Computational Science and Engineering
(International Master's Program)

School of Computation, Information and Technology

Technische Universität München

Master's Thesis

Machine learning potentials using higher order interactions

Rahul Manavalan

Computational Science and Engineering (International Master's Program)

School of Computation, Information and Technology

Technische Universität München

Master's Thesis

Machine learning potentials using higher order interactions

Author:	Rahul Manavalan
1st examiner:	Prof. Dr. Christian Mendl
2nd examiner:	Dr. Felix Dietrich
Submission Date:	November 22, 2023

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

November 22, 2023 Rahul Manavalan

Acknowledgments

The importance of community was evident during the pandemic of 2019-2022. I am elated to have found solace in the Julia community for Scientific Machine Learning and Computational Chemistry during these trying times. My contributions, as little as they were, made me feel valued in an otherwise apathetic world. Special thanks is due to Chris Rackauckas for introducing me to SciML and Michael Herbst and co. for patiently teaching me DFT during the summer school at Sorbonne.

Thanks is due to the PhD candidates of SCCS for some insightful discussions, playful banter and for making me feel at home.

I would also like to thank Christian Mendl and Felix Dietrich for agreeing to supervise this thesis.

Abstract

Ab-initio molecular dynamics simulations are the primary computational tool to analyze natural phenomena that are inexplicable using macroscopic modeling. However, this comes with the caveat that one should solve for the ground state of a many-body electronic Hamiltonian several times during the course of a simulation for evaluating the forces. Machine learning potentials have emerged as a popular alternative in the last decade, with efforts to train large machine learning models on massive chunks of data underway. The hope is that these models would generalize, at least for most practical purposes.

This work adds to the toolbox of machine learning potentials by using function approximators based on random features for approximating the forces and energies of molecular systems. Additionally, it investigates inference schemes for learning functions on sets. Lastly it demonstrates that higher order (interaction) representation is necessary for some molecules that are considered here.

Contents

1 Introduction

Richard Feynman begins his famous lectures on physics with the atomic hypothesis.

> **Postulate 1** (Atomic hypothesis)**.** *All things are made of atoms – little particles that move around in perpetual motion, attracting each other when they are a little distance apart, but repelling upon being squeezed into one another [34].*

It suffices to say that the study of all natural sciences concerns the study of atoms – on how they are composed; on how they interact with each other; on how in the limit of large numbers, groups of atoms exhibit emergent behaviour.

The scientific collective has spent the last century coming up with "the" universal law of nature (may gravity pass over in silence) and to quote Feynman:

> **Postulate 2** (Law of nature)**.** *"Quantum mechanics" is the description of the behavior of matter and light in all its details and, in particular, of the happenings on an atomic scale [35].*

Clearly, the contemporary consensus is to model all atomic interactions using quantum mechanics. This viewpoint is rightly justified, as molecular dynamics has played a pivotal role in understanding a wide variety of physical systems ranging from transient behaviour of non-ideal gases [89], modeling spike proteins of viruses [11] and even electronic energy transitions in atto-second lasers [8]. Scientific curiosity aside, modeling atomic interactions is crucial in the field of medicine, where computational models can exponentially accelerate a drug development cycle [10]. In lieu of this, there is an ever increasing need to improve the computational models, from the perspective of both accuracy and computational efficiency.

Statistical models based on deep neural networks – "machine learning potentials" [16], have catalyzed this progress in the last two decades. They have enabled molecular dynamics enabled investigations of systems that were previously not possible [101]. Yet, there is much frailty in their constitution, the way they are trained and their predictive accuracy for out-of-sample distributions and the costs incurred in re-training the models [33]. This work investigates, if alternative training strategies could be adopted for such statistical models, whilst maintaining the accuracy of the state of the art training methods.

More precisely, I am interested in the following questions:

1. Can function approximators based on random features [73], faithfully approximate high dimensional functions such as $a)$ Inter-atomic potentials and $b)$ Force fields of molecular systems?

2. By what means can inductive bias [52] be incorporated into these empirical models?

3. Do higher order interactions [88] improve the predictive accuracy of such empirical models?

In Part 2, I will briefly introduce the ideas that are central to this work. One should be wary that the sections therein are research disciplines on their own. A more interested reader should seek out the listed references, for a wholesome understanding of the subject matter. Part 3 describes the methods, lists the algorithms that were used, developed in this work. The results from the numerical experiments are presented and also interpreted here. Part 4, summarizes the findings and provides a future outlook.

2 Background and State of the art

2.1 On the quantum many body problem

Molecular systems are modelled atomistically using the laws of quantum mechanics. When such laws are used to model complex systems, it constitutes a many body problem. There exist several coarse-grained versions of its formulation [97, 60, 41]. This section builds up the foundation to understand one of them.

2.1.1 Objects of Quantum Mechanics

When modelling a physical system, there are three questions that we should concern ourselves with, namely:

1. Which mathematical object best represents the state of the system?
2. How to extract physically meaningful observables from this object?
3. What are the set of equations that best model the evolution of this system?

With classical mechanics, systems are represented on a manifold $\mathcal{M}$, observables are inferred as functions defined on $\mathcal{M}$ and most systems can be faithfully modelled using Lagrangian mechanics [63]. The quantum mechanical counterparts to these entities are the wavefunction, the measurement operator and the time dependent Schrodinger's equation respectively [90].

Definition 1 (Wavefunction). *A wavefunction for a physical system $\mathcal{S}$ is an element of a **Hilbert space** $\mathcal{H}$, consisting of anti-symmetric functions [90].*

Definition 2 (Measurement operator). *Quantum measurement is described by a set of measurement operators $\{M_m\}$ acting on the wavefunction Ψ. The probability of a measurement result m occuring is $\langle\Psi|M_m^\dagger M_m|\Psi\rangle$ and the state of the system after measurement is $\frac{M_m|\Psi\rangle}{\sqrt{\langle\Psi|M_m^\dagger M_m|\Psi\rangle}}$.*

Definition 3 (Schrödinger's equation). *The evolution of a quantum mechanically described system is postulated to evolve according to*

$$i\hbar\frac{\partial|\Psi\rangle}{\partial t} = H\,|\Psi\rangle, \tag{2.1}$$

$$H = -\frac{\hbar^2}{2m}\nabla^2 + V. \tag{2.2}$$

This is the time dependent Schrödinger's equation [90].

Similar to classical mechanics, there are alternative formulations of quantum mechanics such as the density matrix formulation and the Wigner formulation. Here, other mathematical objects are used to describe the system's state.

2.1.2 A many-body problem

For a molecular system eq. (2.1), can be formulated as follows:

[Many Body Problem] Consider a molecular system $\mathcal{S}$ with N_e electrons and N_a atoms such that $N = N_e + N_a$. Let the position of an electron/atom be denoted by r_k^i, its charge number Z_k^i, where $k = \{e, a\}$ and $i \in \mathbb{N}$. Let $t \in \mathbb{R}^+$ be time and $r \in \Omega \subset \mathbb{R}^{3N}$.

$|\Psi\rangle : \Omega \times \mathbb{R}^+ \mapsto \mathbb{C}$, represents the many-body wavefunction for $\mathcal{S}$.

Its evolution is prescribed by:

$$i\bar{h}\frac{\partial|\Psi\rangle}{\partial t} = (H + V_{ext})\ |\Psi\rangle, \tag{2.3}$$

$$H = -\frac{1}{2}\sum_{j=1}^{N_e}\nabla_{r_e^j}^2 - \frac{1}{2}\sum_{i=1}^{N_a}\nabla_{r_a^i}^2 - \sum_{i=1}^{N_a}\sum_{j=1}^{N_e}V_{ae}(r_a^i, r_e^j) + \sum_{i,j=1}^{N_e}V_{ee}(r_e^i, r_e^j) + \sum_{i,j=1}^{N_a}V_{aa}(r_a^i, r_a^j), \tag{2.4}$$

$$V_{kl}(p, q) = \frac{Z_k Z_l e^2}{|p - q|} \quad k, l \in \{e, a\}, \tag{2.5}$$

$$V_{ext} : \Omega \times \mathbb{R}^+ \mapsto \mathbb{R}. \tag{2.6}$$

$|\Psi\rangle$ is postulated to contain all the relevant information about the state of $\mathcal{S}$. Consequently, it depends on a number of parameters. It is easy to see that, in the macroscopic limit, the wavefunction $|\Psi\rangle$ of an arbitrary system $\mathcal{S}$ has a very high dimensional domain Ω. This has four intertwined consequences.

1. Analytical solutions to eq. (2.1) rarely exist.
2. Numerical approximations of $|\Psi\rangle$ suffers from the curse of dimensionality [22].
3. Measuring observables O are infeasible due to the cubic scaling of most numerical algorithms for solving eigenproblems [93].
4. Ignoring 2, time integration of eq. (2.1) is extremely inefficient due to the stiffness of the ODE system ensuing from the spatial discretization of this parabolic partial differential equation [49].

2.1.3 Born-Oppenheimer approximation

Bearing these limitations in mind, several simplifications have been proposed. Chronologically, the first simplification of eq. (2.3)-eq. (2.6) is due to Born and Oppenheimer. Based on the observation that the mass of an electron and the mass of a hydrogen nucleus vary by a factor of about 1800, the Born-Oppenheimer approximation [20] posits that:

1. Electrons evolve at a faster timescale than nuclei.
2. The evolution of nuclear centers can be approximated with Hamiltonian dynamics, i.e. as governed by a potential energy surface induced by a sea of fast moving electrons.

Its principal implication is an effective decoupling of the electronic and nuclear dynamics, meaning eq. (2.3) can be effectively dropped and substituted by Newton's equations of motion. That is

$$\frac{dv_a(t)}{dt} = -\nabla_{r_a} E_P(r_a) + V_E(r_a, t), \tag{2.7}$$

$$\frac{dr_a(t)}{dt} = v_a(t), \tag{2.8}$$

$$H - \sum_{i,j=1}^{N_a} V_{aa}(r_a^i, r_a^j) = -\frac{1}{2}\sum_{j=1}^{N_e} \nabla^2_{r_e^j} - \sum_{i=1}^{N_a}\sum_{j=1}^{N_e} V_{ae}(r_a^i, r_e^j) + \sum_{i,j=1}^{N_e} V_{ee}(r_e^i, r_e^j) + V_{ext}(r_e, r_a). \tag{2.9}$$

This shifted operator in eq. (2.9) is the electronic Hamiltonian. The potential energy surface E_P is obtained by evaluating its lowest eigenenergy.

$$H_E|\Psi\rangle = E_P|\Psi\rangle. \tag{2.10}$$

$$H_E = -\frac{1}{2}\sum_{j=1}^{N_e} \nabla^2_{r_e^j} - \sum_{i=1}^{N_a}\sum_{j=1}^{N_e} V_{ae}(r_a^i, r_e^j) + \sum_{i,j=1}^{N_e} V_{ee}(r_e^i, r_e^j) + V_{ext}(r_e, r_a). \tag{2.11}$$

The Cauchy's problem eq. (2.7)-eq. (2.8) can be solved numerically : given initial conditions $[r_a(0), v_a(0)]$ and the closure external forcing potential V_E; using symplectic time stepping methods due to Stromer and Verlet [39]. In addition, the influence of external environment in the simulations can be accounted for, by formulating stochastic counterparts to eq. (2.7). [64] provides a detailed exposition of a litany of methods to solve such models. The eigenproblem in eq. (2.10) represents another physically motivated approximation, as for most practical purposes, higher energy levels (higher eigenvalues) are almost always inconsequential. This facilitates the use of Krylov subspace methods and other tricks that can sometimes elude its cubic complexity wall.

For all practical purposes in chemical systems, it can be assumed that the Born-Oppenheimer approximation is as good a model as the time dependent Schrödinger's equation. And henceforth in this thesis, this is my assumption as well.

2.2 Numerical methods for the many body problem

Despite the simplifications of the BO approximation, the resulting model still need to be simulated numerically. Here, the algorithmic details of two common numerical methods that approximate $E_P - a)$ Exact diagonalization (ED) and $b)$ Density functional theory (DFT) are discussed. Each section ends with a discussion on their convergence and scaling properties.

2.2.1 Exact digonalization

Exact diagonalization (ED) is a family of methods that is used to estimate the potential energy surface of a many body system. While the host of methods constituting ED have their differences; their underlying computational philosophy is common, viz:

1. Define a suitable linear bases set $\mathcal{B}$, whose coefficients represent a discretized Ψ, say $\hat{\Psi} \in \mathbb{C}^d$.
2. Write down the transformed version of $H_E - H_E^{\mathcal{B}}$ for the chosen $\mathcal{B}$.
3. Discretize the operators in $H_E^{\mathcal{B}}$ appropriately (easier said than done).
4. Solve the resulting matrix-eigenvalue problem using a numerical algorithm $\mathcal{N}$.

Depending upon $(\mathcal{B}, \mathcal{N})$, there exist different flavours of ED. The simplest case is where $\mathcal{B}$ is the Euclidian bases and $\mathcal{N}$ is either an Lanczos/Ritz method [93]. Iterative methods such as LOBPCG [75] have been used in recent works [46].

Algorithm 1 Exact diagonalization

Require: Electronic Hamiltonian H_E, "Eigen" algorithm $\mathcal{N}$
1: Call a suitable eigenvalue-solver $E_P = \mathcal{N}(H_E)$. (Lanczos, LOBPCG)
2: Evaluate quantities of interests $F(E_P), \Sigma(E_P)$.
3: **return** Energy E_P, Forces F, Stress tensor Σ

The Lanczos method is a Krylov subspace method that presents significant advantages over a direct solver (QR method for instance). A Lanczos solver requires two vectors $v, w \in \mathbb{R}^d$ to be stored in memory. The Hamiltonian H_E need not necessarily be assembled, rather a function that performs the action of H_E on the vectors is sufficient. The most significant, computational cost incurred in the Lanczos iteration is matrix multiplication $\mathcal{O}(d^2)$. Over the course of m iterations, the overall computational cost is of the order of $\mathcal{O}(md^2)$, making it a suitable alternative to a direct solver $\mathcal{O}(d^3)$.

When d becomes very large, matrix multiplication is no-longer data efficient. As a result, the computational complexity component of the Lanczos iteration becomes prohibitively large. The locally-optimal-blocked-preconditioned-conjugate-gradient (LOBPCG) method

Element	Atomic number	d
Be	4	q^{12}
Mg	12	q^{36}
Ca	20	q^{60}
Sr	38	q^{114}
Ba	56	q^{168}
Ra	88	q^{264}

Table 2.1: Second group elements and the size of their discretized wavefunctions (q points per dimension). Note that this representation for $|\Psi\rangle$ suffers from the curse of dimensionality.

[58] is used in such scenarios, as this method allows for trivial parallelization of the eigencomputations. Thanks to its favourable scaling, LOBPCG has become a mainstay in several mainstream electronic structure software [45, 38].

A discussion on how the quantities of interest in algorithm 1 are evaluated can be found in [69] and references therein. [93] give a more detailed exposition of the Lanczos iteration concerning numerical stability and convergence. Different choices of preconditioners in LOBPCCG for electronic structure calculations is addressed in [44]. Recently, there is also work addressing the use of operator-adapted wavelet ("gamblets") based precondtioners for eigenproblems [77].

However, for larger electronic systems; using ED to estimate E_P is not favourable. For instance, consider elements from the second group of the periodic table in table 2.1. Here d scales exponentially with the atomic number of the elements. Consequently, H_E has a memory requirement of $\mathcal{O}(d^2)$. This motivates the need for coarse grained electronic structure computations. Density functional theory is most widely used coarse-graining scheme.

2.2.2 Density functional theory

Exact diagonalization belongs to the family of wave-function based methods. Parallel to this, the density function approach [54] has been developed over the past 50 years very successfully. Here the object of interest is the electron density ρ:

$$\rho(r) = N \int dr_2 \ldots \int dr_N \langle \Psi(r, r_2, ..., r_N) | \Psi(r, r_2, ..., r_N) \rangle. \tag{2.12}$$

From the property of the wave function,

$$\int dr_1 \int dr_2 \ldots \int dr_N \langle \Psi(r, r_2, ..., r_N) | \Psi(r, r_2, ..., r_N) \rangle = 1, \tag{2.13}$$

it follows that:

$$\int dr\rho(r) = N. \tag{2.14}$$

Unlike $|\Psi\rangle$, which depends on $3N$ coordinates; ρ depends on three input coordinates. As a result, systems that were previously intractable with ED can now be approximated with density functional theory (DFT). The basic tenets of DFT are the Hohenberg-Kohn theorems [51].

Theorem 1. *Hohenberg-Kohn Theorems*

1. *The external potential $V := V_{ae} + V_{ext}$ is a unique functional of $\rho(r)$ up to a constant. Since V in turn determines H_E, the many body ground state E_P is a unique functional of the ground state density $\rho_0(r)$.*

2. *For a positive density function $\hat{\rho}_0$ satisfying eq. (2.14), $E_P(\hat{\rho}_0) \geq E_P(\rho_0)$.*

Consequently, one can evaluate the energy functional as,

$$E_P(\hat{\rho}_0(r)) = T(\hat{\rho}_0(r)) + E_V(\hat{\rho}_0(r)) + E_{ee}(\hat{\rho}_0(r)); \tag{2.15}$$

$$E_V(\hat{\rho}_0(r)) = \int dr\ V(r)\ \hat{\rho}_0(r); \tag{2.16}$$

$$E_{ee}(\hat{\rho}_0(r)) = \int dr'\ dr\ \frac{\hat{\rho}_0(r')\hat{\rho}_0(r)}{\|r' - r\|}. \tag{2.17}$$

Notice, that the functions $\hat{\rho}_0, T$ are still to be defined. Depending upon their modelling choices, there are different formulations of DFT.

2.2.3 Kohn-Sham DFT

Kohn-Sham DFT (KSDFT) is argubaly the most popular version [65]. Here, $\hat{\rho}_0$, T are implicitly defined based on the following assumption – Electrons in a system are non-interacting. This assumption is non-physical. As a result, closure terms can be introduced in the energy functional that corrects for this modeling error. The closure term E_{XC} in the energy functional is the exchange correlation energy, while V_{xc} is the corresponding correction to the potential.

Consequently, one may write down the Kohn-Sham energy functional and its depen-

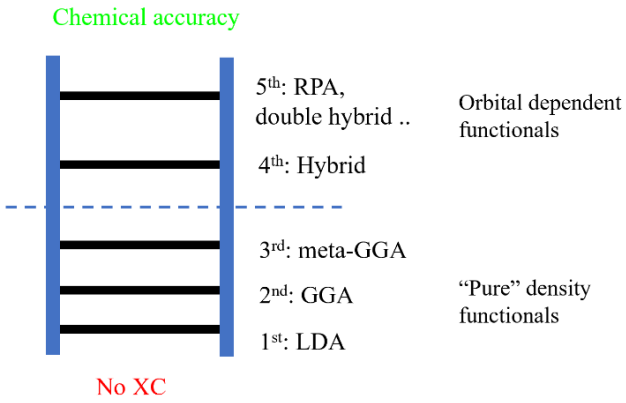

Figure 2.1: Jacob's ladder of exchange correlation functionals. Ascending the ladder, one finds families of exchange correlation functionals that increase the accuracy of DFT computations. Taken from [65].

dencies based on the non-interacting orbital functions ϕ_i:

$$\hat{\rho}_0(r) = \sum_{i=1}^{N} \langle \phi_i(r) | \phi_i(r) \rangle; \tag{2.18}$$

$$\left(-\frac{1}{2}\nabla_{r_i}^2 + V(r) + V_{ee}(\hat{\rho}_0(r)) + V_{xc}(\hat{\rho}_0(r)) \right) \phi_i(r) = \epsilon_i \phi_i(r); \tag{2.19}$$

$$V_{ee}(\hat{\rho}_0(r)) = \int dr' \, \frac{\hat{\rho}_0(r')}{\|r - r'\|}; \tag{2.20}$$

$$V_{xc}(\hat{\rho}_0(r)) = \frac{\delta E_{xc}}{\delta \hat{\rho}_0}; \tag{2.21}$$

$$E_{KS}(\hat{\rho}_0(r)) = T_{NI}(\hat{\rho}_0(r)) + E_{ee}(\hat{\rho}_0(r)) + E_V(\hat{\rho}_0(r)) + E_{XC}(\hat{\rho}_0(r)); \tag{2.22}$$

$$T_{NI}(\hat{\rho}_0(r)) = -\frac{1}{2} \sum_{i=1}^{N} \langle \phi_i(r) | \nabla_{r_i}^2 | \phi_i(r) \rangle. \tag{2.23}$$

E_{ee}, E_V are evaluated with expressions in eq. (2.15). There are different families of E_{XC} in literature. Depending upon their complexity, they are placed on Jacob's ladder (fig. 2.1). Each model on this ladder, provides an explicit expression for E_{XC} in terms of the electron density $\hat{\rho}_0(r)$, effectively closing the system of equations in eq. (2.18)-eq. (2.23). An approximation to E_P say E_{KS} can in-turn be obtained with the self-consistent field iterations in Algorithm 2.

There are several details that are pushed under the rug in algorithm 2. For instance, solving the eigenproblem in Step 4, requires one to choose a suitable bases for representing the orbitals. A common choice is plane wave bases representations [24]. Furthermore, the

Algorithm 2 KSDFT - Self consistent field iterations

Require: Initial guess for density $\hat{\rho}_0^{[0]}(r)$, Exchange correlation functional E_{XC} and its corresponding potential V_{xc}, Convergence threshold ϵ, Density mixing scheme g.

1: $\hat{\rho}_0^{\text{old}}(r) \leftarrow \hat{\rho}_0^{[0]}(r)$
2: Evaluate $V_{ee}(\hat{\rho}_0^{[\text{old}]}(r))$ using eq. (2.20) , $V_{xc}(\hat{\rho}_0^{[\text{old}]}(r))$.
3: Assemble the pseudo-Hamiltonian in eq. (2.19).
4: Solve the eigen-problem in eq. (2.19) to obtain $\{\phi_i^{[\text{new}]}\}_{i=1}^N$.
5: Estimate new density $\hat{\rho}_0^{[\text{old}]}$ using eq. (2.18).
6: **if** $\|\hat{\rho}_0^{[\text{old}]} - \hat{\rho}_0^{[\text{new}]}\| < \epsilon$ **then**
7: **return** $E_{KS}(\hat{\rho}_0^{[\text{new}]}(r))$
8: **else**
9: $\hat{\rho}_0^{[\text{old}]}(r) \leftarrow g(\hat{\rho}_0^{[\text{new}]}(r), \hat{\rho}_0^{[\text{old}]}(r))$
10: Go to Step 2.
11: **end if**

ensuing matrix eigenvalue problem is solved using a LOBPCG algorithm. Preconditioning the eigenproblem is another open question that is actively being investigated [44]. Lastly, the density mixing scheme is another modeling choice that has to be considered. For a more detailed exposition of these different questions see [65].

With the KSDFT algorithm in algorithm 2, electronic systems of the order of 1000 is tractable. This has been leveraged, to simulate several chemical systems over the last three decades [23]. However, with the end of Dennard scaling [30] and the stagnation of Moore's law; investigation of larger molecular systems (as in systems, relevant for solid state physics) remains out of reach.

2.3 Empirical force fields

Electronic structure computations (PES solvers) discussed in section 2.2.1, section 2.2.2 can be used to solve the Cauchy problem in eq. (2.7)-eq. (2.8). However there are two difficulties that prevent their application for practical purposes.

1. Many body systems can be chaotic [64]. Therefore their numerical integration is not feasible, unless very small time steps of the order of $10^{-15}s$ are used.

2. Generally, phenomena of interest such as phase transitions occur over $10^{-3}s - 100s$ [57]; implying several calls to a PES solver over the course of a single MD simulation.

Additionally, there is also increasing need for uncertainty quantification of such simulations, which pose further computational issues. This necessitates the need for surrogate models that are orders of magnitude cheaper than first principle calculations. Empirical

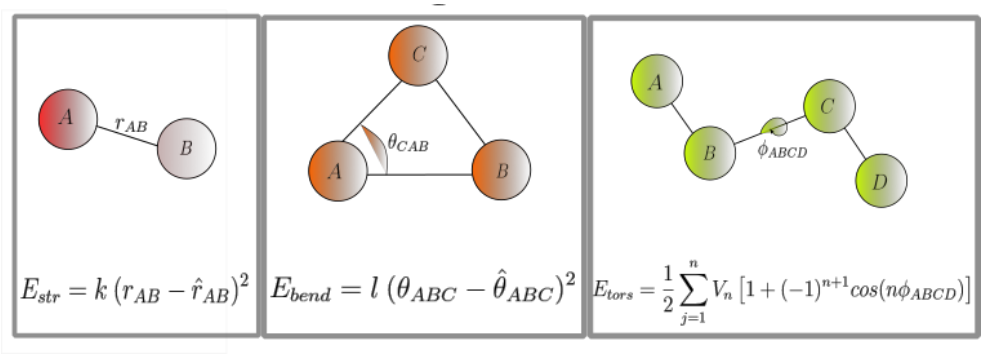

Figure 2.2: **Bonding interactions**: Force fields typically used for a bonded molecule. From left: harmonic potential, bending potential, torsional potential. Note that the torsional potential needs to be represented in the Fourier bases for convenience.

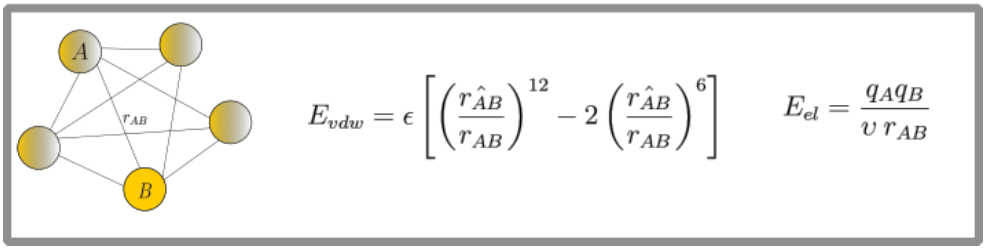

Figure 2.3: **Non-bonding interactions** exist between any two atoms in a molecule. Shown here are the Lennard-Jones potential (representing van der Wall's interaction) and a Coulombic potential for electrostatic exchange.

potentials [54] (force fields – used interchangeably) are such models. Formally, they may be represented as:

$$E_{\mathrm{E}} = E_{\mathrm{str}} + E_{\mathrm{bend}} + E_{\mathrm{tors}} + E_{\mathrm{vdw}} + E_{\mathrm{el}} \tag{2.24}$$

Each of the terms to the right, are factors of the total energy. They are based on physical considerations such as bond length, bond angles (three atoms), torsional angles (four atoms), electrostatic interactions, van der Waals interactions and so on. A pictorial description of the different forces and their generic expressions are shown in Figure 2.2, Figure 2.3.

It is obvious that the parameters $(k, l, V_n, \epsilon, \upsilon)$ of empirical potentials depend on the molecular system under consideration. Experimental observations or results from first principle calculations are usually used as reference to tune the parameters of the specific molecular system. For a complete overview of empirical force field potentials see [54, 25].

It should be noted that the ensuing models are only as accurate as the expressivity of the inference ansatz in Equation (2.24). And rightly so, this is the main drawback of using

empirical force fields for chemical discovery. This search for explicit feature functions that improves the accuracy of the potentials is an open ended problem. In recent times, machine learning algorithms are used to learn these representations. This is discussed in Section 2.4.

2.4 Machine Learning Potentials

Machine learning potentials (MLPs) are glorified empirical force field models. Contrary to empirical force fields, machine learning potentials are not crafted based on physical assumptions. Instead, ab-initio electronic structure calculations are used as a reference to train regression models. Staples in machine learning are favourable representations of input data and inference schemes that learn the correlation among the representations. In the MLP community, such a representation is called a descriptor and the inference scheme is called the regressor. Depending upon the choice of descriptor or regressor there are different flavours of MLPs.

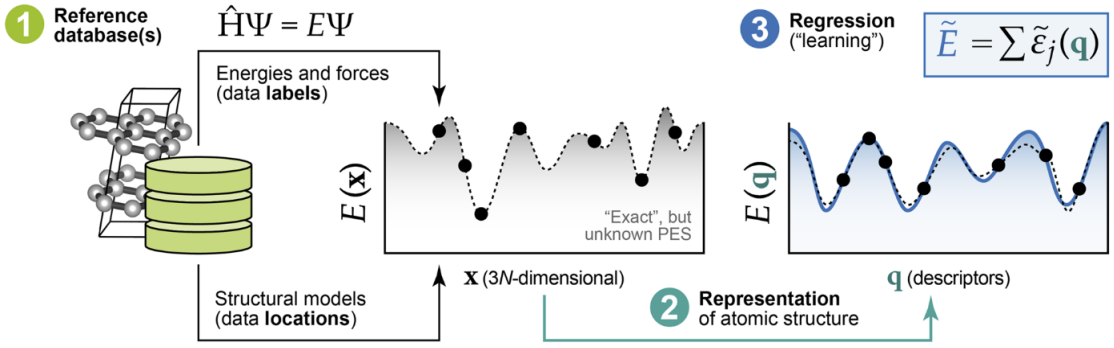

Figure 2.4: Workflow for machine learning potentials – Coordinates of molecules in configuration space (1) is transformed into suitable intermediate representations (2). Predictors for energies and forces (3) are obtained using regression models. Taken from [31].

Since physical considerations are side-stepped to enable "human-out-of-loop" discovery, there are a number of hyperparameters that have to be tuned to guarentee adequate performance of MLPs (See fig. 2.5). After all, typical machine learning algorithms also require tuning of such parameters.

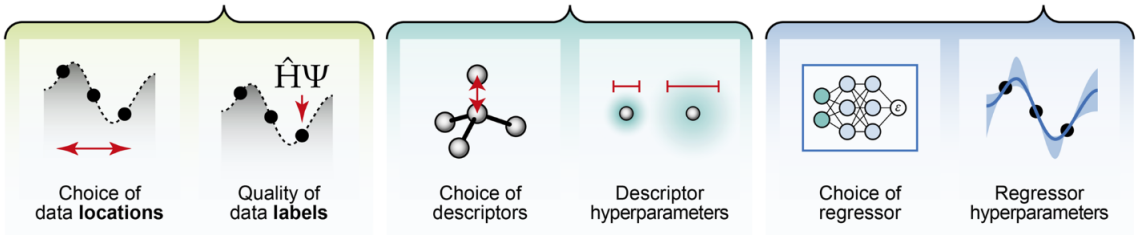

Figure 2.5: Overview of modeling choices in machine learning potentials – Sampling informative parts of E, F, suitable choices of descriptors, regressors are crucial for obtaining a good estimate of the interatomic forces. Taken from [31].

Crucial to quantum chemistry applications is the choice of training data points. Trivially, one would train on all the available points. However, there are a number of complications that speak against this strategy. Active learning methods [92] have been proposed that helps one address this issue. Further, suitable descriptors are instrumental in simplifying the inference procedure. A good descriptor regularizes the already ill-conditioned problem of inference. As a result, the descriptor and regressor are highly dependent entities. Lastly the regressor determines the overall computational cost of the inference procedure. Different approaches exist in the machine learning community [66]. Several of these have been adpoted for MLPs [14, 59], but for random feature models [73]; which is the subject of this thesis. The rest of this section, discusses each of these components in more detail.

2.4.1 Descriptors

Dictionaries [16] (Koopman operator theory), representations, embeddings, feature maps, (deep learning), descriptors [80, 72] (quantum chemistry) are different names for suitable inputs to regression models. Regardless, their main purpose is to facilitate learning. There exists no universal descriptor. Its optimality depends on the function underlying data, availability of data, the regression algorithm and so on. Naturally, there are countless descriptors that have been proposed in literature, each tailored for a specific function class. It is safe to say that a complete reference of all the descriptors in literature does not exist. However, there are a few that seem to work well for large function classes that recur in quantum chemical applications. Figure 2.6 illustrates some of the common descriptors.

Notice that these are invariant to isometries of the input Cartesian coordinates, and/or invariant to permuting coordinates of identical atoms. This is on account of the fact that the energy/forces are invariant/equivariant to such transformations. As a result, characterizing these features in the representations can facilitate the learning process. A few of these are explicit descriptors include:

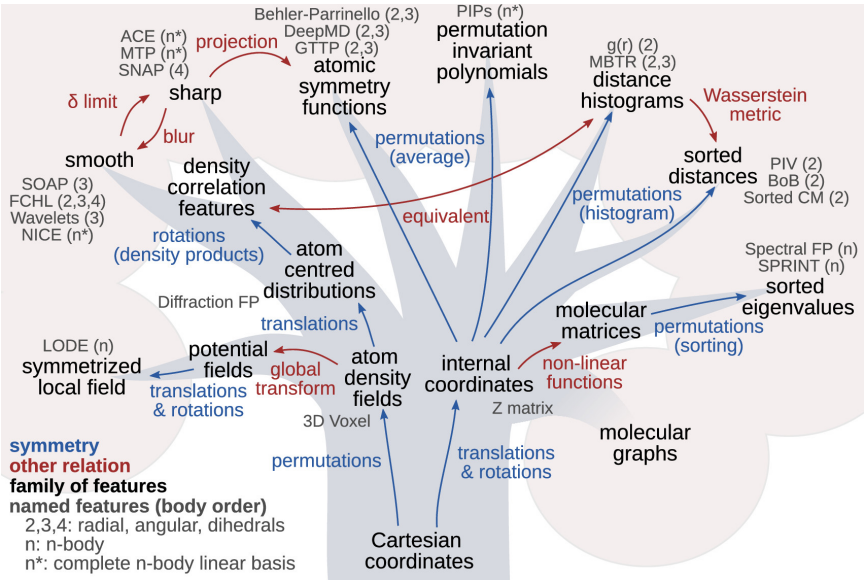

Figure 2.6: Phylogenetic tree of descriptors for materials and molecules. Arrows indicate the relationship between different groups of features. Lists of names, in gray, indicate the most common implementations for each class. Classes that appear as "leaves" of the tree are fully symmetric. Taken from [72].

Coulomb matrix

For atomic configuration coordinates $R \subset \mathbb{R}^{nd}$ and atomic numbers $Z \in \mathbb{N}^n$, the Coulomb matrix [87] is given by the map:

$$\phi_{CM} : R \times Z \mapsto K \subset \mathbb{R}^{n \times n} \tag{2.25}$$

$$K^{ij} = \begin{cases} \frac{Z_i Z_j}{\|R_i - R_j\|} & i \neq j \\ (Z_i Z_j)^{0.24} & i = j. \end{cases} \tag{2.26}$$

Behler-Parrinello two body descriptor

For similar parameters, the Behler Parrinello two body descriptor [15] is given by:

$$\phi_{BP} : R \mapsto G \subset \mathbb{R}^n \tag{2.27}$$

$$G_R^i = \sum_j \exp\left(-\frac{1}{\sigma^2}\left(\|R_i - R_j\| - R_C\right)^2\right). \tag{2.28}$$

Notice the Coulomb matrix representation is isometry invariant, while the BP descriptor is isometry and permutation invariant. In principle, one can encode more prior knowledge into the descriptors, as they do in [21]. However, this comes at the increased cost

of pre-processing the data. For "big" datasets this can be quite tedious. As a result, one needs to judiciously strike a balance between the costs incurred in pre-processing and the costs incurred during inference.

Alternatively, one can learn descriptors that are specific to the inference procedure & dataset [16, 76]. This approach has yielded better results in recent works [100], resulting in increasing proliferation of graph based neural network representations [37] that are seemingly better at representing molecular data, albeit at the cost of reduced interpretability [9]. A more detailed discussion of descriptors can be found in [72, 47].

2.4.2 Regressors

Assuming that an oracle provides a suitable descriptor for the function that one is trying to approximate, the task of the regressor is to fit the available datapoints using a-prior chosen bases-sets. In other words, let $x_i \in \Omega \subset \mathbb{R}^d, y_i \in \Delta \subset \mathbb{R}$, such that there exists $f : \Omega \mapsto \Delta$. Given $\mathcal{D} := (X, Y) := \{(x_i, y_i) : i \in [1, M]\}$, estimate an approximator $\hat{f} : \Omega \mapsto \Delta$; such that $\forall i \in [1, M]$, $\hat{f}(x_i) = y_i$. The three regressors that follow handles this regression problem differently.

Kernel machines

Kernel machines [50] use non-linear feature maps $\phi_i : \Omega \mapsto \Delta$ which constitutes the bases-set to approximate f, i.e

$$\hat{f}(x) = \sum_{j=1}^{F} a_j \, \phi_j(x). \tag{2.29}$$

ϕ_i are generally evaluated from a kernel function κ such that $\phi_i(x) = \kappa(x, x_i)$. a is estimated by solving a linear least squares problem. Formally,

$$\arg\min_{a \in \mathcal{A}} \frac{1}{M} \|Y - \Phi_X \, a\|_2^2 + \lambda^2 \|a\|_2^2. \tag{2.30}$$

Generally, the inference problem (inverse problem) is ill-posed. As a result it is conventional to regularize it. Here Tikhonov regularization [56] with a constant $\lambda \in \mathbb{R}$ is used. Equation (2.30) has an analytical solution, viz:

$$B := (\Phi_X^T \Phi_X + \lambda I) \tag{2.31}$$

$$a^* = B^{-1} \Phi_X^T \, Y. \tag{2.32}$$

Subsequently, predictions can be made using the estimator in eq. (2.29). Kernel machines defined in this form has a complexity of $\mathcal{O}(F^3)$, stemming mostly from matrix inversion. Their expressivity depends directly on the choice of the feature maps ϕ_i. If f is spanned by $\{\phi_i\}_{i=1}^F$, then the approximator is exact. This is rarely the case.

While radial bases kernels are very effective, they suffer from curse of dimensionality when approximating higher dimensional functions, meaning one requires a large number of feature maps F. Nevertheless, kernel machines were used to approximate inter-atomic energies and forces in the early days of the field. See [70, 87] for an exposition on their limitations as machine learning potential regressors.

Gaussian processes

Gaussian processes [42] offer an elegant solution to the curse of dimensionality issue with kernel regression by reformulating the problem in a Bayesian framework. Consider the ansatz in eq. (2.29). The three participating variables are $a \in \mathcal{A} \subset \mathbb{R}^F, x \in \Omega \subset \mathbb{R}^d, y \in \Delta \subset \mathbb{R}$. Denoting all $(x, y)_i$ as (X, Y) and if $p(X, Y, a)$ is the joint probability distribution over these variables, then from Bayes rule

$$p(a|X, Y) = \frac{p(X, Y|a)}{p(X, Y)} p(a) \propto p(X, Y|a)p(a). \tag{2.33}$$

Assuming that the prior and likelihood are Gaussian,

$$p(a) = \mathcal{N}(a; \mu_a, \Sigma_a) \tag{2.34}$$
$$p(X, Y|a) = P(Y|X, a) = \mathcal{N}(Y; \Phi_X a, \Lambda). \tag{2.35}$$

Then posterior over the a, conditioned by X, Y is:

$$p(a|X, Y) = \mathcal{N}(\hat{\mu}, \hat{\Sigma}) \tag{2.36}$$
$$\hat{\Sigma} = (\Sigma_a^{-1} + \Phi_X^T \Lambda^{-1} \Phi_X)^{-1} \tag{2.37}$$
$$\hat{\mu} = \hat{\Sigma}(\Phi_X^T \Lambda^{-1} Y + \Sigma_a^{-1} \mu_a). \tag{2.38}$$

And since eq. (2.29) is a linear map,

$$p(\hat{f}_x) = \mathcal{N}(\hat{f}_x, \Phi_x \hat{\mu}, \Phi_x \hat{\Sigma} \Phi_x^T). \tag{2.39}$$

Notice that the final distribution in eq. (2.39) only implicitly depends on F. Mckay showed in [67] that $\lim_{F \mapsto \infty}$, certain $\phi : \Omega \mapsto \mathbb{R}$ correspond to a positive definite function $\kappa : \Omega \times \Omega \mapsto \mathbb{R}$. In other words, κ - the covariance kernels are infinite dimensional feature maps; which can be evaluated by taking the inner product of the input space with itself. That is, for a given GP prior – $GP(f, \mu, \kappa)$, the posterior over its parameters is given by [43],

$$\hat{\mu}(x) = \mu(x) + K_{xX}(K_{XX} + \Lambda)^{-1}(Y - \mu_X) \tag{2.40}$$
$$\hat{\kappa}(x, x') = \kappa(x, x') - K_{xX}(K_{XX} + \Lambda)^{-1} K_{Xx'}. \tag{2.41}$$

Using Gaussian processes to approximate energies/forces is relatively new. Bartok [13] introduced this first in his thesis. The initial models included a single GP supported over

a class of possible $E : \Omega \mapsto \mathbb{R}$. Later, this was work was expanded in the GDML ecosystem [28, 26, 27], where different aspects of inference was examined and improvements were suggested. One significant contribution of GDML was to learn the forces and energies together using a single statistical model. In effect, they write down the following:

$$\arg\min_{\theta \in \mathcal{H}} \frac{1}{M} \sum_{i=1}^{M} \|E_i - \hat{E}_\theta(R_i)\|_2^2 + \lambda \|F_i - \nabla_R \hat{E}_\theta(R_i)\|_2^2. \tag{2.42}$$

where the forces regularize the energy model. Recently, larger atomic systems with hundreds of atoms have also been investigated with GDML [29], with focus on scaling & efficient inference. However, the fact remains that Gaussian process regression does not scale well for large datasets since the inversion of $G := (K_X X + \Lambda)$ in eq. (2.40), has cubic complexity. As a result, their applicability is restricted to learning functions over a smaller domain Ω'. Fortunately, neural network approximators trained with stochastic gradient descent can be artfully trained on huge datasets at a lower cost.

Neural networks

Formally, a shallow neural network approximating a function f is represented as:

$$\hat{f}(x, \theta) = \sum_{i=1}^{N} a_i \, \sigma(W_i x + b_i). \tag{2.43}$$

Its parameters – θ are obtained by solving a fixed point problem (training)

$$\theta^* = \arg\min_{\theta \in \mathcal{H}} \frac{1}{M} \sum_{i=1}^{M} \|y_i - \hat{f}(x_i, \theta)\|_2^2. \tag{2.44}$$

Typical training algorithms include first order methods such as Adam [55], Nesterov [74]. Second order Hessian based Newton methods [85] are generally more expensive for large models but offer quadratic convergence. Neural networks can approximate high dimensional functions without the curse of dimensionality [12]. Additionally, universal approximation theorems for neural networks guarantee their approximation prowess over a large class of functions. These attributes make them a suitable candidate to learn the interaction potentials and forces of many body systems.

This was realized as early as 1995 when a multilayer perceptron (the other MLP) approximator for the energies was proposed [17]. Since then, incremental modifications to the approximator in eq. (2.43) has been proposed; incorporating more physical information (inductive-bias). Of note, is the idea from [100] where the learning problem in eq. (2.42) is augmented with the virial tensor of a molecule, i.e.

$$\theta^* = \arg\min_{\theta \in \mathcal{H}} \frac{1}{M} \sum_{i=1}^{M} \|E_i - \hat{E}_\theta(R_i)\|_2^2 + \lambda \|F_i - \nabla_R \hat{E}_\theta(R_i)\|_2^2 + \eta \|(R_i \times F_i) - (R_i \times \nabla_R \hat{E}_\theta(R_i))\|_2^2. \quad (2.45)$$

For a detailed review on the different neural network approximators for potentials see [59]. Training neural networks with first order optimizers has been made viable, thanks to the proliferation of GPU devices into mainstream computing [62]. This is especially relevant for functions in quantum chemistry, where databases with billions of reference datapoints have be created with first principle calculations. Owing to its versatility, neural network potentials are arguably the most common machine learning potentials today with applications spanning material science, catalysis, protein-dynamics and the like.

Despite their successful deployment in several ab-initio MD simulations, the brittleness concerning neural network potentials are numerous. Firstly, training neural networks on GPUs, while perfectly tangible, is not necessarily energy efficient. It's all the more worse, as one can seldom give guarantees on the predictions from the network; seriously questioning its deployment in scientific applications with social relevance (medicine, for instance). The issue of uncertainty is addressed by training an ensemble of models on different cross-validation sets [15]. Depending on the variances of the predictions from this ensemble, the decision to re-train the network is made.

2.4.3 Learning descriptors from data

The descriptor and the regressor of a machine learning potential are dependent entities. In recent years, there is increasing work on learning representations that are specific to the dataset and the regressor; in the inference loop [100, 51, 40]. This presents a new challenge of customizing learning architectures to be isometry and permutation invariant.

The question of **isometry invariance**, can be readily addressed from the definition of an isometric operator. Formally for an isometric operator A, $A^T A = \mathbb{I}$. Therefore, a neural network architecture need only transform its input say X, with an intermediate map $\phi : X \mapsto X^T X$.

Permutation invariance is more involved. It connotes that the input to the network is a set. Precisely, let x be a set with cardinality $d = n(x)$, $f : \mathbb{R}^d \mapsto \mathbb{R}^p$. It can be shown [99] that any approximator $\hat{f}$ of f, is a invariant to the permutation of its inputs, if its has the following structure:

$$\hat{f}(x) = \frac{1}{|P_k|} \rho \left(\sum_{y \in P_k} \phi(y) \right) \quad (2.46)$$

where P_k is the set of all possible subsets of x with cardinality $C =: \{1, 2, ..., k\}$. This practice of summing up evaluations of a learned function ϕ over all possible permutations of the input is called Janossy's pooling [71]. It has been shown that $\hat{f}$ is a universal approximator of all permutation invariant functions under certain assumptions [95]. Therefore, *Jannosy's pooling* is a very sound way to learn permutation invariant functions.

Alternatively, one can also rely on heuristics. *Sorting* the inputs based on certain ordering is a trivial way to ensure invariance [70]. However, the question is then : *Which ordering function is best suited for a given dataset and regressor?* This question has a non-trivial answer. Consequently, methods based on heuristics remain far from being competitive with architectures based on Janossy's pooling.

Interestingly, self-attention [94] (fig. 2.7) based on scalar products generalizes Janossy's pooling with $k = 1$. More precisely when $\rho = Id$, $\phi(y) = W_v y\, s((W_k y)^T W_q y)$, s is a suitably normalized softmax function.

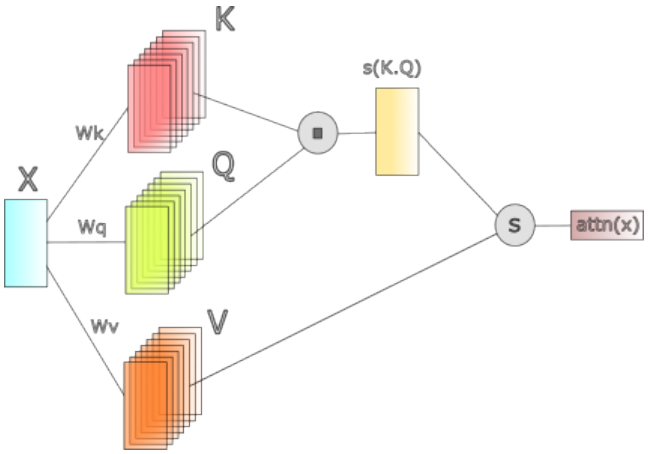

Figure 2.7: Computation graph of self-attention to an input set X. K, Q, V are the query, key and feature vectors respectively. s is a normalized softmax function. S is the summation operation over all elements of X as in eq. (2.46).

In addition, message passing neural networks [35] (MPNN) and its improvements are inherently invariant to permutations and isometries. Promptly, all modern end-to-end machine learning potential are all based on MPNNs [36, 61]. Adjacent to this, kernel learning for data-specific representations is also common [76, 91]. Feature learning with kernels is relatively less explored for machine learning potentials.

2.4.4 Benchmarks and FAIR datasets

Machine learning potentials is a well established field with several FAIR datasets/benchmarks [98]. I mention of a few of them as a reference for posterity.

1. QM7 dataset [18] - Small organic molecules with DFT computed energies

2. GDB17 dataset [86] - Organic molecules with several properties besides the DFT computed energies.
3. MD17 dataset [28] - Small/Medium sized organic molecules with DFT computed energies and forces.
4. Open catalyst [102] datasets - DFT simulations of surfaces occuring in different catalytical applications such as carbon capture with energies, forces and several other properties.
5. The websites Quantum machine [7], NOMAD from FAIRMAT [6] have an assortment of DFT simulated properties for different molecular and material systems.

2.5 Random Feature Models

Random feature models espouse ideas from linear algebra and random network theory for the purpose of function approximation (statistical inference). It is based on the notion that randomization is inherently cheaper than optimization [82].

For an intuitive understanding, consider the following. Given a dataset $\mathcal{D} = \{(x_i, y_i) : i \in [1, m] \subset \mathbb{N}\}$; all x_i can be regarded as the vertices V of a graph – G . Its weighted edges W denote the correlation between x_i and x_j. The function approximators discussed in section 2.4.2 a), section 2.4.2 b), section 2.4.2 c); either directly or indirectly learn this correlation. The connectedness of G directly influences the computational complexity of inferring a function $y \approx \hat{f}(x)$. In the case of a kernel machine, Gaussian process (with isotropic covariance kernel); G is fully connected and their inference procedure has complexity $\mathcal{O}(m^3)$.

Figure 2.8: Conceptual difference between a kernel machine/GP with a random feature model – Random feature models have sparse support over the available reference datapoints. This sparsity accelerates the inference process at the cost of expressivity.

If G were not fully connected but followed a power law distribution, the corresponding covariance matrix would be $K \in \mathbb{R}^{r \times m}$, inducing a complexity of $\mathcal{O}(r^3)$. Therefore, the function f can be now approximated with fewer parameters than before.

A random feature model [82] is a type of kernel machine. Here, the feature maps ϕ_i are chosen apriori. However the parameters of these maps are sampled from a probability distribution. Note that the RFMs have the same problem that is innate to kernel machine, i.e. the class of functions that can be efficiently approximated by these models depends on the choice of the feature map functions ϕ_i. This issue has been addressed with random feature counterparts of neural networks [83].

2.5.1 Random feature neural networks

Random feature neural networks are single hidden layer networks, whose weights and biases are sampled from a suitably chosen probability distribution. The last layer is solved using a linear least-squares procedure – inheriting its favourable convergence properties. One explanation on why random feature networks work; comes from the experiments with neural tangent kernels. It has been observed that, for a randomly initialized extremely wide neural network trained with gradient descent; the magnitudes of weights and biases do not deviate much from their initially assigned values. Furthermore, it can be proved that the neural tangent kernel of an infinitely wide network is constant [53].

Recently, [19] proved that the weights and biases of a neural network can be sampled from a distribution that can be learned from data. This marks a departure from sampling these weights from a normal distribution [96]. A modified version of their method is presented in algorithm 3.

Algorithm 3 Random feature neural networks

Require: Input observations $X \in \mathbb{R}^{d_{in} \times M}$, Output observations $f_X \in \mathbb{R}^{d_{out} \times M}$, Sampling heuristic H, Number of feature maps K, Feature model M, Activation function σ.

$P \leftarrow 2K$.
Sample P pairs $(p^1, p^2)_{j=1}^{P}$ uniformly from $X \times X$.
$it \leftarrow 1$, $\rho_{X_s \times X_s} \leftarrow 0$
while $it < P$ **do**
    $\rho_{X_s \times X_s} \leftarrow \rho_{X_s \times X_s} + H((p^1, p^2)_{it})$
    $it \leftarrow it + 1$
end while
Sample K instances $\{(z^1, z^2)_i : i \in [1, K]\}$ from $p_{X_s \times X_s}$ proportional to $\rho_{X_s \times X_s}$.
Evaluate $\{(W_i, b_i) = M((z^1, z^2)_i) : i \in [1, K]\}$.
Evaluate the feature vectors $\{\phi_{iX} = \sigma(W_i X + b_i) : i \in [1, K]\}$.
Assemble matrix $\Phi = [\phi_1 \ \phi_2 \ ... \ \phi_K] \in \mathbb{R}^{M \times K}$.
Solve for $A = pseudoinv(\Phi, f_X^T)$ with suitable regularization.
return Prediction map $x \mapsto A\sigma([W_1, W_2, ..., W_K]^T x + [b_1, b_2, ..., b_K]^T)$

Here the feature model M evaluates $(W_i, b_i)_{i=1}^{K}$. Notice that M, H are modeling choices for this inference procedure. A suitable choice of M improves the expressivity of the

underlying model. Bolager et al. [19] suggested using a linear model. That is, for any $(z^1, z^2) \sim p_{X_s \times X_s}$:

$$W_i = \alpha \frac{z_i^1 - z_i^2}{\|z_i^1 - z_i^2\|^2}; \qquad b_i = -\beta - \langle W_i | z_i^1 \rangle. \tag{2.47}$$

One should be wary that, there can be no universal definition for W, b. For $relu$ activation functions, $(\alpha, \beta) = (1, 0)$ is optimal. On the other hand, $tanh$ functions require $(\alpha, \beta) = (2\log(1.5), \log(1.5))$ [19]. It is entirely possible to provide an alternative definition for W, b that works just as well for a different σ. Similarly, the optimality of H is again an open question. Bolager et al. [19] provide a definition based on finite differences eq. (2.48).

$$H(x, y) = \frac{\|f_x - f_y\|_2}{\|x - y\|_2}. \tag{2.48}$$

It is also common to sample the space uniformly, i.e $H(x, y) = 1$. For more details concerning the complexity and applicability of random feature neural networks, see [19].

2.5.2 Comparing SLNNs, RFNNs, GPs, Kernel machines

A systematic, generic comparison of the different regressors discussed in this section is not trivial. However, one can make qualitative statements about them. This could facilitate users in making a judicious choice depending on their applications. Such a comparison is attempted in table 2.2. The nomenclature of the symbols used in table 2.2 is given in section 2.5.2.

Symbol	Description
F_1	Number of neurons in a single layer neural network
Θ	Hyperparameters of the training algorithm
q	Number of epochs in the training algorithm
F_2	Number of neurons in a random feature neural network
F_3	Number of feature maps in a kernel machine
M	Number of training points
N	Number of evaluation points
F_M	Feature model of a RFNN (See algorithm 3)
H	Heuristic function for sampling density in a RFNN
κ	Covariance kernel function of a GP
$\phi := \{\phi_i : i \in [1, F_3]\}$	Feature maps of a kernel machine.

Comparison criteria	SLNN	RFNN	GP	Kernel machine
Training complexity	$\mathcal{O}(qMF_1)$	$\mathcal{O}(MF_2 + F_2^3)$	$\mathcal{O}(M^3)$	$\mathcal{O}(MF_3 + F_3^3)$
Evaluation complexity	$\mathcal{O}(NF_1)$	$\mathcal{O}(NF_2)$	$\mathcal{O}(N^2)$	$\mathcal{O}(NF_3)$
Uncertainty of estimates	No	No	Yes	No
Hyper-parameters	Θ, q	F_M, H	κ	ϕ

Table 2.2: A qualitative comparison of the different regressors used in machine learning potentials. For high dimensional functions, it is generally true that $F_1 < F_2 \leq F_3$ which make SLNN seem very viable. However, one has to reckon with the memory footprint of automatic differentiation which can make inference very tedious. In addition, training usually involves a grid search over Θ.

Clearly, RFNNs trade-off computational complexity for reduced expressivity. This leads to the conjecture – "Machine learning potentials with RFNN regressors can accelerate learning without compromising on accuracy". In the rest of this thesis, this conjecture is put to test.

3 Random feature potentials

3.1 Outline

Intuitively, it is clear that the coordinate system that we choose to represent a molecule, should have no influence in either the predictions of the energies or the forces. In other words, the energy and forces are isometric functions. More precisely,

$$E(\mathcal{T}R) = E(R); \quad E(\mathcal{R}R) = E(R). \tag{3.1}$$
$$F(\mathcal{T}R) = F(R); \quad F(\mathcal{R}R) = F(R). \tag{3.2}$$

Here $\mathcal{T}, \mathcal{R}$ are the translation and rotation operators for our chosen coordinate system. Further if one uses a different ordering for similar atoms in a molecule, the energy should be invariant to this permutation and the forces should be equivariant to the same.

$$E(\mathcal{P}R) = E(R). \tag{3.3}$$
$$F(\mathcal{P}R) = \mathcal{P}F(R). \tag{3.4}$$

Therefore merely from a geometric standpoint, one can identify that these isometry /invariant /equivariant properties should be satisfied by any inference model. This chapter examines how such invariances and other physical properties can be leveraged to design physically meaningful statistical models.

Section 3.2 details several inference procedures, in the increasing order of inductive bias. It begins with a baseline RFNN with no descriptors. Isometry and permutation invariant descriptors are subsequently used. Lastly, two models based on physical considerations are discussed. Section 3.3 applies two higher order descriptors and investigates the improvements they offer. Motivated by the shortcomings of existing permutation invariant descriptors for higher order representations, Section 3.4 proposes methods for sampling permutation invariant representations from data. The numerical experiments in each section, have been tested for reproducibility. This entails performing the inference at least five times and ensuring that the variance for the prediction is with a tolerance of 10^{-3}. A full statistical analysis of the predictions remains out of scope of this work.

3.2 Descriptor based models

3.2.1 Black box model

One can approximate the forces and energies using a simple random feature neural network. For a point $R \in \mathbb{R}^{nd}$ in the input space, these models can be described as:

$$E(R) = \sum_{i=1}^{K} a_i^E \, \sigma\left(W_E^i R + b_E^i\right). \tag{3.5}$$

$$F(R) = \sum_{i=1}^{K} a_i^F \, \sigma\left(W_F^i R + b_F^i\right). \tag{3.6}$$

Even though F depends on E, two separate statistical models are chosen for simplicity. The models are trained using algorithm 3. As I demonstrate later, this model is in-adequate when the input space is subject to isometric transformations or when the enumeration order of points in the input vector space is permuted (See tables 3.2, 3.7, 3.12, 3.17 and 3.22).

3.2.2 Black box model with isometry invariance

This problem can be rectified by projecting points in the input space using a isometric kernel $\kappa : R_1 \times R_2 \mapsto \mathbb{R}$. Such kernels are commonly used as covariance functions in Gaussian process regression. The simplest covariance kernel is the inner product $\kappa_{dot}(R_1, R_2) = R_1^T R_2$. A list of more-expressive kernels commonplace is GP literature is provided in table 3.1.

Covariance kernel	$\kappa(R_1, R_2)$
Linear	$R_1^T A R_2$
Polynomial	$\beta \left\|R_1 - R_2\right\|^\gamma$
Squared exponential	$\alpha \ \exp\left(-\frac{\left\|R_1-R_2\right\|^2}{\sigma^2}\right)$
γ-exponential	$\alpha \ \exp\left(-\frac{\left\|R_1-R_2\right\|^\gamma}{\sigma^\gamma}\right)$

Table 3.1: A list of covariance kernels used in Gaussian process regression. Taken from [84].

Notice that the Coulomb matrix representation is a special case of a polynomial kernel; while the Behler Parinello descriptor is exactly equal to the squared exponential kernel. It is amusing that ideas that were mainstream in GP literature in the 2000s, were reinvented much later in the machine learning potential community.

If $K \in \mathbb{R}^{n^2}$ where

$$K^{(i-1)n+j} = \kappa(R_i, R_j) \quad \forall (i,j) \in [1,n]^2. \tag{3.7}$$

Then,

$$E(K) = \sum_{i=1}^{L} a_i^E \, \sigma\left(W_E^i K + b_E^i\right). \tag{3.8}$$

$$F(K) = \sum_{i=1}^{L} a_i^F \, \sigma\left(W_F^i K + b_F^i\right). \tag{3.9}$$

are isometry invariant models; whose parameters can be estimated with algorithm 3. In addition to inducing inductive bias, kernel projections can implicitly linearize the learning problem – allowing the use of less sophisticated regressors (a RFNN with fewer neurons, for instance).

3.2.3 Black box model with permutation invariance

Permutation invariance can be explicity structured into a descriptor using different means. Let $\phi : K \mapsto d$ be the generic notation for all such descriptors. Note that ϕ acts on a isometry invariant input. This bootstrapping allows us sufficient flexibility to use different combinations of representations. Here, I consider three versions of ϕ:

1. The sorted Coulomb matrix ϕ_S [70]
2. Eigenvalues of the Coulomb matrix ϕ_E [87]
3. Singular values of the Coulomb matrix ϕ_Σ

ϕ_S sorts the rows and columns of the Coulomb matrix K such that the row sum of K is in the ascending order. Note that "row-sum" and "ascending order" are heuristics; which ensure permutation invariance by definition. Permutation invariance of the eigenvalues and singular values of K, on the other hand, is not immediately apparent. Consider the following.

Definition 4. *Permutation matrix*
A permutation matrix $P_n \in \mathcal{B}^{n\times n}$ is a square binary matrix that has exactly one entry of 1 in each row and column and 0 elsewhere, where $\mathcal{B} := \{1, 0\}$.

Definition 5. *Permutation transformation*
Let $A \in \mathbb{R}^{n\times n}$ be an arbitrary square matrix, P_n be a permutation matrix. A permutation transformation is defined by the following map $\mathcal{P} : A \mapsto PAP^T$.

Theorem 2. *Permutation invariance of eigenvalues and singular values*
The eigenvalues and singular values of a symmetric positive definite matrix K is invariant to a permutation transformation $\mathcal{P}$.

Proof. From the spectral theorem for real hermitian matrices, $K = U\Lambda U^T$. Here $\Lambda = diagm([\lambda_1, \lambda_2, ..., \lambda_n])$. Let $L := \mathcal{P}(K) = PKP^T$ be the permutation transformed matrix. By substitution, $L = (PU)\Lambda(U^T P^T)$. Notice that L is real and hermitian. Then it follows that the column space of L is spanned by the eigenvectors $[PU_1, PU_2..., PU_n]$ but has the same eigenvalues $diag(\Lambda)$ of K.

A similar argument can be made for the singular values of K. □

Succinctly let $d = \{\phi_S(K), \phi_E(K), \phi_\Sigma(K)\}$. Then,

$$E(d) = \sum_{i=1}^{L} a_i^E \, \sigma\left(W_E^i d + b_E^i\right). \tag{3.10}$$

$$F(d) = \sum_{i=1}^{L} a_i^F \, \sigma\left(W_F^i d + b_F^i\right). \tag{3.11}$$

are permutation and isometry invariant. These models are inferred using algorithm 3. While permutation invariance can be guaranteed with these representations, there does not exist a proof of universal approximation. Furthermore, these transformations result in a loss of information available in the Coulomb matrix. These artifacts are reflected in the accuracy of their predictions (See tables 3.4, 3.9, 3.14, 3.19 and 3.24).

3.2.4 Extensive model

Energy of any thermodynamic system is an extensive quantity. It follows that the total energy of a molecular system is a superposition of energies of individual atoms. Promptly, the inference model can be written down as:

$$E(d) = \sum_{k=1}^{n} E_k(d) = \sum_{k=1}^{n} \sum_{i=1}^{L} a_{i,k}^E \, \sigma\left(W_E^{i,k} d_k + b_E^{i,k}\right). \tag{3.12}$$

$$F(d) = \sum_{k=1}^{n} F_k(d) = \sum_{k=1}^{n} \sum_{i=1}^{L} a_{i,k}^F \, \sigma\left(W_F^{i,k} d_k + b_F^{i,k}\right). \tag{3.13}$$

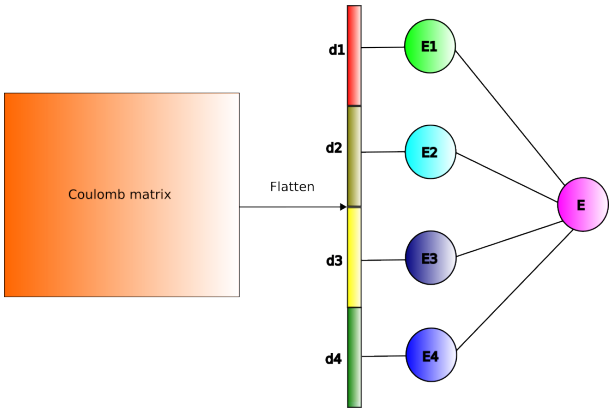

Figure 3.1: Overview of extensive model: A coulomb matrix representation of the molecule is partioned into smaller units such that each partition is specific to an atom in the molecule. This partition is then used to learn a local energy E_i. The sum of all E_i constitutes the total energy E.

Sampling $W_E^{i,k}, W_F^{i,k}, b_E^{i,k}, b_F^{i,k}$ is similar to algorithm 3, only every d_k belongs to a smaller vector space that corresponds to the k^{th} atom (See fig. 3.1). All $a := \{a_{i,k}^{F(or)E} : i \in [1, L], k \in [1, n]\}$ is still obtained from solving a least squares problem. This is detailed in algorithm 4.

Algorithm 4 Extensive random feature potential

Require: Input $d \in \mathbb{R}^{k_{in} \times m}$, Output $F \in \mathbb{R}^{k_{out} \times m}$, Partition factor n, Number of neurons L, Activation function σ.

1: $q \leftarrow k_{in}/n$.
2: $t \leftarrow L/n$.
3: Partition input d into local representations $P \leftarrow [d_1, d_2, ..., d_n]$ where $d_i \in \mathbb{R}^{q \times m}$.
4: Evaluate $[W^{1,1}, ..., W^{L,1}, ..., W^{1,n}, ..., W^{L,n}]$, $[b^{1,1}, ..., b^{L,1}, ..., b^{1,n}, ..., b^{L,n}]$ using eq. (2.47) and the corresponding d_i from P.
5: Evaluate feature vectors $s \leftarrow [s^1, s^2, ..., s^n]$, where $s^i \leftarrow \sigma(W^{:,i} d_i + b^{:,i}) \in \mathbb{R}^{t \times m}$.
6: $Y \leftarrow hstack(s^1, s^2, ..., s^n)$
7: $A \leftarrow pinv(Y^T, F^T)$.
8: $W \leftarrow hstack(W^{:,1}, W^{:,2}, ..., W^{:,n})$.
9: $b \leftarrow hstack(b^{:,1}, b^{:,2}, ..., b^{:,n})$.
10: **return** Prediction map $\hat{F} : d \mapsto A^T \sigma(Wx + b)$

Here $hstack$, $pinv$ are subroutines that perform horizontal concatenation and pseudo-inversion respectively.

3.2.5 Conservative model

Conservative models leverage the fact that forces F is a derived quantity from energy E. Systems with $F(R) = -\nabla_R E(R)$ are called conservative. Forces and energies computed from first principle calculations have this property. As a result, one may write down a single inference model:

$$E(d_R) = \sum_{i=1}^{L} a_i^E \, \sigma\left(W_E^i d_R + b_E^i\right), \tag{3.14}$$

$$F(d_R) = -\sum_{i=1}^{L} a_i^E \, \sigma'\left(W_E^i d_R + b_E^i\right) W_E^i \, \nabla_R d_R. \tag{3.15}$$

Notice that the permutation-isometry invariant descriptors d_R can still be used in this model provided the descriptors' gradient with respect to the coordinates of the atoms $\nabla_R d_R$ is well-defined. Whence eq. (3.14)'s parameters $a := \{a_i\}_{i=1}^L$ can be trained by considering the energies and forces together. Notice that the forces implicitly regularize the energy predictor. The training method is detailed in algorithm 5.

Algorithm 5 Conservative random feature potential

Require: Input $d \in \mathbb{R}^{k_{in} \times m}$, Energy $E \in \mathbb{R}^{1 \times m}$, Force $F \in \mathbb{R}^{nd \times m}$,
Jacobian $J := \nabla_R d \in \mathbb{R}^{k_{in} \times nd \times m}$ Number of neurons L,
Differentiable activation function σ and its derivative σ'.

1: Sample W_E, b_E using eq. (2.47).
2: Evaluate feature maps $\Phi \leftarrow \sigma(W_E d + b_E) \in \mathbb{R}^{L \times m}$, $\Psi \leftarrow -\sigma'(W_E d + b_E) \in \mathbb{R}^{L \times m}$.
3: Compute tensor contraction $T \leftarrow W_E * J \in \mathbb{R}^{L \times nd \times m}$.
4: Compute point-wise product along first and third indices $Q \leftarrow \Psi \odot T \in \mathbb{R}^{L \times nd \times m}$.
5: $Y \leftarrow flatten(hstack(E, F))$.
6: $B \leftarrow hstack(reshape(\Phi, (L, m)), reshape(Q, (L, m \times nd)))$
7: $A \leftarrow pinv(B^T, Y^T)$
8: **return** Prediction maps for *a)* Energy $\hat{E} : x \mapsto A^T \sigma(W_E x + b_E)$
b) Forces $\hat{F} : x \mapsto reshape(A^T(-\sigma'(W_E x + b_E) \odot (W_E * \nabla_R x)), nd, 1)$

A drawback of this training method is its reliance on a for the accuracy of both the energies and the forces; implying the need for an extremely wide network to get sufficiently accurate results. Additionally, differentiable activation functions need to be used.

3.2.6 Numerical experiments

Description of experimental setup

The results for fifteen separate statistical models that predict the forces and energy of five molecular systems ($5 \times 15 = 75$) is presented. Accuracy is measured in terms of the mean

absolute error (MAE) and the root mean squared error (RMSE). Relative error is a less commonly used metric for accuracy for machine learning potentials. However they may be computed from the raw-difference logs in the repository [68].

Five molecules from the MD17 dataset [28] is considered. Training data for these molecules are generated by sampling the trajectories of an AIMD simulation and logging the energies and forces at that instance. Density functional theory with a GGA exchange-correlation functional is used to obtain the energies and forces. More details concerning the data generation workflow can be found in [28]. The models are trained on 2700 atomic snapshots comprising – the positions of atoms R, their atomic numbers Z as inputs; and forces F, energies E as the outputs. Validation is done on 300 different snapshots.

Description of experiments

The following numerical experiments were carried out; to incrementally test the utility of (unless stated otherwise) *tanh* activated-RFNNs to approximate high dimensional functions.

1. Bolager et al. [19] showed that RFNNs could be used in image classification and approximating PDE solution maps. The first experiment uses two vanilla RFNNs to predict the energies and forces. (Effectively re-purposing [19]'s experiments for machine learning potentials.)

2. As an example of isometry-invariant descriptor, the Coulomb matrix is chosen. Note that the Coulomb matrix is symmetric. However, this symmetry was not leveraged while performing the second experiment.

3. The third experiment consisted of three permutation invariant descriptors discussed in section 3.2.3 and RFNN predictors for the energies and forces.

4. The extensive model from section 3.2.4 was used in experiment four, with the Coulomb matrix (CM) and sorted CM as descriptors.

5. The fifth experiment uses a conservative model with no descriptors.

Results

The results from these experiments are presented for five molecules from the MD17 dataset. The presentation of the results follows a template, on which I will briefly elaborate. It begins with a ball-and-stick representation of the molecule. Chemical composition, nomenclature, properties of the molecule are not discussed. Instead appropriate references are cited. Violin plots [48] for the energy and force reference dataset is provided; should a need for a back of the envelope estimation of relative error arise. The tables in each section (that follows) presents the results of the experiments (described above) in order. For experiments 1 and 2, the inputs are transformed via translation, rotation, permutation; and the

predictions from the models are presented (for sanity check and didactic purposes). For all the 75 models, the mean absolute error (MAE) and the root-mean-squared-error (RMSE) over the validation points are shown. Care was taken to ensure that the predictions from the models are sound. This entailed looking at a similarity plot between the ground truth and the predictions, every time a model was inferred and evaluated.

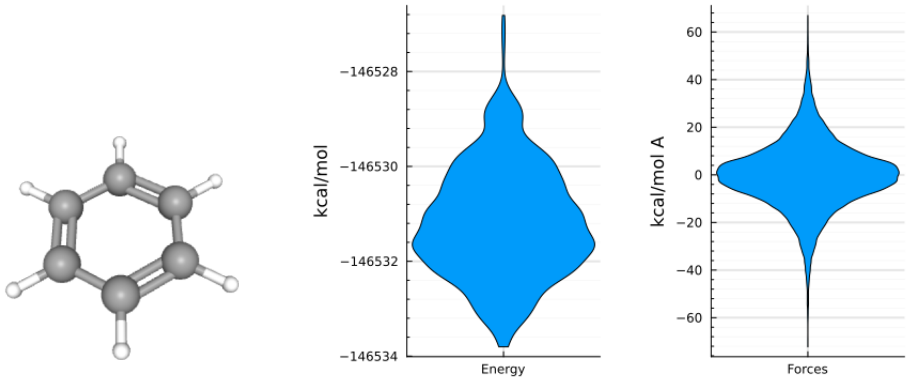

Figure 3.2: Molecular structure of **Benzene** [1] (left). Violin plots of reference data used in training the inference models (right).

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	**0.0221**	**0.0322**	**0.0123**	**0.0223**
*Translation (T)	170132.6540	184559.5941	96158.3846	171916.8738
* Translation (T) and Rotation (R)	195020.5170	199247.4964	100379.8709	180733.8588
*Permutation (P) and TR	221848.1623	222008.0762	97107.3713	172127.9988

Table 3.2: **Benzene** – black box model

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	0.0185	0.02879	0.0026	0.0042
Translation (T)	0.0185	0.02879	0.0026	0.0042
Translation (T) and Rotation (R)	**0.0185**	**0.02879**	**0.0026**	**0.0042**
*Permutation (P) and TR	2.0012×10^6	2.0012×10^6	10899.5915	16439.6504

Table 3.3: **Benzene** – models with isometry invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Eigenvalues	0.05468	0.1696	**0.3252**	1.3239
Singular values	**0.0483**	**0.1192**	0.3569	**1.2279**
Sorted Coulomb matrix	0.6022	1.6420	7.0384	19.5596

Table 3.4: **Benzene** – models with permutation invariant input

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.0259**	**0.0561**	**0.0017**	**0.0048**
Sorted Coulomb matrix	0.3037	0.8914	3.2005	10.5000

Table 3.5: **Benzene** – extensive models with permutation invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
No descriptors	0.5998	0.7352	5.4359	7.1974

Table 3.6: **Benzene** – conservative model

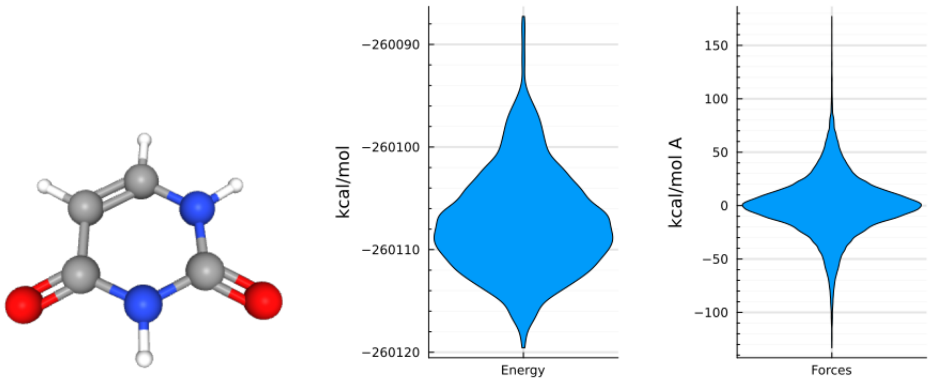

Figure 3.3: Molecular structure of **Uracil** [2] (left). Violin plots of reference data used in training the inference models (right).

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	**0.0862**	**0.2503**	**0.0427**	**0.1704**
*Translation (T)	3.9774×10^6	3.9776×10^6	87500.9885	146041.5868
* Translation (T) and Rotation (R)	3.9571×10^6	3.9572×10^6	82744.4946	137157.1087
*Permutation (P) and TR	3.9571×10^6	3.9572×10^6	77875.8595	130568.2542

Table 3.7: **Uracil** – black box model

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	0.0204	0.0369	0.0197	0.1011
Translation (T)	0.0204	0.0369	0.0197	0.1011
Translation (T) and Rotation (R)	**0.0204**	**0.0369**	**0.0197**	**0.1011**
*Permutation (P) and TR	14763.2457	14772.1628	2527.0183	4284.1704

Table 3.8: **Uracil** – models with isometry invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Eigenvalues	**1.1640**	**1.6392**	3.5249	**7.1756**
Singular values	1.1930	1.6985	4.0401	7.6203
Sorted Coulomb matrix	1.5011	28.1465	**0.8344**	8.5308

Table 3.9: **Uracil** – isometry-Permutation preserving models

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.0107**	**0.04091**	**0.0066**	**0.0564**
Sorted Coulomb matrix	0.4891	2.9036	0.8974	9.0985

Table 3.10: **Uracil** – extensive models

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
No descriptors	1.2548	1.5597	5.1951	6.7914

Table 3.11: **Uracil** – conservative model

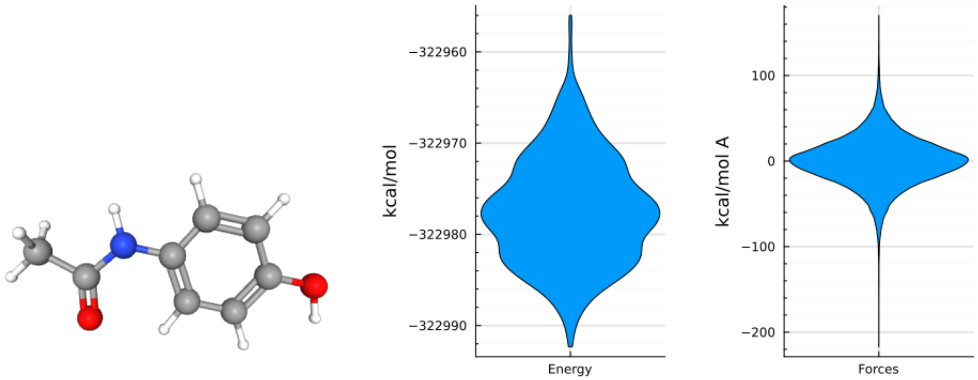

Figure 3.4: Molecular structure of **Paracetamol** [3] (left). Violin plots of reference data used in training the inference models (right).

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	**0.9529**	**3.4998**	**0.7689**	**2.4458**
*Translation (T)	265278.6765	265299.4869	133535.1394	200159.7804
* Translation (T) and Rotation (R)	264677.8583	264689.7260	137258.3965	202921.7452
*Permutation (P) and TR	276248.3042	276253.5043	126694.6834	182643.5324

Table 3.12: **Paracetamol** – black box model

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	0.3133	1.0688	0.8501	3.0006
Translation (T)	0.3133	1.0688	0.8501	3.0006
Translation (T) and Rotation (R)	**0.3133**	**1.0688**	**0.8501**	**3.0006**
*Permutation (P) and TR	13800.8706	13802.0823	3330.4107	5442.1886

Table 3.13: **Paracetamol** – models with isometry invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Eigenvalues	**2.4059**	4.2221	**7.8715**	18.0509
Singular values	2.5003	**3.8092**	8.1050	**17.9362**
Sorted Coulomb matrix	5.7812	20.6717	12.1061	35.2996

Table 3.14: **Paracetamol** – models with isometry and permutation invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.3373**	**0.8387**	**0.8279**	**2.6854**
Sorted Coulomb matrix	7.8740	23.9068	11.7423	34.8863

Table 3.15: **Paracetamol** – extensive models

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
No descriptors	4.0176	4.9413	17.9926	24.4580

Table 3.16: **Paracetamol** – conservative model

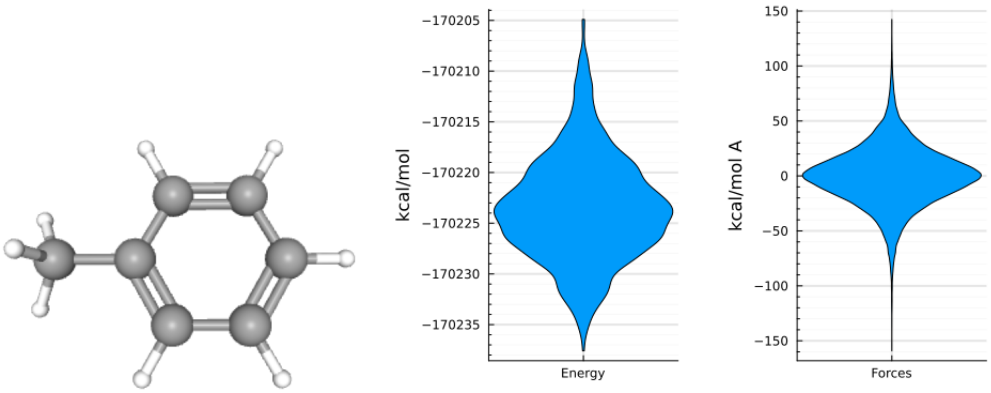

Figure 3.5: Molecular structure of **Toluene** [4] (left). Violin plots of reference data used in training the inference models (right).

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	**0.1850**	**0.4233**	**0.1073**	**0.2887**
*Translation (T)	69497.5651	69518.5747	29682.8295	40032.7038
* Translation (T) and Rotation (R)	65410.3052	65423.4323	29361.2698	41313.1159
*Permutation (P) and TR	78350.6672	78355.8013	28048.1117	37406.8804

Table 3.17: **Toluene** – black box model

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	0.0107	0.0365	0.0168	0.0594
Translation (T)	0.0107	0.0365	0.0168	0.0594
Translation (T) and Rotation (R)	**0.0107**	**0.0365**	**0.0168**	**0.0594**
*Permutation (P) and TR	10158.9147	10338.0666	463552.2690	671876.6283

Table 3.18: **Toluene** – model with isometry invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Eigenvalues	1.9905	2.7833	7.5728	11.6187
Singular values	**1.8161**	**2.5220**	**6.7016**	**10.9791**
Sorted Coulomb matrix	5.9646	14.4071	16.6189	64.9533

Table 3.19: **Toluene** – models with isometry and permutation invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.0114**	**0.0401**	**0.2116**	**1.7623**
Sorted Coulomb matrix	3.5160	17.9604	11.8112	33.1520

Table 3.20: **Toluene** – extensive models

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
No descriptors	2.7661	3.4611	10.6605	14.3088

Table 3.21: **Toluene** – conservative model

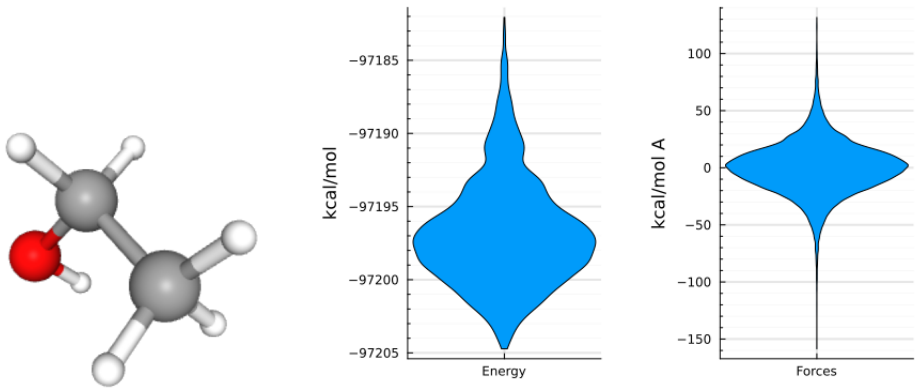

Figure 3.6: Molecular structure of **Ethanol** [5] (left). Violin plots of reference data used in training the inference models (right).

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	**0.0463**	**0.0851**	**0.0654**	**0.1952**
Translation (T)	176138.1227	176850.6669	393966.8690	762423.5326
Translation (T) and Rotation (R)	172538.0114	172771.6819	410544.9175	769435.9447
Permutation (P) and TR	218656.9709	218738.7264	385678.5848	731734.6559

Table 3.22: **Ethanol** – black box model

	Energy $(kcal/mol$		Forces $(kcal/molA)$	
Transformation to the inputs	MAE	RMSE	MAE	RMSE
None	0.0381	0.1358	0.0970	0.7782
Translation (T)	0.0381	0.1358	0.0970	0.7782
Translation (T) and Rotation (R)	**0.0381**	**0.1358**	**0.0970**	**0.7782**
Permutation (P) and TR	953.3074	956.2869	3933.4958	6114.7740

Table 3.23: **Ethanol** – model with isometry invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Eigenvalues	1.6929	**2.5820**	11.2146	23.7170
Singular values	1.8610	4.9771	10.1783	**16.3523**
Sorted Coulomb matrix	**1.5380**	2.9769	**7.2620**	60.2630

Table 3.24: **Ethanol** – models with isometry and permutation invariant inputs

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.01897**	**0.06176**	**0.2944**	**1.1314**
Sorted Coulomb matrix	8.2278	28.6926	12.2748	36.5995

Table 3.25: **Ethanol** – extensive models

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor type	MAE	RMSE	MAE	RMSE
No descriptors	0.9866	1.2523	5.5677	8.2440

Table 3.26: **Ethanol** – conservative model

3.2.7 Observations

The findings from these experimental runs & results can be summarized as:

1. Vanilla RFNNs can approximate $\hat{E}, \hat{F}$, provided the input coordinates are not subject to any transformations (See tables 3.2, 3.7, 3.12, 3.17 and 3.22). In an idealized setting where one has access to infinite data, RFNNs may learn these invariance properties.

2. Coulomb matrix representations which "lifts" the input coordinates; improves on the accuracy of the black-box RFNN models (See tables 3.3, 3.8, 3.13, 3.18 and 3.23).

3. Eigenvalue (EV), Singular-value (SV) descriptors are expensive to evaluate. For molecules with N atoms, their complexity is $\mathcal{O}(N^3)$. In the examples above, N is sufficiently small. However for larger molecules, they have sub-optimal scaling properties. The sorted Coulomb matrix descriptor, on the other hand, scales as $\mathcal{O}(N^2)$. However, it performs worse with respect to EV, SV (See tables 3.4, 3.9, 3.14, 3.19 and 3.24).

4. Sampling local bases functions with extensive models, improve the energy models. Force models do not improve; signalled by the stagnating error with increasing width of the RFNN regressor. This leads one to conclude that while the energy function has local support, forces do not (See tables 3.5, 3.10, 3.15, 3.20 and 3.25).

5. Conservative models with no descriptors perform worse than all other models considered thus far (See tables 3.6, 3.11, 3.16, 3.21 and 3.26). This is counter-intuitive as they have more inductive bias. This contradiction is due to technological issues. The current implementation of conservative models in [68] is not optimized for memory allocation. Consequently, RFNNs with width $K > 10,000$ is too tedious to train. However it can be shown that the errors for this model decay with increasing K. See fig. 3.7 for a scaling experiment performed for the Uracil molecule. Notice the decay of the log-errors. A similar trend is also observed for the other molecules. Therefore, it should be expected that with increasing K arbitrarily accurate models can be obtained. This is currently being pursued.

3.2.8 Issues

Recall that $\hat{F}$ models from the experiments above, need to be eventually deployed in an AIMD simulation. Such many body systems tend to be chaotic. As a result the errors from the forces should be as low as required. Notice that the force predictions using the Coulomb matrix (CM) representation for paracetamol is an order of magnitude greater than the other molecules (Compare table 3.13 with tables 3.3, 3.8, 3.18 and 3.23). Considering that paracetamol is more involved than the other molecules, it leads to the conjecture that the CM representation is not suitable as the complexity of the molecule increases. This motivates the need for higher order descriptors.

Another issue concerns permutation invariant descriptors. Clearly, they are less expressive than CM (See observation 3). This is another avenue that needs to be improved.

3.3 Higher order descriptor based inference models

So far, statistical models that learn E_P from pairwise coordinates is considered. It would seem that augmenting the inputs of these models with coordinates that depend on more than the pairwise interactions (higher order interactions), can increase their accuracy. The N-body decomposition of a many body interaction potential is formally given by [80]:

$$E_P(R_1) = V_1(R_1) + \sum_{j=1}^{N} V_2(R_1, R_j) + ... + \sum_{j_1,...,j_N=1}^{N} V_N(R_1, R_{j_1}, ..., R_{j_{N-1}}). \tag{3.16}$$

Notice that all V_i where $i > 2$, have a complexity of $\mathcal{O}(N^i)$. Here two third-order descriptors are considered – *a)* Angles between the different atoms in a molecule (section 3.3.1) and *b)* a generalized Coulomb matrix representation (section 3.3.2).

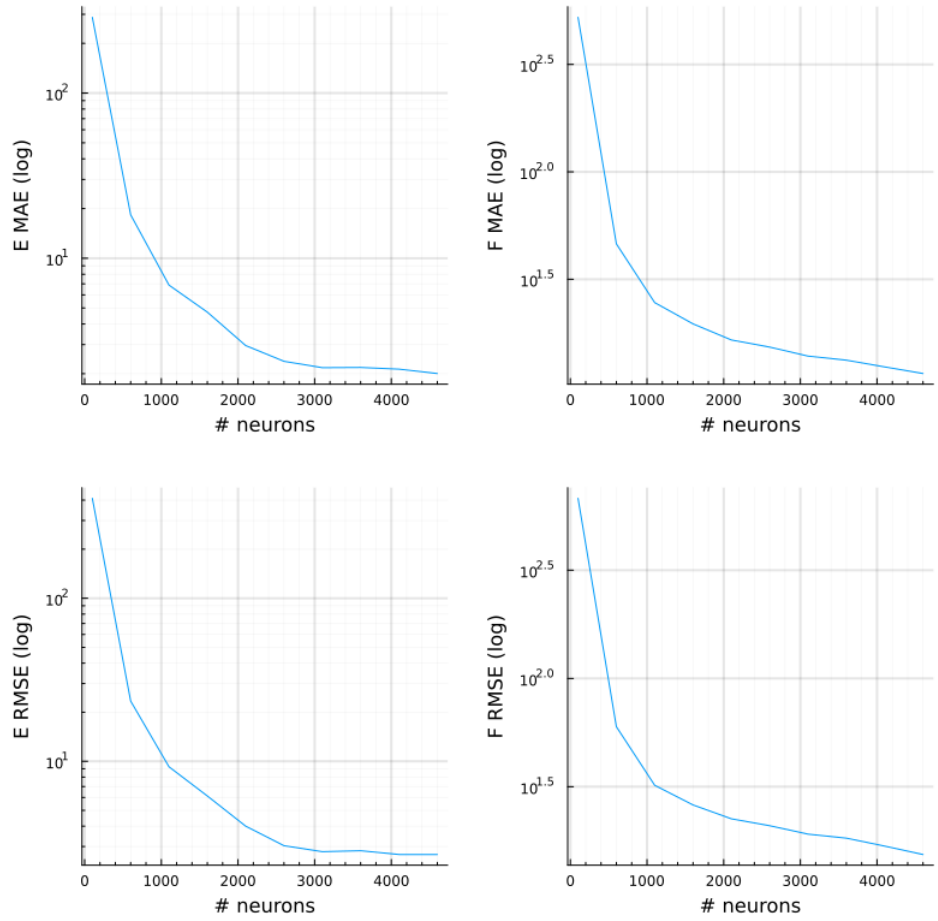

Figure 3.7: Uracil – Decay of MAE and RMSE with increasing K using a conservative model. Notice that the MAE and RMSE for the forces and energies have similar magnitudes. This is a departure from the trend seen in all the previous models, where the RMSE was substantially higher – indicating that model used is implicitly regularized.

3.3.1 Angles

A molecular system can also be completely defined by the angles between the different atoms (fig. 3.8). In this case, α is given by the cosine formula:

$$\alpha = \arccos\left(\frac{a^2 - b^2 - c^2}{2bc}\right). \tag{3.17}$$

A system with N atoms is completely determined by $N\mathrm{C}_3$ coordinates. The angles based descriptor $A \in \mathbb{R}^{N\mathrm{C}_3}$ contains all such α.

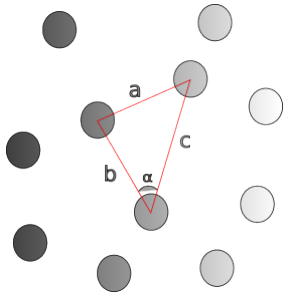

Figure 3.8: A synthetic molecular system with angular degrees of freedom.

3.3.2 Generalized Coulomb matrix

A generalized Coulomb matrix augments the CM (say K) of a molecule with a higher order interactions $G \in \mathbb{R}^{N\mathrm{C}_3}$ defined as:

$$G^{N^2(i-1)+N(j-1)+k} = \hat{G}(R_i, R_j, R_k; \sigma) := \exp\left(-\frac{1}{\sigma^2}(\|R_i - R_j\|^2 + \|R_j - R_k\|^2 + \|R_k - R_i\|^2)\right). \tag{3.18}$$

The complete descriptor is defined as $GCM = [K\, G]^T$. This descriptor is sensitive to the choice of σ; which implicitly defines the cut-off radius for higher order interactions.

3.3.3 Numerical experiments

In the following, the descriptors discussed above are put to test. RFNNs are used as regressors. Unless stated otherwise, *tanh* activation functions were used. Table 3.27–table 3.31 lists the MAE and RMSE of the same molecules from section 3.2. Comparisons are made with the Coulomb matrix descriptor.

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.0082**	**0.0124**	0.0029	0.0049
Angles	0.0096	0.01703	**0.0022**	0.0040
CM + Angles	0.0088	0.0156	0.0022	**0.0037**
Generalized CM	0.0084	0.0125	0.0028	0.0045

Table 3.27: **Benzene** – Higher order descriptors

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.0215**	**0.0556**	**0.0159**	**0.0741**
Angles	0.1053	0.4232	0.0466	0.2201
CM + Angles	0.0285	0.0833	0.0299	0.2246
Generalized CM	0.0266	0.1096	0.0273	0.1982

Table 3.28: **Uracil** – Higher order descriptors

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Coulomb matrix	0.4601	1.4452	0.8081	3.5656
Angles	0.6357	2.4357	0.7743	3.2254
CM + Angles	**0.4413**	**1.3831**	**0.5448**	**2.3379**
Generalized CM	0.6729	2.1980	0.7345	2.7222

Table 3.29: **Paracetamol** – Higher order descriptors

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Coulomb matrix	**0.0370**	0.2398	**0.0199**	0.1447
Angles	0.1033	0.3024	0.1025	0.4671
CM + Angles	0.0412	**0.0863**	0.0477	0.1195
Generalized CM	0.2002	1.8283	0.0159	**0.0905**

Table 3.30: **Toluene** – Higher order descriptors

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Coulomb matrix	0.0391	0.2168	0.1040	0.8240
Angles	0.1015	0.5647	0.1046	**0.2840**
CM + Angles	**0.0204**	**0.0543**	**0.0815**	0.4321
Generalized CM	0.0935	0.8915	0.0816	0.6351

Table 3.31: **Ethanol** – Higher order descriptors

3.3.4 Observations

Notice that the Coulomb matrix descriptor performs better than the higher order descriptors for benzene and uracil (See tables 3.27 and 3.28). On the other hand, models for paracetamol, toluene and ethanol definitely seem to improve with higher order information (See tables 3.29, 3.30 and 3.31). This can be explained based on symmetries. One could infer that the CM representations is sufficient to encode the primitive symmetries for the former, while angular information is required for the latter. The generalized Coulomb matrix, on the other hand, is not a suitable three-body descriptor, as indicated by the sub-optimal predictions (See table 3.27–table 3.31).

3.3.5 Issues

Despite the improvements that higher order descriptors offer for some of the molecules, it is still not permutation invariant. Further, the descriptors are now tensors; meaning the eigen/singular value representations are no longer uniquely defined. While sorting the tensors based on some heuristics is still a viable option, their limitations was already demonstrated for the pairwise cases (section 3.2). This provides additional motivation for section 3.4.

3.4 Random feature shallow sets

The want of expressivity of eigenvalue and sorting based descriptors motivates the need for better permutation invariant descriptors. A common permutation invariant descriptor with guarentees on universal approximation is based Jannousey's k pooling. Here, I examine a special case of $k = 1$ pooling, also know as Deep-Sets [99].

Definition 6 (Deep-Sets)**.** *: Let $X = \{x_1, x_2, ..., x_d\}$ and $f : X \mapsto y \in \mathbb{R}^p$. Here $x_i \in \mathbb{R}^q$. For maps $\Phi : x_i \mapsto I \in \mathbb{R}^r$, $\rho : I \mapsto y \in \mathbb{R}^p$ a permutation invariant approximation of f is given by*

$$\hat{f}_p(x) = \rho\left(\frac{1}{n(\mathcal{P}_X)}\sum_{z\in\mathcal{P}_X}\Phi(z)\right). \tag{3.19}$$

$\hat{f}_p$ can be shown to be a universal approximator of permutation invariant functions [99].

In this chapter, I examine if such permutation invariant functions can be approximated with random feature neural network (RFNNs).

3.4.1 Sampling permutation invariant descriptors

In [99]; Φ, ρ were assumed to be neural networks and trained using back propagation. However, training nested random feature neural networks remains an open problem. As a result, assumptions need to be made for Φ, ρ. In this chapter, I consider the following.

1. A random normal linear projection $\Phi_1(x) := sWx$ where $W^{ij} \sim \mathcal{N}(0, 1), s \in \mathbb{R}$ is a scaling factor and ρ is a RFNN.

2. An activated input space sampled projection $\Phi_3(x) := \sigma(Wx + b)$ where W, b are evaluated using eq. (2.47) and ρ is a RFNN. This corresponds to a sampled two hidden layer neural network.

3.4.2 Numerical experiments

Flattened sets as inputs

For machine learning potentials, the inputs to the random feature shallow sets are similarity matrices. A permutation transformation in this context permutes both the rows and columns of the matrix – necessitating a two step procedure since x_i itself is a set. Trivially, we can avoid all of this by flattening the matrix – making it a single set with higher cardinality.

In the following, the predictive accuracy for the two descriptors approximated with a RFNN for the five molecules in section 3.3 is presented. The predictions from the sorted Coulomb matrix is also provided for reference. Comparisons with either the eigenvalue/singular value descriptors make little sense, as they have a different computational complexity.

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**0.6022**	1.6420	**7.0384**	19.5596
Random normal projection	0.7010	**1.0258**	7.3960	**12.2338**
Sampled-activated projection	0.6599	1.0347	7.6668	15.3476

Table 3.32: **Benzene** – Random feature shallow sets (flattened)

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**1.5011**	28.1465	**0.8344**	**8.5308**
Random normal projection	2.6899	**4.2427**	12.9872	24.2042
Sampled-activated projection	2.7627	5.4885	15.4538	33.8480

Table 3.33: **Uracil** – Random feature shallow sets (flattened)

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	5.7812	20.6717	**12.1061**	35.2996
Random normal projection	3.8723	**5.4715**	16.6822	**26.1975**
Sampled-activated projection	**3.6830**	5.7893	17.0345	32.4314

Table 3.34: **Paracetamol** – Random feature shallow sets (flattened)

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	5.9646	14.4071	16.6189	64.9533
Random normal projection	**2.9743**	4.3371	**14.5249**	**23.7986**
Sampled-activated projection	3.0402	4.3193	17.2807	29.6717

Table 3.35: **Toluene** – Random feature shallow sets (flattened)

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**1.5380**	**2.9769**	**7.2620**	60.2630
Random normal projection	1.9259	3.1354	13.0709	**20.7168**
Sampled-activated projection	3.0617	5.8329	21.0088	48.4880

Table 3.36: **Ethanol** – Random feature shallow sets (flattened)

It is clear from the experiments that introduced descriptors for flattened input sets, do not improve upon sorted Coulomb matrix (CM) descriptors (See table 3.32 – table 3.36). One argument for this, is that upon flattening spatial information is lost.

Nested sets as inputs

This is rectified by considering the CM representation as a nested set. More precisely, the elements of a row of the CM constitute a internal set. There are as many internal sets as

rows of a CM. The internal sets are made permutation invariant by sorting them in their ascending orders. The descriptors from section 3.4.1 can then be used by treating the sorted sets as vectors. With this modification, the same descriptors have now spatial reasoning properties.

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**0.6022**	**1.6420**	7.0384	19.5596
Random normal projection	35.6962	105.7146	**0.1953**	**0.6534**
Sampled-activated projection	16.6882	36.2270	2.4789	5.8013

Table 3.37: Benzene - Random feature shallow sets (nested)

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**1.5011**	**28.1465**	**0.8344**	8.5308
Random normal projection	218.5019	525.5273	1.0613	**3.0019**
Sampled-activated projection	72.3769	141.2698	5.4790	12.3095

Table 3.38: Uracil - Random feature shallow sets (nested)

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**5.7812**	**20.6717**	12.1061	35.2996
Random normal projection	745.8752	1517.1842	**3.7557**	**8.4689**
Sampled-activated projection	48.7088	101.7172	12.0371	24.4677

Table 3.39: Paracetamol - Random feature shallow sets (nested)

	Energy $(kcal/molA)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**5.9646**	**14.4071**	16.6189	64.9533
Random normal projection	209.5065	454.7082	**1.0842**	**3.1677**
Sampled-activated projection	20.3953	41.8933	7.5691	16.4324

Table 3.40: Toluene - Random feature shallow sets (nested)

	Energy $(kcal/mol)$		Forces $(kcal/molA)$	
Descriptor/Model type	MAE	RMSE	MAE	RMSE
Sorted Coulomb matrix	**1.5380**	**2.9769**	7.2620	60.2630
Random normal projection	35.1209	71.8083	**2.9710**	**7.2626**
Sampled-activated projection	6.4901	13.653	9.5293	20.3130

Table 3.41: Ethanol - Random feature shallow sets (nested)

Notice, that the energy predictors with the random normal projection descriptor are worse than the sorted Coulomb matrix. However, the force predictions improve considerably (See table 3.37 – table 3.41). These results are no anomaly. They have been reproduced several times with several random number initializations. An exact explanation of this odd behaviour needs further investigation.

4 Conclusion

4.1 Summary

This work investigated the use of random feature neural networks as function approximators for machine learning potentials. The findings from this work can be summarized as follows.

1. Random feature neural networks (with sampling) are cheap to train, provided the size of the dataset is sufficiently small. With increasing dataset size M, the number of neurons K required for approximating a function upto a certain accuracy increases. When $K > M$, approximation with random feature neural networks is as expensive as an approximation with Gaussian processes albeit without the uncertainty estimates.

2. A-priori defined permutation invariant descriptors based on averaging, pooling; lose information often resulting in worse models than when a Couloumb matrix descriptor is used.

3. Extensive RFNNs are not more expressive than vanilla RFNNs. This belies intuition, as extensive models are based on physical considerations. The why of this artifact needs more investigation.

4. Learning permutation invariant descriptors from data using random feature shallow sets seems promising; but needs more testing & investigation before any conclusive statements can be made.

5. Higher order descriptors improve the surrogate model for certain chemicals. For molecules where the Coulomb-matrix descriptor is sufficiently expressive; no signficant gains are noticed.

4.2 Future work

It is safe to conclude that in their present state; random feature neural networks are not a viable competition to potentials based on other neural network, kernel machine architectures. There are several extensions to this work that can enable this.

1. Random feature networks cannot be trained efficiently on datasets with millions of reference points. An active learning approach based on [92] needs to be adopted; if this method is to scale.

2. Higher order descriptors used in this work, have inherent symmetries. Taking them into consideration can reduce the dimensionality of the input space.

3. Random feature shallow sets with self-attention [79] as feature maps has been shown to work for large language models. This is a promising avenue to further explore in the context of machine learning potentials.

4. When used for an AIMD simulation, confidence over the estimates of the model is necessary [15]. Training an ensemble of RFNNs can aid in this front.

5. In natural science it is not uncommon to have several models for the same phenomena. Embodying this approach, a multi-fidelity inference framework [78] for training the potentials and forces can be adopted.

Once improvements to the model have been carried out, one can deploy them in AIMD simulations. Here, it is certainly possible that the trajectories generated from the learned force fields do not correspond to the ground truth. This is in-fact a known problem in dynamical systems, fluid dynamics, etc. Learning time-intergrator specific force-fields [81, 32] is known to remedy this. To the best of my knowledge, such an approach has not been adopted in the machine learning potential community. This is a direction that is certainly worth investigating.

Bibliography

[1] https://pubchem.ncbi.nlm.nih.gov/compound/Benzene. Accessed: 2023-11-17.

[2] https://pubchem.ncbi.nlm.nih.gov/compound/Uracil. Accessed: 2023-11-17.

[3] https://pubchem.ncbi.nlm.nih.gov/compound/Acetaminophen. Accessed: 2023-11-17.

[4] https://pubchem.ncbi.nlm.nih.gov/compound/Toluene. Accessed: 2023-11-17.

[5] https://pubchem.ncbi.nlm.nih.gov/compound/Ethanol. Accessed: 2023-11-17.

[6] NOMAD — nomad-lab.eu. https://nomad-lab.eu/nomad-lab/. [Accessed 21-11-2023].

[7] Quantum-Machine.org: Home — quantum-machine.org. http://www.quantum-machine.org/. [Accessed 21-11-2023].

[8] L. Adamowicz, S. Kvaal, C. Lasser, and T. B. Pedersen. Laser-induced dynamic alignment of the HD molecule without the Born–Oppenheimer approximation. *The Journal of Chemical Physics*, 157(14):144302, 10 2022.

[9] Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, Mar 2023.

[10] Maral Aminpour, Carlo Montemagno, and Jack A Tuszynski. An overview of molecular modeling for drug discovery with specific illustrative examples of applications. *Molecules*, 24(9):1693, April 2019.

[11] Claudia R. Arbeitman, Pablo Rojas, Pedro Ojeda-May, and Martin E. Garcia. The sars-cov-2 spike protein is vulnerable to moderate electric fields. *Nature Communications*, 12(1):5407, Sep 2021.

[12] Francis Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(19):1–53, 2017.

[13] Albert Bartók-Pártay. *The Gaussian approximation potential*. Springer Theses. Springer, Berlin, Germany, 2010 edition, August 2010.

[14] Albert P. Bartók and Gábor Csányi. Gaussian approximation potentials: a brief tutorial introduction, 2020.

[15] Jörg Behler. Constructing high-dimensional neural network potentials: A tutorial review. *International Journal of Quantum Chemistry*, 115(16):1032–1050, 2015.

[16] Jörg Behler. Perspective: Machine learning potentials for atomistic simulations. *The Journal of Chemical Physics*, 145(17):170901, 11 2016.

[17] Thomas B. Blank, Steven D. Brown, August W. Calhoun, and Douglas J. Doren. Neural network models of potential energy surfaces. *The Journal of Chemical Physics*, 103(10):4129–4137, 09 1995.

[18] L. C. Blum and J.-L. Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database GDB-13. *J. Am. Chem. Soc.*, 131:8732, 2009.

[19] Erik Lien Bolager, Iryna Burak, Chinmay Datar, Qing Sun, and Felix Dietrich. Sampling weights of deep neural networks, 2023.

[20] M. Born and R. Oppenheimer. Zur quantentheorie der molekeln. *Annalen der Physik*, 389(20):457–484, 1927.

[21] Felix Brockherde, Leslie Vogt, Li Li, Mark E. Tuckerman, Kieron Burke, and Klaus-Robert Müller. Bypassing the kohn-sham equations with machine learning. *Nature Communications*, 8(1):872, Oct 2017.

[22] Hans-Joachim Bungartz and Michael Griebel. Sparse grids. *Acta Numerica*, 13:147–269, 2004.

[23] Kieron Burke and Lucas O. Wagner. Dft in a nutshell. *International Journal of Quantum Chemistry*, 113(2):96–101, 2013.

[24] Eric Cancès and Gero Friesecke. *Density functional theory: Modeling, mathematical analysis, computational methods, and applications*. Springer Nature, Cham, Switzerland, July 2023.

[25] David A. Case, Hasan Metin Aktulga, Kellon Belfon, David S. Cerutti, G. Andrés Cisneros, Vinícius Wilian D. Cruzeiro, Negin Forouzesh, Timothy J. Giese, Andreas W. Götz, Holger Gohlke, Saeed Izadi, Koushik Kasavajhala, Mehmet C. Kaymak, Edward King, Tom Kurtzman, Tai-Sung Lee, Pengfei Li, Jian Liu, Tyler Luchko, Ray Luo, Madushanka Manathunga, Matias R. Machado, Hai Minh Nguyen, Kurt A. O'Hearn, Alexey V. Onufriev, Feng Pan, Sergio Pantano, Ruxi Qi, Ali Rahnamoun,

Ali Risheh, Stephan Schott-Verdugo, Akhil Shajan, Jason Swails, Junmei Wang, Haixin Wei, Xiongwu Wu, Yongxian Wu, Shi Zhang, Shiji Zhao, Qiang Zhu, Thomas E. Cheatham III, Daniel R. Roe, Adrian Roitberg, Carlos Simmerling, Darrin M. York, Maria C. Nagan, and Kenneth M. Merz Jr. Ambertools. *Journal of Chemical Information and Modeling*, 63(20):6183–6191, Oct 2023.

[26] Stefan Chmiela, Huziel E. Sauceda, Klaus-Robert Müller, and Alexandre Tkatchenko. Towards exact molecular dynamics simulations with machine-learned force fields. *Nature Communications*, 9(1):3887, 2018.

[27] Stefan Chmiela, Huziel E. Sauceda, Alexandre Tkatchenko, and Klaus-Robert Müller. *Accurate molecular dynamics enabled by efficient physically-constrained machine learning approaches*, pages 129–154. Springer International Publishing, 2020.

[28] Stefan Chmiela, Alexandre Tkatchenko, Huziel E. Sauceda, Igor Poltavsky, Kristof T. Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*, 3(5):e1603015, 2017.

[29] Stefan Chmiela, Valentin Vassilev-Galindo, Oliver T. Unke, Adil Kabylda, Huziel E. Sauceda, Alexandre Tkatchenko, and Klaus-Robert Müller. Accurate global machine learning force fields for molecules with hundreds of atoms. *Science Advances*, 9(2):eadf0873, 2023.

[30] R.H. Dennard, F.H. Gaensslen, Hwa-Nien Yu, V.L. Rideout, E. Bassous, and A.R. LeBlanc. Design of ion-implanted mosfet's with very small physical dimensions. *IEEE Journal of Solid-State Circuits*, 9(5):256–268, 1974.

[31] Volker L. Deringer, Miguel A. Caro, and Gábor Csányi. Machine learning interatomic potentials as emerging tools for materials science. *Advanced Materials*, 31(46):1902765, 2019.

[32] Felix Dietrich, Alexei Makeev, George Kevrekidis, Nikolaos Evangelou, Tom Bertalan, Sebastian Reich, and Ioannis G. Kevrekidis. Learning effective stochastic differential equations from microscopic simulations: linking stochastic numerics to deep learning, 2022.

[33] Marco Eckhoff and Markus Reiher. Lifelong machine learning potentials. *Journal of Chemical Theory and Computation*, 19(12):3509–3525, 2023. PMID: 37288932.

[34] R.P. Feynman, R.B. Leighton, and M. Sands. *The Feynman Lectures on Physics, Vol. I: The New Millennium Edition: Mainly Mechanics, Radiation, and Heat*. Number v. 1. Basic Books, 2015.

[35] R.P. Feynman, R.B. Leighton, and M. Sands. *The Feynman Lectures on Physics, Vol. III: The New Millennium Edition: Quantum Mechanics*. Number v. 3. Basic Books, 2015.

[36] Johannes Gasteiger, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 6790–6802. Curran Associates, Inc., 2021.

[37] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017.

[38] Xavier Gonze, Bernard Amadon, Gabriel Antonius, Frédéric Arnardi, Lucas Baguet, Jean-Michel Beuken, Jordan Bieder, François Bottin, Johann Bouchet, Eric Bousquet, Nils Brouwer, Fabien Bruneval, Guillaume Brunin, Théo Cavignac, Jean-Baptiste Charraud, Wei Chen, Michel Côté, Stefaan Cottenier, Jules Denier, Grégory Geneste, Philippe Ghosez, Matteo Giantomassi, Yannick Gillet, Olivier Gingras, Donald R. Hamann, Geoffroy Hautier, Xu He, Nicole Helbig, Natalie Holzwarth, Yongchao Jia, François Jollet, William Lafargue-Dit-Hauret, Kurt Lejaeghere, Miguel A. L. Marques, Alexandre Martin, Cyril Martins, Henrique P. C. Miranda, Francesco Naccarato, Kristin Persson, Guido Petretto, Valentin Planes, Yann Pouillon, Sergei Prokhorenko, Fabio Ricci, Gian-Marco Rignanese, Aldo H. Romero, Michael Marcus Schmitt, Marc Torrent, Michiel J. van Setten, Benoit Van Troeye, Matthieu J. Verstraete, Gilles Zérah, and Josef W. Zwanziger. The abinit project: Impact, environment and recent developments. *Comput. Phys. Commun.*, 248:107042, 2020.

[39] E Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration*. Springer Series in Computational Mathematics. Springer, Berlin, Germany, 2 edition, December 2006.

[40] Jiequn Han, Linfeng Zhang, Roberto Car, and Weinan E. Deep potential: A general representation of a many-body potential energy surface. *Communications in Computational Physics*, 23(3), 2018.

[41] D. R. Hartree. The wave mechanics of an atom with a non-coulomb central field. part ii. some results and discussion. *Mathematical Proceedings of the Cambridge Philosophical Society*, 24(1):111–132, 1928.

[42] Philipp Hennig, Michael A. Osborne, and Hans P. Kersting. *Probabilistic Numerics: Computation as Machine Learning*. Cambridge University Press, 2022.

[43] Philipp Hennig, Michael A Osborne, and Hans P Kersting. *Probabilistic numerics: Computation as machine learning*. 2022.

[44] Michael F Herbst and Antoine Levitt. Black-box inhomogeneous preconditioning for self-consistent field iterations in density functional theory. *Journal of Physics: Condensed Matter*, 33(8):085503, dec 2020.

[45] Michael F. Herbst, Antoine Levitt, and Eric Cancès. DFTK: The Density-functional toolkit.

[46] Michael F. Herbst, Benjamin Stamm, Stefan Wessel, and Matteo Rizzi. Surrogate models for quantum spin systems based on reduced-order modeling. *Phys. Rev. E*, 105:045303, Apr 2022.

[47] Lauri Himanen, Marc O.J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Yashasvi S. Ranawat, David Z. Gao, Patrick Rinke, and Adam S. Foster. Dscribe: Library of descriptors for machine learning in materials science. *Computer Physics Communications*, 247:106949, 2020.

[48] Jerry L. Hintze and Ray D. Nelson. Violin plots: A box plot-density trace synergism. *The American Statistician*, 52(2):181–184, 1998.

[49] Marlis Hochbruck and Alexander Ostermann. Exponential integrators. *Acta Numerica*, 19:209–286, 2010.

[50] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36(3):1171 – 1220, 2008.

[51] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136:B864–B871, Nov 1964.

[52] Eyke Hüllermeier, Thomas Fober, and Marco Mernberger. *Inductive Bias*, pages 1018–1018. Springer New York, New York, NY, 2013.

[53] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks, 2020.

[54] Frank Jensen. *Introduction to computational chemistry*. John Wiley & Sons, Nashville, TN, 2 edition, January 2007.

[55] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[56] Andreas Kirsch. *An introduction to the mathematical theory of inverse problems*. Applied Mathematical Sciences. Springer, New York, NY, April 2013.

[57] Stefan Klus, Feliks Nüske, Sebastian Peitz, Jan-Hendrik Niemann, Cecilia Clementi, and Christof Schütte. Data-driven approximation of the koopman generator: Model reduction, system identification, and control. *Physica D: Nonlinear Phenomena*, 406:132416, 2020.

[58] Andrew V. Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM Journal on Scientific Computing*, 23(2):517–541, 2001.

[59] Emir Kocer, Tsz Wai Ko, and Jörg Behler. Neural network potentials: A concise overview of methods, 2021.

[60] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133–A1138, Nov 1965.

[61] Arthur Kosmala, Johannes Gasteiger, Nicholas Gao, and Stephan Günnemann. Ewald-based long-range message passing for molecular graphs, 2023.

[62] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[63] L D Landau and E M Lifshitz. *Mechanics*. Elsevier, January 1982.

[64] Ben Leimkuhler and Charles Matthews. *Molecular dynamics*. Interdisciplinary Applied Mathematics. Springer International Publishing, Basel, Switzerland, 2015 edition, May 2015.

[65] Lin Lin, Jianfeng Lu, and Lexing Ying. Numerical methods for kohn–sham density functional theory. *Acta Numerica*, 28:405–539, 2019.

[66] Christopher M. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, NY, August 2016.

[67] David John Cameron MacKay. Introduction to gaussian processes. 1998.

[68] Rahul Manavalan. RandomFeaturePotentials.jl - A julia package for force field surrogates based on random features., January 2023.

[69] Benoit Minisini, Patrick Bonnaud, Qiuping A. Wang, and François Tsobnang. Dft evaluation of thermomechanical properties of scheelite type mlif4 (m=la, ce, pr, nd, pm, sm, gd, tb, dy, ho, er, tm, lu). *Computational Materials Science*, 42(1):156–160, 2008.

[70] Grégoire Montavon, Katja Hansen, Siamac Fazli, Matthias Rupp, Franziska Biegler, Andreas Ziehe, Alexandre Tkatchenko, Anatole Lilienfeld, and Klaus-Robert Müller. Learning invariant representations of molecules for atomization energy prediction. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[71] Ryan L. Murphy, Balasubramaniam Srinivasan, Vinayak Rao, and Bruno Ribeiro. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs, 2019.

[72] Felix Musil, Andrea Grisafi, Albert P. Bartók, Christoph Ortner, Gábor Csányi, and Michele Ceriotti. Physics-inspired structural representations for molecules and materials. *Chemical Reviews*, 121(16):9759–9815, Aug 2021.

[73] Nicholas H. Nelsen and Andrew M. Stuart. The random feature model for input-output maps between banach spaces. *SIAM Journal on Scientific Computing*, 43(5):A3212–A3243, 2021.

[74] Yurii Nesterov. A method for solving the convex programming problem with convergence rate o(1/k^2). $Proceedings of the USSR Academy of Sciences, 269 : 543 - -547, 1983$.

[75] Tommaso Nottoli, Ivan Giannì, Antoine Levitt, and Filippo Lipparini. A robust, open-source implementation of the locally optimal block preconditioned conjugate gradient for large eigenvalue problems in quantum chemistry. *Theoretical Chemistry Accounts*, 142(8):69, Jul 2023.

[76] Houman Owhadi and Gene Ryan Yoo. Kernel flows: From learning kernels from data into the abyss. *Journal of Computational Physics*, 389:22–47, July 2019.

[77] Houman Owhadi and Lei Zhang. Gamblets for opening the complexity-bottleneck of implicit schemes for hyperbolic and parabolic odes/pdes with rough coefficients. *Journal of Computational Physics*, 347:99–128, 2017.

[78] Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, 60(3):550–591, 2018.

[79] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A. Smith, and Lingpeng Kong. Random feature attention, 2021.

[80] Wiktor Pronobis, Alexandre Tkatchenko, and Klaus-Robert Müller. Many-body descriptors for predicting molecular properties with machine learning: Analysis of pairwise and three-body interactions in molecules. *Journal of Chemical Theory and Computation*, 14(6):2991–3003, Jun 2018.

[81] I.G. KEVREKIDIS M.C. KUBE R. RICO-MARTÍNEZ, K. KRISCHER and J.L. HUDSON. Discrete- vs. continuous-time nonlinear signal processing of cu electrodissolution data. *Chemical Engineering Communications*, 118(1):25–48, 1992.

[82] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, page 1177–1184, Red Hook, NY, USA, 2007. Curran Associates Inc.

[83] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008.

[84] Carl Edward Rasmussen and Christopher K I Williams. *Gaussian processes for machine learning*. Adaptive Computation and Machine Learning Series. MIT Press, London, England, June 2019.

[85] Severin Reiz, Tobias Neckel, and Hans-Joachim Bungartz. Neural nets with a newton conjugate gradient method on multiple gpus, 2022.

[86] Lars Ruddigkeit, Ruud van Deursen, Lorenz C. Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database gdb-17. *Journal of Chemical Information and Modeling*, 52(11):2864–2875, Nov 2012.

[87] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*, 108(5), January 2012.

[88] Michael T. Schaub, Yu Zhu, Jean-Baptiste Seby, T. Mitchell Roddenberry, and Santiago Segarra. Signal processing on higher-order networks: Livin' on the edge... and beyond. *Signal Processing*, 187:108149, October 2021.

[89] Baochao Shan, Songze Chen, Zhaoli Guo, and Peng Wang. Pore-scale study of non-ideal gas dynamics under tight confinement considering rarefaction, denseness and molecular interactions. *Journal of Natural Gas Science and Engineering*, 90:103916, 2021.

[90] R Shankar. *Principles of quantum mechanics*. Springer, New York, NY, 2 edition, December 2012.

[91] Aman Sinha and John C Duchi. Learning kernels with random features. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[92] Ganesh Sivaraman, Anand Narayanan Krishnamoorthy, Matthias Baur, Christian Holm, Marius Stan, Gábor Csányi, Chris Benmore, and Álvaro Vázquez-Mayagoitia. Machine-learned interatomic potentials by active learning: amorphous and liquid hafnium dioxide. *npj Computational Materials*, 6(1):104, Jul 2020.

[93] Lloyd N Trefethen and David Bau. *Numerical Linear Algebra: Twenty-fifth anniversary edition*. SIAM, June 2022.

[94] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[95] Edward Wagstaff, Fabian B. Fuchs, Martin Engelcke, Ingmar Posner, and Michael Osborne. On the limitations of representing functions on sets, 2019.

[96] Jian Wang, Siyuan Lu, Shui-Hua Wang, and Yu-Dong Zhang. A review on extreme learning machine. *Multimedia Tools and Applications*, 81(29):41611–41660, Dec 2022.

[97] Steven R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69:2863–2866, Nov 1992.

[98] Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A.C 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1):160018, Mar 2016.

[99] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets, 2018.

[100] Linfeng Zhang, Jiequn Han, Han Wang, Wissam A. Saidi, Roberto Car, and Weinan E. End-to-end symmetry preserving inter-atomic potential energy model for finite and extended systems, 2018.

[101] Hao Zhou, Ya-Juan Feng, Chao Wang, Teng Huang, Yi-Rong Liu, Shuai Jiang, Chun-Yu Wang, and Wei Huang. A high-accuracy machine-learning water model for exploring water nanocluster structures. *Nanoscale*, 13:12212–12222, 2021.

[102] C. Lawrence Zitnick, Lowik Chanussot, Abhishek Das, Siddharth Goyal, Javier Heras-Domingo, Caleb Ho, Weihua Hu, Thibaut Lavril, Aini Palizhati, Morgane Riviere, Muhammed Shuaibi, Anuroop Sriram, Kevin Tran, Brandon Wood, Junwoong Yoon, Devi Parikh, and Zachary Ulissi. An introduction to electrocatalyst design using machine learning for renewable energy storage, 2020.

Appendix

Implementation and Experiments

A complete implementation of the work in this thesis is made available here [68]. This includes implementations of algorithm 4, algorithm 5 along with a re-implementation of vanilla RFNN from [19]. Furthermore, the reults of the experiments can be reproduced from the scripts provided there.

It is natural to enquire why a re-implementation was necessary. This is based on the consideration that [19]'s implementation used *numpy*; which is less flexible when compared with the least squares solvers used in the Julia ecosystem. As detailed in section 4.1, there may be applications where a large number of neurons are needed to approximate certain functions. In this case, Julia's GPU ecosystem tools can be seamlessly interchanged; while the implementation from [19] needs to be re-implemented using different libraries.

Miscellaneous

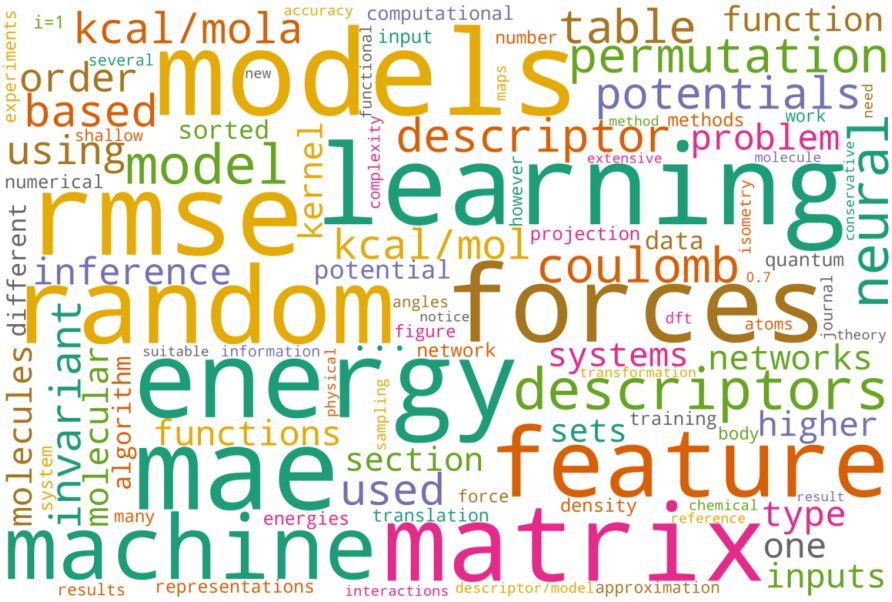

Figure 1: A frequency based wordcloud for the text in this thesis.