BROWN UNIVERSITY

UNDERGRADUATE HONORS THESIS

Force-Explicit Machine Learning Schemes and Interactive Visualization for Atomic Simulations

---

Author:
Kaley BRAUER

Advisors:
Prof. Andrew A. PETERSON
Prof. John Bradley MARSTON

A thesis submitted in fulfillment of the requirements for the degree of Physics Bachelor of Science.

May 4, 2017

Brown University
Catalyst Design Lab

Abstract

Physics Bachelor of Science

Force-Explicit Machine Learning Schemes and Interactive Visualization for Atomic Simulations

by Kaley BRAUER

Quick, direct access to atomic force calculations is essential for efficient geometry optimizations and molecular dynamics simulations, but current machine learning simulations treat force calculations as secondary to energy calculations. For situations when the force is desired more than the energy, a force-explicit machine learning scheme could be used to obtain atomic forces more quickly.

We propose two force-explicit machine learning schemes and compare them to each other and to a previously proposed scheme by Botu and Ramprasad (2015) (and, independently, by Li, Kermode, and De Vita (2015)). The first proposed scheme is based on the derivative of the Gaussian "fingerprint" (a numerical representation of an atom's local environment) that was originally suggested by Behler and Parrinello (2007). This scheme incorporates two-body and three-body interactions in its representation of local environments. The second proposed scheme creates a rotationally invariant force-explicit regression model by using an intrinsic, natural coordinate system (in this case, the principal axes of the moment of inertia tensor of each atom's local environment) as the basis for the model. From testing on two example atomic systems, we find that the scheme based on the derivative Gaussian fingerprint performs best overall. The scheme that uses the moments of inertia basis performs sometimes worse and sometimes better than the previously proposed scheme, and more testing (specifically on a rotating atomic system) is required to determine if this basis is preferable or not in general.

A force-explicit machine learning scheme could allow for robust simulation with "real-time" force predictions in response to perturbations. To demonstrate how such a simulation might work and lay the groundwork, we developed a prototype of interactive atomic simulation software called "The Interactive Atomistic Environment" (IntAE). We also created different multivariate visualization options and interviewed ten potential users to evaluate both the effectiveness of the visualization options and the system as a whole, determining that visualizations which encoded variables into each atomic sphere were more intuitive and thus nearly unanimously preferred over visualizations which changed the coordinates of the space in which the atomic system is displayed.

Contents

Chapter 1

Introduction

1.1 Brief Background on Atomistic Calculations

Calculating forces between atoms is an important part of studying simulated atomic and molecular systems. When systems become large and complicated, however, calculating the forces between atoms can become computationally intensive and often analytically impossible. The Hellmann-Feynman theorem (which shows that the force on an atom is the same as the expectation value of the derivative of the Hamiltonian with respect to the atom's position) can be used in conjunction with the Schrödinger equation or electronic structure methods to determine atomic forces (Hellmann, 1937; Feynman, 1939), but these methods quickly become impractical when the systems of interest have a large number of atoms of different types.

The many-body Schrödinger equation can only be solved exactly for simple systems such as a single hydrogen atom. Since many systems of interest are much more complicated than a single atom, various alternative electronic structure methods have been developed to accurately approximate energies and forces, including perturbation theory, Kohn–Sham density functional theory, and Hartree-Fock theory (Cramer, 2013). All of these, especially density functional theory, have seen significant success in facilitating study of atomic systems (Jones, 2015), but the computational cost of these methods typically grows nonlinearly with size. Additionally, the number of possible geometrical configurations and compositions increases exponentially with the number and identities of atoms in the system. Because of these scaling issues and limitations in current computational facilities, electronic structure methods are impractical for studying systems with more than a few hundred atoms.

To address this problem, many attempts have been made to develop models that can quickly approximate energies and forces of even large systems. Assuming that each atomic energy contribution only depends on the local chemical environment allows a model to learn from the energetics of many local environments and then predict the energetics of other systems with similar local environments. Atomistic Machine-learning Package (Amp), a modular, open-source software package developed here at Brown (Khorshidi and Peterson, 2016), uses

relatively fast machine-learning models to interpolate the potential energy surface of a reference data set (referred to as the "training" data set) which has been prepared by a parent electronic structure calculator. The trained model can then predict the potential energy surface of new atomic systems with similar local environments. After predicting the potential energy, Amp also calculates the forces on each atom by differentiating the energy at each atomic position. More information about how Amp predicts the potential energy surface of atomic systems can be found in Section 2.1.

1.2 Motivation

Regression models based on machine learning (such as those used in Amp) allow for a general framework to predict atomic energies and forces in arbitrarily complicated systems. The currently developed regression models, however, focus on predicting the potential energy surface and then differentiating to obtain the forces. While this works well in most situations, having quicker and more direct access to atomic forces can allow for more efficient geometry optimizations and molecular dynamics simulations. In particular, this thesis will partially focus on how force-explicit machine learning models should be able to give sufficiently quick force predictions to allow for molecular dynamics simulations with real-time interactivity. Additionally, a model which directly trains on and predicts forces may be easier to fit than a model that attempts to predict the potential energy surface.

Once a regression model in Amp has been trained, obtaining forces for new systems can take as little as a few seconds, but even a few seconds is too long for real-time interaction without a feeling that the simulation is lagging. By cutting out the differentiation step and skipping straight to predicting the forces, a force-explicit model in the Amp framework should be able to return atomic forces almost immediately.

Currently, robust atomic simulation software with real-time interactivity does not exist, but such an environment would likely be useful for facilitating understanding of atomistic-level phenomena in both an educational and a research capacity. Educational chemical laboratories with interactivity have been shown to help students understand material. Shin et al. (2002) developed a Web-based, interactive virtual laboratory system for unit operations and process systems engineering education, and Stieff and Wilensky (2003) created an educational chemistry modeling and simulation package called 'Connected Chemistry'. Both of these studies found that students gained a better understanding of the material by interacting with the chemical simulations. While these simulations were only tested in an educational environment, this lends support to our belief that a robust interactive simulation could be used in both education and research to further understanding of the atomic systems depicted in the simulations.

Before a robust interactive atomic simulation is possible, force-explicit machine learning schemes must be developed. Botu and Ramprasad (2015) put forth an initial scheme for directly predicting atomic forces in a machine learning framework, and we expand on this work by investigating their scheme and two novel schemes. The theory behind the different schemes and our machine learning model can be found in Chapter 2, and the details of how the different schemes perform can be found in Chapter 3.

In addition to investigating the abilities and efficiencies of different force-explicit schemes, we created a prototype of a robust, interactive, multivariate atomic simulation visualization software called IntAE (Interactive Atomistic Environment). This prototype was created as a potential application of force-explicit machine learning models, and is modular to be extensible as force-explicit models are developed. In addition to the simulation being interactive (allowing for click-and-drag interactivity with the atoms), we focused on providing multivariate visualization. Intuitive scientific visualization is important to ensure that information in simulations is being expressed efficiently, so we included different visualization options in the software and interviewed potential users to determine their visualization preferences and overall feedback. A description of the software and the evaluation interviews can be found in Chapter 4.

The contributions of this thesis are thus (1) a description of two novel force-explicit machine learning schemes which could be broadly applicable to geometry optimizations and molecular dynamics simulations, (2) an investigation and comparison of the novel schemes and a previously established scheme, (3) an open-source prototype of interactive, multivariate atomic simulation visualization software, and (4) a brief evaluation of multivariate visualization options in atomic simulations.

Chapter 2

Theory

2.1 Amp's Energy-Explicit Scheme

Regression models based on machine learning can be used to accurately approximate potential energies and forces in large atomic systems more quickly than electronic structure methods (such as density functional theory) whose computational costs increase significantly with system size. Atomistic Machine-learning Package (Amp) successfully trains models to predict the potential energy surfaces of atomic systems using a "descriptor" to map the atomic positions to feature vectors (also called "fingerprints") that is used as input into a regression model. The fingerprint for a given atom is named such because it is an identifier of the local environment around the atom. Fingerprints for each atom in an atomic system are input into the regression model, and the model outputs the potential energy surface (or, in the case of force-explicit models, it directly outputs the forces on each atom).

Amp includes three different descriptors: Gaussian, Zernike, and Bispectrum, with Gaussian fingerprints being the default. The three descriptors are described in detail in Khorshidi and Peterson (2016), but I will summarize the Gaussian here. The Gaussian fingerprint was originally suggested by Behler and Parrinello (2007). The fingerprint for atom i, $\mathbf{G_i}$, has two subvectors, $\mathbf{G_i^I}$ and $\mathbf{G_i^{II}}$, respectively representing two-body and three-body interactions with neighboring atoms. The components of $\mathbf{G_i}$ are denoted f_i^I:

$$f_i^I = \sum_{\substack{\text{atoms } j \text{ within } R_c \\ \text{distance of atom } i \\ j \neq i}} e^{-\eta(R_{ij}-R_s)^2/R_c^2} f_c(R_{ij}) \tag{2.1}$$

where $R_{ij} = ||\mathbf{R_j} - \mathbf{R_i}||$ is the distance between atoms i and j, η and R_s are width and center parameters which are varied several times to create the fingerprint vector, and R_c is a cutoff radius that defines the area of interest around atom i.

f_c is a cutoff function that smoothly goes to zero as R_{ij} goes to R_c. The cutoff function is required to be smooth so that no discontinuities arise in the fingerprints. Both f_c and its first derivative must be continuous to ensure that no singularities appear when the energy-explicit model is used to predict forces.

Three-body interactions are described in $\mathbf{G_i^{II}}$ via components denoted f_i^{II}:

$$f_i^{II} = 2^{1-\zeta} \sum_{\substack{j,k \neq i \\ (j \neq k)}}^{\substack{\text{atoms } j,\, k \text{ within } R_c \\ \text{distance of atom } i}} (1 + \lambda \cos\theta_{ijk})^{\zeta} e^{-\eta(R_{ij}^2 + R_{ik}^2)/R_c^2} \\ \text{x } f_c(R_{ij}) f_c(R_{ik}) \tag{2.2}$$

where $\theta_{ijk} = \cos^{-1}(\mathbf{R_{ij}}.\mathbf{R_{ik}}/(R_{ij}R_{ijk}))$ is the angle between all possible sets of atoms i, j, and k centered on atom i within the cutoff radius, and ζ, λ, and η are all parameters that are varied several times to create the fingerprint vector.

This scheme works well when one is directly predicting potential energy, but it involves only distance and angle magnitudes and does not explicitly retain information about the direction of the distances between atoms. When predicting a scalar energy, we can use a model that does not explicitly retain directionality, but directionality is needed to efficiently predict force vectors. We have thus endeavored to investigate several different possible schemes with descriptors that retain directionality.

2.2 Force-Explicit Fingerprints

The development of a scheme that can numerically describe, or "fingerprint", atomic environments while retaining directionality is central to the goal of creating force-explicit machine learning models. Such a scheme should differentiate between different configurations, be rotationally and translationally invariant, and ideally be relatively simple.

2.2.1 Botu-Ramprasad

Botu and Ramprasad (2015) proposed this fingerprint function to represent the k^{th} component of the force on atom i:

$$V_i^k = \sum_{j \neq i}^{\substack{\text{atoms } j \text{ within } R_c \\ \text{distance of atom } i}} \frac{R_{ij}^k}{R_{ij}} e^{-(R_{ij}/\eta)^2} f_c(R_{ij}) \tag{2.3}$$

where R_{ij}^k is the scalar projection of R_{ij} along the direction k and η is a parameter that governs the extent of coordination around atom i that needs to be captured. As in Amp's fingerprints, the η parameter is varied several times to create different components of the feature vector for each V_i^k. Also, once again the cutoff / damping function f_c goes to zero as R_{ij} approaches R_c (where R_c is the cutoff radius that defines the area of interest around each atom):

$$f_c(R_{ij}) = \begin{cases} 0.5[\cos(\frac{\pi R_{ij}}{R_c}) + 1] & \text{, if } R_{ij} \leq R_c \\ 0 & \text{, otherwise} \end{cases} \tag{2.4}$$

This fingerprint requires three k-components along non-parallel directions; a natural choice is the x, y, and z directions of your coordinate system. Additionally, the fingerprint leads to symmetry-adapted forces. For example, if an atom is in a centrosymmetric position and thus experiences no force, its fingerprint will consist of all zeros.

Li, Kermode, and De Vita (2015) independently proposed a similar fingerprint.

$$V^k_{i,Li} = \sum_{j \neq i} \frac{R^k_{ij}}{R_{ij}} e^{-(R_{ij}/R_c)^p} \tag{2.5}$$

where R_c and p are varied. By selecting $p = 2$ and including a damping function, this fingerprint is the same as that proposed by Botu and Ramprasad.

At a glance, equation (2.3) resembles equation (2.1), the f^I_i functions from $\mathbf{G^I_i}$ in the Behler Gaussian fingerprint. In fact, the Botu-Ramprasad fingerprint closely matches the derivative of the components of $\mathbf{G^I_i}$. Ignoring the cutoff function and the R_s parameter for the moment,

$$f^I_i = \sum_{j \neq i} e^{-\frac{\eta}{R_c^2}(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2})^2} \tag{2.6}$$

$$\begin{aligned} \frac{df^I_i}{dx_i} &= \sum_{j \neq i} \frac{-\eta}{R_c^2} 2(x_i - x_j) e^{-\frac{\eta}{R_c^2} R^2_{ij}} \\ &= \frac{-2\eta}{R_c^2} \sum_{j \neq i} R^k_{ij} e^{-\frac{\eta}{R_c^2} R^2_{ij}} \end{aligned} \tag{2.7}$$

When you multiply $\frac{df^I_i}{dx_i}$ by a cutoff function, you get almost the same functional form as the Botu-Ramprasad fingerprint. Botu-Ramprasad has an additional factor of $1/R_{ij}$. (Note: When testing the schemes, we briefly investigated whether including or excluding this factor resulted in different results. We found that the results were about the same; sometimes slightly better, sometimes slightly worse. This was not conclusive or intensely investigated, though.)

Because forces are obtained by taking derivatives of energy, having a force-explicit fingerprint that is the derivative of an accepted energy-explicit fingerprint makes intuitive sense. Thus, while Botu and Ramprasad do not appear to have created this fingerprint while thinking about the derivative of the Behler Gaussian fingerprint, the Botu-Ramprasad fingerprint's similarity to the derivative of f^I_i gives it some additional theoretical justification.

2.2.2 Gaussian Derivative

Inspired by the similarity of the Botu fingerprint to the derivative of one component of the Gaussian fingerprint, we decided to adapt the Botu fingerprint to make it an exact derivative and to use this insight as a tool to create three-body force-explicit fingerprints, which to our knowledge have not previously been produced. This fingerprint (occasionally shortened to "dG" in this thesis) has two components: $V_i^{I,k}$, essentially the Botu-Ramprasad fingerprint, and $V_i^{II,k}$, created from the derivative of the f_i^{II} function in the Behler Gaussian fingerprint.

Temporarily ignoring the cutoff function again,

$$f_i^{II} = 2^{1-\zeta} \sum_{\substack{j,k\neq i \\ (j\neq k)}} (1+\lambda\cos\theta_{ijk})^{\zeta} e^{-\eta(\frac{R_{ij}^2+R_{ik}^2}{R_c^2})} \tag{2.8}$$

$$\begin{aligned} \frac{df_i^{II}}{dx_i} =& 2^{1-\zeta} \sum_{\substack{j,k\neq i \\ (j\neq k)}} (1+\lambda\cos\theta_{ijk})^{\zeta-1} [\zeta\lambda e^{-\eta(\frac{R_{ij}^2+R_{ik}^2}{R_c^2})} \frac{d(\cos\theta_{ijk})}{dx_i} \\ &+ (1+\lambda\cos\theta_{ijk}) \frac{d(e^{-\eta(\frac{R_{ij}^2+R_{ik}^2}{R_c^2})})}{dx_i}] \end{aligned} \tag{2.9}$$

Remembering that $\cos\theta_{ijk} = \frac{\mathbf{R_{ij}}.\mathbf{R_{ik}}}{R_{ij}R_{ijk}}$, taking the derivatives, and then rearranging, we obtain:

$$\begin{aligned} \frac{df_i^{II}}{dx_i} =& 2^{1-\zeta} \sum_{\substack{j,k\neq i \\ (j\neq k)}} (1+\lambda\cos\theta_{ijk})^{\zeta-1} e^{-\eta(\frac{R_{ij}^2+R_{ik}^2}{R_c^2})} \\ &\text{x } [\zeta\lambda \frac{\alpha_{ijk}^x + \alpha_{ikj}^x}{R_{ij}^2 R_{ik}^2} - \frac{2\eta}{R_c^2}(R_{ij}^x + R_{ik}^x)(1+\lambda\cos\theta_{ijk})] \end{aligned} \tag{2.10}$$

where $\alpha_{ijk}^x = R_{ij}^x(R_{ij}R_{ik} - R_{ik}^2\cos\theta_{ijk})$ is defined to allow for a more succinct expression for the derivative. Generalizing to any $k = x, y$, or z and including the cutoff functions:

$$\begin{aligned} V_i^{II,k} =& 2^{1-\zeta} \sum_{\substack{j,l\neq i \\ (j\neq l)}} (1+\lambda\cos\theta_{ijl})^{\zeta-1} e^{-\eta(\frac{R_{ij}^2+R_{il}^2}{R_c^2})} \\ &\text{x } [\zeta\lambda \frac{\alpha_{ijl}^k + \alpha_{ilj}^k}{R_{ij}^2 R_{il}^2} - \frac{2\eta}{R_c^2}(R_{ij}^k + R_{il}^k)(1+\lambda\cos\theta_{ijl})] \\ &\text{x } f_c(R_{ij}) f_c(R_{il}) \end{aligned} \tag{2.11}$$

This expression is much lengthier than the other proposed fingerprint functions, but its computational run time is not significantly longer because R_{ij}, R_{il},

and $\cos\theta_{ijl}$ can all be calculated once and reused. Additionally, its relation to the $\mathbf{G_i^{II}}$ in the Behler Gaussian fingerprint makes it promising as a way to represent three-body interactions in atomic local environments.

As mentioned, this function ($V_i^{II,k}$) would be used in conjunction with the adjusted Botu-Ramprasad fingerprint (denoted $V_i^{I,k}$):

$$F_i^k = \left[V_i^{I,k}(\eta_1), ..., V_i^{I,k}(\eta_n), V_i^{I,k}(\zeta_1, \lambda_1, \eta_1), ..., V_i^{I,k}(\zeta_n, \lambda_n, \eta_n)\right] \tag{2.12}$$

where the feature vector includes several components of $V_i^{I,k}$ and $V_i^{I,k}$, varying the η, ζ, and λ parameters.

Because equation 2.11 is not particularly simple, a possible alternative would be to use a simpler function with a similar functional form. For example, fingerprint functions that drop the term with α's or the term with $\frac{2\eta}{R_c^2}$ could be tested and compared to each other and to the full function. Some of our minor testing suggests that the term with α's is less important than the term with $\frac{2\eta}{R_c^2}$, but this testing was not stringent. For the purposes of this thesis, we kept the entire function when implementing this fingerprint.

2.2.3 Moments of Inertia

We also developed an alternative approach based on an initial transformation out of Cartesian coordinates. The previous fingerprints require any three non-parallel directions, typically taken to be the x, y, and z of your coordinate space. Rather than fingerprinting for any arbitrary three directions, however, it might be more natural and fruitful to fingerprint along the directions of an intrinsic, natural coordinate system. Such a model would thus not depend on the coordinate system we happened to choose for our atomic system. For this force-explicit scheme, we chose to use the principal axes of the moment of inertia tensor of each local environment to give us an intrinsic coordinate system that can be used as the basis for a regression model.

The principal moment of inertia axes (MOI axes) are determined for the rigid system of particles in the local environment around each atom. If we are determing the MOI axes for atom i, a sphere of cutoff radius R_c is drawn around the atom and we "collect" all of the atoms in the local environment. We then calculate the principal axes for that sub-system about the position of atom i (rather about than the center of mass). Figure 2.1 illustrates how this looks. Mathematically, each tensor is calculated:

$$I_{\text{atom i}} = \sum_{j=1}^{N} m_j[(\mathbf{r_j} * \mathbf{r_j})\mathbf{E} - \mathbf{r_j} \otimes \mathbf{r_j}] \tag{2.13}$$

where N is the number of atoms in the local environment around atom i, m_j is the mass of atom j, $\mathbf{r_j}$ is the distance of atom j from atom i (the center of our

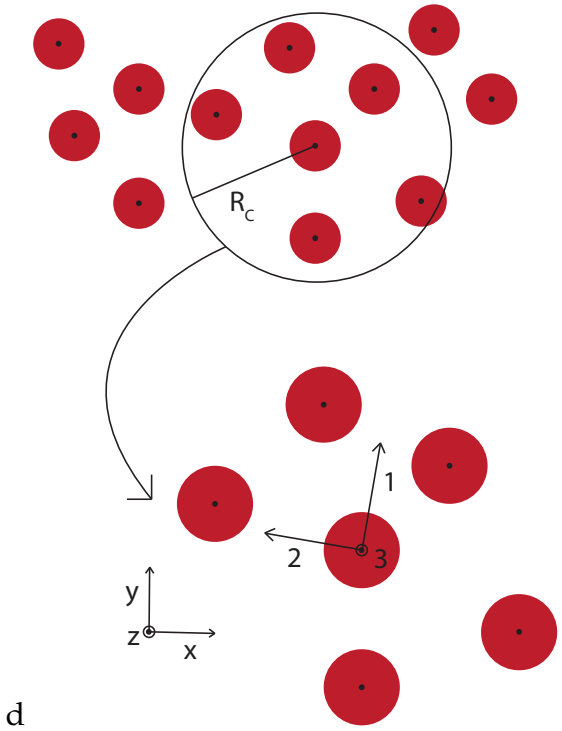


FIGURE 2.1: This diagram shows how a local environment is selected (based on a given cutoff radius, R_c) and then the principal axes (of the moment of inertia tensor) of the local environment are calculated. The machine learning model is based around the force components in the MOI basis, not in the x, y, z basis. Note: These axes are drawn as an example and are not necessarily correct for the pictured system.

local environment), and $\mathbf{E}$ is the identity tensor.

$$\mathbf{E} = \mathbf{e_1} \otimes \mathbf{e_1} + \mathbf{e_2} \otimes \mathbf{e_2} + \mathbf{e_3} \otimes \mathbf{e_3} \tag{2.14}$$

where $\mathbf{e_i}$, $i = 1, 2, 3$ are the three orthogonal unit vectors defining the original coordinate system of the environment (aka x, y, z directions).

The components of the tensor are thus:

$$I_{11} = \sum_{j=1}^{N} m_j(y_j^2 + z_j^2) \tag{2.15}$$

$$I_{22} = \sum_{j=1}^{N} m_j(x_j^2 + z_j^2) \tag{2.16}$$

$$I_{33} = \sum_{j=1}^{N} m_j(x_j^2 + y_j^2) \tag{2.17}$$

$$I_{12} = I_{21} = -\sum_{j=1}^{N} m_j x_j y_j \tag{2.18}$$

$$I_{13} = I_{31} = -\sum_{j=1}^{N} m_j x_j z_j \tag{2.19}$$

$$I_{23} = I_{32} = -\sum_{j=1}^{N} m_j y_j z_j \tag{2.20}$$

The eigenvectors of I are the principal moment of inertia axes of the system.

We hypothesized that transforming into this rotationally invariant basis should allow us to train more reliably. If we base our model in the x, y, z basis, the model is not rotationally invariant; it changes as our system is rotated. If we use an innate coordinate system, however, our model no longer depends on our external coordinate system and the outputs will remain the same even when the entire system is rotated.

While the moments of inertia provide a good innate coordinate system, we have to verify that they are giving us what we want: a rotationally invariant basis. As a system is rotated, we need to be sure that the MOI axes of the local environments remain the same and do not flip directions. To check this, we created an ozone molecule (O_3) and rotated it while we continually checked the force components on one of the atoms in both the x, y, z basis and the MOI basis. The top graph in Figure 2.2 shows what happened to the force components in the MOI basis as the system was rotated: the MOI axes occasionally switched which direction was positive, resulting in the force components in the MOI basis occasionally turning negative.

To have a unique definition of the principal axes and ensure that they do not flip directions, we adopt the convention that coordinates of the center of mass in the MOI basis (denoted $\tilde{x}$, $\tilde{y}$, and $\tilde{z}$) must be positive. (Remember that the center of mass is *not* the origin of the MOI basis because we calculated the inertia tensor about the position of atom i; atom i is our origin.)

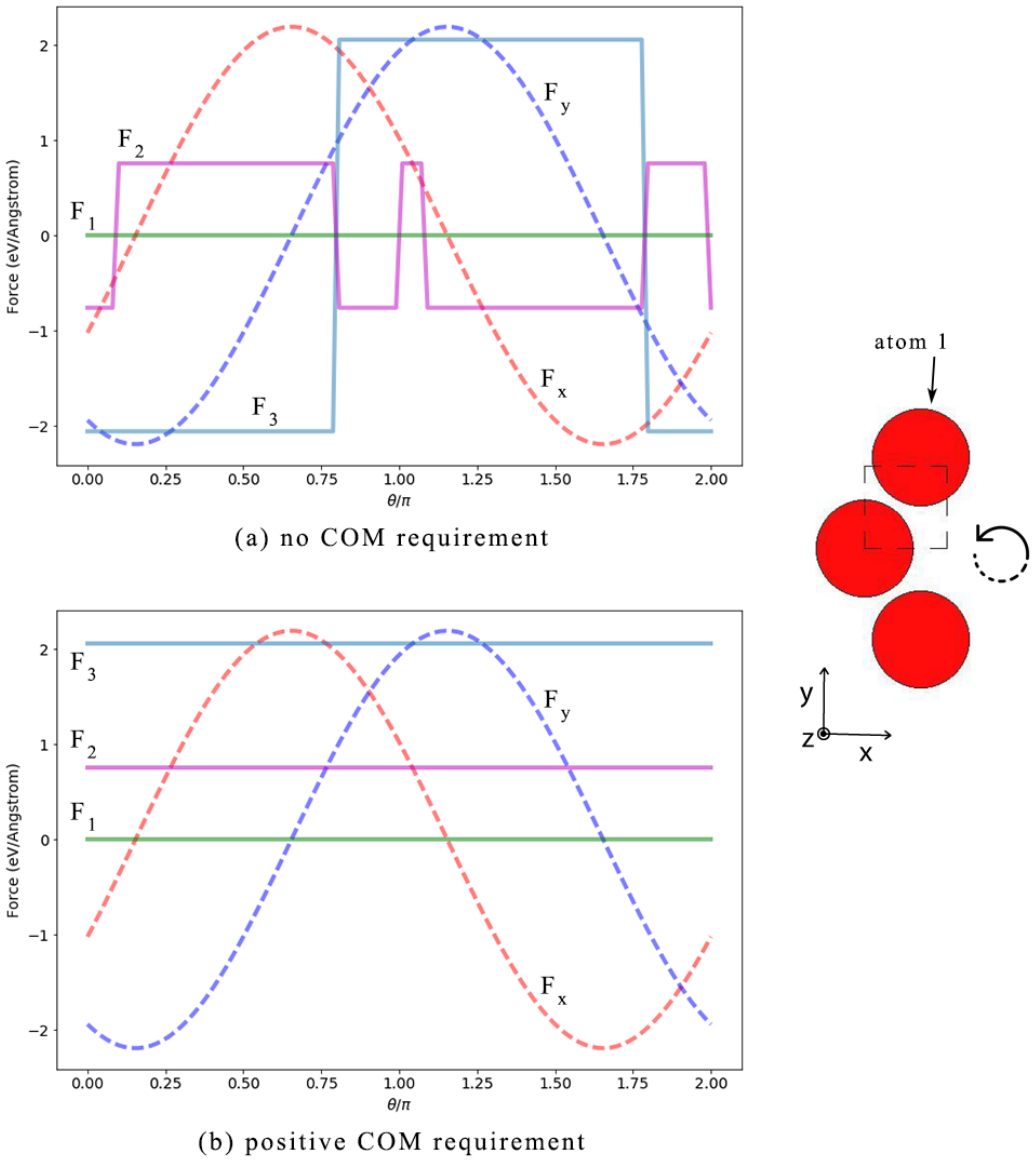
$$\tilde{x} \geq 0 \tag{2.21}$$

$$\tilde{y} \geq 0 \tag{2.22}$$

$$\tilde{z} \geq 0 \tag{2.23}$$

When this requirement is enforced, the force components in the MOI basis remain constant as the ozone molecule is rotated (the bottom graph in Figure 2.2). It is possible that the flipping of the axes would not cause significant problems because the fingerprint and force component would gain a negative sign together, but for easier model fitting and consistency, we enforced a positive COM requirement

For our MOI fingerprint, we keep the same functional form as the Botu-Ramprasad fingerprint (equation 2.3), but select our k directions for atom i to



(a) no COM requirement

(b) positive COM requirement

FIGURE 2.2: The magnitudes of the force components on atom 1 in an ozone molecule (O_3) in the x, y, z basis and the MOI basis as the system is rotated. The top graph (a) shows how the principal axes in the MOI basis can flip direction as the system rotates (causing the force components to gain negative signs), and the bottom graph (b) shows how we can prevent the principal axes from flipping directions when we require that the coordinates of the center of mass in the MOI basis be positive, giving us a unique definition of the principal axes that is rotationally invariant.

be $k = 1, 2, 3$ corresponding to the MOI axes of the local environment around atom i. The regression model is trained in the basis of each local environment's principal moments of inertia axes and outputs force predictions in this base. Thus, when predicting the forces of new systems, an extra step is required to transform the model outputs back into the x-y-z basis of our external coordinate system.

A potential problem with using the MOI axes of local environments as the basis for a regression model is how the principal axes of a given environment can abruptly change when a neighboring atom enters or exits the cutoff radius. Say one configuration of the entire system has atom j just outside of the cutoff radius for atom i, but in another configuration, atom j is just inside of the cutoff radius. The position of atom j relative to atom i has not significantly changed, but when calculating the MOI axes for atom i's local environment, atom j goes from giving no contribution to giving a full contribution. This could cause problems when training a model.

To avoid this problem, we apply a damping function when calculating the MOI axes so that the contribution of each atom in the local environment goes smoothly to zero as they leave the local environment. In our implementation, we damped the mass of each atom so that the mass of atom j used in calculating the moments of inertia around atom i goes to zero as the distance between atom i and atom j goes to (and beyond) R_c.

$$m_{j,damped} = m_j * f_c(R_{ij}) \tag{2.24}$$

where $f_c(R_{ij})$ is the same damping function used in the fingerprint calculations, equation 2.4.

Additionally, the principal moment of inertia axes were chosen as our intrinsic coordinate system because they are easy to calculate and give reliable, orthogonal directions. Moment of inertia axes are based on the mass and positions of the neighboring atoms, though; while the positions are relevant to the forces, the masses are not explicitly physically relevant to the forces. Another intrinsic coordinate system that is more physically relevant (perhaps a coordinate system based on charge, vibrational modes, etc) therefore might be a preferable choice over the moments of inertia. For this thesis, though, for ease, reliability, and lack of a better choice, we stick with the principal axes of the inertia tensor.

2.3 Machine Learning Model

The model in our machine learning scheme is a functional form containing adjustable parameters that can be regressed to link the fingerprints with the force components for each atom. Many different types of machine learning models can be used, such as neural networks, Gaussian processes, kernel ridge regression, or support vectors. While our group's previous works on atomistic machine learning have used a neural network model, we chose to use Gaussian

processes (GPs). We chose this because neural networks require random initialization and GPs are deterministic, and we want to directly compare different fingerprint functions without having the results affected by a random component we cannot control.

Models that require random initialization can be inconsistent. On one run, the initialization may be good and the model trains to a set of parameters that can predict forces with great accuracy. On another run, the initialization may be bad and the model gets stuck in a local minima of the loss function, resulting in poorer predictions. To avoid the possibility of one run being poor, neural networks and other schemes that require random initialization must be run many times. Gaussian processes avoid this problem altogether, however, and this makes directly comparing different schemes easier.

Another advantage is that the prediction is probabilistic (Gaussian), so that we can obtain an indication of the uncertainty for our predictions. Additionally, the model is versatile in that we can specify any kernel function we desire. A kernel is a general name for a function that maps a pair of inputs into a single real number. In this context, it determines the form of the prior and posterior probability distributions of the model. We use a standard kernel called the radial-basis function (RBF) or Gaussian kernel.

$$k(x_i, x_j) = e^{\frac{-(x_j - x_i)^2}{2r^2}} \tag{2.25}$$

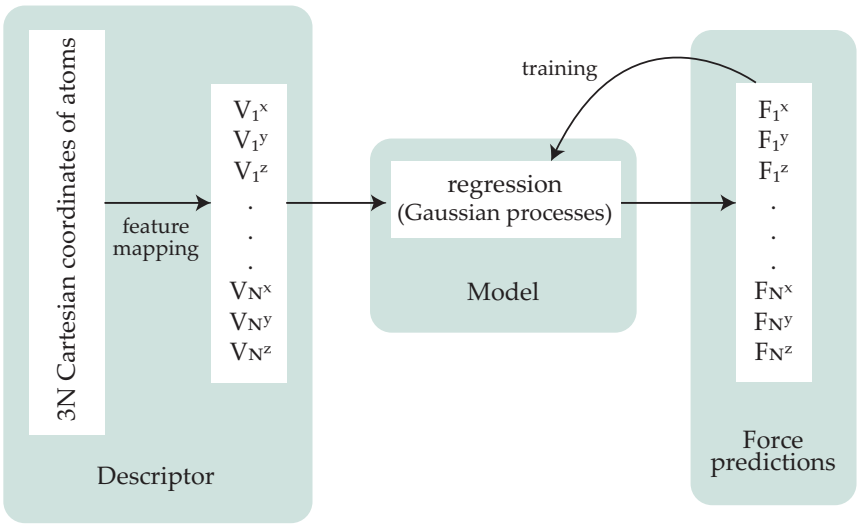
where r is a length-scale parameter ($r > 0$).

This kernel is infinitely differentiable, leading to smooth GPs. By using the kernel as a measure of similarity between points and assuming that data points with similar inputs should give similar outputs, the model can predict outputs for never-before-seen inputs.

A significant disadvantage of using Gaussian processes is that the model size grows with the size of the training set, causing the calculation time required when using the model to predict the forces of new systems to grow. Because of this, Gaussian processes are poorly suited for large (e.g., 100,000 images) training sets. Gaussian processes meet our needs, however, for directly comparing the performance of our different force-explicit schemes.

To see a schematic of how the descriptors (fingerprints) and regression model work together, see Figure 2.3.

for Botu and dG schemes:



for MOI scheme:

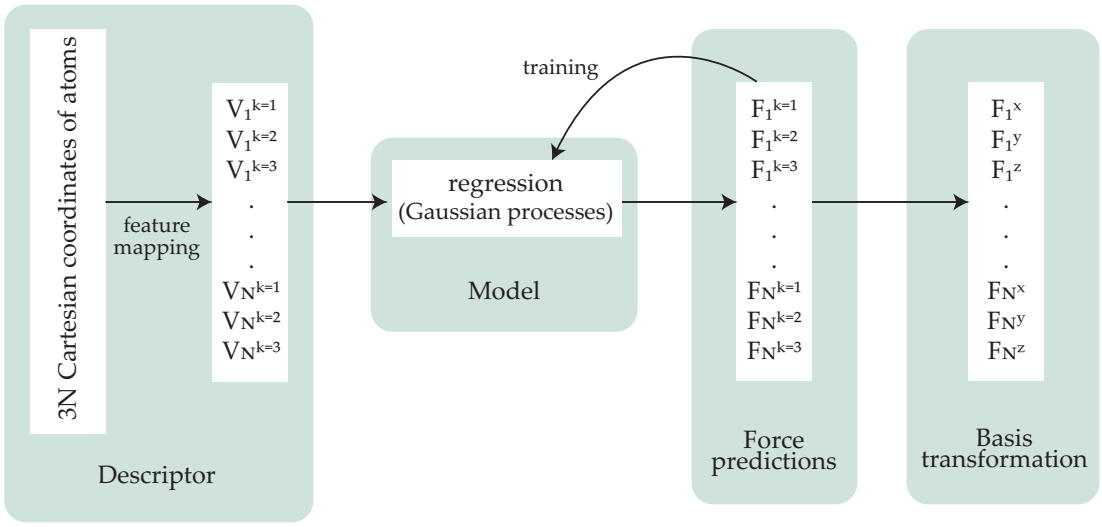


FIGURE 2.3: This schematic shows how our force-explicit machine learning schemes work. The atomic positions of N atoms are mapped to "fingerprints" for each atom and each $k = x$, y, and z component (or in the case of the MOI fingerprint, $k = 1$, 2, and 3). The descriptor is used as input into the regression model, which then predicts each force component for each atom. For the MOI scheme, the regression is done in the MOI basis, so the outputs of the model must be transformation back into the x, y, z basis by determining the MOI of each atom's local environment and relating that coordinate system to the x, y, z basis.

Chapter 3

Force-Explicit Scheme Comparison

We implemented each of the force-explicit fingerprints from Section 2.2 using the Gaussian processes regression model described in Section 2.3. The schemes were then trained and tested on two example systems.

3.1 Example Systems

These two example systems were originally prepared as part of another project in Brown's Catalyst Design Lab (see Peterson, Christensen, and Khorshidi (2017)).

The first example system is a simulated water molecule (H_2O). Over the course of 400 static "images" of the system, we varied both the H-O-H angle, θ, and one of the O-H bond lengths, r. (see Figure 3.1). The other O-H bond was kept at its equilibrium length, taken to be $r_0 = 0.969$ Å. r/r_0 was varied at 20 values between 0.9 and 2.0, and θ/π was varied at 20 values between 0.5 and 1.0.

The atomic structure of the water molecule was prepared in the Atomic Simulation Environment (ASE) (Bahn and Jacobsen, 2002). For each configuration, or image, of the system, the forces were found by density functional theory in the NWChem calculator in ASE with the B3LYP exchange–correlation functional (Becke, 1988; Lee, Yang, and Parr, 1988) and the 6-31+G* basis set (Ditchfield, Hehre, and Pople, 1971).

The second example system is a platinum face-centered cubic (111) surface with two layers of six atoms per layer. For a total of 10,000 5 fs trajectory steps, we propagated Langevin dynamics. In this process, a vacancy was created; one of the atoms was moved from its position in the lattice to an adatom position (see Figure 3.2). The calculations were done in planewave/pseudopotential DFT with the Dacapo calculator (Bahn and Jacobsen, 2002) and the RPBE functional (Hammer, Hansen, and Nørskov, 1999).

3.2 Training

To train, we randomly selected 10-100 images from both example systems and trained regression models with those images. A model was trained for each type of atom in each system, resulting in three models for each scheme: (1) hydrogen

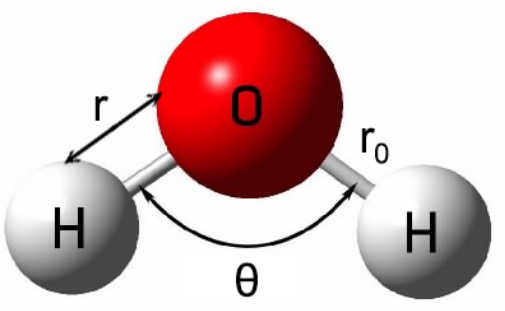


FIGURE 3.1: In the example system of a water molecule, one of the O-H bond lengths and the H-O-H angle was varied while the other O-H bond was kept at equilibrium length.

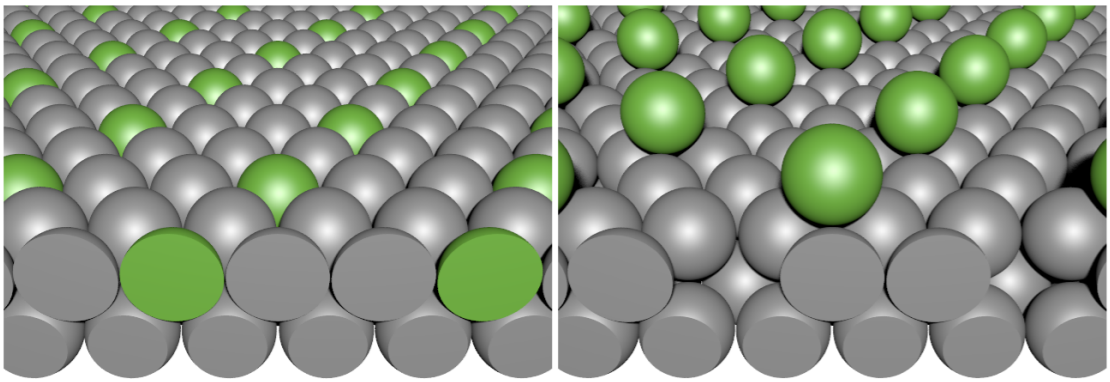


FIGURE 3.2: An example system of a Pt fcc (111) surface. Over the course of the simulation, one of the Pt atoms is moved from its lattice position to an adatom position. This atom is highlighted green for emphasis. This image is courtesy of Peterson, Christensen, and Khorshidi (2017).

in the water molecule, (2) oxygen in the water molecule, and (3) platinum in the platinum surface. This resulted in nine total regression models (from the three atom types and the three schemes). For consistency, the same training images from each example system were used for all of the different models.

When making the fingerprints, we varied η from 0.1 to 100 eight times with log spacing. For the Gaussian Derivative fingerprint, we varied λ and ζ from 0.01 to 1. These were chosen through trial and error. Additionally, when an atom had neighboring atoms of different types (eg. a hydrogen in the water molecule has an oxygen and a hydrogen neighbor), its fingerprint vector was comprised of individual fingerprints for each atom type.

$$\text{fingerprint}^{k}_{\text{hydrogen}} = \left[\text{fingerprint}^{k}_{\text{from other hydrogen}},\ \text{fingerprint}^{k}_{\text{from oxygen}}\right] \tag{3.1}$$

We used Scikit-learn's Gaussian processes library to implement a GP model

with a radial-basis function kernel (Pedregosa et al., 2011). To calculate the moments of inertia for local environments in the MOI scheme, we used an edited version of ASE's moments of inertia function (Bahn and Jacobsen, 2002). For the full code for all schemes, see our code repository.[1]

3.3 Testing Results

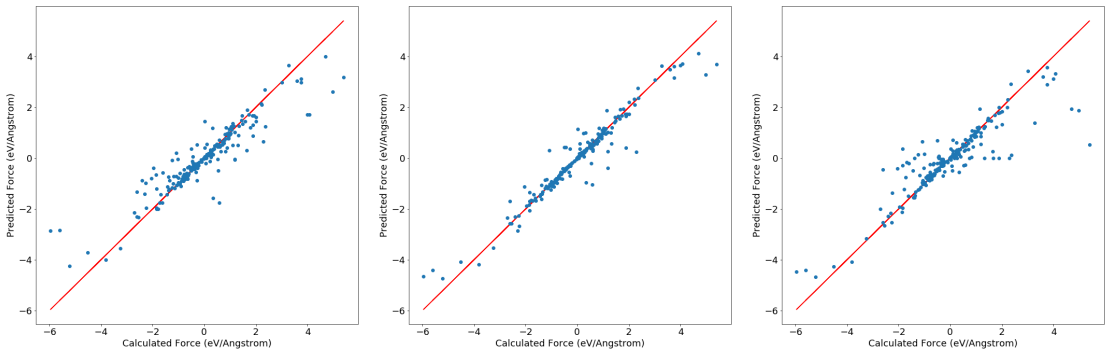
After each model was trained, we tested it on new images selected from the example systems. In general, the predictions made from these models matched well with the actual calculated forces. Figures 3.4 and 3.5 show parity plots that display the results of using the Botu-Ramprasad, Gaussian Derivative, and Moments of Inertia schemes to predict each component of the force on the different atoms. The x-axis shows the actual, calculated force component on each atom (from density functional theory), and the y-axis shows the predicted force component (from our trained regression model).

If the models had perfectly predicted each force component, all points in the parity plots would lie perfectly along the reference line (slope = 1) provided. Because the predictions were not perfect, however, points can be seen that do not lie on the line. The prediction errors strongly depend on which images are chosen as training images and which images are chosen as testing images. For example, in Figure 3.4b, all three schemes under-predict the higher magnitude forces. When other training and testing images are used with the schemes, however, the models sometime over-predict, under-predict, or perfectly predict. One exception to the random errors is that the MOI scheme on the platinum surface frequently predicted force components close to zero on some atoms (this can be seen on 3.5a and c) no matter what training and testing images were chosen. The number of the atom (of the twelve atoms in the lattice) that was being incorrectly predicted differed, however, and the exact reason for the trend in the errors is currently unknown.

In addition, more training resulted in better force predictions, but because we only care about relative performance between schemes and we trained/tested all schemes identically, this is not relevant to the results. The results in Figures 3.4 and 3.5 are shown because they are generally indicative of the relative magnitude of errors we obtained when using the different schemes.

Looking at Figure 3.3 (F_x on Pt, 3.5a split into three graphs for the different schemes) as an example, the MOI or Botu schemes have the most points that are far from the slope = 1 line; the errors from the two schemes are different, with the MOI scheme having errors that tend to push the force predictions closer to zero (a previously mentioned flaw whose cause is currently unknown), but the mean magnitude of errors is similar. In comparison, the dG scheme performs the best. The This trend extends, for the most part, to the overall results for all the force components on all of the atom types.

[1] https://github.com/kaleybrauer/brown-thesis



(a) Botu Scheme (b) dG Scheme (c) MOI Scheme

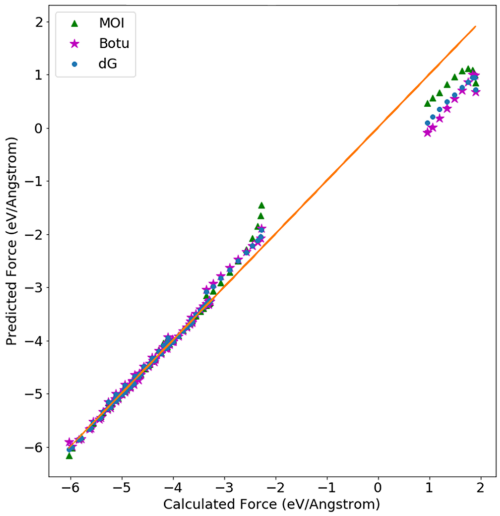
FIGURE 3.3: Parity plots showing the results of predicting the x-component of force on the platinum atoms in the platinum surface. Each model was trained on 5 images and tested on 20 images (12 Pt atoms per image). A line with slope = 1 is provided for reference.

The mean and median absolute errors for the different schemes are listed in Table 3.1 and graphically shown in Figure 3.6. For all the atom types (whether looking at the mean or median error), dG performs better than Botu, especially for the platinum in the platinum surface. The MOI scheme sometimes performs worse and sometimes performs better than the Botu scheme.

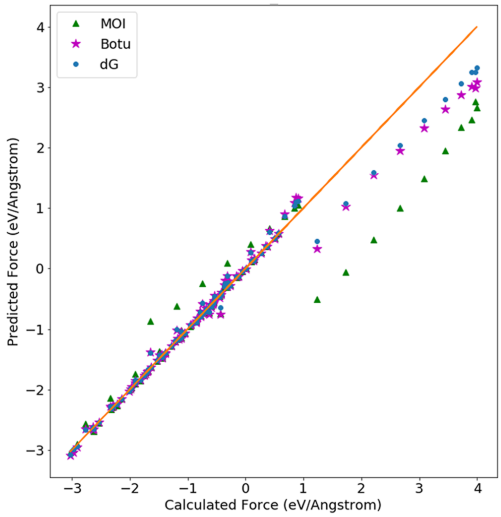
We expect dG to perform better than Botu because it incorporates representations of two-body *and* three-body interactions, while the Botu scheme only represents two-body interactions. The platinum surface allows the differences between the two schemes to become most apparent because the platinum surface involves more three-body interactions than the water molecule.

The MOI scheme's performance in comparison to the Botu scheme (sometimes worse, sometimes slightly better) helps us confirm that the MOI basis can be used as the basis for a force-explicit regression model, but does not allow us to determine if the MOI scheme would be better or worse than the Botu scheme in general. One point of interest is that the strength of the MOI scheme is its rotational invariance (the model does not depend on the x, y, and z of our external coordinate system). For that reason, it may be able to perform better than the dG or Botu schemes when training on and predicting forces for a system that has significant rotation between images. Because the water molecule and platinum surface were not rotating, we have not yet been able to test this. Additionally, more testing is required to determine the cause of some of the force predictions tending towards zero.

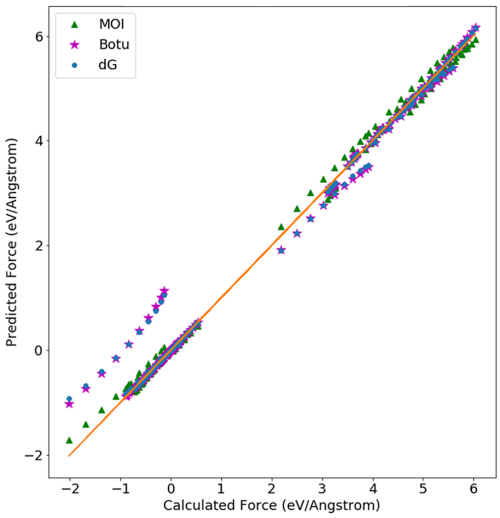
Based on these results, we conclude that the dG scheme has a fingerprint that represents the local environment better than the fingerprint in the Botu scheme and thus is better at predicting forces. We also conclude that the MOI basis can be used as the basis for a force-explicit regression model, but that more testing is required to determine if using the MOI basis is preferable in general.



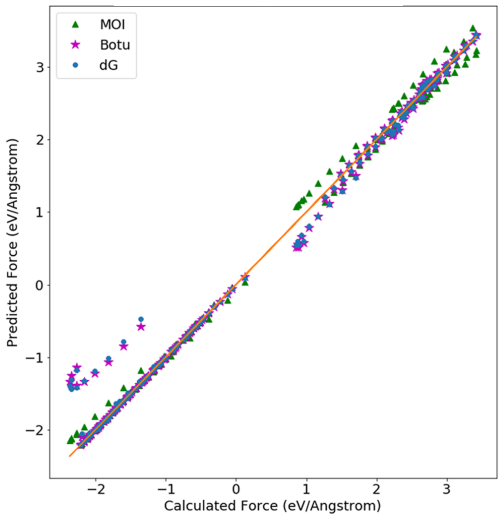
(a) F_y on O



(b) F_z on O

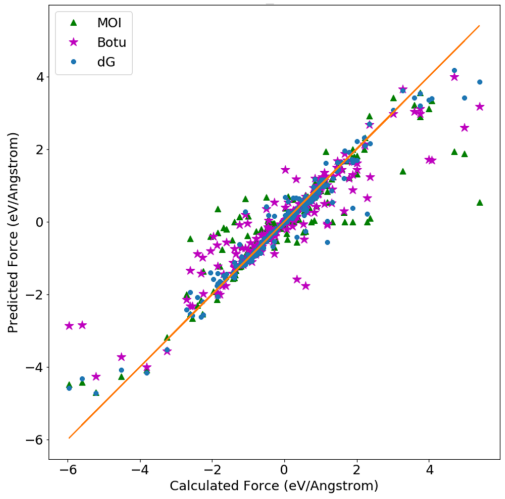


(c) F_y on H

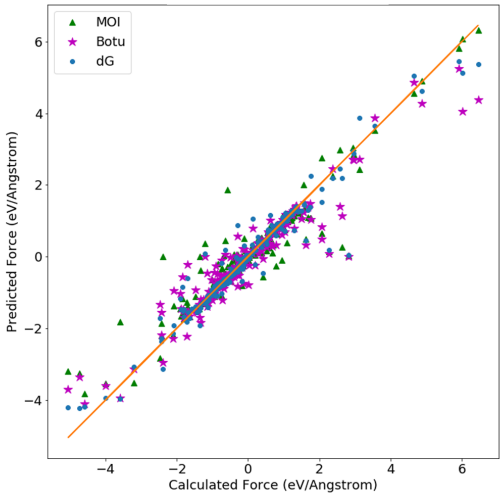


(d) F_z on H

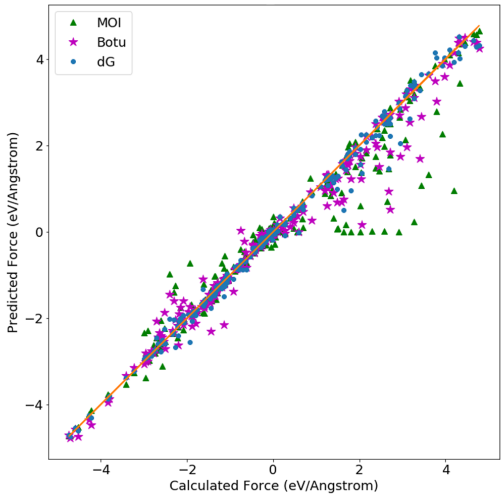
FIGURE 3.4: Parity plots showing the results of predicting the forces on oxygen and hydrogen in the water molecule. Each model was trained on 50 images and tested on 100 images; the testing results for each scheme are shown. The predictions for F_x are not included because the water molecule was oriented in the $y-z$ plane and experienced no force in the x-direction.



(a) F_x on Pt



(b) F_y on Pt

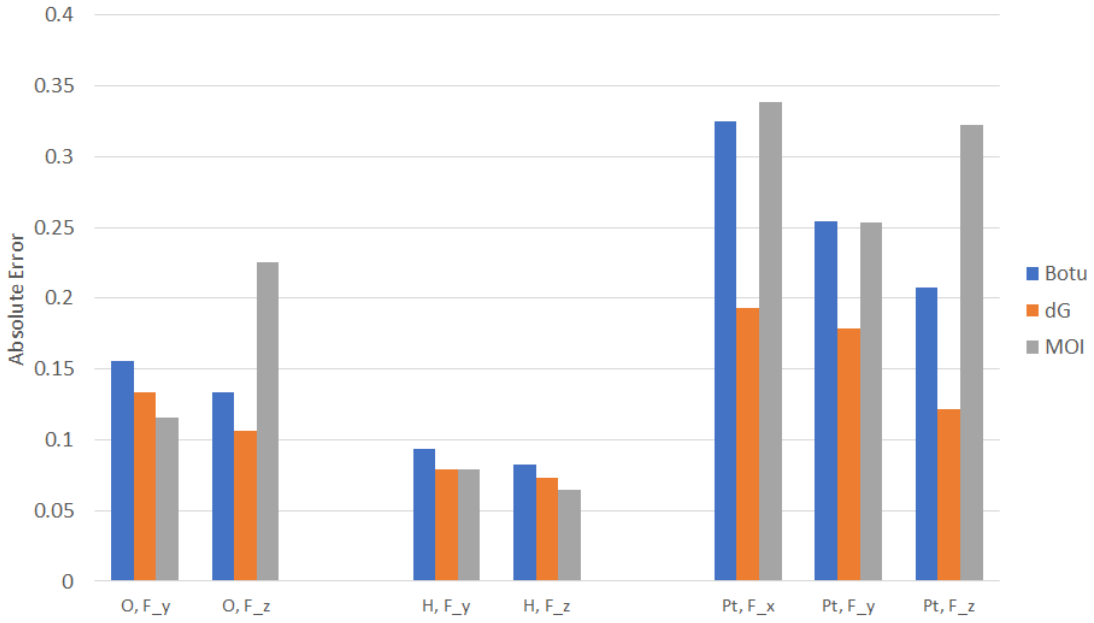


(c) F_z on Pt

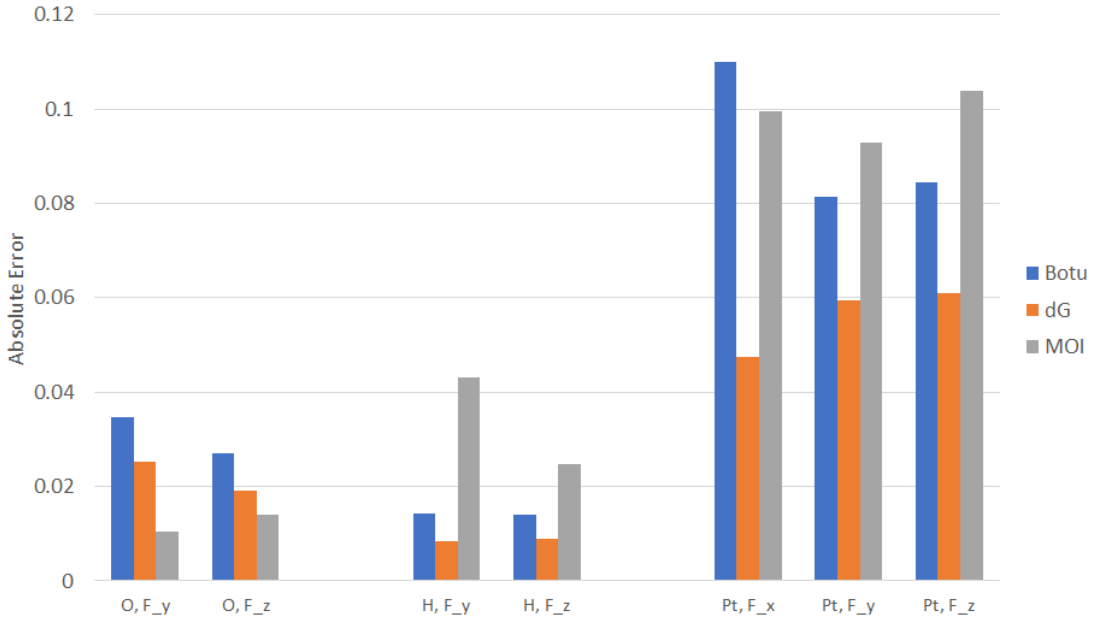
FIGURE 3.5: Parity plots showing the results of predicting the forces on the platinum atoms in the platinum surface. Each model was trained on 5 images and tested on 20 images (12 Pt atoms per image); the testing results for each scheme are shown.

TABLE 3.1: The mean and median absolute errors ($error = |F_{pred} - F_{calc}|$) for predictions of the force components on the different atoms in the example systems. The errors for F_x on oxygen and hydrogen are zero because the water molecule was oriented in the $y - z$ plane and experienced no force in the x-direction, and the models perfectly reproduced the zero force.

Atom = O	Mean			Median		
Scheme	F_x	F_y	F_z	F_x	F_y	F_z
Botu-Amprasad	0.0	0.1557	0.1332	0.0	0.03478	0.02712
Gaussian Derivative	0.0	0.1338	0.1062	0.0	0.02537	0.01903
Moment of Inertia	0.0	0.1159	0.2254	0.0	0.01044	0.01391
Atom = H	Mean			Median		
Scheme	F_x	F_y	F_z	F_x	F_y	F_z
Botu-Amprasad	0.0	0.0933	0.0827	0.0	0.01418	0.01391
Gaussian Derivative	0.0	0.0788	0.0732	0.0	0.00847	0.00893
Moment of Inertia	0.0	0.0791	0.0650	0.0	0.04301	0.02473
Atom = Pt	Mean			Median		
Scheme	F_x	F_y	F_z	F_x	F_y	F_z
Botu-Amprasad	0.3251	0.2546	0.2077	0.10997	0.08132	0.08436
Gaussian Derivative	0.1927	0.1787	0.1216	0.04758	0.05933	0.06098
Moment of Inertia	0.3383	0.2538	0.3224	0.09944	0.09287	0.10379



(a) Mean Absolute Error



(b) Median Absolute Error

FIGURE 3.6: The mean and median absolute errors of each of the three schemes when predicting the force components of oxygen and hydrogen (in a water molecule) and platinum (in a platinum surface). The higher magnitude of errors in the platinum surface is primarily due to those models being trained on fewer images than the water molecule models.

Chapter 4

Interactive Visualization

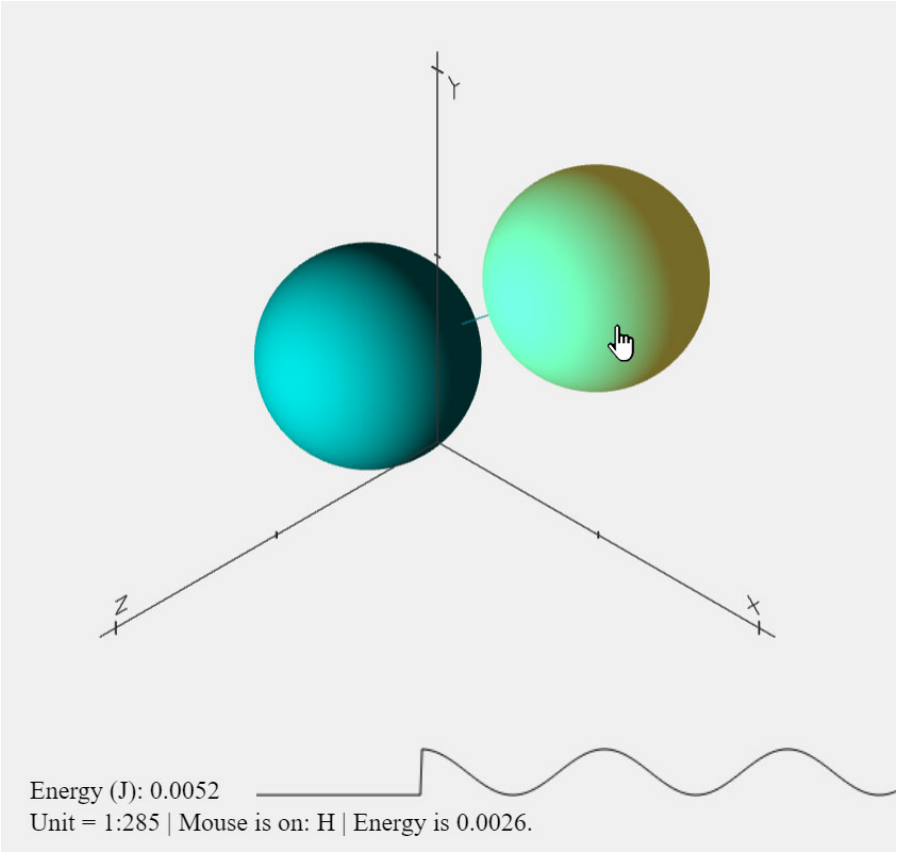


FIGURE 4.1: A hydrogen molecule in IntAE. The right hydrogen atom has been selected and displaced from equilibrium.

Atomic simulation visualizations help scientists study complicated systems at atomic and molecular scales, but currently available visualization software has only limited interactivity and multivariate visualization options. Real-time interaction with simulations is roadblocked by the time required to obtain atomic forces via (1) electronic structure methods or (2) derivatives of energies predicted from regression models. When systems become large and complicated, even taking derivatives of predicted potential energies can take a few seconds

- too long to allow for real-time interaction with lagging. A force-explicit machine learning scheme such as those discussed in the previous chapters, however, would provide quicker, direct access to atomic forces. This should make fully interactive simulations of even complicated systems possible.

Additionally, scientific visualization is an important component of simulation software. Multiple variables are associated with each atom at each time step in the simulation (potential and kinetic energy, position, force, momentum, charge, etc), and being able to quickly and intuitively understand how variables of interest are changing leads to more efficient and thorough understanding of the atomic system. Visualizing multiple variables in a simulation, or multivariate visualization, has not been a focus of currently available atomic visualization software, however. We therefore chose to evaluate user's preferences of different multivariate visualization options as a first step to a more dedicated focus on multivariate visualization in atomic simulations.

To demonstrate how an interactive simulation might work, investigate multivariate visualization options, and provide the coding framework for an accessible, robust, interactive, multivariate atomic simulation visualization software, we developed a modular prototype of Interactive Atomistic Environment (IntAE). In this chapter, we present the prototype of IntAE, focusing on the interactivity and the multivariate visualization. We also evaluate the overall visualization and potential usefulness of the software.

4.1 The Environment

IntAE is a prototype of the first robust, interactive, multivariate atomic simulation visualization software. The code is open-source and modular to ensure extensibility, and currently is browser-based for accessibility[1]. Fig. 4.1 shows an example of what one might see when they load up the software. The hydrogen molecule in the figure is arbitrarily colored, but the atomic colors in the final version will be based on ASE's standard color library in order to keep the colors in agreement with other accepted visualization programs. The atoms are displayed in three-dimensional position space as spheres with a radius equal to the atom's covalent radius. An alternative option would be displaying the atoms as spheres with a radius equal to the atom's van der Waals radius.

The potential energy of the entire system is continually plotted as a function of time at the bottom of the screen, and to obtain information about a particular atom, one can mouse over any atom in the simulation. Additionally, while not visible in Fig. 4.1, the top right corner of the screen has a drop down control panel for adjusting the time step, visualization options, or loading in a new atomic system.

[1] View the prototype at kaleybrauer.github.io/intae/iae.html?

4.1.1 Interactivity

After an atomic system has been loaded into IntAE, the user can interact with the system by clicking an atom and dragging it in the space. This displaces the atom from equilibrium and "sets off" the simulation. All variables (forces, energies, positions, etc) are calculated for every user-specified time step using Verlet integration.

$$x^{i+1} = x^i + v^i * \Delta t \tag{4.1}$$
$$v^{i+1} = v^i + a^i * \Delta t \tag{4.2}$$

where x^{i+1} is the position of an atom at time step $i+1$, x^i is the position at time step i, v is velocity, Δt is the time step, and a is the acceleration as obtained from the force calculation.

Because the force-explicit machine learning schemes require further development before they are fully integrated with Amp and used in IntAE, the force calculations in this prototype are done with Hook's law, a simplification loosely based on atomic vibrational modes.

$$F_{Hooks} = k * x \tag{4.3}$$

Due to the modular nature of IntAE, switching to using a force-explicit regression model as soon as such a model is fully developed will be as simple as changing a single line of code.

While the simulation is running and atomic positions, energies, etc are being continually updated, the user can continue to interact with the system. If an atom is selected, the simulation pauses and the user can move the atom as desired. Additionally, users can rotate the space and zoom in and out to view the system from different angles and distances.

4.1.2 Multivariate Visualization

To investigate how best to incorporate multivariate visualization in the simulation, we created two different options for visualizing the energy and momentum of each atom (Fig. 4.2). The first set of options encodes the energy or momentum into the visualization of the sphere representing each atom. The second set of options changes the coordinates of the three-dimensional space to plot the atoms according to their energy or momentum. The variables being visualized (energy and momentum) were simply chosen as examples; the visualization options could extend to any variable desired, and a final version of this software would allow the user to select which variables they are interested in visualizing.

The first option for visualizing energy encodes the energy of each atom into the atom's opacity; high opacity corresponds to high kinetic energy. The opacity of each atom dynamically changes as its energy changes, allowing users to

glance at the system and identify which atoms are most energetic. The first option for visualizing momentum displays a dynamic momentum vector at the center of each atom that corresponds to the magnitude and direction of the atom's momentum.

The options that encode the energy or momentum into the visualization of each sphere can be stacked (ex. the opacity can be changing while a dynamic vector is simultaneously displayed in the center, allowing both energy and momentum to be visualized). One potential worry, however, is that the visualization will become too busy. The color and the size of each sphere are already used to represent the atomic number and covalent radius, respectively. When information is also encoded into the opacity, into an internal vector, and possibly into the texture (another potential visualization option), the visualization may quickly become over-saturated with visual encodings. This is one danger and limitation of these options.

The second options for visualizing energy and momentum change the coordinates of the space in which the system is displayed. By default, the atoms are plotted according to their position. We created the option, however, to change to an energy phase space (Fig. 2c) or a momentum phase space (Fig. 2d). To provide simultaneous visualizations of different variables, the phase spaces could be displayed side-by-side. This is even more limited than using visual encodings, however, because a user cannot simultaneously view many different phase spaces. These visualizations could thus be used primarily to investigate a single variable in depth rather than as a means to visualize many different variables.

4.2 Evaluation

After completing the prototype of IntAE, we conducted interviews to evaluate which multivariate visualization options were most effective and what features users would find useful in our system. The interviewees were my primary advisor, Professor Andrew Peterson, and nine fourth-year undergraduates in Chemistry, Engineering, and Physics. Of the undergraduates, four out of the nine had previous experience working with ASE or other atomic simulation software. Participants were shown the features of the software and allowed to play with the system before they were asked a series of questions. The themes of the questions can be seen in Table 4.1.

Responses were overwhelmingly positive with all participants saying that they liked the system and could foresee themselves using it in an educational and/or research capacity. A few participants cited specific problems that they would be interested in investigating with such a system, e.g. watching how stresses and forces in a carbon sheet change in response to small atomic displacements. Participants also had numerous ideas for other useful features such as moving multiple atoms at once, exporting videos for presentations, and allowing other types of interaction (e.g. applying electromagnetic fields). Assorted

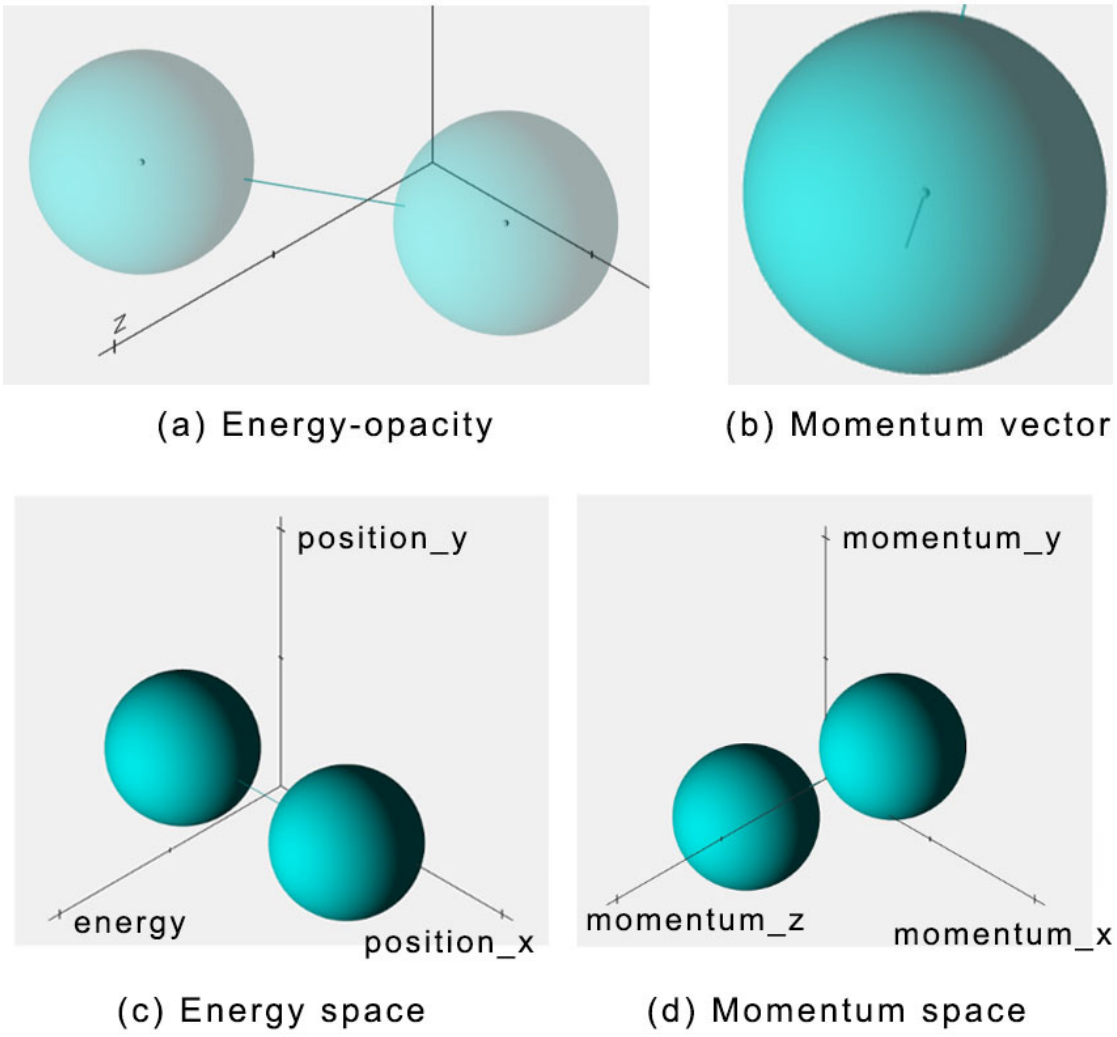


FIGURE 4.2: Different visualization options. (a) and (b) encode energy or momentum into the visualization of each atom, while (c) and (d) plot the atoms into energy or momentum phase spaces.

feedback was also given about how to improve the overall visualization (such as font changes and increasing the size of the momentum vector so that it is easier to see).

The most significant result was that participants were nearly unanimous in preferring the opacity and vector visualizations over the phase spaces, as seen in Figure 4.3. When asked an open-ended question about why they had those preferences, seven of the ten participants said that they preferred the opacity and vector visualizations because they found them more intuitive than the phase spaces. As one participant said about the momentum visualizations, "vectors are more intuitive for someone without experience in momentum space." Another participant simply said "the phase spaces are confusing." The second most popular reason for preferring the opacity and vector visualizations was that they

TABLE 4.1: Themes of Evaluation Questions

Section	Question Themes
Demographics	What is your specialty? Highest degree?
	Do you have experience working with atomic simulations?
Visualization	What are your thoughts on the visualization options?
	Which visualizations do you prefer and why?
General	What are other useful features to implement?
	How would you use such software?
	Do you have any other feedback?

allowed users to still easily see the positions of all of the atoms.

The phase spaces are likely unintuitive because simulations typically display in only position space. Additionally, we live in position space, so our intuitive understanding of that coordinate system has been developed throughout our entire lives while our experience with other coordinate systems is necessarily limited in comparison. One participant who had experience with looking at momentum phase space graphs actually preferred momentum space over the vector visualization, however, saying the phase space "could be a bit more useful to see patterns." The participant who labeled the vector and momentum space visualizations as equally useful also cited momentum space's potential for seeing patterns.

Thus, while participants overall strongly preferred the opacity and vector visualizations, this could be because users are unacquainted with using phase spaces in simulations and not necessarily because the phase spaces are inherently worse multivariate visualization options. Additionally, we evaluated these options using a small system (hydrogen), so the results may not map to systems with many atoms. Re-evaluating the visualization options later with a large system and participants who have worked more with phase spaces would give more insight into the effectiveness of these (and potentially other) multivariate visualization options. For now, our evaluation results imply that phase space visualizations are less naturally intuitive than options that encode variables into the visualization of the atomic spheres. In particular, participants expressed an extreme preference for visualizing energy as opacity and relayed that they found it highly intuitive.

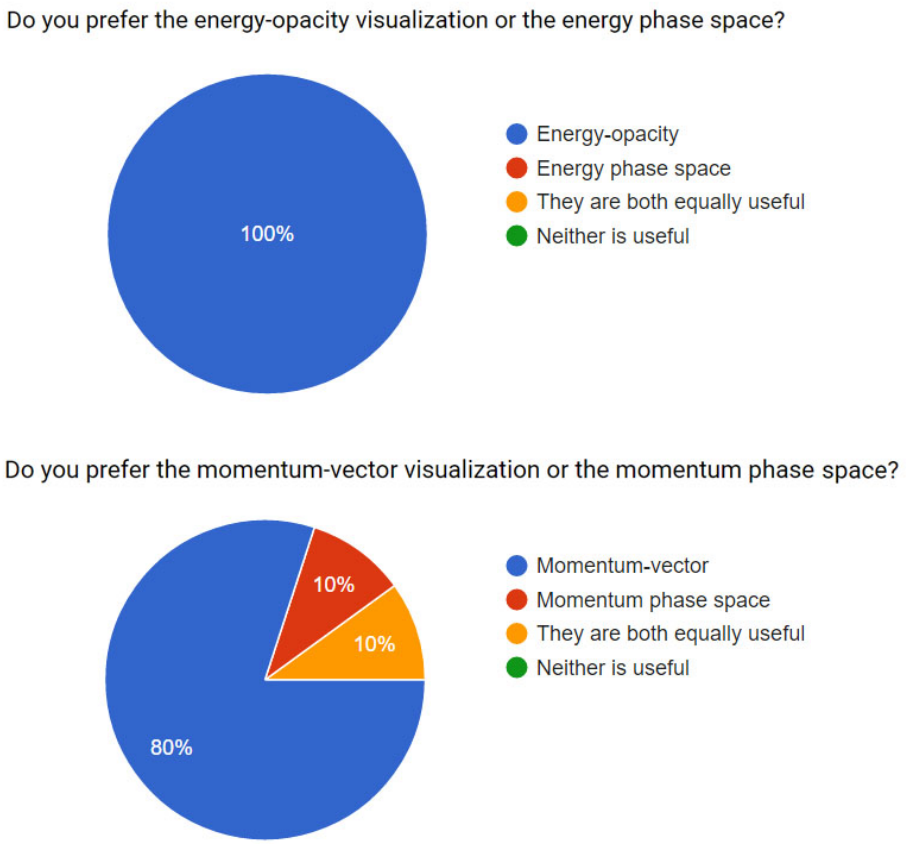


FIGURE 4.3: Multivariate visualization preferences of ten users. Users reported that energy-opacity and momentum-vector visualizations were more intuitive.

Chapter 5

Conclusions and Future Work

5.1 Force-Explicit Schemes

While atomic simulations generally treat forces as secondary to energy, force-explicit models can provide quicker, more direct access to the forces and allow for more efficient geometry optimizations or molecular dynamics simulations. In particular, a force-explicit regression model based on machine learning could allow forces to be obtained quickly enough for a simulation with real-time interaction. The user could displace an atom in an ongoing simulation and watch as the system immediately responds.

Energy-explicit regression models have been developed (such as the Atomistic Machine-Learning Package, or Amp, developed in part by my primary advisor Andrew Peterson), but little work has been done on developing schemes for force-explicit regression models. Botu and Ramprasad (2015) proposed an initial scheme for numerically representing (or "fingerprinting") the local environments of atoms for use in a machine learning based regression model. We expand on this work by proposing two additional schemes.

The first of our novel schemes is based on the idea that because force is the derivative of energy, a good fingerprint (or numerical representation of an atom's local environment) for a force-explicit scheme could come from the derivative of an accepted energy-explicit fingerprint. This idea is supported by the fact that Botu and Ramprasad's fingerprint has almost the same functional form as the derivative of the two-body components of the Behler Gaussian fingerprint, the standard fingerprint used in Amp. To expand on this idea, we took the derivative of the three-body components of the Behler Gaussian fingerprint and combined that with the Botu fingerprint. Thus, this novel Gaussian Derivative fingerprint has a two-body and three-body component for each atom, and both have similar (or in the case of the three-body component, the same) functional form to the derivatives of the Behler Gaussian fingerprint. Because this fingerprint takes into account three-body interactions, it should capture the local environment better than the Botu-Ramprasad fingerprint alone.

The second of our novel schemes is based on the idea that a rotationally invariant regression model that does not depend on our choice of coordinate system is preferable to a model that changes when we rotate our atomic system. In

their current form, the Botu-Ramprasad and Gaussian Derivative fingerprints depend on our choice of x, y, and z. To avoid this dependency, we chose to use the moments of inertia of each atom's local environment as an innate, rotationally invariant (when certain conventions are enforced) coordinate system. This scheme uses a fingerprint of the same form as the Botu-Ramprasad fingerprint, but it trains in the moments of inertia basis and not the x, y, z basis. To end up with forces in the x, y, z basis, we must perform a basis transformation on the model outputs.

We implemented, trained, and tested the three different schemes on two example systems: a water molecule that has the H-O-H angle and one H-O bond length varied, and a platinum fcc (111) surface which has one of its atoms gradually moved into an adatom position. When the schemes were tested on these systems, the Gaussian Derivative scheme performed the best. The Moments of Inertia scheme performed sometimes worse and sometimes better than the Botu-Ramprasad scheme, and usually not by much. Figure 3.6 shows the mean absolute errors for the predictions from each scheme on each system.

These results lead us to conclude that the Gaussian Derivative scheme has a fingerprint that represents the local environment better than the fingerprint in the Botu-Ramprasad scheme and thus is better at predicting forces. We also conclude that the Moments of Inertia basis can be used as the basis for a force-explicit regression model, but that more testing is required to see if this basis is preferable or not in general.

Because the primary strength of the Moments of Inertia scheme is its rotational invariance, future work will involve testing the schemes again on a system that significantly rotates to determine if the Moments of Inertia scheme performs better in that context. Additionally, the Moments of Inertia scheme could be adapted to use different fingerprints/models while still keeping the model in the MOI basis; for example, we could try using the Gaussian Derivative fingerprint or to try an energy-explicit fingerprint with a neural network that outputs all three components of the force simultaneously. The Gaussian Derivative fingerprint function could also be adjusted and tested to determine which portions of the function are most important when representing three-body interactions.

5.2 Interactive Visualization

In addition to proposing and testing force-explicit machine learning schemes, we developed a prototype of 3D interactive atomic simulation software with multivariate visualization. The Interactive Atomistic Environment (IntAE) software supports real-time click-and-drag interaction with the simulation and was developed as a potential application of the force-explicit schemes. While the software currently only uses Hook's law for forces, after the force-explicit schemes have been fully developed and integrated with Amp, IntAE can use the schemes

to quickly obtain forces and allow for real-time interaction with the robust simulation.

Additionally, we used IntAE as an opportunity to evaluate different multivariate visualization options. Scientific visualization is an important component of atomic simulations because effective, intuitive visualization is needed to help users quickly and thoroughly understand what is happening in the simulation. In particular, atomic simulations have a number of different, potentially insightful variables associated with each atom: energy, position, force, charge, etc. Being able to visualize these different variables should help users gain insight into the system.

We included two different sets of multivariate visualization options in IntAE (see Figure 4.2). We chose energy and momentum as the variables being visualized, but these are simply examples and the visualizations could be extended to other variables. The first set of options to visual energy and momentum involves encoding the variables into the visualizations of the sphere representing each atom. Energy is encoded as the opacity of the sphere (such that the opacity of the atom's sphere changes dynamically as the atom's energy changes), and momentum is encoded as a vector at the center of each atom (the magnitude and direction of the vector corresponds to the magnitude and direction of the momentum).

The second set of visualization options changes the coordinate system displayed in IntAE. Instead of showing the atoms in position space (plotted according to their position), the user can choose to display the atoms in energy space (the z-axis becomes energy) or momentum space (the axes become the x-, y-, and z-components of momentum).

To determine which options users find more effective and to get an overall evaluation of the system, we conducted ten interviews with potential users. The participants were my primary advisor, Andrew Peterson, and nine fourth-year undergraduates in Chemistry, Engineering, and Physics. The results of this evaluation were overwhelmingly positive, and all the participants reported that they enjoyed using the software and could foresee themselves using a fully functional version of the software in either an educational or research capacity. Additionally, the participants were in nearly unanimous agreement in preferring the opacity and vector visualization options to the energy space and momentum space visualization options. To explain this preference, seven of the ten participants expressed that they found the energy and momentum space visualizations unintuitive and/or confusing, while they found the opacity and vector visualizations very intuitive.

Users with more experience with phase spaces such as momentum space could potentially find the coordinate-change visualizations more effective (the one participant who reported a prior familiarity with momentum space preferred that visualization over the vector visualization), but the nearly unanimous result of our evaluation implies that such visualizations are less naturally intuitive than options that encode variables into the visualization of the atomic

spheres. Because of the small sample size (ten participants), limited visualization options (two visualizations in each set of options), and small atomic system (a hydrogen molecule) used in the evaluation, however, more testing must be done before we can say with any certainty that encoding variables into the visualization of the atomic spheres is a more effective visualization option than phase spaces.

Additional future work will of course involve taking IntAE from a prototype to fully functional simulation software. This will require implementing the force-explicit machine learning schemes with IntAE (so the simulation has quick, robust force calculations), adding additional features (the ability to create custom systems, output videos and numerical results, etc), and fixing bugs.

Acknowledgements

I would like to give a special thanks to Alireza Khorshidi for his ideas and continual assistance with the force-explicit machine learning schemes. I would also like to give a special thanks to Fumeng Yang for creating the base code of the IntAE prototype with me. In addition, thank you to my advisors, Andrew Peterson and Brad Marston, for the help and guidance; especially thanks to Professor Peterson for helping me with the theory of atomic forces and machine learning techniques. Lastly, thank you to everyone in the Catalyst Design Lab for the support!

Bibliography

Bahn, Sune R and Karsten W Jacobsen (2002). "An object-oriented scripting interface to a legacy electronic structure code". In: *Computing in Science & Engineering* 4.3, pp. 56–66.

Becke, Axel D (1988). "Density-functional exchange-energy approximation with correct asymptotic behavior". In: *Physical review A* 38.6, p. 3098.

Behler, Jörg and Michele Parrinello (2007). "Generalized neural-network representation of high-dimensional potential-energy surfaces". In: *Physical review letters* 98.14, p. 146401.

Botu, Venkatesh and Rampi Ramprasad (2015). "Learning scheme to predict atomic forces and accelerate materials simulations". In: *Physical Review B* 92.9, p. 094306.

Cramer, Christopher J (2013). *Essentials of computational chemistry: theories and models*. John Wiley & Sons.

Ditchfield, RHWJ, W J_ Hehre, and John A Pople (1971). "Self-consistent molecular-orbital methods. IX. An extended Gaussian-type basis for molecular-orbital studies of organic molecules". In: *The Journal of Chemical Physics* 54.2, pp. 724–728.

Feynman, Richard Phillips (1939). "Forces in molecules". In: *Physical Review* 56.4, p. 340.

Hammer, BHLB, Lars Bruno Hansen, and Jens Kehlet Nørskov (1999). "Improved adsorption energetics within density-functional theory using revised Perdew-Burke-Ernzerhof functionals". In: *Physical Review B* 59.11, p. 7413.

Hellmann, H (1937). "Einfuhrung in die Quantenchemie". In: *F. Deuticke, Leipzig*.

Jones, Robert O (2015). "Density functional theory: Its origins, rise to prominence, and future". In: *Reviews of modern physics* 87.3, p. 897.

Khorshidi, Alireza and Andrew A Peterson (2016). "Amp: A modular approach to machine learning in atomistic simulations". In: *Computer Physics Communications* 207, pp. 310–324.

Lee, Chengteh, Weitao Yang, and Robert G Parr (1988). "Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density". In: *Physical review B* 37.2, p. 785.

Li, Zhenwei, James R Kermode, and Alessandro De Vita (2015). "Molecular dynamics with on-the-fly machine learning of quantum-mechanical forces". In: *Physical review letters* 114.9, p. 096405.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Peterson, Andrew A., Rune Christensen, and Alireza Khorshidi (2017). "Addressing uncertainty in atomistic machine learning". In: *Phys. Chem. Chem. Phys.* Pp. –.

Shin, Dongil et al. (2002). "A web-based, interactive virtual laboratory system for unit operations and process systems engineering education: issues, design and implementation". In: *Computers & chemical engineering* 26.2, pp. 319–330.

Stieff, Mike and Uri Wilensky (2003). "Connected chemistry—incorporating interactive simulations into the chemistry classroom". In: *Journal of Science Education and Technology* 12.3, pp. 285–302.