UNIVERSITEIT VAN AMSTERDAM

Machine learning for molecular simulations: Representations and Symmetries

Tristan Bereau
Van 't Hoff Institute for Molecular Sciences & Informatics Institute
University of Amsterdam

Multiscale simulations

time

$$E_{\text{bend}} = \int_{\text{membrane}} \mathrm{d}A \left\{ \frac{1}{2}\kappa K^2 - \bar{\kappa} K_{\text{G}} \right\}$$

Helfrich, *Z. Naturforsch. C* **28** (1973)

$$\mathbf{F} = m\mathbf{a}$$

$$\mathscr{H}\psi = E\psi$$

Continuum Mechanics

Coarse-Grained Molecular Dynamics

All-Atom Molecular Dynamics

Quantum Mechanics

10 ns

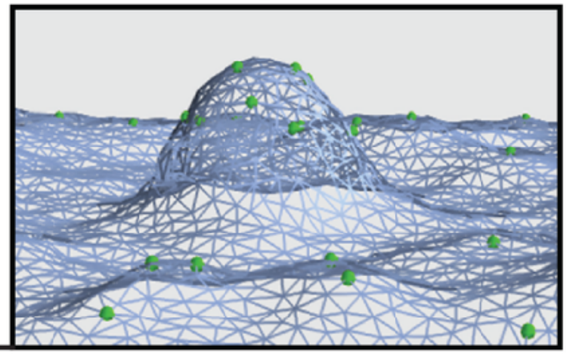
1 ns

1 ps

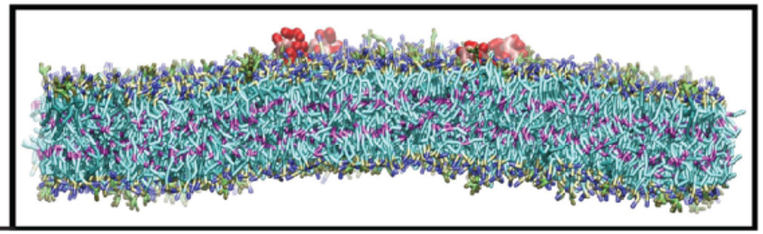
1 Å 1 nm 10 nm 100 nm 1 µm 10 µm

length

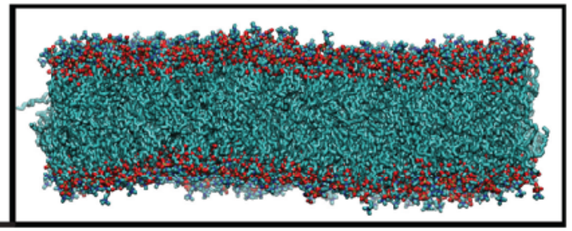
Vesicle budding



Coarse-grained peptides and membrane

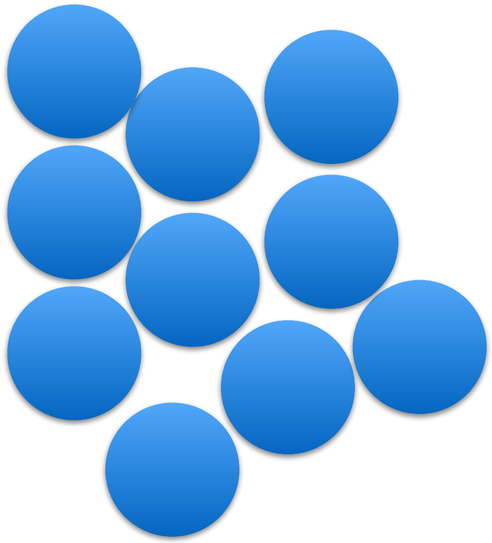


All-atom lipid bilayer



Bradley and Radhakrishnan, *Polymers* **5** (2013)

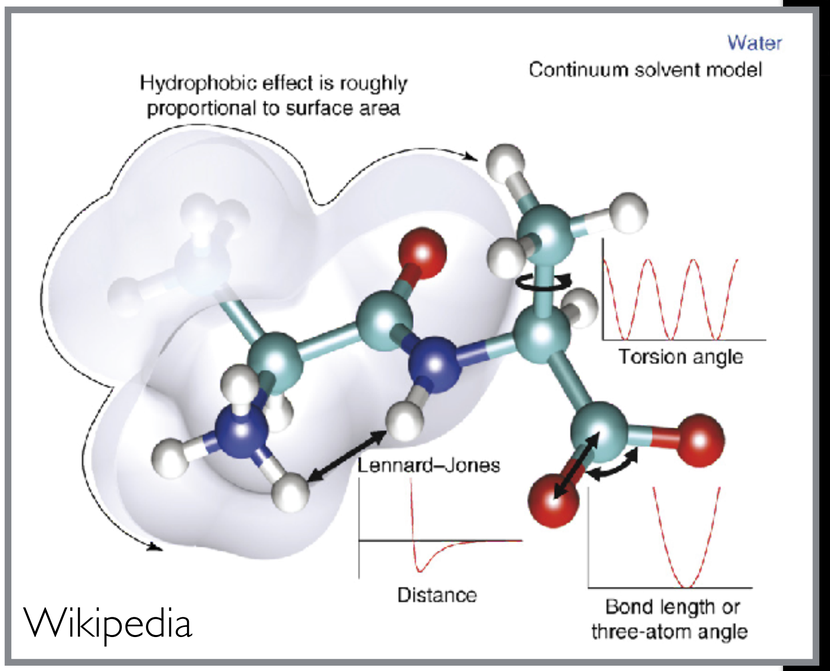
Molecular dynamics



Numerically integrate particle positions

$$\mathbf{F} = m\mathbf{a}$$

Specify interparticle forces: “force field”



integration time step

Timescales of interest

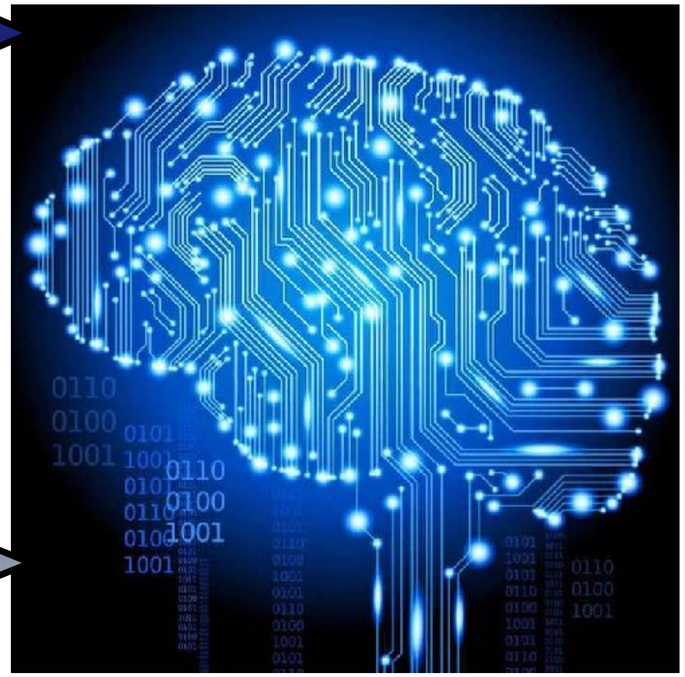
Emergent complexity

fs ps ns μs ms s

Data science

Machine learning

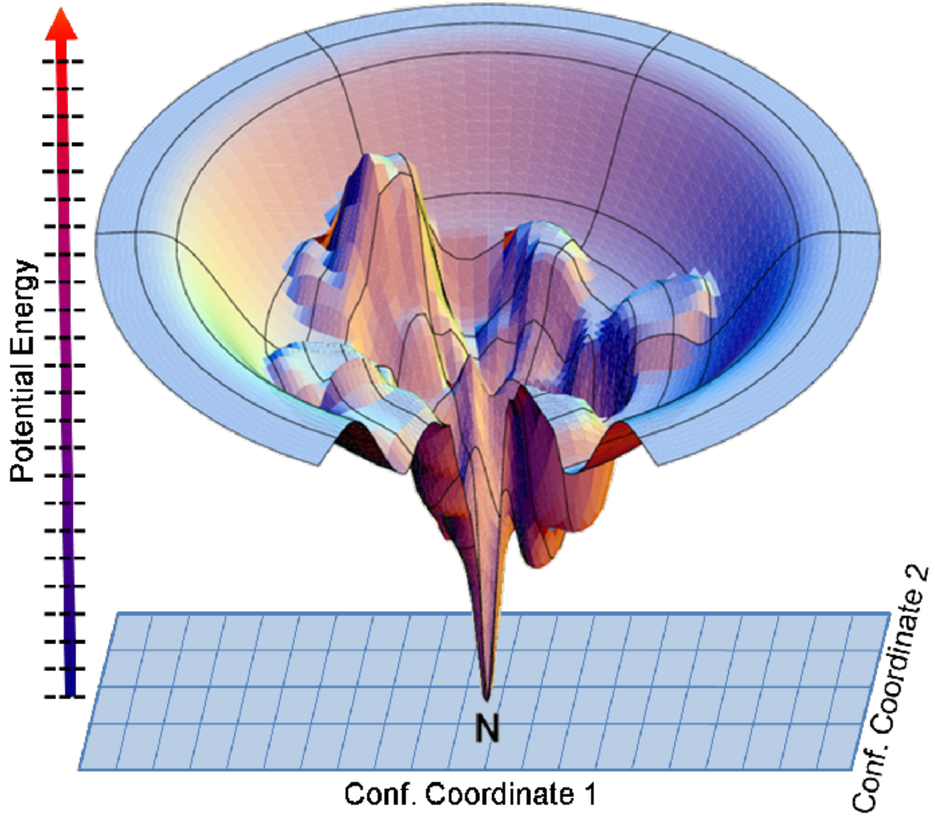
Databa



Links to machine learning

Interpolation of a high-dimensional function

Potential energy surface



Can we build a more accurate PES?

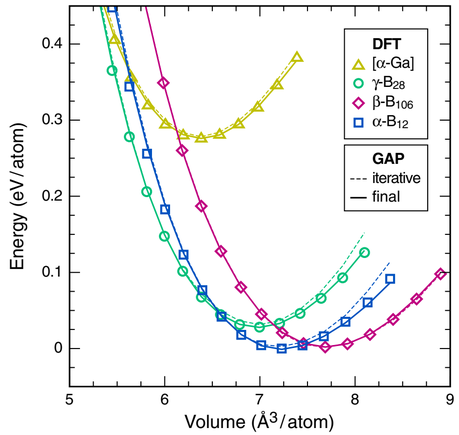
Can we **easily** build an accurate PES?

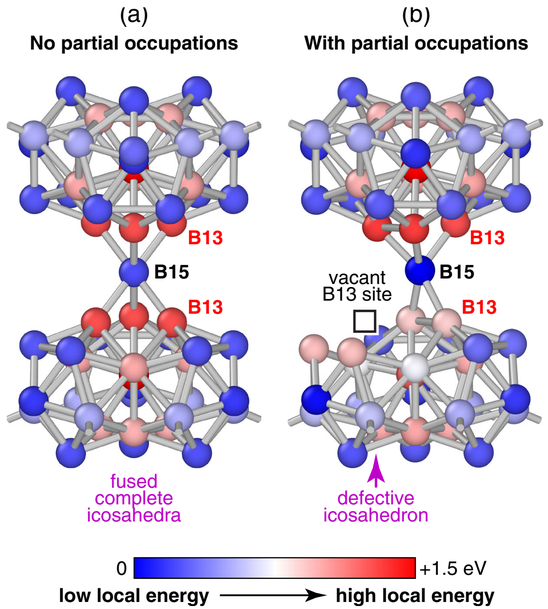
Can we make the numerical integration faster and/or more efficient?

…

Impressive developments in ML-based potentials

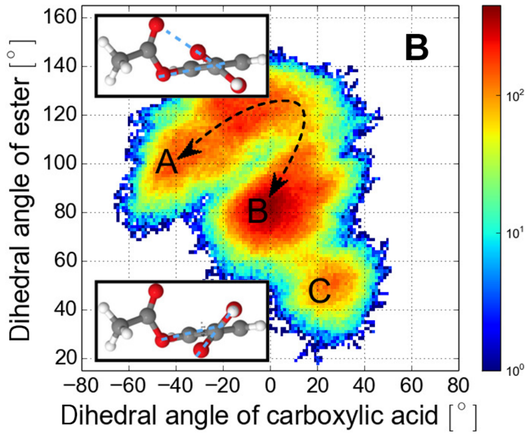
Elemental Boron



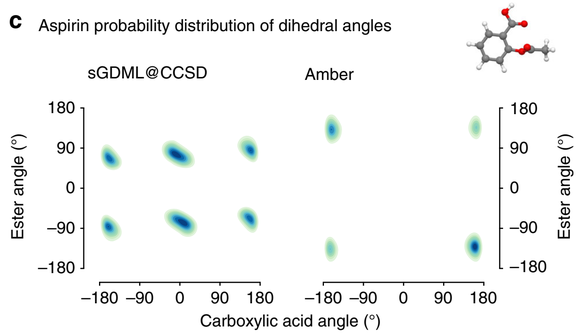


Deringer, Pickard, Csányi, *PRL* **120** (2018)

Small molecules

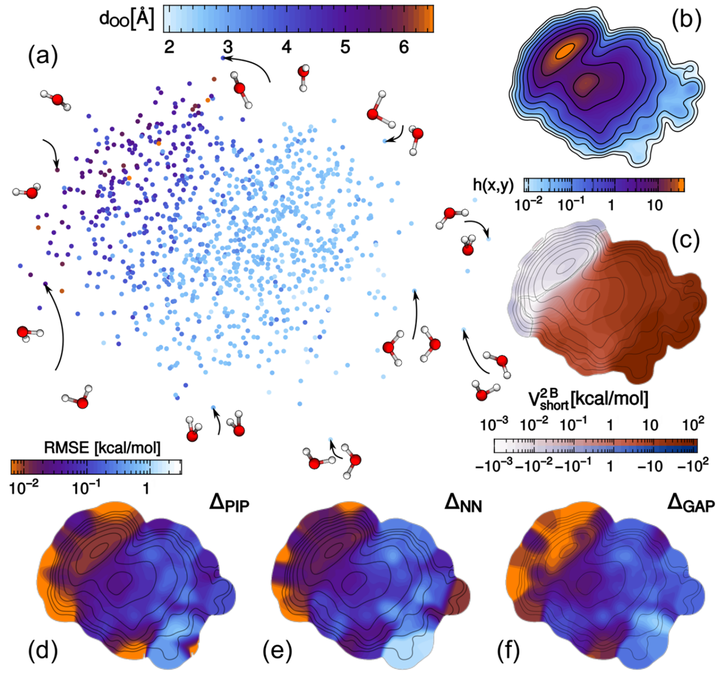


Chmiela et al., *Science Adv.* **3** (2017)



Chmiela et al., *Nat. Comm.* **9** (2018)

Water



Nguyen et al., *JPC* **148** (2018)

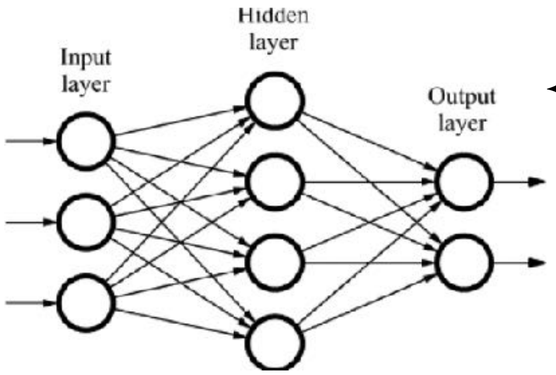
Kernel methods are vintage

Kernel

$$\mathbf{K}\alpha = p$$

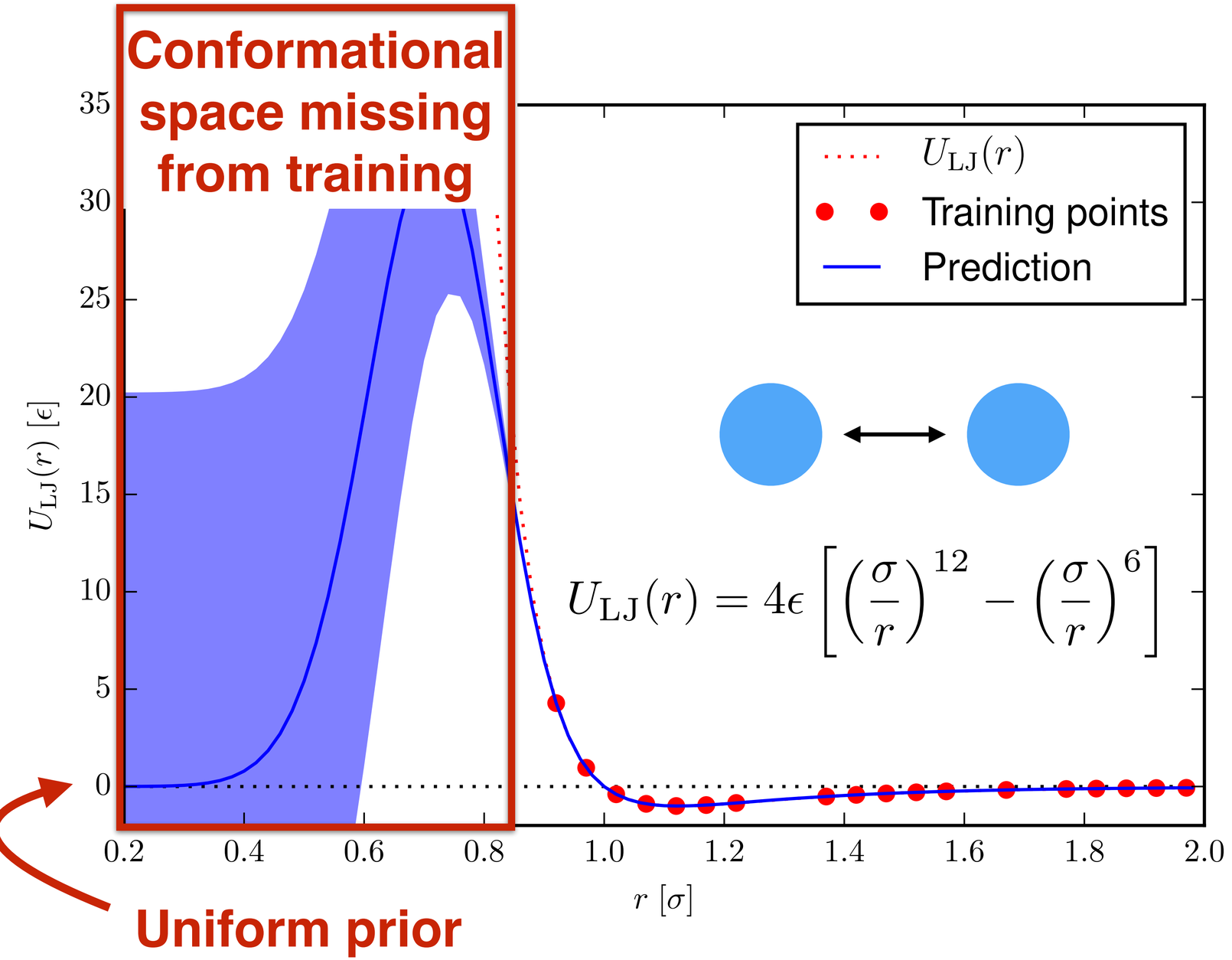
- needs a representation
- linear algebra
- can be efficient with small data

Deep learning



- learns the representation
- complex mathematical structure
- data hungry

Extrapolation in machine learning



Define kernel

$$K(r, r') = \exp\left(-\frac{(r - r')^2}{2\sigma^2}\right)$$

Make a prediction:

$$U(r) = \sum_i \alpha_i K(r_i^*, r)$$

Predicted energy

Training points

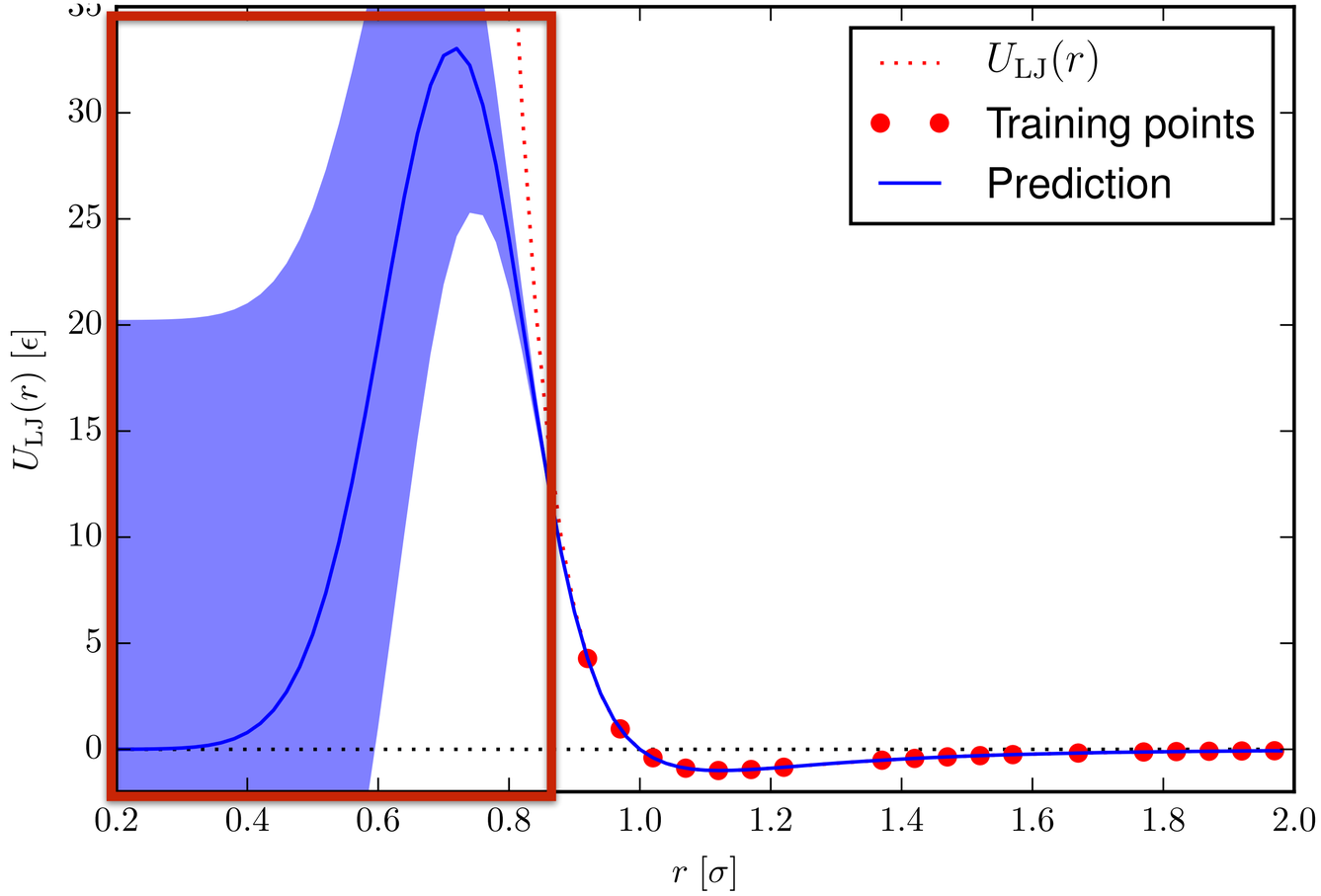
Query sample

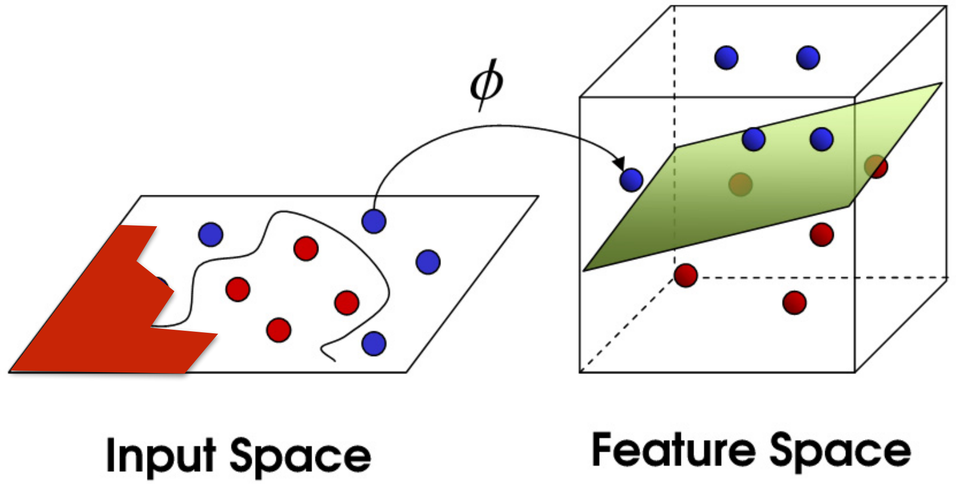
Train your model:

$$\alpha = (K + \lambda \mathbb{I})^{-1} U$$

Linking conformational and interpolation spaces

Conformational space missing from training

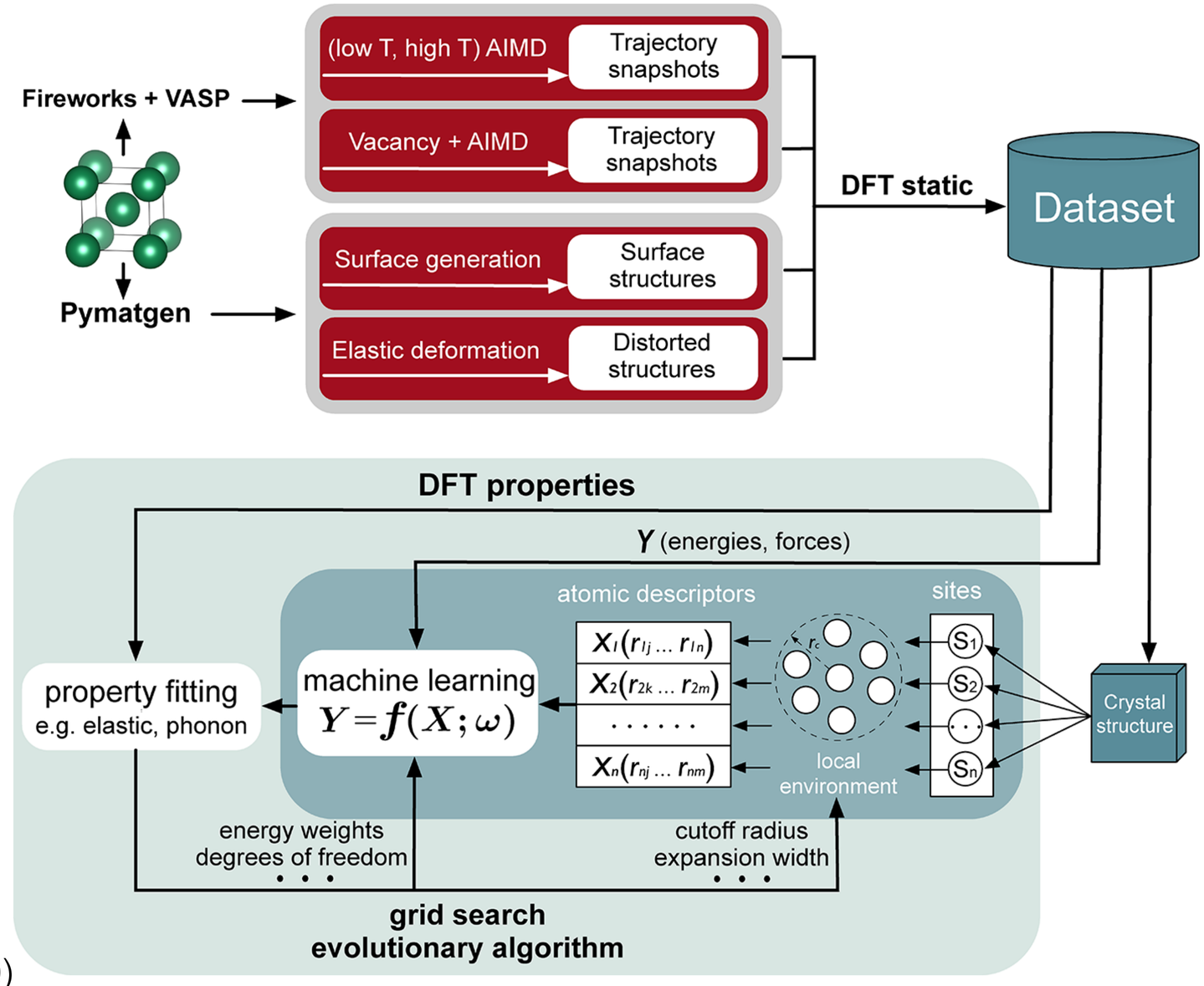




ML training set size is limited (kernels!)

Use physics to reduce the interpolation space

ML-PES workflow



Zuo et al., *JPCA* **124** (2020)

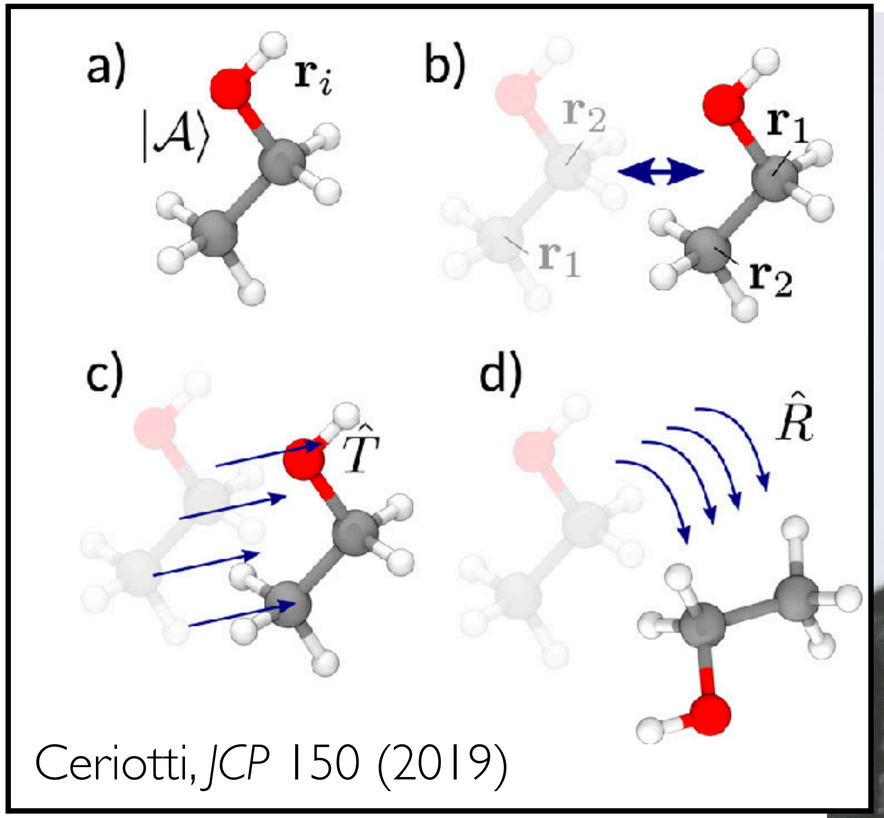
Symmetries and conservation laws

Noether's theorem

To every differentiable symmetry generated by local actions there corresponds a conserved quantity

3 examples:

- Translational symmetry: **Linear momentum conservation**
- Rotational symmetry: **Angular momentum conservation**
- Time translation: **Energy conservation**

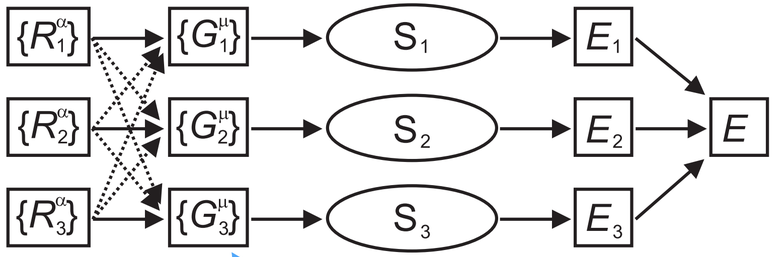


Ceriotti, *JCP* 150 (2019)

Encoding symmetries in the representation

Translational and rotational symmetries

Behler-Parrinello



Cartesian coordinates

Symmetry functions

$$G_i^1 = \sum_{j \neq i}^{\text{all}} e^{-\eta(R_{ij}-R_s)^2} f_c(R_{ij})$$

Distances

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

Angles

Behler & Parrinello, *Phys Rev Lett* **98** (2007)

Coulomb matrix

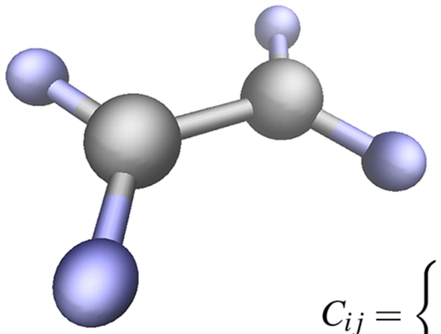
$$C_{ij} = \begin{cases} \frac{1}{2} Z_i^{2.4} & \forall \ i = j \\ \frac{Z_i Z_j}{|\mathbf{R}_i - \mathbf{R}_j|} & \forall \ i \neq j \end{cases}$$

Distances

Rupp, Tkatchenko, Müller, von Lilienfeld, *Phys Rev Lett*, **108** (2012)

Representation: the Coulomb matrix

Symmetries of the representation should emulate symmetries of the system



$$\longrightarrow \quad \mathbf{C} =$$

	H	H	C	C	H	H
H	0.5	0.3	2.9	1.5	0.2	0.2
H	0.3	0.5	2.9	1.5	0.2	0.2
C	2.9	2.9	36.9	14.3	1.5	1.5
C	1.5	1.5	14.3	36.9	2.9	2.9
H	0.2	0.2	1.5	2.9	0.5	0.3
H	0.2	0.2	1.5	2.9	0.3	0.5

$$C_{ij} = \begin{cases} 0.5\, Z_i^{2.4} & \forall i = j \\ \frac{Z_i Z_j}{|\mathbf{R}_i - \mathbf{R}_j|} & \forall i \neq j. \end{cases}$$

1. Translation
2. Rotations
3. Mirror reflection

~ Coulomb's law $E = \dfrac{q_i q_j}{|\mathbf{r}_i - \mathbf{r}_j|}$

Problems:

1. **Dimensionality from # atoms**
2. **Ordering of the atoms**

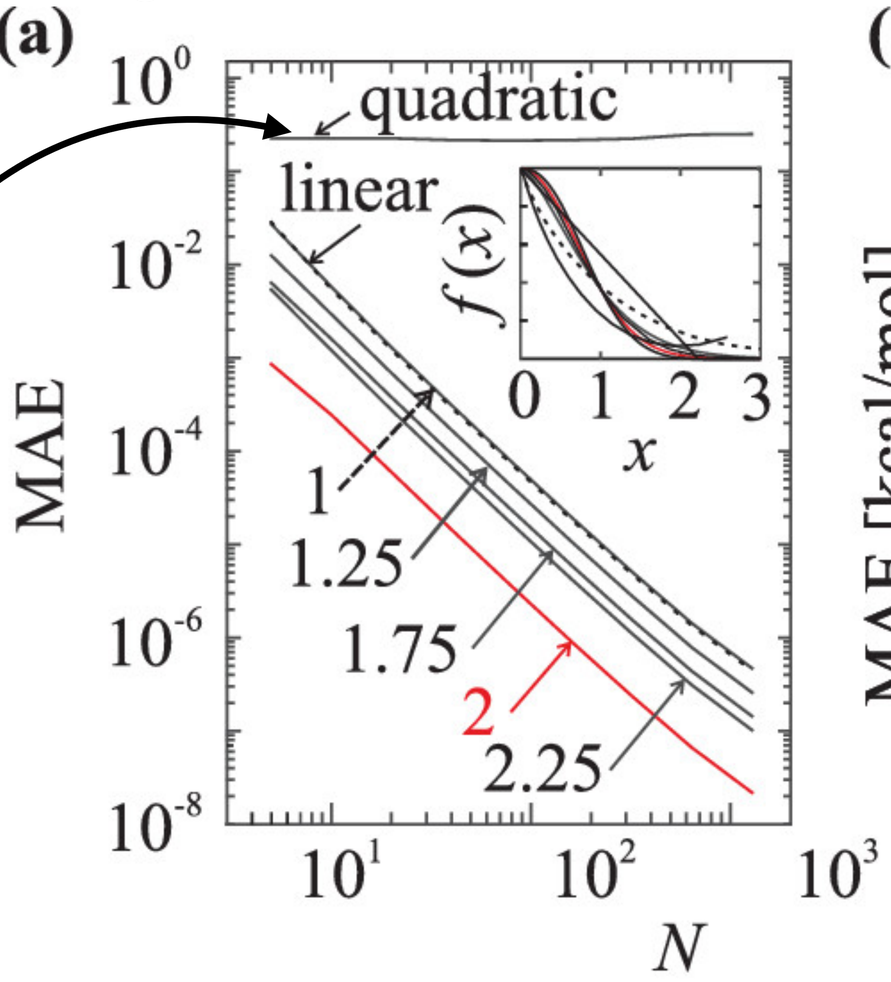
Hansen et al., *J Chem Theory Comput*, **9** (2013)

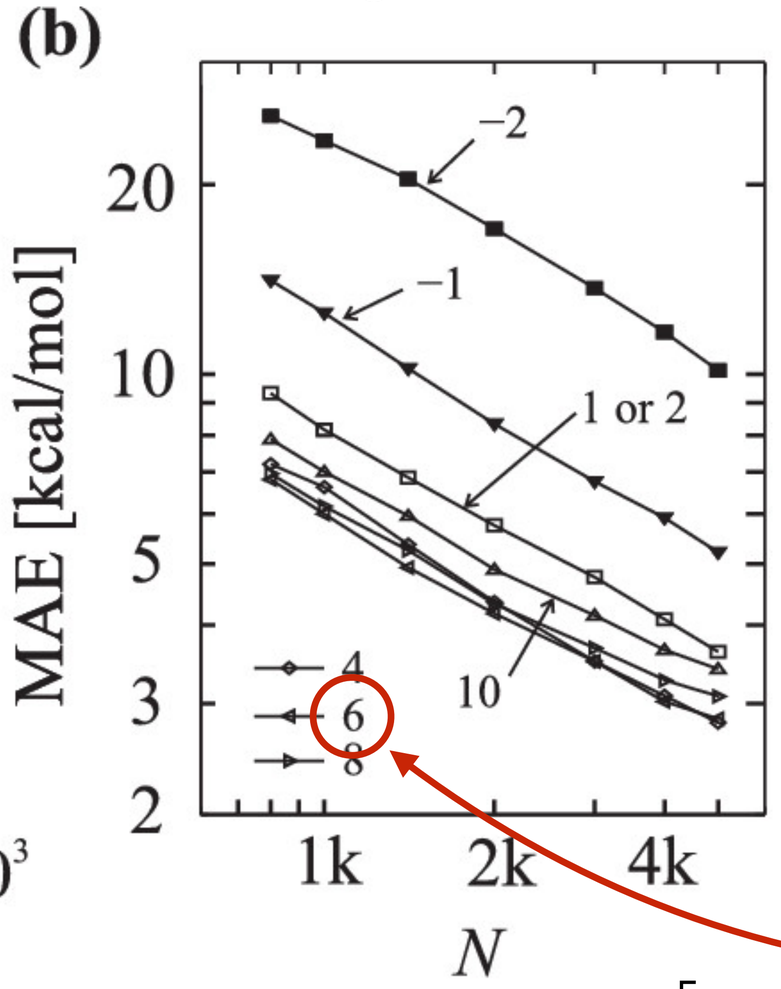
Optimizing the representation links to the physics

Learning a Gaussian function

Learning atomization energies

Non-unique representation!





Empirically test for relevant physics

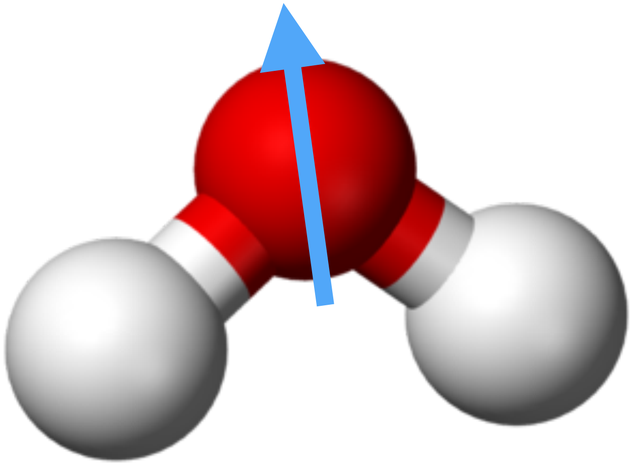
Huang and von Lilienfeld, *J Chem Phys* **145** (2016)

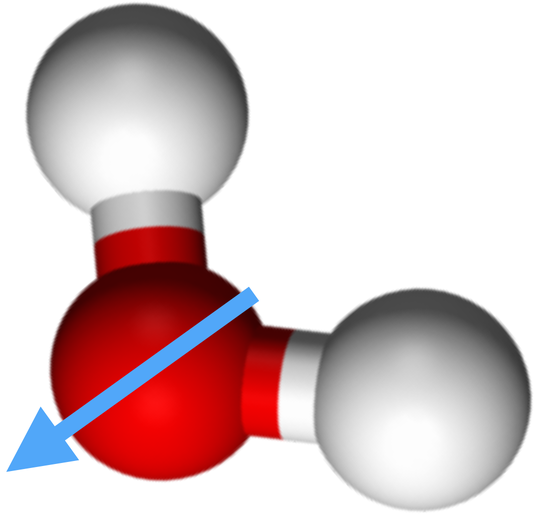
$$U_{\mathrm{LJ}}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

Encoding symmetries in the ML model

Invariant vs. covariant properties

Tensorial property (e.g., dipole moment, force) **rotates** with the sample





"Build kernel so as to encode the rotational properties of the target property"

Glielmo, Sollich, De Vita, *Phys Rev B* **95** (2017)

Covariant kernels

Encode rotational properties of the target property in the **kernel**

$$\hat{\mathbf{f}}(\mathcal{S}\rho \mid \mathcal{D}) = \mathbf{S}\hat{\mathbf{f}}(\rho \mid \mathcal{D})$$

Force prediction

Descriptor

Training data

Transformation (rotation/inversion)

"Transform the configuration, and the prediction transforms with it"

Glielmo, Sollich, De Vita, *Phys Rev B* **95** (2017)

Covariant kernels

$$\mathbf{K}(\mathcal{S}\rho, \mathcal{S}'\rho') = \mathbf{S}\mathbf{K}(\rho, \rho')\mathbf{S}'^{\mathrm{T}}$$

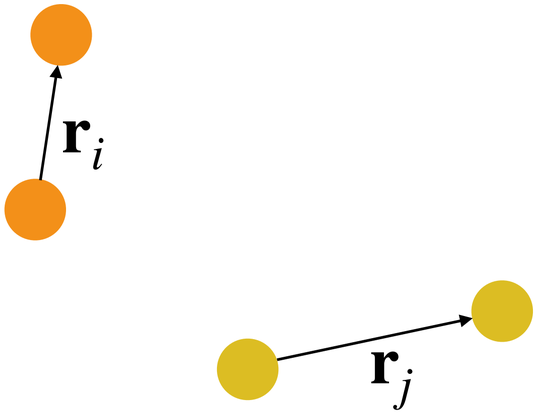
Kernel

Configurations

Transformations (rotation/inversion)

$$\mathbf{K}(\rho, \rho') = \int d\mathcal{R}\, \mathbf{R} k_b(\rho, \mathcal{R}\rho')$$

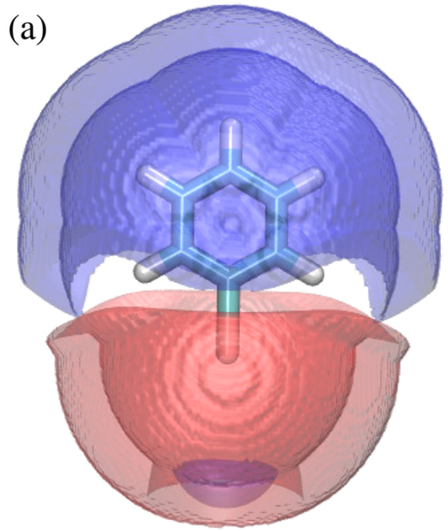
$$\mathbf{K}^{\mu}(\rho, \rho') = \frac{1}{L} \sum_{ij} \phi(r_i, r_j) \mathbf{r}_i \otimes \mathbf{r}_j'^{\mathrm{T}}$$

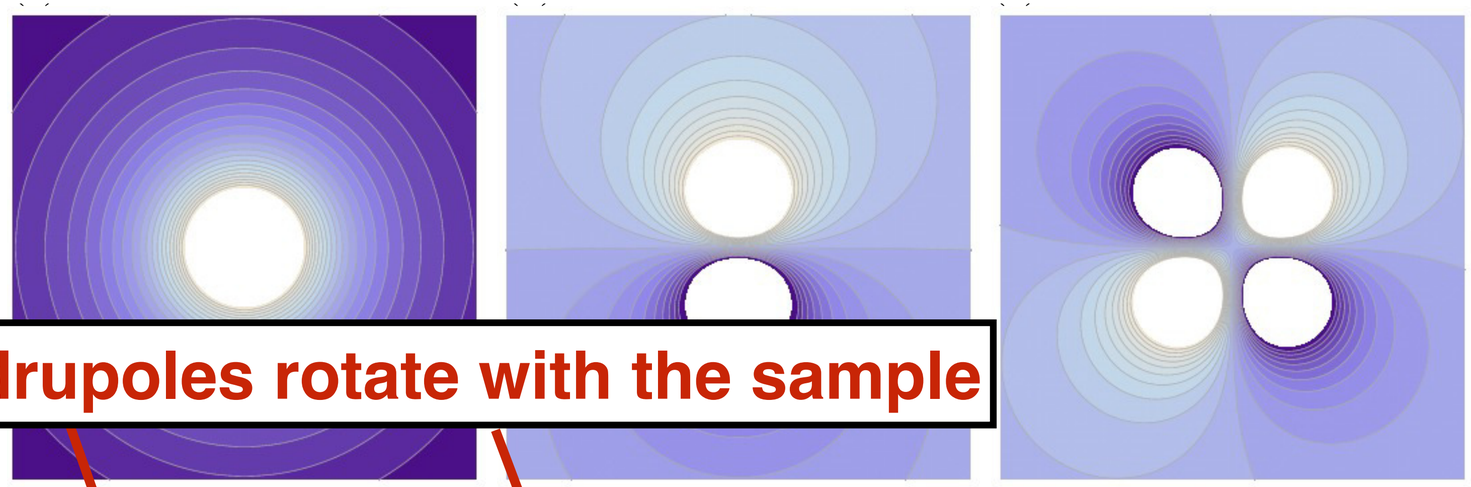


Glielmo, Sollich, De Vita, *Phys Rev B* **95** (2017)

Static multipole electrostatics

$$V_{\mathrm{Coulomb}}(r) = \frac{1}{4\pi\epsilon_0}\frac{q_i q_j}{r}$$



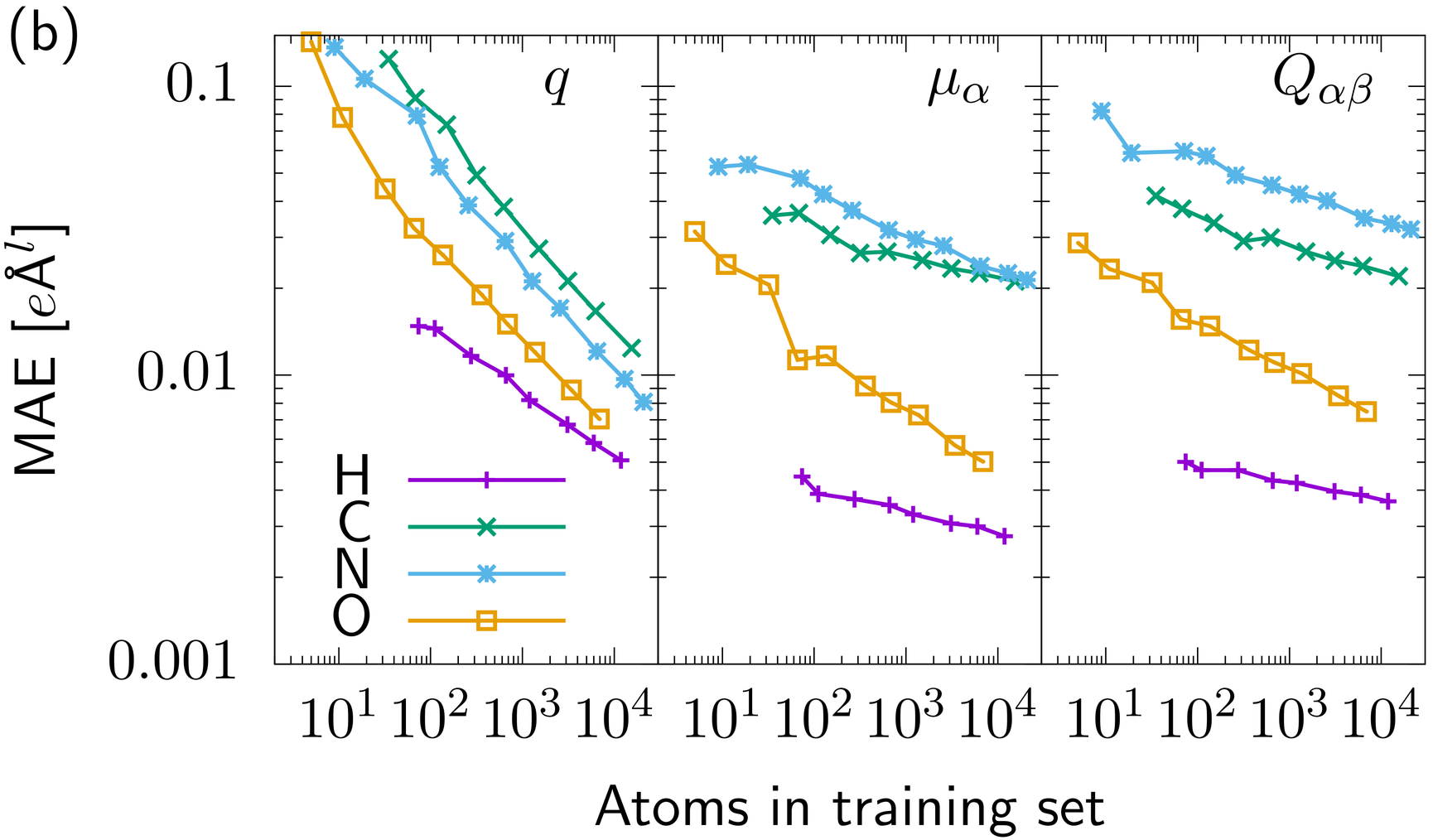


dipoles, quadrupoles rotate with the sample

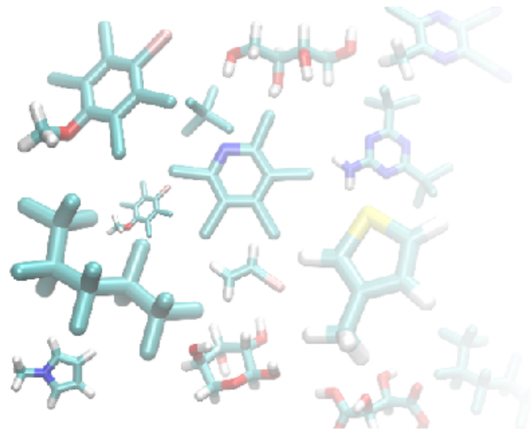
$$4\pi\varepsilon_0\Phi(\mathbf{r}) = \frac{q}{R} + \frac{\mu_\alpha R_\alpha}{R^3} + \frac{1}{3}\frac{\Theta_{\alpha\beta}\left(3R_\alpha R_\beta - R^2\delta_{\alpha\beta}\right)}{R^5} + \ldots$$

Stone, *The Theory of Intermolecular Forces*
Bereau and Meuwly, *Many-Body Effects and Electrostatics in Biomolecules*

Multipoles: Learning curves



Easier to learn H,O than C,N



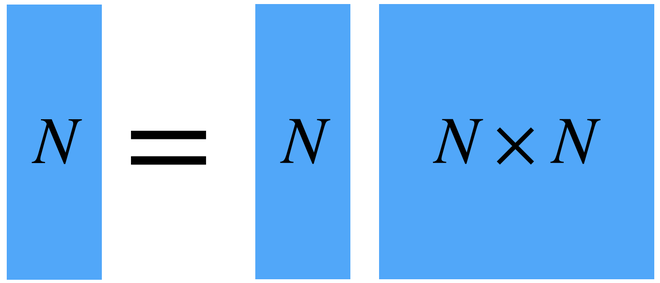
Bereau, DiStasio Jr., Tkatchenko, von Lilienfeld, *JCP* **148**, 241706 (2018)

Energy conservation

Learning a vector field: matrix-valued kernels

Up to now: learning a *scalar* field $f : \mathbb{R}^n \to \mathbb{R}$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$

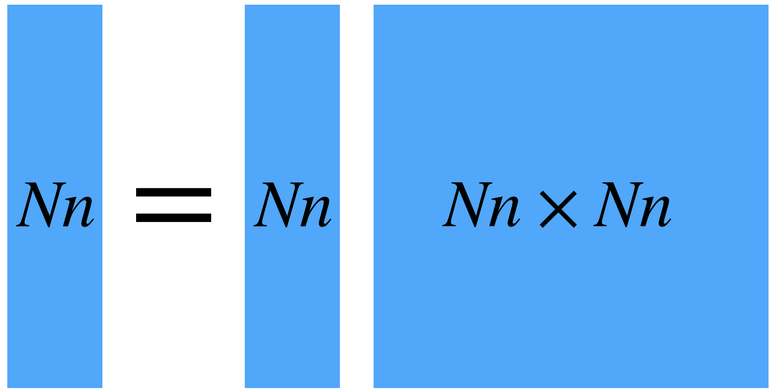


$$\alpha_i \in \mathbb{R}$$

$$K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$$

Learning a vector field $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$



$$\alpha_i \in \mathbb{R}^n$$

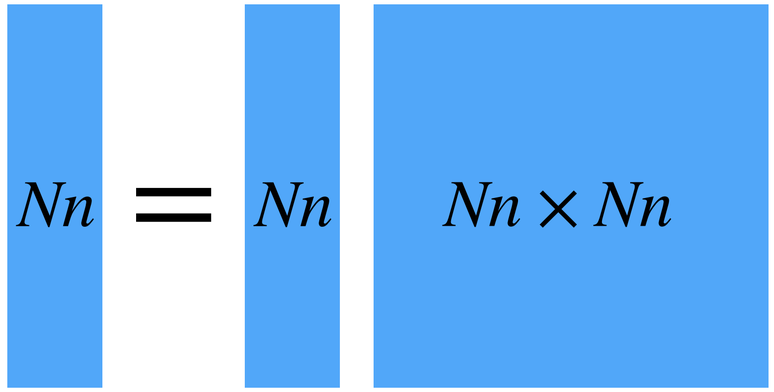
$$K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{n \times n}$$

Macêdo and Castro, *Learning divergence-free and curl-free vector fields with matrix-valued kernels, IMPA* (2010)
Micchelli and M. Pontil. *Neural Computation*, 17(1):177–204, 2005.

Matrix-valued kernel

Learning a vector field $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$



$$\alpha_i \in \mathbb{R}^n$$

$$K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$$

By default: components learned *independently*. **No prior on the vector field**

Time invariance leads to an **energy-conserving** force field (curl-free):

$$\nabla \times \mathbf{f} = 0$$

Design matrix-valued kernel that is *also* curl free.

Enforcing structure onto the vector field: Matrix-valued radial basis functions

Recall the (translation-invariant) kernel:

$$K(\mathbf{r}, \mathbf{r}') = \exp\left(-\frac{\|\mathbf{r}-\mathbf{r}'\|^2}{2\sigma^2}\right) = \varphi\left(\|\mathbf{r}-\mathbf{r}'\|\right)$$

scalar RBF $\Phi : \mathbb{R}^n \to \mathbb{R}$

Construct matrix-valued RBF $\Phi : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ from a scalar RBF $\phi(\mathbf{x}) = \varphi(\|\mathbf{x}\|)$

Apply linear differential operator: $\Phi(\mathbf{x}) := (\mathscr{L}\phi)(\mathbf{x})$

Example: curl-free $$(H\phi)_{ij} := \frac{\partial^2 \phi}{\partial x_i \partial x_j}$$

E. Fuselier. PhD thesis, Texas A&M University (2006)

Vector fields: curl-free and divergence-free

divergence-free

$$\Phi_{\mathrm{df}}(\mathbf{x}) = (H\phi)(\mathbf{x}) - \mathrm{Tr}\{(H\phi)(\mathbf{x})\} \cdot \mathbb{I}$$

curl-free

$$\Phi_{\mathrm{cf}}(\mathbf{x}) = -(H\phi)(\mathbf{x})$$

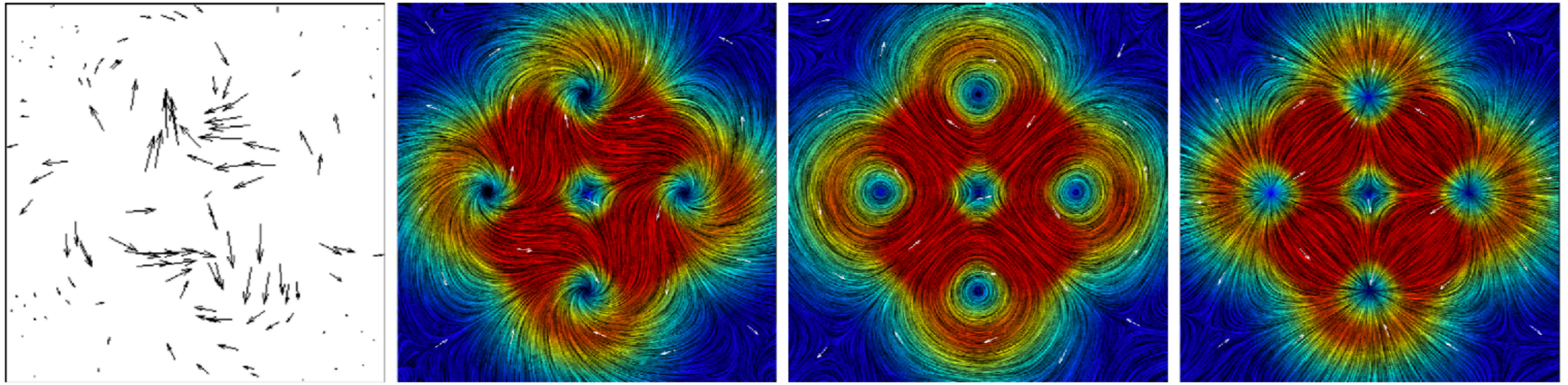
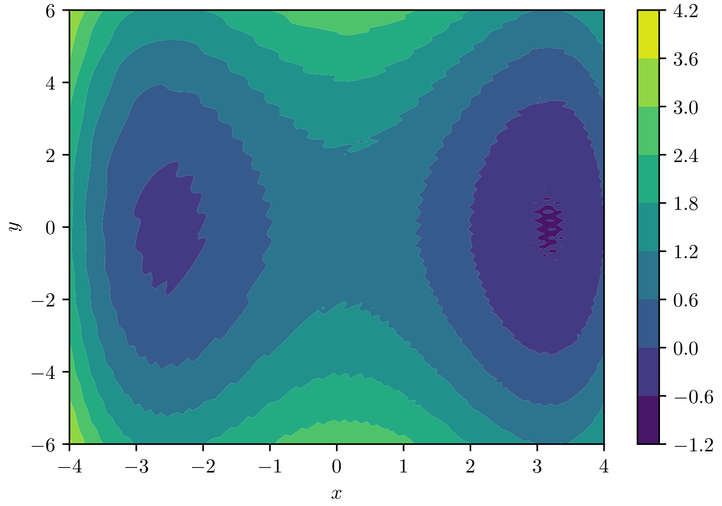


Figure 1. Learning a vector field decomposition: samples, learned field, divergence- and curl-free parts.

Macêdo and Castro, *Learning divergence-free and curl-free vector fields with matrix-valued kernels, IMPA* (2010)

Kernel learning of a 2D potential

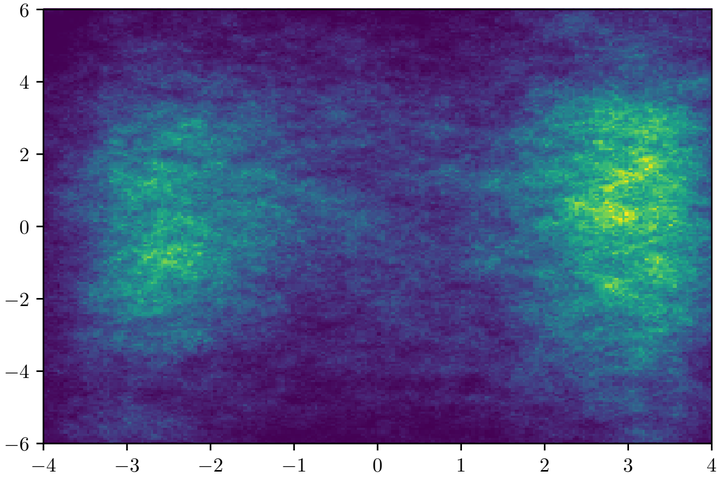
Potential



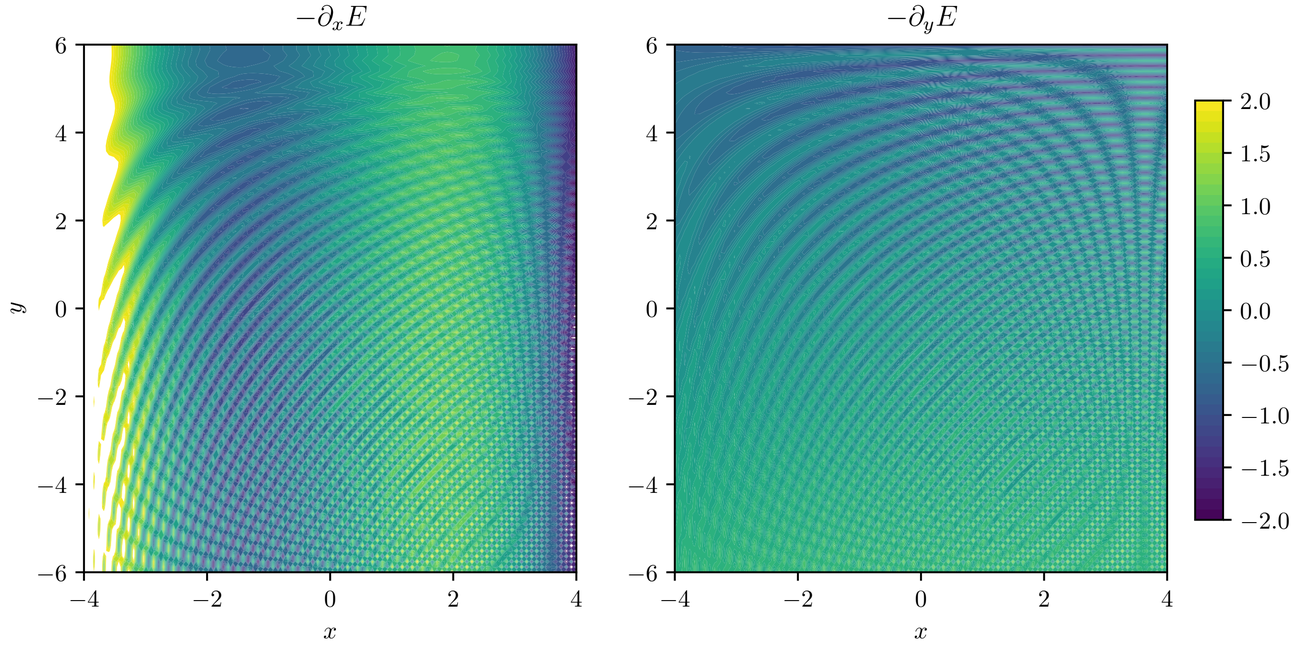
$$\frac{V(x,y)}{k_{\mathrm{B}}T} = \frac{1}{50}(x-4)(x-2)(x+2)(x+3) + \frac{1}{20}y^2 + \frac{1}{25}\sin(3(x+5)(y-6)$$

Potential from: Wang et al., *ACS Cent. Sci.* **5** 755–767 (2019)

Distribution (Brownian dynamics)

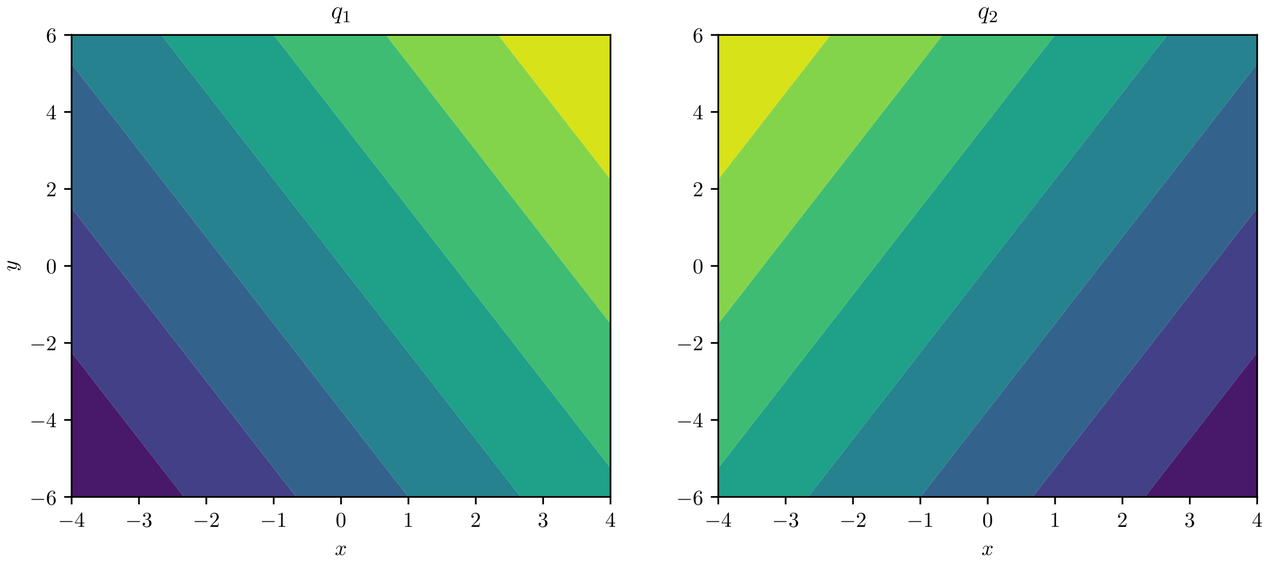


Force field



Standard kernel

$$\mathbf{q} = \begin{pmatrix} 2(x+y) \\ x-y \end{pmatrix}$$



Learn the instantaneous forces

$$(\{f^x\}, \{f^y\})$$

and assume **independence**

$N = N \quad N \times N$ $\qquad$ $N = N \quad N \times N$

$$K(\mathbf{q}, \mathbf{q}') = \exp\left(-\frac{(\mathbf{q}-\mathbf{q}')^2}{2\sigma^2}\right)$$

$$f_j^x = \sum_i \alpha_i^x \left(K(\mathbf{q}_i, \mathbf{q}_j) + \lambda \mathbb{I}\right) \qquad f_j^y = \sum_i \alpha_i^y \left(K(\mathbf{q}_i, \mathbf{q}_j) + \lambda \mathbb{I}\right)$$

Energy-conserving kernel

Learning in the gradient domain:

$$\mathrm{Cov}\left(\frac{\partial E(\mathbf{q}_i)}{\partial r^k}, \frac{\partial E(\mathbf{q}_j)}{\partial r^l}\right) = \underbrace{\frac{\partial \mathbf{q}}{\partial r_i^k} \cdot \frac{\partial^2 K(\mathbf{q}_i, \mathbf{q}_j)}{\partial \mathbf{q} \partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial r_j^l}}_{\mathbf{K}_{\mathrm{Hess}}(q_i^k, q_j^l)} + \cancel{\frac{\partial K(\mathbf{q}_i, \mathbf{q}_j)}{\partial \mathbf{q}} \cdot \frac{\partial^2 \mathbf{q}}{\partial r_i^k \partial r_j^l}}$$

$$K(\mathbf{q}, \mathbf{q}') = \exp\left(-\frac{(\mathbf{q} - \mathbf{q}')^2}{2\sigma^2}\right)$$

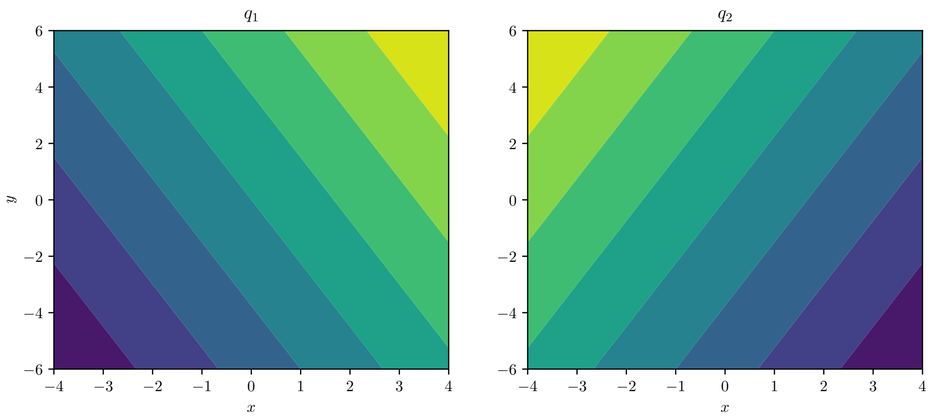
$$\mathbf{K}_{\mathrm{Hess}}(q_i^k, q_j^l) = \frac{\partial \mathbf{q}}{\partial r_i^k} \cdot \frac{\partial \mathbf{q}}{\partial r_j^l} \frac{\partial}{\partial \mathbf{q}} \frac{\partial}{\partial \mathbf{q}} \exp\left(-\frac{(\mathbf{q}_i - \mathbf{q}_j)^2}{2\sigma^2}\right)$$

$$= \boxed{\frac{\partial \mathbf{q}}{\partial r_i^k} \cdot \frac{\partial \mathbf{q}}{\partial r_j^l}} \frac{1}{\sigma^2}\left(1 - \frac{(\mathbf{q}_i - \mathbf{q}_j)^2}{\sigma^2}\right) \exp\left(-\frac{(\mathbf{q}_i - \mathbf{q}_j)^2}{2\sigma^2}\right)$$

enables energy conservation

Mathias, Master Thesis, Bonn (2015); Glielmo et al., *arXiv:1905.07626; Csanyi; Tkatchenko; Müller…*

Energy-conserving kernel



$$\mathbf{q} = \begin{pmatrix} 2(x+y) \\ x-y \end{pmatrix} \qquad \frac{\partial \mathbf{q}}{\partial x} = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \qquad \frac{\partial \mathbf{q}}{\partial y} = \begin{pmatrix} 2 \\ -1 \end{pmatrix}$$

$$\mathbf{q} = \begin{pmatrix} x \\ y \end{pmatrix} \qquad \frac{\partial \mathbf{q}}{\partial x} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad \frac{\partial \mathbf{q}}{\partial y} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

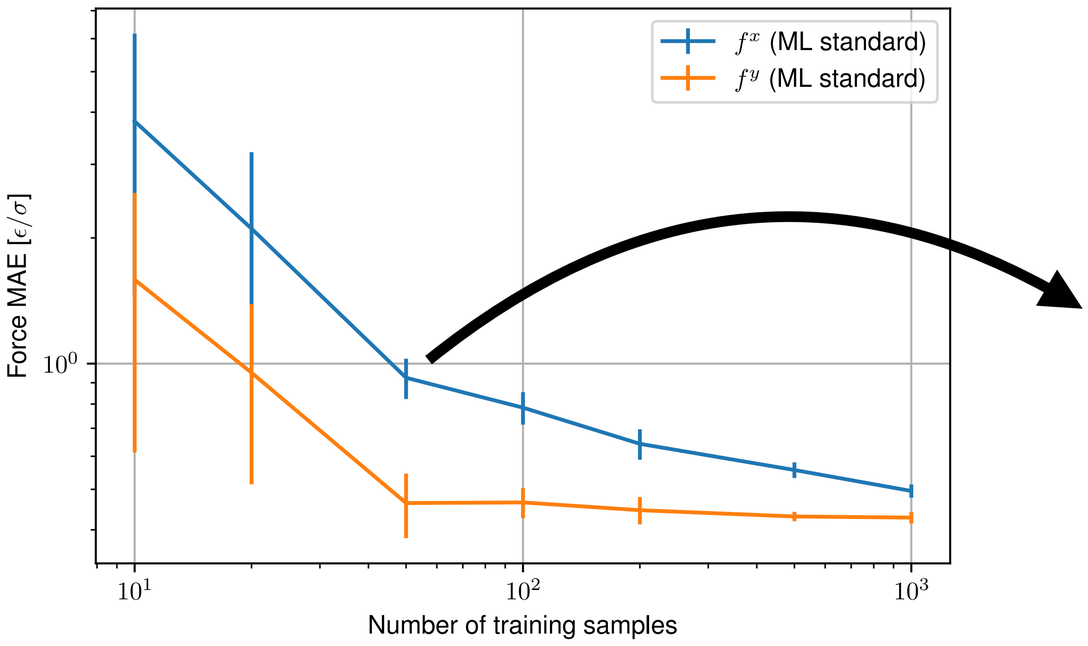
$$K(\mathbf{q}, \mathbf{q}') = \exp\left(-\frac{(\mathbf{q}-\mathbf{q}')^2}{2\sigma^2}\right)$$

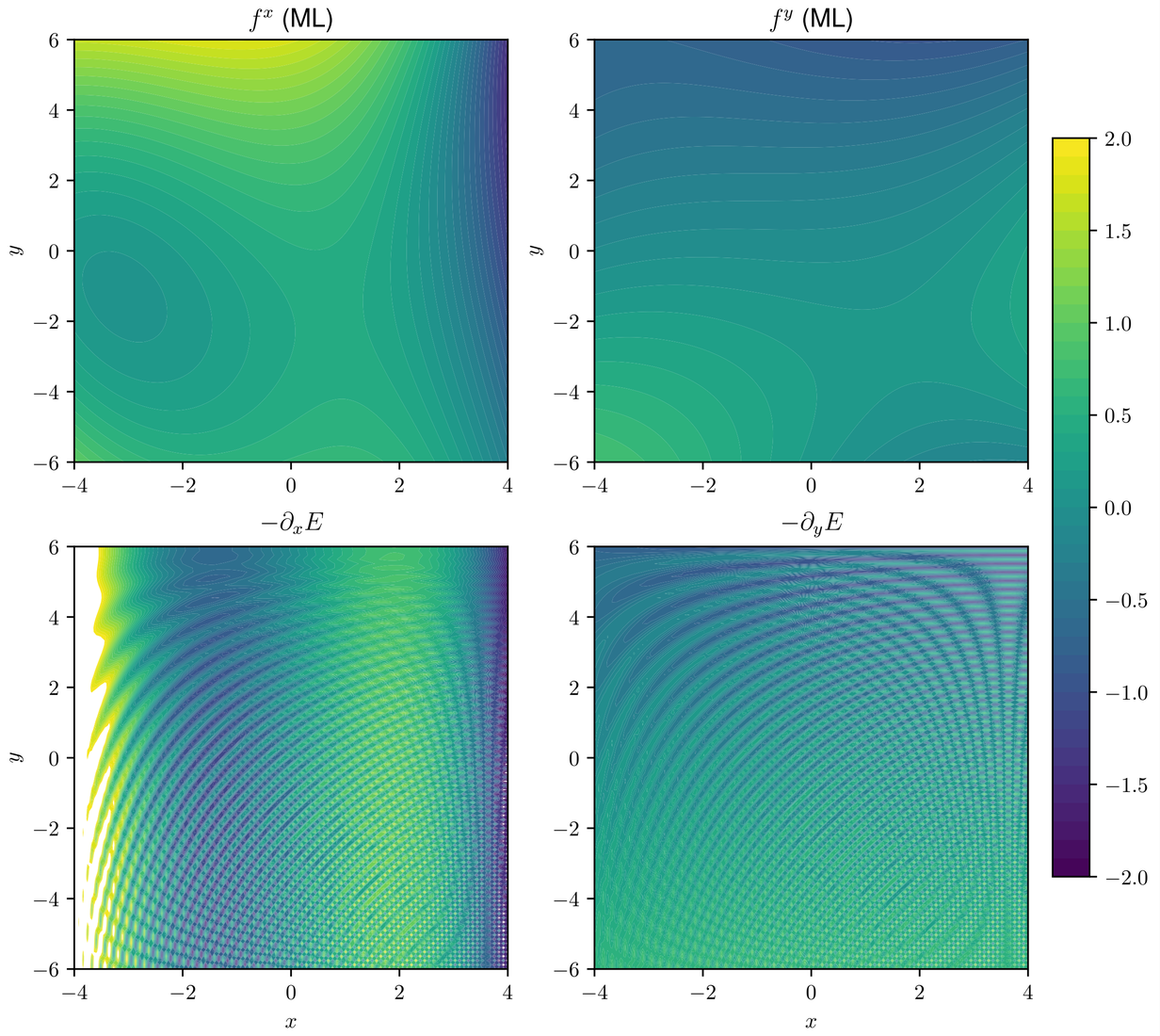
$$\mathbf{K}_{\text{Hess}}(q_i^k, q_j^l) = \frac{\partial \mathbf{q}}{\partial r_i^k} \cdot \frac{\partial \mathbf{q}}{\partial r_j^l} \frac{\partial}{\partial \mathbf{q}} \frac{\partial}{\partial \mathbf{q}} \exp\left(-\frac{(\mathbf{q}_i-\mathbf{q}_j)^2}{2\sigma^2}\right)$$

$$= \frac{\partial \mathbf{q}}{\partial r_i^k} \cdot \frac{\partial \mathbf{q}}{\partial r_j^l} \frac{1}{\sigma^2} \left(1 - \frac{(\mathbf{q}_i-\mathbf{q}_j)^2}{\sigma^2}\right) \exp\left(-\frac{(\mathbf{q}_i-\mathbf{q}_j)^2}{2\sigma^2}\right)$$

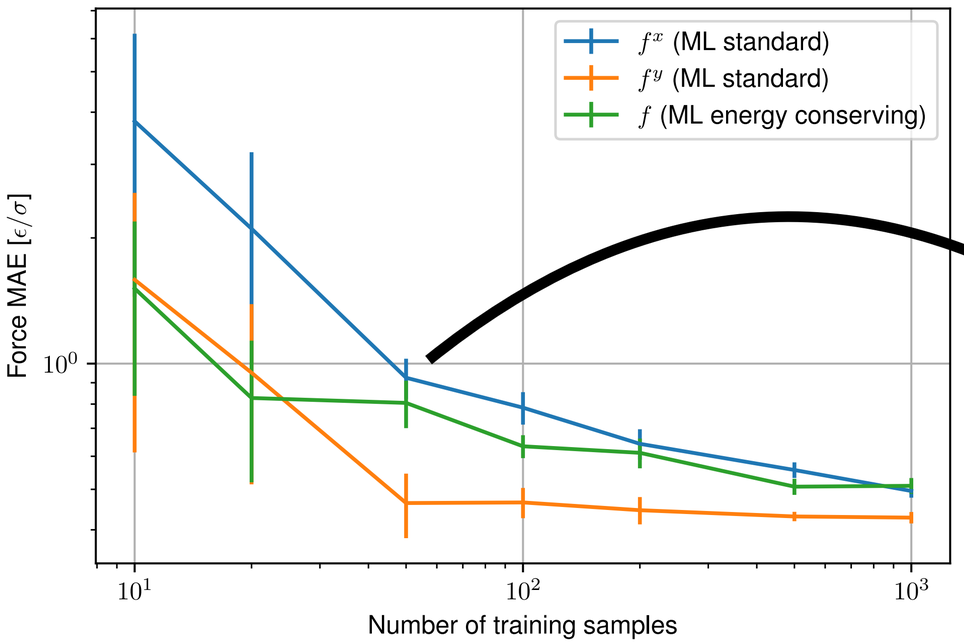
enables energy conservation

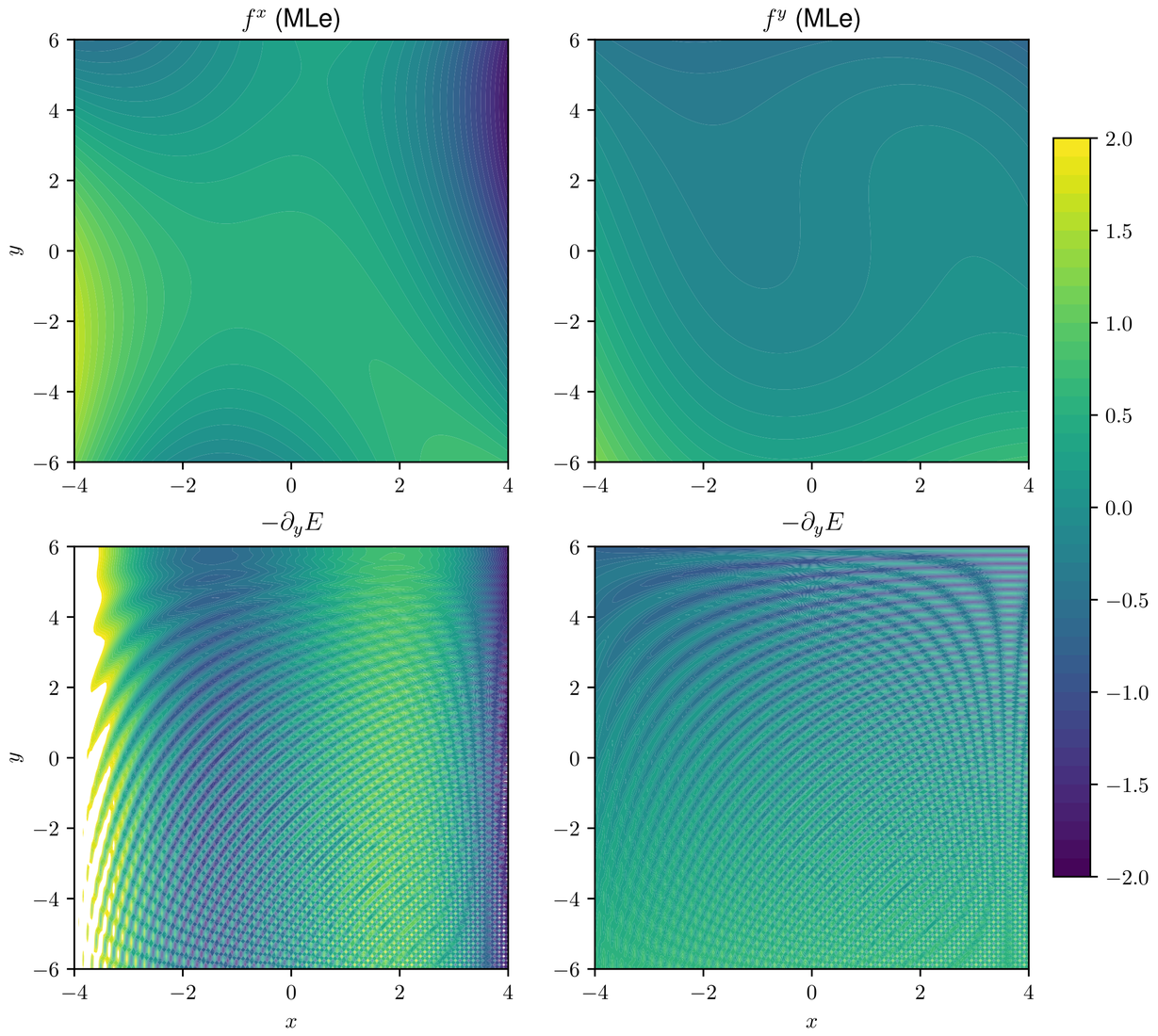
Standard kernel



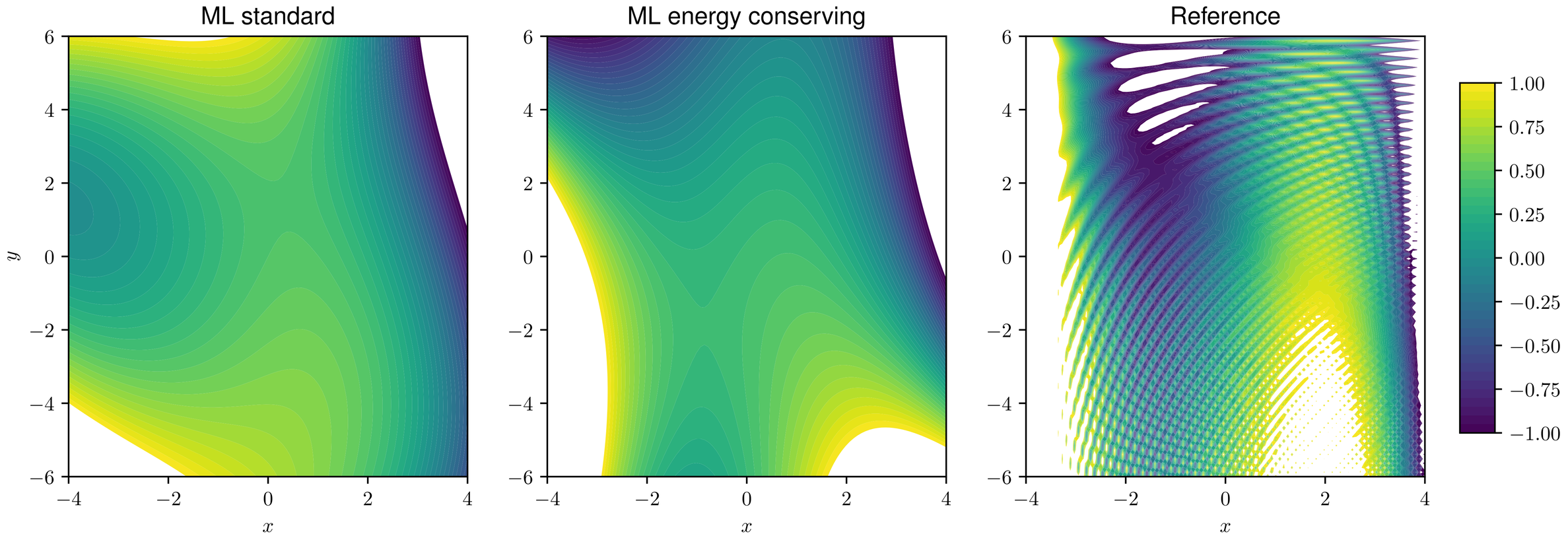


Energy-conserving ML





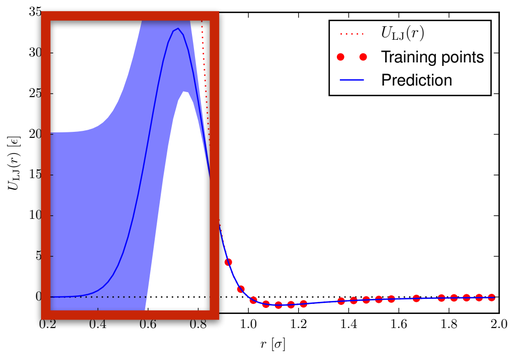
Comparison between ML models



Learning from only 50 samples

Energy-conserving force field useful in the low-data regime

Conclusions

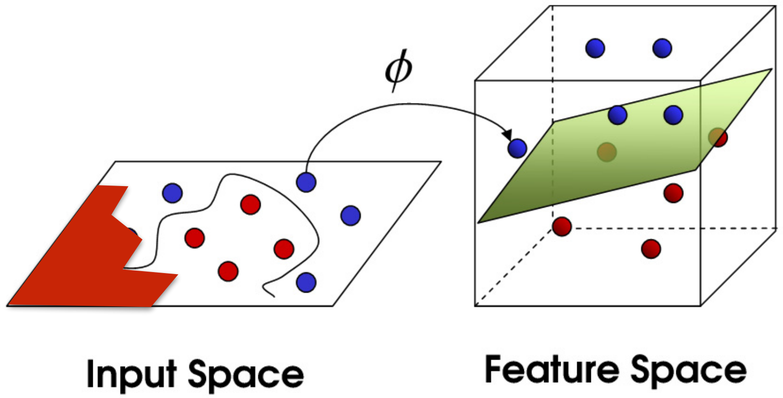


Extrapolation in ML models of energy landscapes

Can lead to catastrophic physics

Take advantage of symmetries

Noether: symmetry leads to conservation law



$$\mathbf{K}(\mathcal{S}\rho, \mathcal{S}'\rho') = \mathbf{S}\mathbf{K}(\rho, \rho')\mathbf{S}'^{\mathrm{T}}$$

Build symmetries in ML model

Work with subset of kernels that a priori satisfy conservation law