its surroundings



## Downside

anything beyond  $r_c$  is missed. Stacking  $T$  message-passing layers extends the reach to  $T \times r_c$ , and long-range terms can be added separately

# Periodic graphs for crystals



Edges from the highlighted atom. Amber edges go to copies in neighbouring cells.



## Neighbours can be in the next cell

Atoms near the edge of the cell have neighbours that are copies of atoms on the other side.



## Each edge stores a cell offset

Edge  $(i, j, \mathbf{n})$  with  $\mathbf{n} \in \mathbb{Z}^3$ :  $r_{ij} = \|\mathbf{r}_j + n_1 \mathbf{a}_1 + n_2 \mathbf{a}_2 + n_3 \mathbf{a}_3 - \mathbf{r}_i\|$



## Small cells give multigraphs

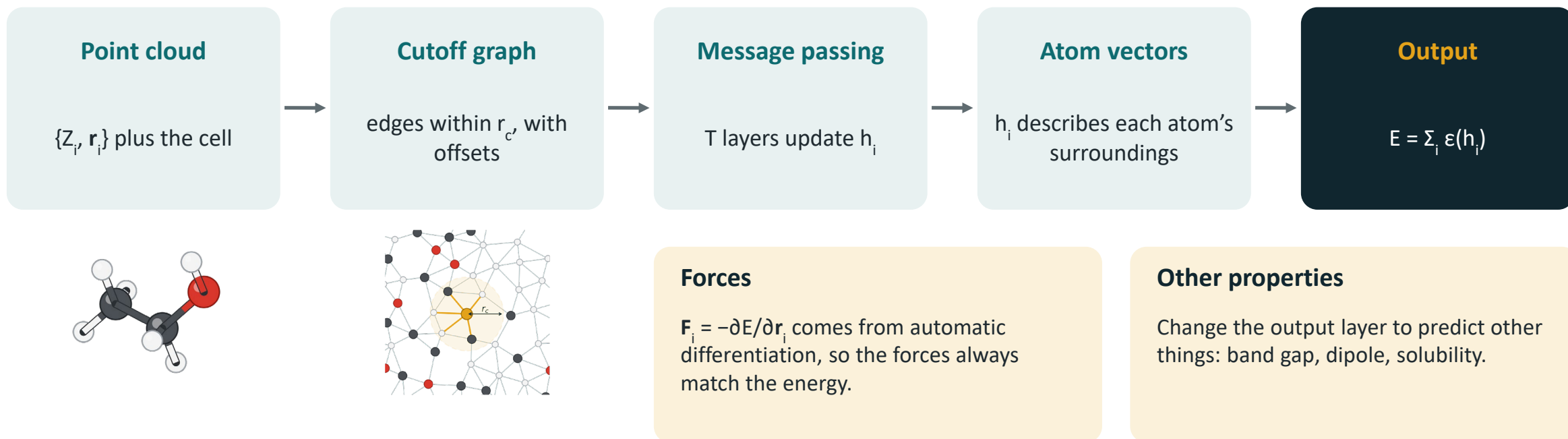
An atom can connect to its own copy, and the same pair can be connected more than once.



## Libraries do this for you

ASE's `neighbor_list` and `pymatgen` build these graphs, including the offsets.

# From point cloud to property



The energy is a **sum over atoms**, so a model trained on small structures also works on bigger ones. This is why GNN potentials can be used for large simulations.

# What a geometric GNN has to respect

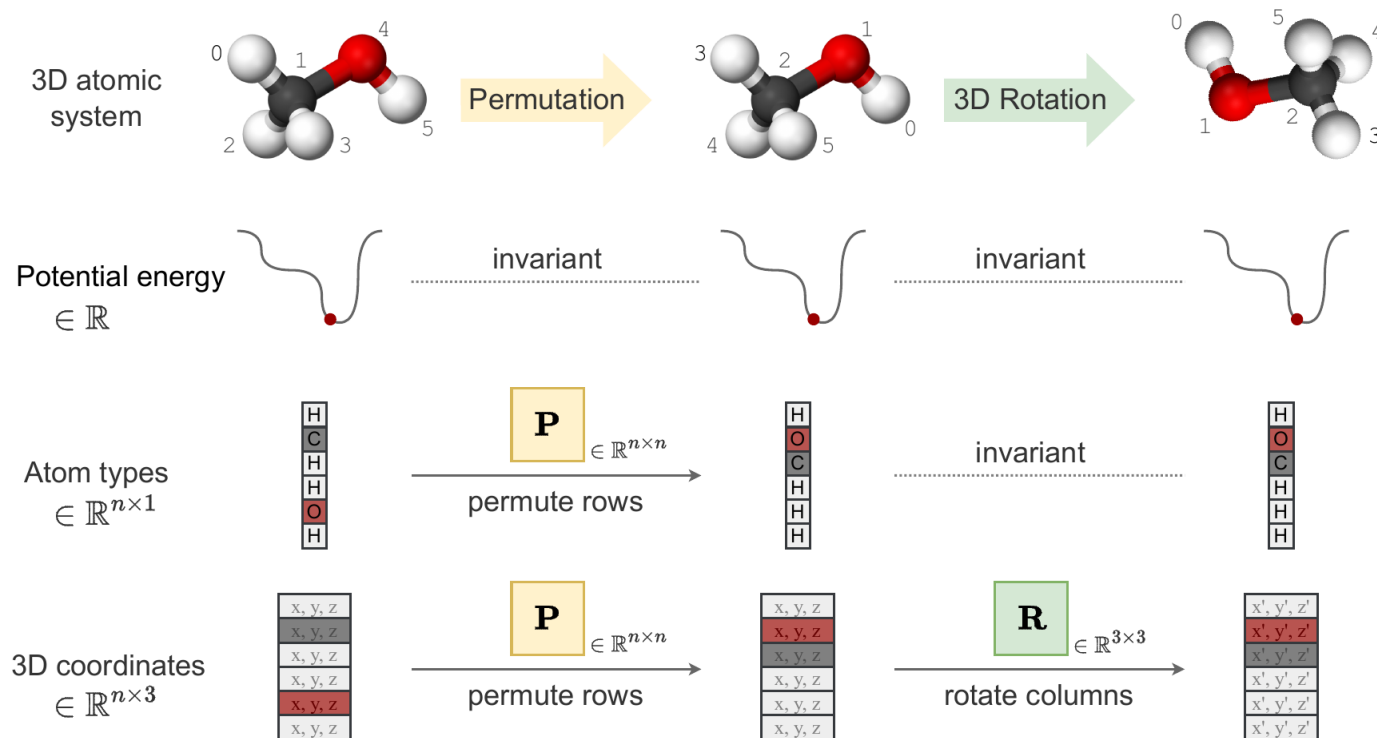


Figure 2 from Duval, Mathis, Joshi et al. (2023), arXiv:2312.07511

**Invariant:** stays the same. Energy, band gap, solubility.

**Equivariant:** rotates with the molecule. Coordinates, forces, dipole moment.

## What to notice

- **Permutation:** renumbering the atoms reorders the rows of atom types and coordinates together. The energy stays the same.
- **Rotation:** the coordinates change, the atom types don't, and the energy stays the same.
- **Translation** (not shown) works the same way.
- **Standard GNNs** only handle permutations. Geometric GNNs also handle rotations and translations.

# Geometric GNN families over time

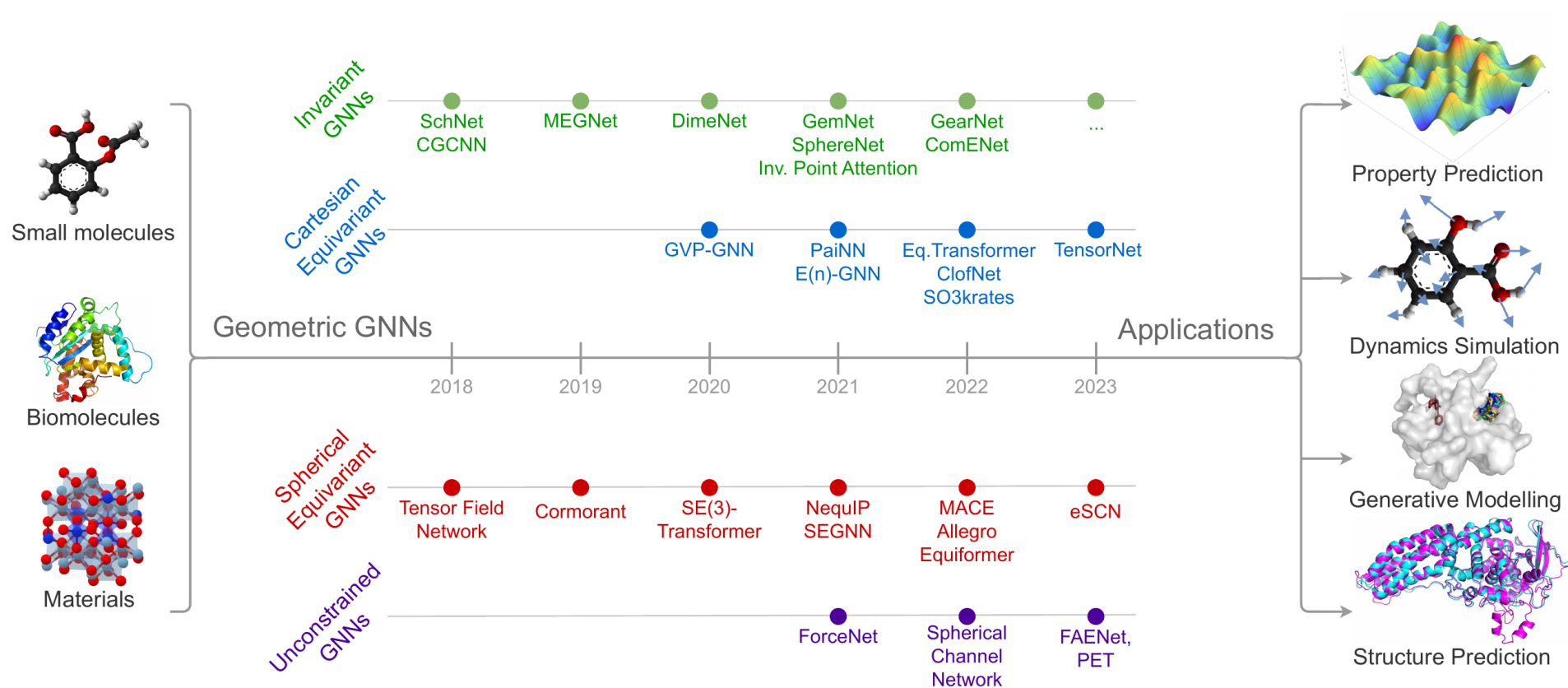


Figure 1 from Duval, Mathis, Joshi et al. (2023), arXiv:2312.07511

**Rows:** four families, sorted by what the features inside the network are

**Left to right:** 2018 to 2023: from invariant to equivariant models, and lately some unconstrained ones

**Left and right:** the same models are used for molecules, proteins and materials, and for four tasks

# Inside one layer of each family

## ● Invariant

*SchNet, DimeNet*

$$\mathbf{h}_i \leftarrow \mathbf{h}_i + \sum_{j \in \mathcal{N}(i)} \mathbf{h}_j \odot W(r_{ij})$$

Only distances (and angles in DimeNet) enter the filter  $W$ , so everything stays invariant. Simple and fast.

## ● Equivariant, Cartesian

*PaiNN, EGNN, GVP-GNN*

$$\Delta \vec{\mathbf{v}}_i = \sum_j \phi_v(\mathbf{s}_j, r_{ij}) \odot \vec{\mathbf{v}}_j + \phi_r(\mathbf{s}_j, r_{ij}) \hat{\mathbf{r}}_{ij}$$

$$\Delta \mathbf{s}_i = \phi_s(\mathbf{s}_i, \|\vec{\mathbf{v}}_i\|)$$

Vectors are only scaled by invariant numbers and added, so they rotate correctly. Their norms feed back into the scalars.

## ● Equivariant, spherical

*TFN, NequIP, MACE*

$$\mathbf{h}_i^{(l_3)} \leftarrow \sum_j R(r_{ij}) \left[ \mathbf{Y}^{(l_1)}(\hat{\mathbf{r}}_{ij}) \otimes \mathbf{h}_j^{(l_2)} \right]^{(l_3)}$$

Directions enter through spherical harmonics  $Y$ . Features of different order  $l$  are combined with Clebsch–Gordan tensor products.

## ● Unconstrained

*FAENet, PET*

$$\Phi(\mathbf{X}) = \frac{1}{|\mathcal{F}(\mathbf{X})|} \sum_{g \in \mathcal{F}(\mathbf{X})} g \cdot f_\theta(g^{-1} \cdot \mathbf{X})$$

Any network  $f$ . Symmetry comes from averaging over a few frames  $g$  (for example from PCA), or roughly from training on rotated copies.

In all four, the layer is still message passing. What changes is **the type of the features, and how they can be combined**.

# How expressive is each family?

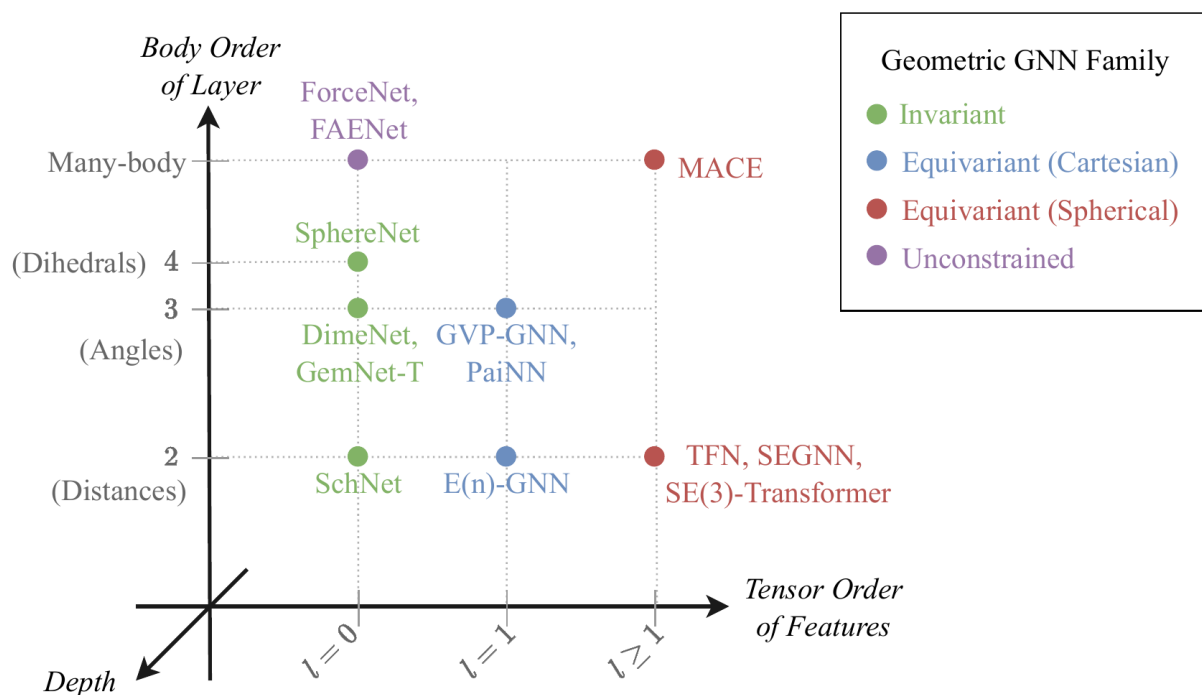


Figure 9 from Duval, Mathis, Joshi et al. (2023), arXiv:2312.07511

- 1 Body order:** how many atoms one message looks at together. 2 = distances (SchNet), 3 = angles (DimeNet), 4 = dihedrals (SphereNet), many-body (MACE).
  - 2 Tensor order:** what the features are.  $l = 0$  is scalars,  $l = 1$  is vectors, and spherical models also use  $l = 2$  and higher.
  - 3 Depth:** the number of layers, so how far information travels, roughly  $T \times r_c$ .
- **Unconstrained models** avoid this trade-off by not enforcing equivariance inside the layers.

**Link back:** the ACSF radial terms are 2-body and the angular terms 3-body. SOAP is 3-body as well.

**Question:** can a model that ignores reflections tell (R)- from (S)-thalidomide? When does that matter?

# Moving along the three axes

## 1 Body order

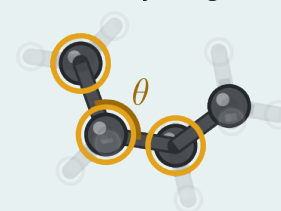
How many atoms one message looks at together: **2** = distance (SchNet), **3** = angle (DimeNet), **4** = dihedral (SphereNet).

MACE gets many-body terms cheaply: sum over the neighbours first, then multiply the sums.

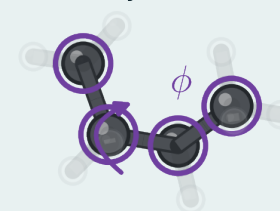
2-body: distance



3-body: angle



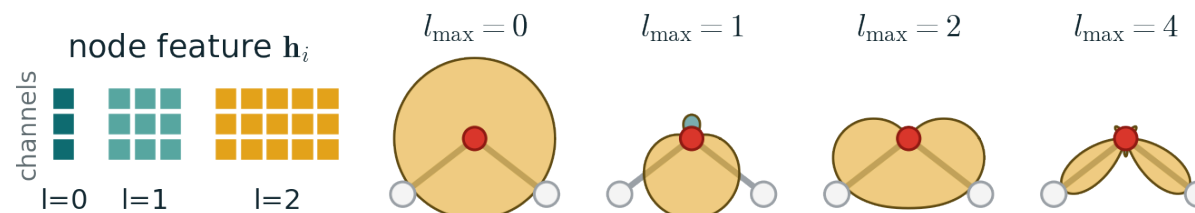
4-body: dihedral



## 2 Tensor order $l$

Each atom stores a stack of features. Per channel: **1** number for  $l = 0$ , **3** for  $l = 1$ , **5** for  $l = 2$ , so  $2l + 1$  each.

Together they describe a shape around the atom, like s, p and d orbitals. A higher  $l_{\max}$  resolves the two O-H directions more sharply.



## 3 Message iterations $T$

How far information travels from atom  $i$  (amber). Each iteration adds one hop, about one cutoff:  $T$  iterations reach about  $T \times r_c$ .

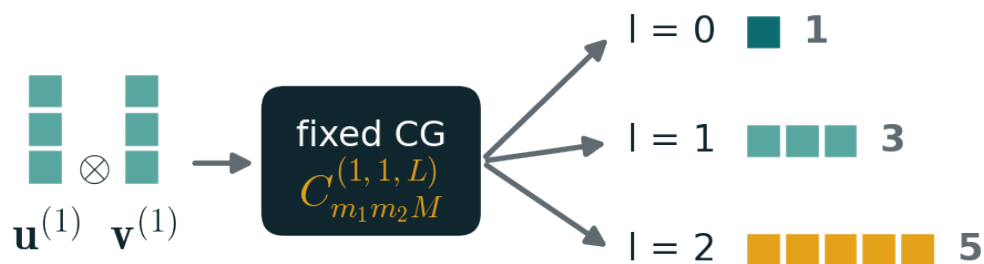
Many iterations squeeze far-away information through few edges (over-squashing).



**MACE's choice:** 4-body terms in every layer, moderate  $l$ , and only two iterations. That keeps it accurate and fast.

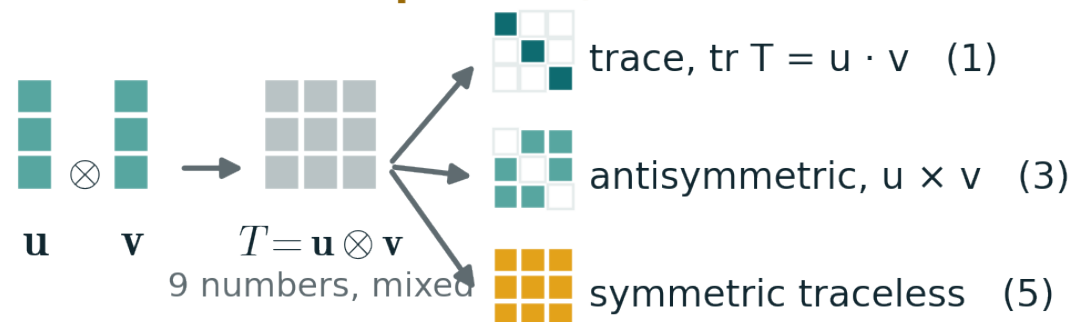
# Two ways to build higher-order equivariant features

## Spherical: Clebsch-Gordan coupling



Each output order comes out directly: 1 + 3 + 5 numbers, with one learnable weight per path (1, 1  $\rightarrow$  L).

## Cartesian: outer product, then contract



The same 9 numbers, split afterwards with contractions (trace, dot, matrix-vector product):  $T = I + A + S$ .

### Spherical basis *TFN, NequIP, MACE, Equiformer*

$$(u \otimes v)_{m_3}^{(l_3)} = \sum_{m_1, m_2} C_{m_1 m_2 m_3}^{(l_1, l_2, l_3)} u_{m_1}^{(l_1)} v_{m_2}^{(l_2)}$$

One learnable weight per path. Only  $2l + 1$  numbers per order, but products get expensive at high  $l$ .

### Cartesian basis *PaiNN, TensorNet*

$$s \mapsto s, \quad v \mapsto Rv, \quad T \mapsto RTR^\top$$

Plain matrix maths, fast on GPUs. But rank  $c$  has  $3^c$  numbers, so models usually stop at rank 2.

**Both give the same 1 + 3 + 5 numbers.** Spherical models keep the features split by order all the time. Cartesian models keep full tensors and split them only when needed.

# Four ways geometric GNNs are used



## Property prediction

cutoff graph

Structure in, a number out.

**Biomolecular examples:** binding affinity (PDBbind), toxicity, solubility, pKa



## Dynamics simulation

cutoff graph

The GNN gives the forces, and an MD engine moves the atoms.

**Biomolecular examples:** MACE-OFF, AI2BMD, SO3LR



## Generation and design

global graph

Learn a distribution, then sample new molecules or proteins.

**Biomolecular examples:** RFdiffusion, Chroma, DiffDock



## Structure prediction

global graph

Sequence in, 3D structure out.

**Biomolecular examples:** AlphaFold, RoseTTAFold, Boltz

**The paper's observation:** local tasks use cutoff graphs, while tasks that need one consistent global structure use fully connected graphs.

# What people compute from energy and forces

| Property                         | How it's computed                                              | Where it matters                           |
|----------------------------------|----------------------------------------------------------------|--------------------------------------------|
| Battery voltage                  | Energy difference between the charged and discharged electrode | Li-ion and sodium-ion batteries            |
| Ion mobility                     | Diffusion from MD, or migration barriers                       | Electrolytes, solid-state batteries        |
| Adsorption and reaction barriers | Energies of molecules on surfaces, and saddle points           | Catalysts for fuel cells and electrolyzers |
| Reaction thermodynamics          | Energy differences plus corrections from vibrations            | Synthesis, degradation, stability          |
| Phonons and thermal properties   | Second derivatives of the energy                               | Thermoelectrics, heat transport            |
| Band gap and light absorption    | Electronic structure, so it needs more than E and F            | Solar cells, LEDs, photocatalysis          |

Most of these need energies and forces **many, many times**. That's why a fast and accurate model is so useful.

# GNNs are used for much more than energies

from single molecules to industrial processes



## Single molecules

Solubility, toxicity,  
lipophilicity

Spectra

Reactivity and synthesis  
planning



## Molecules with proteins

Protein–ligand binding

Docking and structure  
prediction

Antibiotic discovery  
(Halicin, 2020)



## Mixtures and liquids

Solvation energies

Electrolyte recipes

Polymers and blends



## Crystals and surfaces

Formation energy, band  
gap

Catalysts

Battery materials



## Engineering scale

Reaction networks

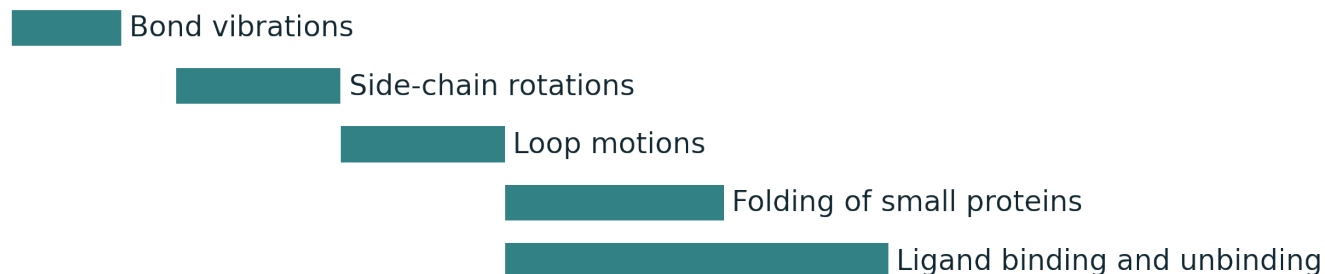
Process models

Product formulations

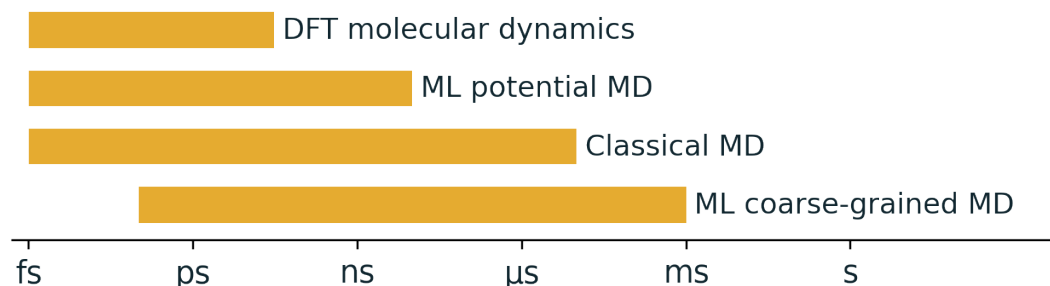
For simulations, the key model is the **ML interatomic potential**, which predicts energy and forces.

# Why biomolecules are hard to simulate

*What happens in a protein*



*What methods reach (roughly)*



- **Size:** a protein in water is  $10^4$  to  $10^6$  atoms.
- **Time:** an MD step is about 1 fs, but folding and binding take microseconds to seconds. That is  $10^9$  steps or more.
- **Environment:** water, ions and membranes matter, and electrostatic forces reach far.
- **Today's tools:** classical force fields (AMBER, CHARMM) are fast but fixed. DFT is accurate but only works for small systems.

GNNs can help by giving **more accurate energies**, by making **cheaper models**, or by **skipping the dynamics** altogether.

# What has changed since 2023, and what is still hard

## Since 2023

- **Much bigger data:** SPICE 2, and OMol25 (2025) with over 100 million DFT calculations, including biomolecules, electrolytes and metal complexes
- **General molecular models:** MACE-OFF, AIMNet2 (handles charged molecules), SO3LR, and UMA trained on OMol25
- **In standard tools:** ML potentials plug into OpenMM, ASE and LAMMPS, alone or as ML/MM

## Still hard

- **Long-range electrostatics** and charged groups, which a short cutoff can't see
- **Stable long MD:** a low force error does not guarantee a stable simulation
- **Speed:** still roughly 100 to 1000 times slower than a classical force field
- **Water and ions:** many biomolecular GNNs still leave them out

**Open question from the paper:** forces as  $-\nabla E$  give more stable MD, while predicting forces directly is faster. Which one to use is still debated.

# Notebook 1: one notebook, six parts

## 1 Molecules and crystals

Ethanol written four ways, the pairs of structures that some representations can't tell apart, and one crystal in three unit cells.

*RDKit · ASE · pymatgen*

## 2 Energy and forces

Relax ethanol and run MD, then scan the central bond of butane and find the gauche–anti energy difference.

*ASE · GFN2-xTB · MMFF*

## 3 Fingerprints and descriptors

Solubility with a random forest, random versus scaffold split; ACSF and SOAP; small distortion or real change?

*RDKit · scikit-learn · DScribe*

## 4 Building graphs

Cutoff graphs, edges that grow linearly, and periodic offsets in NaCl and copper.

*ASE · MACE*

## 5 A MACE potential

Train MACE on rMD17 ethanol in about two minutes, test it, check its symmetry, run MD, and look at what it learned.

*MACE · rMD17*

## 6 Equivariance with e3nn

Spherical harmonics around an atom, the rotation check, and Clebsch–Gordan versus Cartesian giving the same 1 + 3 + 5 numbers.

*e3nn*

**Start Part 5 first:** MACE trains while we go through Parts 3, 4 and 6 together. The data and a trained model come with the notebook.

# Reviews to start with

## Overviews

**Butler et al., Nature 559, 547 (2018).** *Machine learning for molecular and materials science.* A short, readable overview of the whole field.

**Wang et al., Nature 620, 47 (2023).** *Scientific discovery in the age of artificial intelligence.* Where AI fits into science more broadly.

**Keith et al., Chem. Rev. 121, 9816 (2021).** *Combining machine learning and computational chemistry for predictive insights into chemical systems.* A good bridge for non-chemists.

## Density functional theory

**Burke, J. Chem. Phys. 136, 150901 (2012).** *Perspective on density functional theory.* A clear, informal introduction.

**Jones, Rev. Mod. Phys. 87, 897 (2015).** *Density functional theory: its origins, rise to prominence, and future.* More history and theory (optional).

## Representations and descriptors

**Musil et al., Chem. Rev. 121, 9759 (2021).** *Physics-inspired structural representations for molecules and materials.* The main reference for ACSF, SOAP and their links to GNNs.

**Langer, Goeßmann, Rupp, npj Comput. Mater. 8, 41 (2022).** *Representations of molecules and materials for interpolation of quantum-mechanical simulations via machine learning.* Compares descriptors side by side.

**David et al., J. Cheminform. 12, 56 (2020).** *Molecular representations in AI-driven drug discovery: a review and practical guide.* SMILES, fingerprints and graphs, from the drug-discovery side.

**Wigh, Goodman, Lapkin, WIREs Comput. Mol. Sci. 12, e1603 (2022).** *A review of molecular representation in the age of machine learning.* SMILES, SELFIES, InChI, graphs and fingerprints.

**Huang, von Lilienfeld, Chem. Rev. 121, 10001 (2021).** *Ab initio machine learning in chemical compound space.* Datasets like QM9, and how representations affect data efficiency.

# Reviews on graph neural networks and equivariance

## GNNs for molecules and materials

**Reiser et al., Commun. Mater. 3, 93 (2022).** *Graph neural networks for materials science and chemistry.* A broad review from the Friederich group here at KIT.

**Atz, Grisoni, Schneider, Nat. Mach. Intell. 3, 1023 (2021).** *Geometric deep learning on molecular representations.* A short overview for molecules and drug design.

**Wieder et al., Drug Discov. Today Technol. 37, 1 (2020).** *A compact review of molecular property prediction with graph neural networks.* Message passing on 2D molecular graphs.

**Choudhary et al., npj Comput. Mater. 8, 59 (2022).** *Recent advances and applications of deep learning methods in materials science.* Covers periodic GNNs such as CGCNN, MEGNet and ALIGNN.

## Geometric GNNs and equivariance

**Duval, Mathis, Joshi et al., arXiv:2312.07511 (2023).** *A hitchhiker's guide to geometric GNNs for 3D atomic systems.* The paper behind Figures 1, 2 and 9.

**Bronstein, Bruna, Cohen, Veličković, arXiv:2104.13478 (2021).** *Geometric deep learning: grids, groups, graphs, geodesics, and gauges.* The symmetry view of deep learning. Chapters 3 to 5 are enough.

**Gerken et al., Artif. Intell. Rev. 56, 14605 (2023).** *Geometric deep learning and equivariant neural networks.* Group representations and equivariant layers, written for mathematicians.

**Geiger, Smidt, arXiv:2207.09453 (2022).** *e3nn: Euclidean neural networks.* A practical guide to spherical harmonics and Clebsch–Gordan products.

**Zhang et al., arXiv:2307.08423 (2023).** *Artificial intelligence for science in quantum, atomistic, and continuum systems.* Very long; use it like a reference book.

# Reviews on ML potentials and applications

## ML interatomic potentials

**Behler, Chem. Rev. 121, 10037 (2021).** *Four generations of high-dimensional neural network potentials.* From ACSF-based networks to message passing.

**Unke et al., Chem. Rev. 121, 10142 (2021).** *Machine learning force fields.* Practical: data, training, testing and pitfalls.

**Deringer et al., Chem. Rev. 121, 10073 (2021).** *Gaussian process regression for materials and molecules.* The SOAP and GAP side.

**Batatia et al., Nat. Mach. Intell. 7, 56 (2025).** *The design space of  $E(3)$ -equivariant atom-centred interatomic potentials.* Puts ACE, NequIP, MACE and Allegro in one framework.

**Thiemann et al., J. Phys.: Condens. Matter 37, 073002 (2025).** *An introduction to machine learning potentials for atomistic simulations.* A step-by-step tutorial for newcomers.

## Biomolecules and drug discovery

**Noé, Tkatchenko, Müller, Clementi, Annu. Rev. Phys. Chem. 71, 361 (2020).** *Machine learning for molecular simulation.* Potentials, coarse-graining and sampling.

**Durumeric et al., Curr. Opin. Struct. Biol. 79, 102533 (2023).** *Machine learned coarse-grained protein force-fields: are we there yet?* A short, recent review for proteins.

**Vamathevan et al., Nat. Rev. Drug Discov. 18, 463 (2019).** *Applications of machine learning in drug discovery and development.* The wider drug-discovery context.

## Materials and benchmarks

**Schmidt et al., npj Comput. Mater. 5, 83 (2019).** *Recent advances and applications of machine learning in solid-state materials science.* Databases, descriptors and applications.

**Dunn et al., npj Comput. Mater. 6, 138 (2020).** *Benchmarking materials property prediction methods: the Matbench test set and Automatminer reference algorithm.* How to compare models fairly.