Accelerating Electronic Structure Calculations with Machine Learning

Daniel Rothchild
Aditi Krishnapriyan
Joseph Gonzalez

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2023-222
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-222.html

August 11, 2023

Accelerating Electronic Structure Calculations with Machine Learning

By

Daniel Rothchild

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Associate Professor Joseph Gonzalez, Chair
Professor Ion Stoica
Assistant Professor Aditi Krishnapriyan

Summer 2023

Accelerating Electronic Structure Calculations with Machine Learning

Abstract

Accelerating Electronic Structure Calculations with Machine Learning

by

Daniel Rothchild

Doctor of Philosophy in Computer Science

University of California, Berkeley

Associate Professor Joseph Gonzalez, Chair

New chemicals and new materials have transformed modern life: pharmaceuticals, pesticides, surfactants, alloys, catalysts, polymers, battery electrodes, and countless other materials play critical roles in healthcare, construction, energy, and other wide-ranging industries. New materials are not generally stumbled upon by happenstance, but rather are discovered through a long process that involves extensive physics-based computer simulations at the atomic level. Electronic structure calculations play an important role in the discovery process, but they can be extremely computationally expensive. As such, there is a long history of approximation methods that trade off speed and accuracy.

Machine learning has the potential to open a new frontier on this speed-accuracy trade-off, and in doing so, significantly accelerate discovery of new materials. In this dissertation, we first cover the quantum mechanical background necessary to understand the problem setting, written with the machine learning community in mind as the audience. Next, we survey the learning-based methods that are pushing the speed-accuracy frontier, along with some foundational non-learning-based methods. Lastly, we investigate self-supervised learning as a mechanism for understanding the shape of the potential energy surface without expensive-to-obtain supervision on energies and forces.

To My Mother, Who I'm Pretty Sure Thinks There's Only a 60% Chance I Actually Submit This

Contents

List of Figures

List of Tables

Acknowledgments

Research. I'd first like to acknowledge and thank my advisors and dissertation committee members Ion Stoica, Joey Gonzalez, and Aditi Krishnapriyan. Ion is probably the biggest single reason I came to Berkeley: he and his students recruited me at visit days, and he made me excited about the prospect of graduate school and of working with him in RISELab. Ion took me on as one of his students when I arrived, and he guided me to work on communication-efficient distributed learning, which I continued to work on for the first two years of my PhD. Ion's signature refrain—always focus on choosing good problems to work on—is advice that I will keep with me for the rest of my career.

I was also excited to work with Joey after meeting him at visit days, and I added him as a second advisor soon after arriving at Berkeley. Joey's versatility as an advisor and his willingness to get excited about a diverse range of projects made him exactly the advisor I needed as I jumped around from project to (often unrelated other) project throughout graduate school. Without an advisor like Joey, I probably would not have made it through the last five years.

Lastly, I'd like to thank Aditi, who took me on as a student when she started as a new professor last year. Aditi and I started working together when she was a postdoc at Lawrence Berkeley Lab, and her advice has helped me explore ideas in machine learning for computational chemistry, which has culminated in this thesis. Aditi believed in and helped refine my ideas in a space that was very new to me, and her optimism helped propel me over the finish line.

I'd also like to thank Christian Borgs, who has given me helpful research and career advice and who served on my qualifying exam committee; Alyosha Efros and Michael Mahoney, who helped me explore research areas early in my PhD; and Sylvia Ratnasamy, who supported me in writing a very silly computer networking paper based on my final project in her class.

Next, I'd like to thank my research collaborators, whose ideas and contributions were crucial to all the work I carried out while at Berkeley. First, Nikita Ivkin, Enayat Ullah, and Vova Braverman, who, together with Ion, originated the idea behind my work on communication-efficient distributed learning, and who were an absolute pleasure to work with. I'd also like to thank Raman Arora and Ashwinee Panda, who joined the project to help extend our work to federated learning, and whose contributions were invaluable. Next, I'd like to thank Alex Tamkin, who helped originate the idea behind C5T5 and whose positivity propped me up in the middle years of the PhD. More recently, I have been working with Andrew Rosen and Eric Taw, who have been infinitely patient with my ignorance of their field and have been incredibly generous with their time and expertise.

Lastly, I'd like to thank my undergraduate research advisors, Stuart Shieber and Chris Stubbs, who mentored me as an undergraduate and who still respond to my texts when I'm visiting Boston. I'd like to especially thank Chris for giving me the opportunity to work on an important project, for engaging with me like an adult even though I was a junior in college, for sharing his great wisdom with me, and for sponsoring me to fly around the world

presenting my research through my year in England and even into the beginning of the PhD.

My path through graduate school was made much more bearable by the often-heroic efforts of a number of staff members, including Kattt Atchley, Boban Zarkovich, Ivan Ortega, Jon Kuroda, Shane Knapp, Angie Abbatecola, Jean Nguyen, and Shirley Salanio. Thanks to all of you for keeping the GPU machines running, wrangling the room schedules, plugging the funding sources into my bank account, planning retreats, and most of all, for keeping the faculty in line.

Teaching. I'd like to thank Jennifer Listgarten and Jitendra Malik, who entrusted me to help run the undergraduate machine learning course, who allowed me to give experimental lectures on graph neural networks, and who were remarkably understanding during the UC-wide strike. I was also very lucky to get to work alongside fellow GSIs and uGSIs Michael Janner, Forest Yang, Kartik Mangalam, Ameesh Shah, Mrunali Manjrekar, Arvind Rajaraman, Catherine Gai, and Jim Wang. Lastly, I'd like to thank Cindy Conners, without whom instruction in EECS would completely collapse within about 48 hours.

Neither Research Nor Teaching. First, I'd like to acknowledge Tanzil Chowdhury, Jess Banks, Garrett Shishido Strain, Rikhav Shah, Jean-Luc Watson, and the countless others who helped form a union for GSRs and who have worked tirelessly to uphold the rights of graduate students in my department and across UC. Thank you for involving me in a cause worth fighting for.

Next, I'd like to thank my labmates and friends who have helped make my life more enjoyable over the last five years (any friends listed above included here by reference): Audrey Cheng, Ajay Jain, Alvin Wan, Colorado Reed, Conor Power, Charles Packer, David Gold, Alejandro Escontrela, Evan Gastman, Frances Ding, Geoff Negiar, Eyal Sela, Jiwon Park, Julien Piet, Justin Kerr, Kevin Lin, Laura Power, Lexi Ding, Lisa Dunlap, Manish Shetty, Medhini Narasimhan, Mia Bladin, Nilesh Tripuraneni, Nithin Chalapathi, Paras Jain, Sam Kumar, Samyu Yagati, Shadaj Laddad, Shishir Patil, Suzie Petryk, Sukrit Kalra, Tess Despres, David Chu, Norman Mu, Tianjun Zhang, Vivian Fang, Justin Wong, and countless others. Also Michael Lauricella and Noah Yonack. Thanks to Alex Krentsel, Ben Mildenhall, Dave Moore, Henry Burnam, Leyla Kabuli, Lily Tsai, Mariel Werner, Mindy Perkins, Yu Sun, and others for making music with me over the last five years (and earlier!).

And thank you to Hank Lauricella and Mary Pickering, for sailing with me, taking me to nice restaurants, celebrating with me, cooking with me, and for welcoming me unconditionally into their home.

Next, I'd like to thank Sarah Wooders, who has enriched my life for the last two years, who has shown me new perspectives and new ways of thinking, who stands up to me when I am wrong, and who has tirelessly encouraged me to be a better person.

Lastly, I'd like to thank my parents, who taught me how to learn and how to think critically, who removed every obstacle to my success that they could, and who successfully steered me away from going to law school (really dodged a bullet there). I'd also like to

thank my sister, who taught me that hitting people will get you in trouble, who was good at the things I didn't want to be good at, which saved my fragile ego, and who served as my dry run of life one year ahead of every decision I had to make.

Funding Sources. I'd like to thank the American taxpayer for supporting me over the last five years via the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1752814. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and are unlikely to reflect the views of the National Science Foundation. In addition to NSF CISE Expeditions Award CCF-1730628, this research was supported by gifts from Alibaba, Amazon Web Services, Ant Financial, CapitalOne, Ericsson, Facebook, "Futurewei", Google, Intel, Microsoft, Nvidia, Scotia-bank, Splunk and VMware.

Chapter 1

Quantum Mechanics Background

1.1 Electrons

Wavefunctions

Electrons are the central object of interest for the behavior of atomic systems. If we could quickly and accurately figure out what electrons would do in any situation, we would immediately increase the pace of discovery across chemistry and materials science. However, modeling electrons is difficult because their behavior is governed by the Schrödinger equation, which has no known analytical solution for any but the simplest of systems. Physicists and chemists have come up with many ways to approximate and numerically solve the Schrödinger equation at various speed/accuracy trade-offs, and machine learning has the potential to significantly improve on this tradeoff compared to existing methods. The rest of this thesis will explore a number of these ML methods.

To provide a foundation upon which to understand these methods, this chapter first gives an overview of how we model electrons (and other particles) in quantum mechanics. Electrons are often described as being "both" waves and particles. This analogy is supposed to help students of quantum mechanics understand electrons by comparing them to familiar objects from classical mechanics: waves – like sound waves or water waves – and particles – like billiard balls. In fact, electrons are neither waves nor classical particles. In some situations, they behave like waves we are familiar with; in some situations, like particles; and in plenty of situations, like neither.

When we say that a billiard ball behaves like a classical particle, we mean that a system of billiard balls can be well described using the laws of classical mechanics: the state of the system at any given time is the locations and momenta of each of the balls, and the system changes over time according to Newton's laws ($F = ma$, etc.). By modeling the system in this way, we can carry out the two most basic functions of scientific modeling: predicting the outcome of measurements, and using the state of the system at one time to predict the

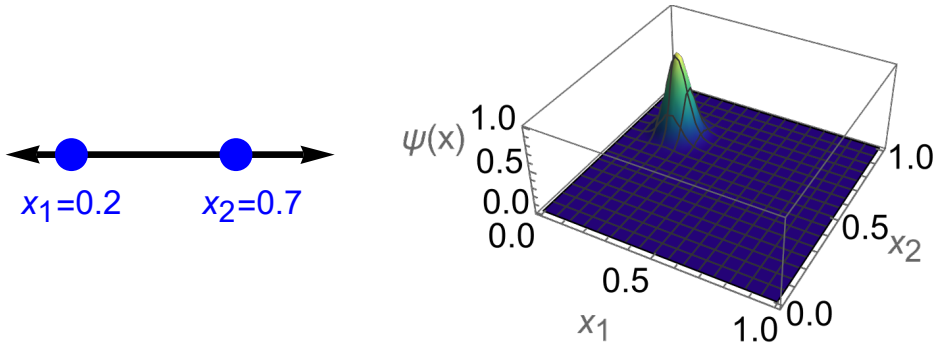

Figure 1.1: Left: example of a classical state describing two particles at $x_1 = 0.2$ and $x_2 = 0.7$. Right: example of a quantum state describing two particles localized around $x_1 = 0.2$ and $x_2 = 0.7$. $\psi(x)$ is assumed to be real-valued in this example.

state at future times.[1]

When we say an electron is a quantum-mechanical particle, we mean that a system of electrons can be well described using the laws of quantum mechanics, but not by the laws of classical mechanics. In other words, one could try to assign positions and momenta to each electron in the system and then evolve those over time using Newton's laws, however, this results in a very poor match between predictions and measurements in the lab. Instead of a list of particle positions and momenta, the state of a quantum mechanical system is called a "wavefunction", and the system's wavefunction evolves over time according to Schrödinger's equation. As in classical mechanics, what we want is a scientific model that allows us to predict the outcome of measurements and to propagate a system's state forward in time. The system's wavefunction, combined with the laws of quantum mechanics, allow us us to do both of these.

A system's wavefunction is a function over the degrees of freedom of the system. For example, the wavefunction of a system of two electrons in three dimensions could be expressed as a function over six dimensions—three degrees of freedom (x, y, and z) for each particle. For every possible pair of positions of the particles, the wavefunction takes on some value, and these values can be real or complex. The wavefunction is commonly denoted by $\Psi(x, t)$, or $\psi(x)$ when we're only considering Ψ at a particular time. In other words, a state in classical mechanics might look like $x_1 = 2, x_2 = -1 \ldots$; a state in quantum mechanics might instead look like $\psi(x_1, x_2) = e^{-(x_1^2+x_2^2)}$. A state in classical mechanics evolves over time by changing the positions and momenta of the particles according to Newton's laws. A quantum state evolves over time by changing the system's wavefunction according to Schrödinger's equation, which I will discuss below. The difference between a classical and quantum system state is illustrated in Figure 1.1

[1] "Predict" in this context refers to what a physics model asserts will happen, not the output of any kind of learned model, as discussed later in the thesis.

To start making sense of the idea of a wavefunction, consider a simple example of a particle in a one-dimensional box. The box starts at $x = 0$ and ends at $x = 1$. Another way of saying this is that a particle between $x = 0$ and $x = 1$ has zero potential energy, and a particle outside of this range has infinite potential energy. In classical mechanics, the state of this system at any moment in time consists of the particle's position and momentum. The particle could have any position between $x = 0$ and $x = 1$, and it can have any momentum whatsoever. The particle will follow Newton's laws, which predict that it will bounce back and forth within the box forever as long as the initial momentum is non-zero. The energy of the particle consists solely of kinetic energy, since the potential inside the box is zero. Therefore, since the particle can have any momentum, it can also have any energy $E = \frac{1}{2}mv^2 = \frac{p^2}{2m}$, where p, v, and m are the momentum, velocity, and mass of the particle.

For a quantum particle in our one-dimensional box, the system's state at any moment in time is the particle's wavefunction $\psi(x)$, which is a complex-valued function over all coordinates x. As you might expect, for the one-dimensional box, $\psi(x) = 0$ for $x \leq 0$ and $x \geq 1$. Within the box, $\psi(x)$ can be any function (though it must be suitably normalized). Some example wavefunctions are shown in 1.2. Although any suitably normalized function is possible, the wavefunctions shown in Figure 1.2 have the property that $|\psi(x)|$ does not change over time as $\psi(x)$ evolves according to the Schrödinger equation. In other words, $|\Psi(x,t)| = |\Psi(x,0)|$ for all t. These wavefunctions are called "stationary states", which are discussed below.

By itself, the wavefunction at a particular point in time doesn't tell us much. What's missing is 1) an understanding of how to use the wavefunction to predict what the outcome of a measurement will be, and 2) a mechanism to predict what the wavefunction will be in the future. I'll cover these in the following two sections. The concepts of wavefunctions, measurement, and time evolution are important because we want to replace slow, physics-based calculations involving these concepts with machine learning.

Measurement

In classical mechanics, predicting the outcome of measurements is simple and intuitive. If we are modeling a system of billiard balls and we want to predict, say, the outcome of a measurement of the number of balls that have x positions less than 0, we simply look at each ball in the system's state and check whether its x position is less than 0. If we build an apparatus that measures the number of balls in the real world with an x position less than zero, it will measure the same value every time, and that value will match what the model predicts.

Measurement in quantum mechanics is not so straightforward. As in classical mechanics, we can use the state of the system to predict what the outcome of a measurement in the lab will be. However, unlike in classical mechanics, the system's state gives us probabilities that certain values will be measured rather than predicting a particular outcome of a measurement with certainty. In the case of a system of particles, a classical model of a system of billiard balls will predict that a particular number of billiard balls have x position less than

0; a quantum model of a system of electrons will yield a probability distribution over the number of electrons that have an x position less than 0. This distinction between how we model classical and quantum systems reflects a difference in how measurements actually work in the lab. If you make a measurement on a particular classical state, your measurement apparatus will always give you the same answer unless the equipment is malfunctioning. If you make a measurement on a particular quantum state, your correctly functioning measurement apparatus will in general give you different answers each time you make the same measurement.

The wavefunction of a system at a given time tells us the probability of measuring each possible value. For example, if we build an apparatus that measures the position of an electron, then given an electron with a wavefunction $\psi(x)$, the probability of measuring the electron at position x_0 is $|\psi(x_0)|^2$, where $|\cdot|$ denotes the complex modulus. Position is called an "observable", and other observables include a particle's momentum, its spin, and its energy. As it turns out, all observables of physical quantities are linear operators on the space of all possible wavefunctions, and as such, they have eigenvectors and eigenvalues. Measuring the value of an observable always results in an eigenvalue of the observable, and after the measurement is complete, the quantum state of the system "collapses" to the eigenvector (or eigenstate) corresponding to that eigenvalue. The probability of a measurement returning any particular eigenvalue is given by the square of the inner product between the system's state before the measurement and the eigenstate corresponding to that eigenvalue.

Using quantum mechanics to predict the outcome of a measurement is therefore a two-step process. Say we are measuring an observable $\hat{O}$. The first step is to find the eigenvalues and eigenstates of $\hat{O}$ by solving the eigenvalue problem

$$\hat{O}\psi = o\psi.$$

The result of this step is a set of eigenstates ψ_a and corresponding measurement values o_a. Note that there can be infinite ψ_a, and that a can be either continuous or discrete depending on the observable. For the position observable, the eigenstates are $\psi_a(x) = \delta_a(x)$ for every real value a—*i.e.* the eigenstates are dirac delta function centered at every possible x—and the corresponding measurement values are $o_a = a$. For the energy observable, the eigenstates will depend on the system's potential function. For the one-dimensional particle in a box discussed above, the eigenstates are indexed by integers $n \geq 1$, and are given by

$$\psi_n(x) = \begin{cases} \sqrt{2}\sin(\pi n x) & 0 < x < 1 \\ 0 & o/w \end{cases}.$$

The corresponding measurement values are $E_n = \frac{n^2\pi^2\hbar^2}{2m}$. These $\psi_n(x)$ are the wavefunctions shown in figure 1.2.

The second step of predicting the outcome of measuring $\hat{O}$ is to compute the square of the inner product between the system's wavefunction $\psi(x)$ and each of the eigenstates $\psi_a(x)$. The inner product between two wavefunctions ψ_1 and ψ_2 is denoted $\langle\psi_1|\psi_2\rangle$, and is given by

$$\langle\psi_1|\psi_2\rangle = \int_{-\infty}^{\infty} \psi_1(x)^*\psi_2(x)dx,$$

where $\psi_1(x)^*$ denotes the complex conjugate of $\psi_1(x)$. For the position observable, the probability of measuring x is

$$|\langle\psi|\delta_x\rangle|^2 = \left|\int_{-\infty}^{\infty} \psi(x')^* \delta_x(x') dx'\right|^2 = |\psi(x)|^2.$$

Once a value x is observed, the wavefunction ψ collapses to δ_x. In other words, if we measure a particle's position twice in a row, we'll get the same answer the second time as we got the first time. For the particle-in-a-box energy observable, the probability of measuring E_n is

$$|\langle\psi|\psi_n\rangle|^2 = \left|\int_0^1 \psi(x)^* \sqrt{2}\sin(\pi n x) dx\right|^2.$$

If $\psi(x)$ is equal to one of the eigenstates ψ_n, then the inner product will be 1 and we will measure the value E_n with probability 1. In other words, the eigenstates ψ_n are the wavefunctions with a definite energy value. Any wavefunction that isn't one of these eigenstates will stochastically return an E_n depending on the inner product above, and it will collapse to the corresponding ψ_n after measurement.

Time Evolution

Energy and time are intertwined in quantum mechanics, and as such, the energy observable, called the Hamiltonian and denoted $\hat{H}$, is of central importance in determining how a system evolves over time. The time-dependent Schrödinger equation governs how wavefunctions evolve over time:

$$i\hbar\frac{d}{dt}|\Psi(x,t)\rangle = \hat{H}|\Psi(x,t)\rangle$$

We can solve this equation by assuming the solution $\Psi(x,t)$ can be separated into spatial and temporal components: $\Psi(x,t) = \psi(x)\tau(t)$. If we make this assumption, then the solution to the time-dependent Schrödinger equation above is $\Psi(x,t) = \psi(x)\tau(t)$ (as assumed), where $\tau(t) = e^{-iEt/\hbar}$ and $\psi(x)$ is the solution to the time-independent Schrödinger equation:

$$\hat{H}|\phi(x)\rangle = E|\phi(x)\rangle.$$

This equation is the same eigenvalue equation used above to determine the wavefunctions of definite energy. Solutions to this differential equation are called stationary states, since their time evolution is very simple (*i.e.*, multiply by $e^{-iEt/\hbar}$ to advance to time t), and the probability of finding a particle at position x, $|\phi(x)e^{-iEt/\hbar}|^2 = |\phi(x)|^2|e^{iEt/\hbar}|^2 = |\phi(x)|^2$, does not depend on time. In other words, the wavefunctions of definite energy are exactly those which vary over time only by a global phase $e^{-iEt/\hbar}$. In contrast, states that are not energy eigenstates will have different probability densities $|\psi(x)|^2$ depending on what time a measurement is taken.

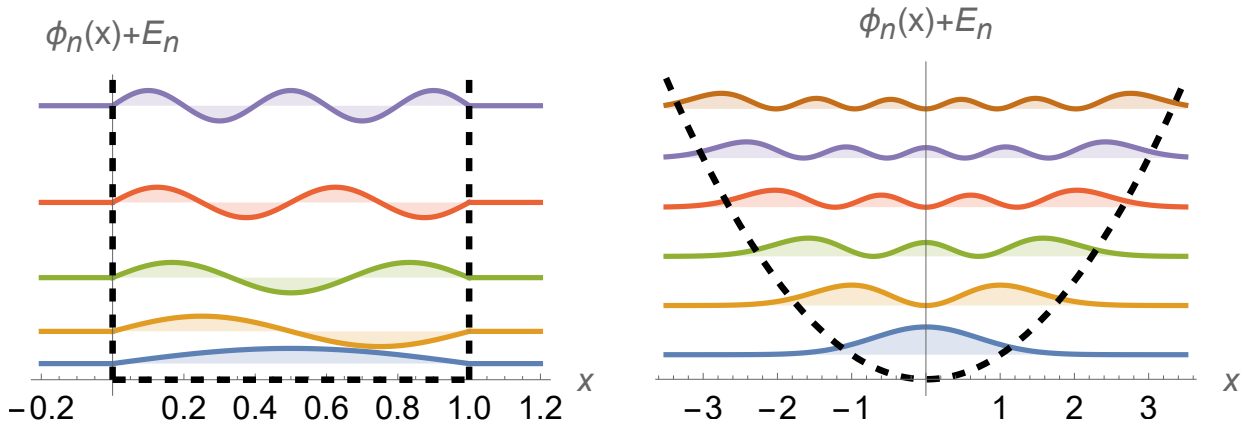

Figure 1.2: Left: eigenstates $\phi(x)$ for the particle in a box potential. Right: eigenstates $\phi(x)$ for the harmonic potential. In both figures, the potential is indicated by a dashed black line, and the wavefunctions ϕ_n are offset vertically by E_n.

Example: Harmonic Oscillator

Above, we considered the example of a particle in a one-dimensional box. This potential function is of limited use for modeling atomic systems, since real-world potentials rarely look like the particle in a box. In this section we'll consider a much more useful example: the harmonic oscillator. The one-dimensional harmonic oscillator is the quadratic potential function $V(x) = \frac{1}{2}kx^2$, where k is analogous to the classical spring constant and determines how quickly the potential increases as you move away from the origin. The harmonic oscillator is of particular interest because it is the lowest-order approximation to any potential function around a local minimum. Local minima in the potential function are important because systems in the real world tend to occupy low energy states.

To find out how particles behave in a harmonic potential, we need to find the energy eigenstates, or stationary states of an electron situated in the potential. The energy of an electron is the sum of its kinetic and potential energy, and so the Hamiltonian (also known as the energy observable) can be written as

$$\hat{H} = KE + PE = \frac{\hat{p}^2}{2m} + \frac{1}{2}k\hat{x}^2.$$

As explained above, the stationary states are the solutions to the eigenvalue problem $\hat{H}\psi = E\psi$, and it turns out that, for the harmonic oscillator, there is one stationary state for each integer $n \geq 0$. If we ignore normalization and choose units judiciously, the stationary states are given by:

$$\psi_n(x) \propto e^{-x^2/2} H_n(x),$$

where H_n are polynomials of order n called Hermite polynomials. In other words, the n^{th} stationary state is a polynomial of degree n multiplied by a Gaussian envelope. The

associated energies E_n are (in the same judiciously chosen units as above)

$$E_n = n + \frac{1}{2}.$$

These stationary states of the quantum harmonic oscillator are shown in Figure 1.2.

There are a number of differences between the classical and quantum harmonic oscillator that are worth noting. An analogous classical system to consider is that of a ball rolling back and forth in a quadratically shaped valley. If the ball is released from some place on the hill, it will oscillate sinusoidally back and forth within the valley forever (assuming no friction). Compared to the classical harmonic oscillator, where the ball can be released from any height and therefore can have any amount of total energy, the states of definite energy in the quantum harmonic oscillator are quantized, meaning that only a discrete set of energy measurements are possible. A consequence of this is that an electron in energy level n—*i.e.* whose wavefunction is given by $\psi_n(x)$—cannot absorb or release an arbitrary amount of energy in order to move to some other energy level n'. Instead, it must absorb or release energy in packets of size $\hbar\omega$, or some integer multiple thereof. Electrons can absorb or release energy by absorbing or releasing photons, so this phenomenon is the reason why emission and absorption spectra exhibit lines at particular wavelengths.

Another noteworthy difference between the quantum and classical harmonic oscillators is that, in the classical harmonic oscillator, there is a hard maximum x value at which you might find the ball in. In the quantum case, on the other hand, there is non-zero probability of measuring the quantum particle in the classically disallowed region. This probability decays exponentially as $|x|$ increases, but is non-zero nonetheless. This feature of quantum mechanics is what allows quantum tunneling to occur, where a particle surpasses an energy barrier that, from the standpoint of classical mechanics, should not be possible to cross.

Example: Hydrogen Atom

The hydrogen atom is also an important model system, and for a single electron, it can be solved analytically. In a hydrogen atom, an electron experiences the Coulomb potential in three dimensions due to its electrostatic interaction with the nucleus: $V(\vec{r}) \propto \frac{1}{\vec{r}}$. Plugging this potential into the usual Hamiltonian $\hat{H} = \frac{p^2}{2m} + V(r)$ and solving the eigenvalue problem $\hat{H}\phi = E\phi$ gives the energy eigenstates for the Hydrogen atom with one electron. For the Hydrogen atom in particular, the energy eigenstates are also referred to as "orbitals", since before quantum mechanics was fully developed, electrons in atoms were believed to be particles orbiting a nucleus.

Similar to the particle in a box and the harmonic oscillator, these energy eigenstates for the Hydrogen atom are quantized, but for Hydrogen they are indexed by three integers rather than just one. A full treatment of the Hydrogen atom is beyond the scope of this thesis, but in brief, the three integers are n, the "principle quantum number", which determines the spatial extent of the orbital; l, the "azimuthal quantum number", which determines the

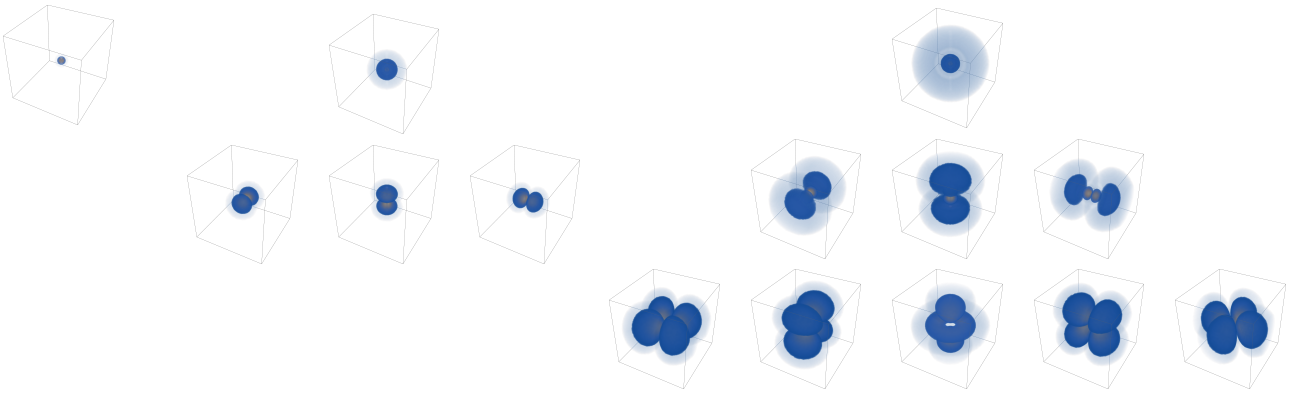

Figure 1.3: Density plots of $|\psi_{n\ell m}|^2$ for $n = 1$ (left-most plot), $n = 2$ (middle pyramid), and $n = 3$ (right-most pyramid). The rows of each pyramid denote $\ell = 1$, 2, and 3, and the columns are the m values, from $m = -\ell$ to $m = \ell$

angular momentum of the orbital, and m, the "magnetic quantum number", which sets a tradeoff between the oscillation frequency of the orbital along lines of latitude vs. longitude.

Ignoring normalization and dropping fundamental constants, the energy eigenstate (or stationary state, or orbital) associated with integers n, ℓ, and m can be written as follows:

$$\psi_{n\ell m}(r, \theta, \phi) \propto e^{-r/n} r^{\ell} L_{n-\ell-1}^{2\ell+1}\left(\frac{2r}{n}\right) Y_{\ell}^{m}(\theta, \phi).$$

Here, $L_{n-\ell-1}^{2\ell+1}(x)$ is a polynomial in x of degree $n - \ell - 1$ called a generalized Laguerre polynomial, and Y_{ℓ}^{m} is a spherical harmonic function, which is analogous to a sinusoid but on a sphere, where ℓ determines how quickly Y_{ℓ}^{m} changes as a function of θ and ϕ. Notice that n determines the spatial extent of $\psi_{n\ell m}$ because the wavefunction is exponentially cut off by $e^{-r/n}$. ℓ determines both how many times $\psi_{n\ell m}$ oscillates with increasing r before dying out (through $L_{n-\ell-1}^{2\ell+1}$) and also how fast $\psi_{n\ell m}$ changes as a function of θ and ϕ (through Y_{ℓ}^{m}.

Visualizations of $\psi_{n\ell m}$ are shown in Figure 1.3.

Spin

So far, I have introduced wavefunctions as functions of three spatial degrees of freedom. However, electrons have an additional non-spatial degree of freedom called spin. A spatial degree of freedom x can take on any real value, but the spin degree of freedom can only take on two discrete values: "up" or "down". Just like the position $\hat{x}$ of an electron is an observable, so too is the spin $\hat{s}$ an observable that can return either up or down. The associated eigenstates are denoted $\psi_{\uparrow}(s)$ and $\psi_{\downarrow}(s)$, respectively.

If we assume that there is no interaction between an electron's spin and its orbital, the total wavefunction of an electron $\psi(x, s)$ can be factored into a spatial and a spin component

$\psi_x(x)\psi_s(s)$. The spin component $\psi_s(s)$ can be written as any (suitably normalized) linear combination of $\psi_\uparrow(s)$ and $\psi_\downarrow(s)$. So, for example, an electron in the lowest energy orbital of the Hydrogen atom with completely unknown spin can be written as:

$$\psi(r,\theta,\phi,s) = \frac{1}{\sqrt{2}}\psi_{000}(r,\theta,\phi)(\psi_\uparrow(s) + \psi_\downarrow(s)).$$

Because I will assume that there is no interaction between the spatial and spin components of an electron's wavefunction, spin can largely be treated separately from the spatial wavefunction, so for ease of exposition, I will mostly ignore spin in this thesis.

1.2 Approximation Methods

Multiple Electrons and Antisymmetry

For larger atoms with multiple electrons, there is no known analytical solution for the energy eigenstates. The reason for this is that the potential becomes more complicated. For two electrons, the potential depends not only on the distances between each electron and the nucleus r_1 and r_2, but also on the distance between the two electrons r_{12}:

$$V(\vec{r}_1,\vec{r}_2) = \frac{p_1^2}{2m} + \frac{p_2^2}{2m} - \frac{Z}{r_1} - \frac{Z}{r_2} + \frac{1}{r_{12}},$$

where Z is the charge of the nucleus (e.g. $Z = 2$ for Helium). This extra term means it is impossible to factor the problem into two independent one-electron problems, and the problem only gets worse with more electrons.

As a very first approximation, we can model the 2-electron wavefunction by simply ignoring the $\frac{1}{r_{12}}$ term entirely. Doing so allows us to factor the problem into two separate one-electron problems. By this logic, if the energy eigenstates for the 1-electron problem are $\phi_i(x)$ with energies E_i, then the energy eigenstates for the 2-electron system are products $\phi_i(x_1)\phi_j(x_2)$, and the associated energies are $E_i + E_j$. Extending this to the N-electron case would mean that the energy eigenstates are products $\phi_i(x_1)\phi_j(x_2)\phi_k(x_3)\cdots$ with associated energies $E_i + E_j + E_k + \ldots$. In particular, the lowest-energy state for a system of N electrons would be $\phi_0(x_1)\phi_0(x_2)\phi_0(x_3)\cdots$, where $\phi_0(x)$ is the lowest-energy stationary state for the 1-electron system.

Atoms are systems of N electrons, so by this reasoning, the lowest-energy state of every atom would consist of all electrons occupying the orbital ψ_{000}. In other words, the electronic wavefunction for an atom with N electrons would be $\psi(x_1,x_2,\ldots,x_N) = \psi_{000}(x_1)\psi_{000}(x_2)\cdots\psi_{000}(x_N)$. This product of one-electron wavefunctions is called a Hartree product. In reality, the Hartree product is a qualitatively incorrect model for an atom with N electrons. What we observe in the real world is that the binding energy between electrons and nucleii decreases as more electrons are added. In contrast, if the Hartree product were a

good model for N electrons, we would expect each electron to have the same binding energy to the nucleus.

The reason why the Hartree product is so bad is not because we ignored the $1/r_{12}$ term in the Hamiltonian. Instead, it is due to one last feature of quantum mechanics that I'll introduce in this chapter, namely that the wavefunction of any system of multiple electrons must be what's called "antisymmetric". An antisymmetric wavefunction has the property that if you swap the coordinates of two electrons, the overall wavefunction picks up a negative sign. In other words, if r_1 is the location of electron one and r_2 is the location of electron two, then (ignoring spin)

$$\psi(r_1, r_2) = -\psi(r_2, r_1).$$

This fact about electrons is highly unintuitive: electrons are identical, so from the standpoint of classical mechanics, it should make no difference if two of the particles swap places. Nevertheless, this negative sign is extremely important, since it means that the Hartree product is not a physically allowed wavefunction for a system of N electrons. For example, if $\psi(x_1, x_2) = \phi(x_1)\phi(x_2)$, then antisymmetry requires

$$\phi(x_1)\phi(x_2) = \psi(x_1, x_2) = -\psi(x_2, x_1) = -\phi(x_2)\phi(x_1).$$

This can only be satisfied if $\phi(x) = 0$ everywhere, which is not possible because the wavefunction must be normalized so that $|\psi|^2$ is a probability distribution.

In fact, not only is the Hartree product $\psi(x_1, x_2) = \psi_{000}(x_1)\psi_{000}(x_2)$ not allowed, but there is no way whatsoever to combine $\psi_{000}(x_1)$ and $\psi_{000}(x_2)$ into a two-electron antisymmetric wavefunction. This fact is another way to state the Pauli exclusion principle, which says that it is impossible for two electrons to occupy the same quantum state.

Hartree-Fock

Evidently, the Hartree product is not a valid way to combine N one-electron wavefunctions into an N-electron wavefunction. Instead, to form antisymmetric N-electron wavefunctions, we can combine N one-electron wavefunctions using what's called a Slater determinant. For the case of two electrons with one-electron wavefunctions $\phi_1(x)$ and $\phi_2(x)$, the Slater determinant of ϕ_1 and ϕ_2 is

$$\psi(x_1, x_2) = \frac{1}{\sqrt{2}} \left(\phi_1(x_1)\phi_2(x_2) - \phi_1(x_2)\phi_2(x_1)\right).$$

For N electrons, the Slater determinant is the determinant of a matrix:

$$\psi(x_1, x_2, \ldots, x_N) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_N(x_1) \\ \phi_2(x_1) & \phi_2(x_2) & \cdots & \phi_N(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_N(x_1) & \phi_N(x_2) & \cdots & \phi_N(x_N) \end{vmatrix}.$$

The Slater determinant satisfies antisymmetry because the determinant of a matrix picks up a negative sign if you swap two of the rows.

Armed with an antisymmetric way to combine one-electron orbitals, we can now make progress on finding the energy eigenstates for a system of interacting electrons (i.e. without ignoring the $1/r_{12}$ term in the potential). To start, we'll focus on finding the energy eigenstate corresponding to the lowest energy eigenvalue. This eigenstate is particularly important because it represents the lowest-energy state that the N electrons might find themselves in. The approach we'll take will be remarkably familiar to anyone who has studied machine learning. The key insight is that, given a guess for the ground-state wavefunction is, we can pretty easily compute the actual energy of that wavefunction. Then, because we're trying to find the ground-state wavefunction, we can adjust our initial guess until the energy is minimized. More concretely, we'll guess that the wavefunction is a Slater determinant of N one-electron wavefunctions ${\phi_i}_{i=1}^N$. The energy of this Slater determinant is a mess of integrals of various combinations of the ϕ_i, which I won't write out here. But if we take the derivative of this expression for the energy with respect to ϕ_i and set it to zero, we get out the equation

$$f(x)\phi_i(x) = \epsilon_i \phi_i(x),$$

where $f(x)$, called the Fock operator, consists of several integrals involving all of the N different wavefunctions $\phi_i(x)$. If we treat $f(x)$ as fixed, we now have a set of N eigenvalue problems that we can solve to get a new set of N wavefunctions $\phi_i'(x)$. The Slater determinant of these new $\phi_i'(x)$ must have as low or lower energy than the Slater determinant of the $\phi_i(x)$, since we found the $\phi_i'(x)$ by minimizing the system's energy. However, the new Slater determinant is not necessarily the wavefunction with lowest possible energy, since in order to find the $\phi_i'(x)$ we had to treat $f(x)$ as fixed, even though in reality it depends on all the $\phi_i(x)$. Nevertheless, this procedure allows us to take any set of $\phi_i(x)$ and improve them. As such, we can start with an initial guess for the $\phi_i(x)$, and then repeatedly apply this method to find wavefunctions with lower and lower energy until the energy converges.

This algorithm is called the Hartree-Fock algorithm. To summarize, the steps in the algorithm are as follows:

1. Choose an initial set of one-electron wavefunctions $\{\phi_i(x)\}_{i=1}^N$.
2. Calculate the Fock operator $f(x)$. If we choose the $\phi_i(x)$ to be a linear combination of basis functions, we can write $f(x)$ as a matrix F.
3. Solve each of the N eigenvalue problems $\mathbf{F}\vec{c_i} = \epsilon\vec{c_i}$.
4. Calculate each of the new $\phi_i'(x)$ as a linear combination of the basis functions using the $\vec{c_i}$ calculated in the previous step.
5. Return to step 2, replacing $\phi_i(x)$ with the new $\phi_i'(x)$. Continue iterating until the energy converges.

The Hartree-Fock method will find the lowest-energy N-electron wavefunction out of all possible Slater determinants. However, we unfortunately haven't fully solved the Schrödinger equation, since there's no guarantee that the energy eigenstates are in fact Slater determinants. A number of methods, called post-Hartree-Fock methods, remove the assumption that the energy eigenstates can be written as a Slater determinant. These post-Hartree-Fock methods attain much higher accuracy than Hartree-Fock, and in principle it's even possible to fully solve the Schrödinger to arbitrarily good accuracy. However, these methods are extremely expensive. The computational complexity of Hartree-Fock scales as $n_{electrons}^4$, and post-Hartree-Fock methods scale as $n_{electrons}^6$, $n_{electrons}^8$, or even higher. As such, these methods are only suitable for small systems.

Density Functional Theory

For larger systems, we can turn to a method called density functional theory (DFT), which makes a different set of approximations as Hartree-Fock. In DFT, we make use of a fact about electronic ground states that was mathematically proven by Walter Kohn and Pierre Hohenberg in the 1960's. Hohenberg and Kohn proved two theorems about the electronic ground state. The first theorem says that the ground state properties of an atomic system are completely determined by the electron density in the ground state. In other words, it says that it's possible to do any calculation you want about the ground state even if you don't know what the wavefunction of the system is—you only have to know the electron density. The second theorem gives an expression for the system's energy as a function of the electron density. Minimizing the energy yields the true ground-state electron density. The first theorem is important because the electron density is a function of three spatial coordinates, whereas the N-electron wavefunction $\psi(x_1, x_2, \ldots, x_N)$ is a function of $3N$ spatial coordinates. This means that calculations using the density are much more computationally tractable than calculations with the wavefunction. The second theorem, combined with later work from Walter Kohn and Lu Jeu Sham, gives a method—in principle—to find the ground-state electron density, from which (by the first theorem) all properties can be computed. The expression for the Kohn-Sham energy as a functional of the electron density ρ is

$$E_{KS-DFT}[\rho] = T[\rho] + E_{eN}[\rho] + J[\rho] + E_{xc}[\rho].$$

Each term depends on ρ, which itself is a function, so the expression for the energy is called a functional. The first term, $T[\rho]$, is the kinetic energy of the wavefunction that generated ρ. The second term, $E_{eN}[\rho]$, is the energy due to Coulomb attraction between the electron cloud and the nucleii. Coulomb interactions are the normal electrostatic attraction or repulsion between positive and negative charges, and they decay proportionally to the inverse of the distance between the charges. The third term, $J[\rho]$, is the Coulomb repulsion between electrons. And the last term, $E_{xc}[\rho]$, is called the "exchange-correlation energy", and it consists of two terms. The first term, called the "exchange" term, represents the energy arising from the requirement that the N-electron wavefunction $\psi(x_1, x_2, \ldots x_N)$ be antisymmetric. As

opposed to Hartree-Fock, where we only considered antisymmetric wavefunctions, in DFT we instead remain more flexible but penalize wavefunctions that violate antisymmetry. The second term, called the "correlation" term, arises from the fact that we don't know how to compute $T[\rho]$ exactly. There's a standard way to approximate $T[\rho]$, which I won't explain, but the correlation term within E_{xc} is an attempt to make up the difference between this approximation and the true $T[\rho]$.

We don't know how to calculate either of the exchange or correlation terms within E_{xc}, so instead we have to substitute in our best guess for what those terms should look like as a functional of ρ. It's possible to analytically compute E_{xc} for some toy systems. For example, we can compute what E_{xc} would be for a uniform gas of electrons, and then we can apply that same E_{xc} for calculations on real systems. This particular functional is referred to as the local density approximation (LDA), and there are many other functionals of increasing sophistication. Besides functionals derived from first principles, several authors have proposed learning the functional with machine learning [48, 30].

Once an E_{xc} is chosen, we can write down a full expression for the energy as a functional of the electron density ρ. Then, we find the ρ that minimizes $E[\rho]$ which, by the Hohenberg Kohn theorem, is the true ground-state electron density, up to whatever error is introduced by our approximation of the true E_{xc}. The details of how we actually minimize $E[\rho]$ are similar to how we minimized the energy in Hartree-Fock, and in fact much of the code is reused between Hartree-Fock and DFT calculations. However, the computational scaling of DFT is in principle only $n^3_{electrons}$, which is better than Hartree-Fock. In practice, DFT can scale faster or slower than that, depending on whether additional approximations are made or whether more advanced (and expensive) E_{xc} functionals are used.

Chapter 2

Machine Learning Background

2.1 Interatomic Potentials

Instead of deriving how electrons behave from first principles, we can also write down much simpler models and fit the parameters of those models to match empirical data or data derived from first-principles calculations. For many types of calculations, we are interested primarily in the energy of a system or the forces experienced by the nuclei. We can obtain these two quantities by defining a potential function that depends on nuclear coordinates. To evaluate a given configuration of nuclei, we plug the nuclear coordinates into the potential in order to get the energy of that configuration, and we take the derivative of the potential with respect to the nuclear coordinates to get the forces experienced by the nuclei. These potentials, or "interatomic potentials", range from the simple two-parameter Lennard-Jones model all the way up to machine learning interatomic potentials with millions of parameters.

Force Fields

Originally, domain experts would come up with interatomic potentials by hand based on a combination of physical principles and heuristics. Perhaps the most famous interatomic potential is the Lennard-Jones potential. Given a set of nuclear coordinates $\{x_i\}_{i=1}^N$, the Lennard-Jones potential is

$$V_{LJ}(x_1, x_2, \ldots, x_N) = \sum_{i=1}^{N} \sum_{j=i+1}^{N} 4\epsilon \left(\left(\frac{\sigma}{||x_i - x_j||} \right)^{12} - \left(\frac{\sigma}{||x_i - x_j||} \right)^{6} \right).$$

In other words, the potential of the system is a sum over all pairs of nuclei, where each pair—separated by a distance r—contributes energy proportional to $(\sigma r)^{-12} - (\sigma r)^{-6}$. This potential is shown in Figure 2.1 for two nuclei separated by a distance r.

Although the Lennard-Jones potential is not derived from basic physical laws, it does have some physical motivation: the r^{-6} term matches how we would expect the potential to behave at large r for uncharged particles with no dipole. This attractive force is called the

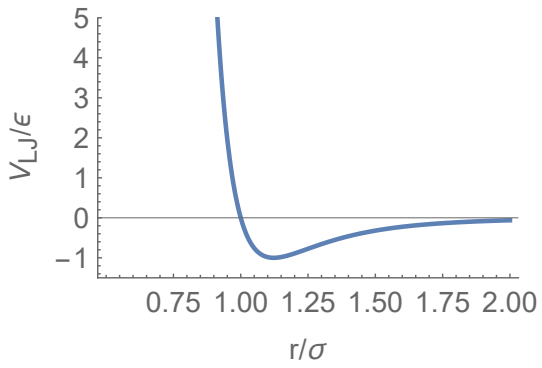

Figure 2.1: Lennard-Jones potential between two nuclei separated by a distance r.

London Dispersion force. However, the r^{-12} term is more arbitrary: we know that nuclei can't get too close together or else the electron orbitals will overlap and start to disobey the Pauli exclusion principle, so we need some function that increases steeply as $r \to 0$. However, the exponent of r^{-12} was chosen specifically because it can be easily calculated in a computer as the square of r^{-6}, not because it comes from a basic physical law.

Besides the exponents on r, the Lennard-Jones potential has two configurable parameters: ϵ and σ. ϵ controls the energy scale of the potential, and σ controls the effective size of each particle. These parameters can be set fairly easily to match empirical data or more sophisticated quantum chemical calculations. Despite having only two parameters, the Lennard-Jones potential can accurately reproduce a range of physical phenomena for some simple materials, such as noble gasses and methane. However, for materials that don't have spherical symmetry or that experience interactions besides London dispersion forces, and for mixtures of multiple atoms or molecules, the basic Lennard-Jones potential fails to model real-world behavior.

There are many other interatomic potentials that go beyond the functionality of Lennard-Jones. Other potentials use a different dependence on r, add additional parameters to handle multiple atom types, add three-body terms, add terms to explicitly consider bonds between atoms, add dependence on bond angles, etc. Modern force fields are complex, have many parameters, and are known to work well with certain types of materials. For example, the Merck Molecular Force Field, or MMFF, is commonly used for organic chemistry calculations, the ReaxFF force field is used when modeling chemical reactions where bonds break and form, and the Universal Force Field, or UFF, is commonly used as a general-purpose force field that sacrifices accuracy on any specific task for generality. Force fields have the advantage of being computationally simple to calculate, but they generally lack the accuracy of ab initio quantum mechanical calculations.

Local Descriptor Methods

To move towards models with higher accuracy, we need to increase the capacity of the interatomic potential functions and employ more sophisticated methods to set parameters based on reference data. A popular way to accomplish this is to write the potential as a sum over atoms:

$$V(x_1, x_2, \ldots) = \sum_{i=1}^{N_{atoms}} v(\{r_{ij}\}_{j=1}^{N_{atoms}}),$$

where $r_{ij} = x_j - x_i$ is the vector pointing from nucleus i to nucleus j. Then, we parameterize v however we like. A popular paradigm is to first construct a description of the local neighborhood of atom i using the vectors $\{r_{ij}\}$, and to then pass that description to a neural network or other machine learning model, which predicts the total energy.

Dividing v in this fashion is advantageous because the potential must be invariant to rotations and translations of the coordinate system. If the potential were not invariant, then different rotations/translations of the same system could yield different energies, which is unphysical. In local descriptor methods, the description of an atom's local neighborhood computed from the $\{r_{ij}\}$ is generally chosen to be invariant to translations and rotations. Then, the machine learning model that predicts energy based on the local descriptors doesn't need to worry about invariance, since there's no way it could accidentally re-introduce a dependence on rotation or translation once that information has been eliminated from the local neighborhood description. The only constraint on the energy-prediction model is that it must be differentiable, since in order to calculate forces we need to take the derivative of the potential with respect to the atomic coordinates. The descriptors must also be differentiable for the same reason.

Descriptors

Many papers have explored different local descriptor methods. The transformations that we need to be invariant to are translations and rotations in 3D space, reflections, and permutations of atoms of the same type (e.g. swapping one carbon with another). Local descriptor methods work by identifying properties of an atom's local neighborhood that are invariant to these transformations, or by computing functions of non-invariant properties that are provably invariant. For example, a simple invariant descriptor is a vector containing the number of neighboring atoms of each atom type that are closer than some cutoff distance. For each of the transformations mentioned above, this descriptor remains unchanged. As a more realistic example, Smith, Isayev, and Roitberg [47] propose a descriptor with both a radial and an angular component. The radial component is a vector, where each element of the vector (roughly speaking) counts the number of neighboring atoms within an annulus at some distance from the central atom. The angular component is a matrix, where each element (roughly speaking) counts how many pairs of neighbors both fall within a certain annulus and also form an angle with the central atom within a certain range. The overall descriptor combines both the radial and angular components, where each component is

repeated for different sizes of the annulli and angular buckets. This descriptor is invariant to translations and rotations because it only considers distances and angles between atoms, and it is invariant to permutations because it counts up the total number of atoms within each radius/angular bin.

More sophisticated methods make use of properties of spherical harmonics to achieve invariance. For example, Bartók, Kondor, and Csányi [4] project a smoothed version of the density onto a basis of spherical harmonics, and then they compute invariant features from the projection coefficients by summing up the modulus squared of all coefficients corresponding to each value of ℓ. Thompson et al. [51] and Bartók et al. [5] also project the neighbor density onto a basis of spherical harmonics, but they use a slightly different basis set and use a different method to compute invariant features from the projection coefficients. Drautz [15] generalizes many existing descriptor methods by proposing the atomic cluster expansion (ACE). Studying local atomic descriptors continues to be an active area of research, with several recent papers for example improving the computational efficiency of descriptor calculation [13, 16, 36].

Energy Prediction

Once these per-atom descriptors have been calculated, there are a range of techniques used to combine descriptor values from all atoms in the system to predict the system's energy. Linear regression [51] and Gaussian process regression [5, 14, 33, 6], and shallow neural networks [9, 8, 47] are common options. Interatomic potential models are typically trained on energies obtained from ab initio calculations like DFT, though some also train on forces [33].

Neural Network Interatomic Potentials (NNIPs)

As an outgrowth of the line work by [9] and others, which put shallow neural networks on top of local atomic descriptors, several authors proposed using increasingly complex neural network architectures using a similar invariant featurization of the nuclear coordinates that uses only distances and in some cases angles [47, 58, 35, 44]. These architectures make use of "interaction" layers, which combine features of nearby atoms several times throughout the network instead of only once in the descriptor-calculation stage.

One particularly simple example of this approach is SchNet, which uses a continuous version of convolutions for its interaction layer [43]. Standard convolutional layers, like those used for image processing, cannot be directly applied to atomic systems, since the nucleii don't lie on a grid. To address this, SchNet uses spherically symmetric continuous convolutional filters that compute the features $x_i^{\ell+1}$ for atom i at layer $\ell + 1$ as

$$x_i^{\ell+1} = \sum_j x_j^\ell W^\ell(||r_j - r_i||),$$

where $W^\ell(r)$ is a learned function that serves as the convolutional filter at layer ℓ, and the sum is over neighbors of atom i. This convolution operator is invariant to translations and rotations by means of only using interatomic distances, and it is invariant to permutations because it sums over all neighbors.

Graph Neural Network Formulation

Although the aforementioned papers did not identify as such, a useful abstraction for describing these and other interatomic potentials is the graph neural network (GNN). In a GNN, the raw atomic coordinates are first transformed into a graph, where nodes represent nuclei and edges connect neighboring nuclei. Edges can connect either atoms that are bonded together or atoms that are closer in Euclidean space than some cutoff radius. Each node is assigned node features corresponding to the atomic species of the nucleus, and each edge can be assigned a feature as well, such as the Euclidean distance between the two connected nuclei. This graph representation is invariant to rotations and translations because it only contains distances between pairs of nuclei. Once the graph has been constructed, any number of GNN architectures can be used to predict the energy of the atomic system, and a prediction for the forces acting on the nuclei can be calculated by taking the gradient of the energy with respect to atomic positions.

Duvenaud et al. [17] introduced graph convolutional networks for molecular modeling, where atom features are updated in each layer by summing up neighboring atom features and passing the result through a linear layer. Gilmer et al. [23] unify this and many other models into a framework called message-passing neural networks (MPNN). In an MPNN, each node i has features x_i and each edge has edge features e_{ij}. At each layer ℓ in the network, node i has a hidden state h_i^ℓ, which gets updated in the message-passing step. The first step to updating h_i^ℓ is to compute "messages" from all the neighboring nodes, and then to sum them up to form $m_i^{\ell+1}$:

$$m_i^{\ell+1} = \Sigma_j M_\ell(h_i^t, h_j^t, e_{ij}),$$

where the sum is over neighboring nodes, and M_ℓ is a learnable function. Then, the hidden states for each node are updated as

$$h_i^{\ell+1} = U_\ell(h_i^t, m_i^{t+1}),$$

where U_ℓ is another learnable function. Finally, after all N layers are complete, there is a readout function R that computes the final prediction based on the final hidden states h_i^N.

A number of other interatomic potentials based on GNNs have been proposed, including DimeNet [21], which additionally gives angles between neighbors to the GNN, GemNet [20], which adds dihedral angles to DimeNet, PhysNet [54], which predicts energies by summing up the final output with intermediate outputs, and others.

Equivariant Neural-Network Interatomic Potentials

Training a GNN on top of an invariant representation of the atomic system guarantees that the resulting predictor will also be invariant. The invariant descriptors described above don't necessarily discard any relevant information, but several authors hypothesized that requiring the input to the neural network to be invariant makes it more difficult for the network to reason about geometric relationships. For example, while it is possible to reconstruct the full geometry of the system using only the pairwise distances between nuclei, it is nontrivial to calculate, say, the angle between two neighbors using only pairwise distances. Doing so would require multiple layers of computation in an MPNN and for the network to learn how to do trigonometry. Even if we add angles into the invariant descriptor, it is still difficult for the neural network to calculate, for example, dihedral angles, or distances to atoms outside the cutoff radius.

While it is always possible to add more and more features into the invariant descriptor, the next wave of neural network interatomic potentials instead took a different approach: give the network the raw displacement vectors between neighbors, which are not invariant to rotations, but limit network to architectures that only compute equivariant functions of the inputs. Interatomic displacement vectors are not invariant to rotations, but unlike invariant descriptors, displacement vectors can easily be used to calculate geometric quantities like angles between neighbors (take a dot product within one layer), dihedral angles (take a dot product across two layers), and multi-hop distances (add displacement vectors across multiple layers). Furthermore, while displacement vectors are not invariant to rotations, they are equivariant, meaning that rotating the atomic system leads to a rotation of the displacement vectors. If the neural network is constrained to only carry out equivariant computations, then the final energy prediction will also be equivariant, as desired (for scalars, equivariance to rotation is the same as invariance).

Examples of rotation-equivariant calculations include the dot product and cross product; if we were to limit a neural network to only compute dot products and cross products, then we would be guaranteed that its output is equivariant to rotations. Thomas et al. [50] generalize cross products and dot products in a way that both extends to higher dimensions and also allows the introduction of learnable weights. They then propose a convolutional neural network architecture for point clouds that only makes use of these equivariant calculations. Batzner et al. [7] propose NequIP, a GNN interatomic potential that uses the equivariant calculations introduced by Thomas et al. [50] and improves upon their results, and many other equivariant neural network interatomic potentials have also been proposed [37, 38, 10, 49, 25, 42]. A full treatment of equivariant neural networks is beyond the scope of this thesis, but I recommend Geiger and Smidt [22] to the interested reader.

2.2 Generative Modeling

Problem Setting

Predicting the energy of a structure is not usually an end goal in and of itself. Instead, interatomic potentials are typically used for a number of downstream tasks, like finding low-energy geometries, running molecular dynamics or MCMC simulations, estimating transition states, or finding reaction rates. All of these tasks require some understanding of the potential, but each individual task may not require as holistic an understanding as is provided by a NNIP. Noting this, a number of machine learning methods have been proposed that specialize to these individual tasks (e.g. [55] for ground-state estimation, [12] for transition-state estimation, and [18] for molecular dynamics simulations). As a representative example, we consider the foundational task of predicting low-energy geometries. This is a particularly important task because structures corresponding to local minima of the potential function are the ones most likely to appear in nature, and finding ground states is often a first step before carrying out further analysis.

Methods take a variety of approaches to the task of finding low-energy geometries. The simplest setup is that of *de novo* generation, where a model is trained on a distribution of ground state geometries and learns to produce similar geometries without any additional constraints. This problem applicable for example in the drug discovery setting, where we might have a dataset of existing drug-like molecules and we want to generate additional candidates for further study. A relatively simple extension of this setting is to condition generation on desired property values. For example, we may want to generate drug-like moleclues that have a strong binding affinity to a particular protein, and we have a training dataset of drug-like molecules labeled with their binding affinities. Additional complication arises when we want to condition generation on more complex information. For example, Li, Pei, and Lai [32] propose a method to generate geometries of drug-like molecules conditioned on the binding pocket of a protein. Shi et al. [45] condition generation of 3D geometries on a molecular graph, which allows them to generate conformers for particular molecules instead of generating new molecular structures.

Aside from varying problem settings, a number of modeling techniques have been proposed for generating geometries. Simm, Pinsler, and Hernández-Lobato [46] place individual atoms in 3D using reinforcement learning; Ragoza, Masuda, and Koes [39] make use of a 3D convolutional neural network on voxels representing the 3D density of atoms; and Garcia Satorras et al. [19] generate 3D geometries using equivariant normalizing flows. Recently, motivated by their success in other domains, denoising diffusion models have been gaining popularity for modeling atomic systems.

Denoising Diffusion Models

Denoising diffusion models for generative modeling were popularized by the success of Ho, Jain, and Abbeel [26], who used these models to generate high-quality images. Diffusion

models have also become popular for generative modeling of 3D geometries of atomic systems [56, 28, 27, 45, 29]. Diffusion models train on a dataset of low-energy geometries, and they can be conditioned on labels in the dataset to control the generative process (e.g. to generate geometries with high binding affinity to a protein). During training, noise is added to the geometries from the dataset, and the model is tasked with de-noising its input by predicting what noise was added to the training set geometry to obtain the model's input. Instead of predicting the entire noise vector in one step, diffusion models break down the problem into a series of steps, where at each step, the model predicts a structure that is slightly less noisy than its input. At inference time, these steps are applied in order starting from entirely random noise, and at each step, the model makes the input look a little more like an example from the training distribution and a little less like noise. See [27] and [26] for a more formal introduction to diffusion models.

For atomic systems, training examples are more structured than for images, where diffusion models first saw widespread adoption: to represent an atomic system, one needs to include the 3D coordinates of the nuclei, the atomic species of each nucleus, and the charge of the molecule (or perhaps formal charges on individual atoms), whereas for images, one needs only a 2D grid of pixel values. Atomic species is a categorical variable, and nuclear coordinates are continuous, whereas pixel values are ordinal. Atomic systems also exhibit translational and rotational equivariance, which are much less relevant for image datasets. Hoogeboom et al. [27] adapt the denoising diffusion models of [26] to account for both of these requirements. Their method, called "E(3) Equivariant Diffusion Model" (EDM), incorporates categorical features by representing atomic species as a one-hot vector, which is a vector of ordinal values, and then by using the same machinery developed for pixel values. The method incorporates continuous positions by showing that they can be treated similarly to the transitions between latent states but with an additional normalization factor. EDM also ensures equivariance to translations and rotations is respected by working in the center-of-mass frame (to achieve translation invariance) and by using an equivariant neural network to predict each denoising step.

We investigate EDM in more detail in Chapter 3.

Chapter 3

Self-Supervised PES Learning

This Chapter contains work co-authored with Andrew Rosen, Eric Taw, and Connie Robinson, in addition to my advisors Aditi Krishnapriyan and Joseph Gonzalez.

3.1 Introduction

The potential energy surface (PES) is a fundamental quantity for understanding the behavior of atomic systems. Finding stable geometries, estimating reaction rates, and predicting transition states all require an understanding of the shape of the PES. Accurate first-principles methods for calculating points on the PES are computationally expensive or even entirely infeasible for larger systems, so a number of cheaper alternatives have been developed, such as classical force fields, tight-binding methods, and machine learning methods, the last of which we focus on here. The vast majority of machine learning methods aimed at understanding the shape of the PES are learned inter-atomic potentials, which today largely consist of neural-network interatomic potentials (NNIPs). NNIPs are trained in a supervised fashion using a dataset of geometries that are annotated with energies and forces derived from a more expensive physics-based method. The result is a machine learning model that can predict, given a particular geometry of atoms, the energy of that configuration and the forces experienced by each nucleus.

While machine-learned interatomic potentials are a powerful tool for understanding the shape of the PES, they suffer from the major limitation that they require a supervised dataset of geometries that have been annotated with energies and forces. Relying on datasets based on first-principles calculations is problematic for several reasons:

- **Computational expense.** It is extremely computationally expensive to obtain this sort of dataset. For example, the OC20 dataset includes 1.3M relaxations, each of which was allowed up to 1728 core-hours of compute time [11].
- **Lack of inclusion of all geometries.** The space of off-equilibrium geometries is combinatorially large, and there is no clear way to choose which geometries to include

in the dataset. If important types of geometries are excluded, models trained on the dataset may generalize poorly.

- **Limited by the level of theory.** Because of how expensive it is to produce the dataset, we are limited in the level of theory that can be used for generating the energies and forces. For example, authors often generate geometries using tight binding calculations, which are significantly less accurate than DFT [1, 2]. Generating a labeled dataset experimentally is not feasible, as it would require measuring energies and forces for arbitrary off-equilibrium geometries. As such, learned interatomic potentials will always be limited by the level of theory they were trained on.

To help alleviate these concerns, we explore in this chapter an alternative method, based on self-supervised learning, to understand the shape of the PES that does not involve learning an interatomic potential.

Self-supervised learning has powered many of the recent successes in the machine learning community, from image generation to style transfer to ChatGPT. In more traditional supervised learning such as the NNIPs described above, the dataset consists of examples (e.g. images, or molecular structures) and labels (e.g. what object is depicted, or the energy of the structure), and models are trained to predict the labels from the inputs. In self-supervised learning, the dataset consists only of the examples themselves, with no corresponding labels. With no labels to train on, we have to be more creative about how to train models. One popular strategy is to treat part of the input as a label, and use the rest of the input to predict the label as in supervised learning. Predicting a patch of an image given the rest of the image, or predicting the position of a missing atom given all the other atom positions are two such examples. Another popular approach, which we focus on here, is to add noise to the input and train a model to predict the original, de-noised version. More specifically, we investigate adding noise to the 3D coordinates of each nucleus in their local energetic minima, and then we predict the original positions.

A self-supervised approach helps to alleviate the problems with learning from a labeled dataset described above:

- **Computational expense.** It may be less expensive to generate a dataset of only ground states than to generate a training dataset for NNIPs, since we can initialize calculations at a higher level of theory with the best guess from a lower level of theory. In contrast, NNIP training data must include many off-equilibrium structures at a high level of theory so that they learn to generalize beyond the immediate neighborhood of the ground state.

- **Lack of inclusion of all geometries.** There is no need to choose which geometries should be included, since a self-supervised dataset includes only ground states.

- **Limited by the level of theory.** Unlike an NNIP, a self-supervised method can in principle be trained on experimental data. Training ML models on experimentally

measured geometries is challenging, and we do not pursue experimental structures in this dissertation, but we think even the possibility of training on experimental data is a major advantage for self-supervised methods over NNIPs.

Our goal is not to train a better interatomic potential; prior work has already investigated denoising as a way to improve supervised learning, including for NNIPs [57, 24, 34]. We are also not proposing a new way to train denoising models on chemical systems, rather opting for an off-the-shelf Equivariant Diffusion for Molecules (EDM) model [27]. Instead, our goal here is to understand, from a scientific standpoint, how much can be learned about the PES using only a denoising objective, without any supervision on energy or forces.

Instead of using a supervised dataset of geometries labeled with energies and forces, we train only on unlabeled geometries of ground-state structures, or structures corresponding to local minima of the PES. The EDM models we use were trained to generate new ground state geometries given a training set of known ground states. To assess how much the model has learned about the PES, we re-purpose it in three different ways: to carry out structure relaxations, to estimate the force experienced by the nuclei, and to sample structures from the Boltzmann distribution.

We investigate the model's performance on small organic molecules, where the PES is relatively easy to reason about. What we find is that 1) EDM can find low-energy structures within an accuracy of $\sim$ 1kcal/mol, even when the starting structure is out of the training distribution; 2) The path EDM takes to the ground state is a noisy estimate of the direct path from the current geometry to the ground state, rather than an estimate of the local forces at the current geometry, suggesting that the model has learned about local curvature in the PES; and 3) EDM can draw samples from a distribution similar to the Boltzmann distribution, even though it was not trained on any off-equillibrium structures, and it can capture the relationship between energy and temperature (i.e. heat capacity).

To summarize, we make the following contributions:

- We undertake a study of a pretrained EDM model, finding that its inference procedure can be roughly divided into an "exploration" regime and a "relaxation" regime (Section 3.2).
- We demonstrate that the relaxation regime of the EDM model can be used to robustly carry out structure relaxations (Section 3.3).
- We compare EDM's predictions for how to de-noise the nuclear positions with the actual forces experienced by the nuclei (Section 3.4).
- We use EDM to sample from a molecule's Boltzmann distribution, establishing a correspondence between diffusion steps and temperature (Section 3.5).

3.2 Physical Intuition Behind EDM

We investigate how the diffusion process progresses once trained. We aim to gain a physical understanding of the model's operation that will inform what other chemical tasks we might be able to carry out besides the intended de-novo molecular generation.

General procedure. Inference on an EDM model proceeds as follows. First, we sample a random distribution over chemical species and a random set of 3D coordinates for each nucleus. Then, for each of T timesteps, we use an equivariant neural network to predict how the atom species distribution and nuclear positions should change, and we sample the next structure from a Gaussian distribution around that predicted delta.

We test on QM9, a dataset of small molecules with up to nine heavy atoms among CNOF [41, 40]. To match QM9, we carried out all DFT calculations at the B3LYP/6-31G(2df,p) level of theory. We use Psi4 [53] version 1.8, and relaxations were carried out with the Atomic Simulation Environment (ASE) [31] version 3.22.1 using BFGS with `fmax` of either 0.01 or 0.03eV/A.

Results. At the beginning, the atomic species and 3D positions are completely scrambled, but by the end, the model has decided on which atomic species to use and where to place them. This progression is shown in Figure 3.1, which plots as a function of diffusion steps the fraction of atomic species that have been finalized, the fraction of atoms that have the correct valence, and how close the interatomic distances are to their final values.

Considering first the plot of what fraction of atomic species have been finalized, it is clear that the diffusion process can be divided into two regimes: an "exploration" regime, from step 0 to step ~ 950, where the model is still figuring out the atomic identities, and a "relaxation" regime, starting at step ~ 950, where the model is moving around the atoms while holding the atom types fixed. The transition from 0% finalized to 100% finalized is fairly abrupt, suggesting that the model is deciding on all the atomic species at once instead of first deciding on, say, the carbon structure and then deliberating about which functional groups to add. Note also that the model finalizes geometries decidedly after choosing the atomic species: the curve showing which fraction of molecules have the correct valence doesn't increase until after almost all atomic species have been decided on, and the RMSD of interatomic distances rapidly decreases starting at step 950, right after the atomic species have been decided upon.

Looking more closely at the relaxation regime, we calculate the energy of each structure along the diffusion path, starting with the first structure where all atom types are finalized. As seen in Figure 3.2, the energy consistently decreases during the relaxation regime, suggesting that the diffusion process is largely following the potential energy surface to the ground state rather than moving atoms through each other or taking a more erratic path. We investigate this further in Section 3.4.

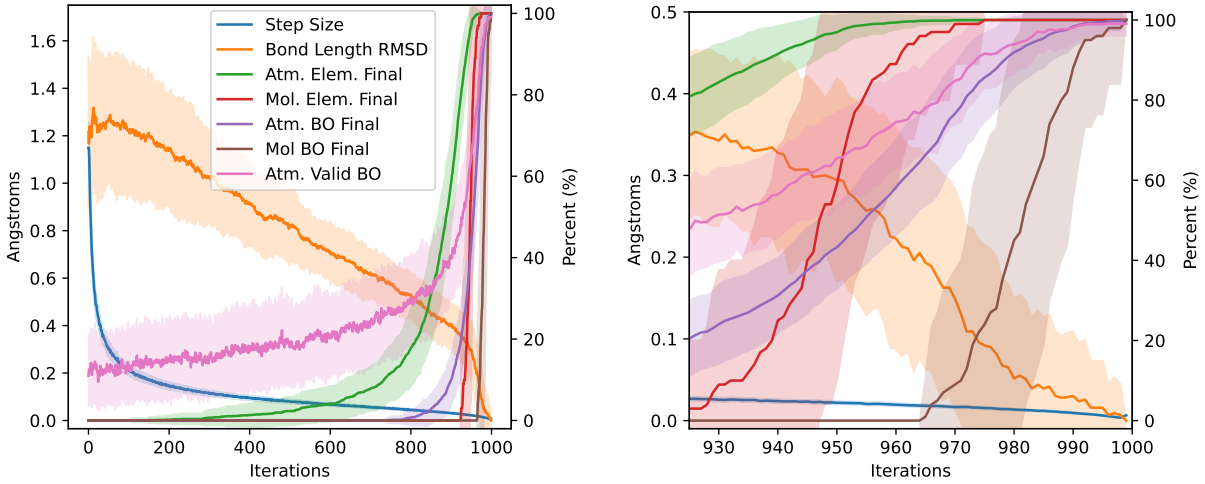

Figure 3.1: Left: ℓ_2 norm of the step size taken at each step in diffusion ("Step Size"); root-mean-squared deviation between interatomic distances of the current structure compared to the final structre, considering only distances between atoms bonded in the final structure ("Bond Length RMSD"); fraction of atoms whose chemical species is finalized ("Atm. Elem. Final"); fraction of molecules where every atom's chemical species is finalized ("Mol. Elem. Final"); fraction of atoms that have the same number of bonds as they do in the final structure ("Atm. BO Final"); fraction of molecules where all bond orders have been finalized ("Mol BO Final"); fraction of atoms that have a valid number of bonds (e.g. 4 for carbon, "Atm. Valid BO"). Right: zoom of figure on the left.

3.3 EDM Can Relax Structures

Noting the separation of the diffusion process into these two regimes, and that the energy of generated structures consistently decreases during the relaxation regime, a natural question to ask is whether we can successfully relax arbitrary structures by running diffusion steps $\sim 950 \to 1000$, starting at a user-chosen unrelaxed structure. During training, the model only sees structures that are a Gaussian perturbation away from the ground state, so any initial structure we provide the model not generated in this way will be outside of the training distribution. As such, there's no guarantee that the relaxation regime will actually find a low-energy geometry if it is initialized at one of these out-of-distribution states. To test this, we initialize the diffusion process with the ground state produced by running an optimization using the Merck Molecular Force Field (MMFF94) [52] on molecules from the QM9 validation set, and then we run the last N steps of diffusion. To find the MMFF ground state, we use RDKit to embed the molecule in 3D given only the molecular graph, and then we relax this structure using MMFF.

The MMFF-optimized structures are already a fairly good approximation of the true

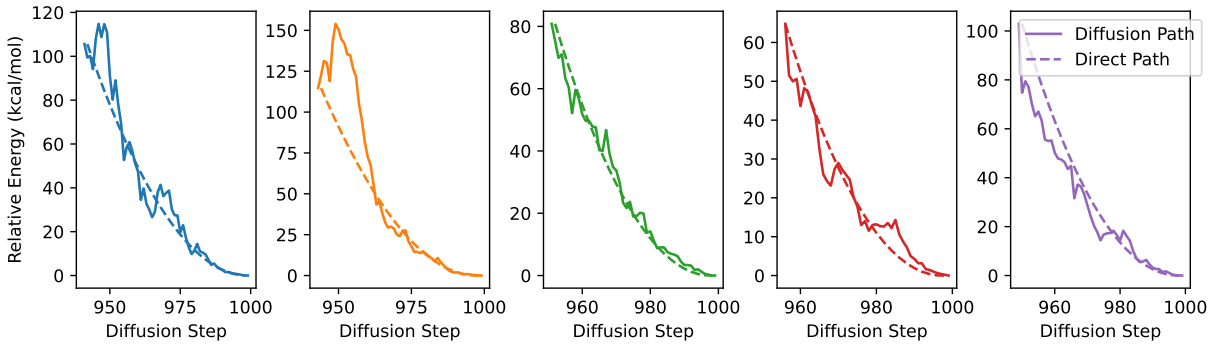

Figure 3.2: DFT-computed relative energy compared to the final generated geometry for the N final images in the chain that have the correct atom types. Each plot corresponds to a different generated molecule. Solid lines are geometries predicted by diffusion, and dashed lines are geometries that are linearly interpolated between the initial and final geometries.

ground state, but we seek to further improve them using EDM. To evaluate the diffusion-generated structures, we compare the DFT-computed energies of the diffused structures to the MMFF structures. We also calculate how many steps are required to carry out a DFT relaxation when initialized at the diffused structure vs. the MMFF structure.

Diffusion-relaxed Structures Have Lower Energies

The question remains what diffusion step N to start at when carrying out the relaxation. As shown in Figure 3.1, the model consistently makes smaller/larger steps for earlier/later diffusion steps, so we need to choose N carefully: too large, and the model won't have enough steps to move the distance required to reach the ground state; too small, and the model will drastically re-arrange the molecule instead of finding the nearest local minimum. Most likely, we should use $N >\sim 950$, since before that point, the model has not finalized atom species. As such, we try three values of N: 950, 970, and 980.

Figure 3.3 shows DFT-computed relative energies between the DFT-relaxed structure and all structures along the diffusion path for each of these values of N. For all values of N, the diffused structures are on average equally good, on average about 1 kcal/mol worse than the DFT-computed ground state. On the high end, at $N = 980$, the model immediately begins improving the structure, whereas on the low end, at $N = 950$, the model first worsens the structures, increasing the energy, but eventually finds as good or better final structures as the lower values of N. Because in practice we do not know how far away our initial state is from the true ground state, it is important that this method is robust to choosing too many steps, since if so, we can safely overestimate N and still arrive at high-quality ground states.

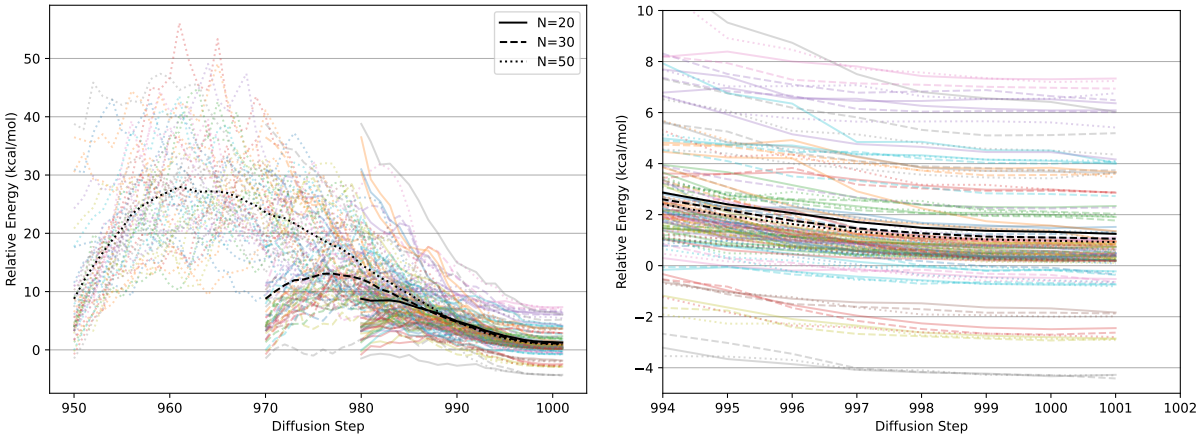

Figure 3.3: Left: DFT-computed energies relative to the ground state for all structures on the diffusion-generated "relaxation" paths, for 45 randomly chosen molecules from the QM9 validation set. Linestyle indicates how many diffusion steps were used ($N = 980$, 970, and 950). Different colors represent different individual relaxations. Black lines are averages of the corresponding colored lines. Right: zoom of left figure.

Diffusion-relaxed Structures Accelerate DFT Relaxations

Next, we compare the number of iterations required to relax structures using DFT when initialized at the MMFF ground state vs. the diffusion-generated geometry. We perform DFT relaxations from three different starting points: the MMFF-optimized structure, and the structures obtained by further "relaxing" this MMFF-optimized configuration using diffusion with both $N = 980$ and $N = 950$. The number of DFT iterations required for $N = 980$ and $N = 950$ are shown in Figure 3.4, plotted vs. the number of steps required when starting at the MMFF-optimized structure.

For both $N = 980$ and $N = 950$, the DFT relaxations are consistently faster: the median number of steps required is 40% lower for $N = 980$ and 57% lower for $N = 950$. On each point in Figure 3.4, we plot an arrow showing the difference in energy between the DFT-relaxed structures when starting at each of the diffusion-generated geometries and the DFT-relaxed structure when starting at the MMFF-optimized structure. If the diffused geometries stay within the same local minimum of the DFT PES as the MMFF-optimized structures, then we would expect these arrows to be approximately zero.

This is the case in every structure but one, in which diffusion with $N = 950$ steps moved the geometry to a better local minimum than MMFF found, as indicated by the negative arrow of magnitude ~ 1 kcal/mol (scale bar in lower right). This structure is also one of the few where the relaxation took longer when starting from the diffusion-generated geometry. Interestingly, when using only $N = 980$ steps on the same structure, diffusion does not stray from the MMFF minimum, which is unsurprising given the results in Figure 3.3.

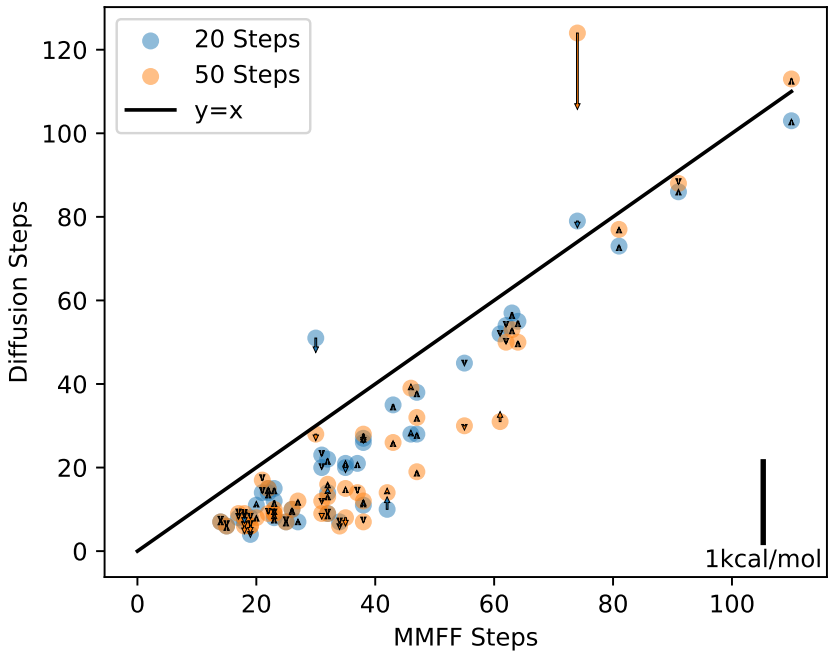

Figure 3.4: Each point in the plot is a structure that we relaxed using DFT. The value on the x-axis is the number of steps it took for the relaxation to converge when initializing DFT with the structure predicted by MMFF. The value on the y-axis is the number of steps when initializing with the structure predicted by diffusion. Orange dots indicate structures that were generated using 50 steps of diffusion (i.e. $N = 950$), while blue dots are for 20 steps (i.e. $N = 980$). Arrows indicate the difference between the energy of the DFT-relaxed structure when starting from the MMFF structure vs. starting from the diffused structure. An arrow with zero magnitude indicates that the DFT-relaxed structures had the same final energy when starting with either initial geometry. Negative-pointing arrows indicate that the relaxed structure had lower energy when the DFT relaxation was initialized with diffusion than with MMFF.

3.4 Alignment of Denoising Steps with Forces

As seen in Figure 3.2, the relaxation regime of the diffusion process consistently improves structures down to the ground state. However, although the energy mostly decreases as diffusion proceeds, and the final structure is close to the DFT-computed ground state, it is unclear which exact path the model takes to the ground state. One simple hypothesis is that it follows the forces down to the minimum; another is that it takes the most direct path to the minimum.

To differentiate between these hypotheses, we compute the cosine similarity between the steps that diffusion makes ("Δ"), the DFT-computed forces at each geometry along the

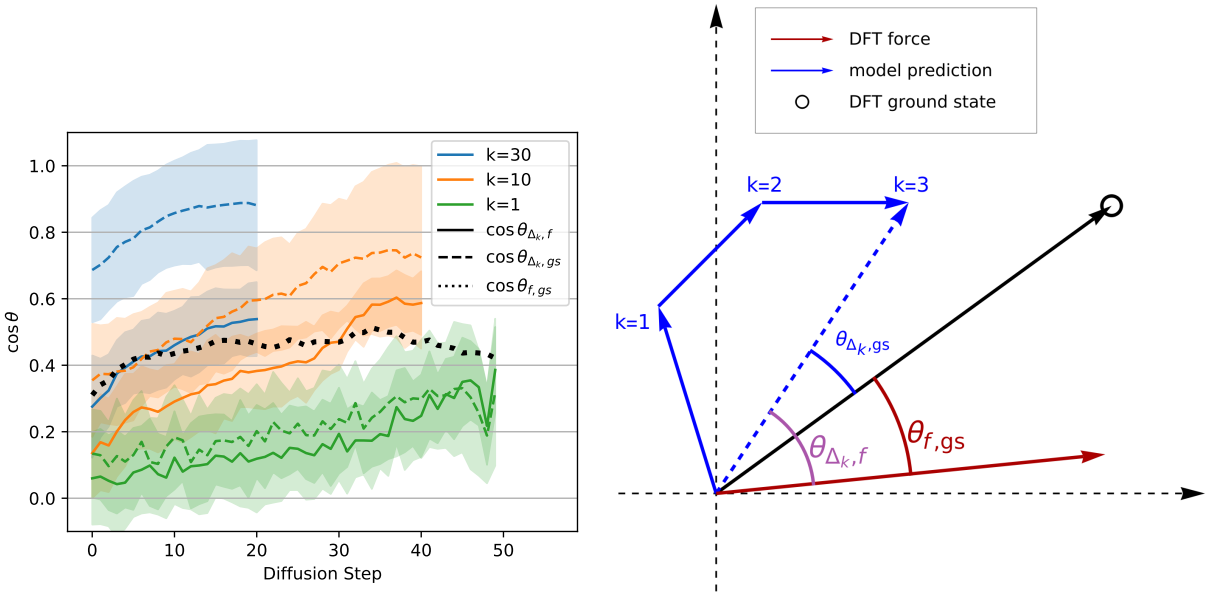

Figure 3.5: Left: cosine of the angles between the sum of the next k steps predicted by diffusion (Δ_k), the DFT-computed forces (f), and the path directly from the current geometry to the DFT-computed ground state (gs). Curves are averaged across atoms from 45 molecules from the QM9 validation set. Shaded regions represent one standard deviation above and below the mean. (The shaded region can exceed 1.0 because the distribution is not Gaussian.) Right: schematic showing what different values of k mean.

diffusion path ("f"), and the direction straight from each geometry to the DFT-relaxed ground state ("gs"). If the model tends to follow the forces down to the minimum, then we expect the angle between Δ and f, ("$\theta_{\Delta,f}$") to be small, or $\cos\theta_{\Delta,f}$ to be large. In particular, it should be large compared to $\cos\theta_{\Delta,gs}$. Similarly, if the model heads straight for the ground state, ignoring bumps in the PES along the way, then we expect $\cos\theta_{\Delta,gs}$ to be large compared to $\cos\theta_{\Delta,f}$.

Results are plotted in Figure 3.5 (left). The green lines show $\cos\theta_{\Delta,f}$ and $\cos\theta_{\Delta,gs}$ as diffusion progresses. At first glance, it appears that diffusion is taking a path to the minimum that differs from both following the forces and the direct path, since $\cos\theta$ tops out around 0.3. However, noting that the diffusion process is inherently noisy, we also plot results when we consider the model prediction to be the sum of the subsequent k steps of diffusion (the green line represents $k = 1$). This idea is depicted schematically on the right side of Figure 3.5, and results are shown for $k = 10$ and $k = 30$ on the left side.

As k increases, alignment increases between the model prediction and both the forces and the direct path to the ground state. The model predictions are substantially more aligned with the direct path to the ground state than with the DFT-predicted forces, suggesting that the model is finding a more efficient path to the ground state than following the forces

gradient-descent style. Although there is some alignment between the model predictions and the forces, this can be explained by the fact that the forces themselves are somewhat aligned with the direct path to the ground state (black dotted line in Figure 3.5). $\cos\theta_{\Delta_k,f}$ is never substantially higher than $\cos\theta_{f,gs}$, so any alignment between the model predictions and the forces can be explained by the alignment between the forces and the straight path to the ground state.

Given that these results are on molecules that were unseen during training, the fact that the diffusion path aligns better with the direct path to the ground state rather than the forces suggests that the model has learned about local curvature of the PES, rather than only learning about local gradients. Non-learning-based relaxations also attempt to take the most direct path to the ground state; for example, BFGS preconditions the gradients with second-order information in order to move straight towards the minimum instead of following the gradients directly.

3.5 EDM as Boltzmann Generator

As seen in Figure 3.1, the step size taken by diffusion decreases monotonically for larger timesteps. At any given diffusion time step, the model does not always move directly towards the local minimum in the PES, both because there is inherent stochasticity in the diffusion process and because the model moves by a certain step size even if the geometry is closer to the local minimum than the step size. (This can also be seen experimentally by the bumpiness in Figure 3.2.) Given that the model has a preference for low-energy conformations, we might hope that repeatedly applying the same diffusion timestep to a structure might sample low-energy geometries more often than high-energy geometries. In particular, we might hope that the distribution of geometries follows the Boltzmann distribution, and that different diffusion timesteps N would correspond to particular temperatures T.

To investigate this possibility, we carry out MCMC simulations on ten molecules randomly from the QM9 validation set after filtering to select for flexible and linear molecules, where we expect more interesting behavior at non-zero temperature.[1] We run the Metropolis-Hastings algorithm with an isotropic Gaussian proposal distribution—initialized at the DFT ground state with 5,000 steps of burn-in—using eighteen temperatures T between 10K and 400K. On the diffusion side, we repeatedly apply a single step of diffusion to the structures, initialized at the DFT ground state with 1,000 steps of burn-in (we observed that the chain stabilizes faster than for Metropolis-Hastings). We try fourteen different steps N, from the last step down to step 960. In both cases, we use GFN2-xTB [3] to measure energies, as implemented by python's XTB package (version 22.1).

For very low T and N, we expect both the MCMC simulation at temperature T and the diffusion chain with step N to have a relative energy of 0 relative to the ground state. How-

[1] In particular, we use a simple filter based on the molecules' SMILES strings: we filter out SMILES containing "=" or "#", SMILES containing numbers, and SMILES with more than 15 "(" (to bias towards more linear molecules)

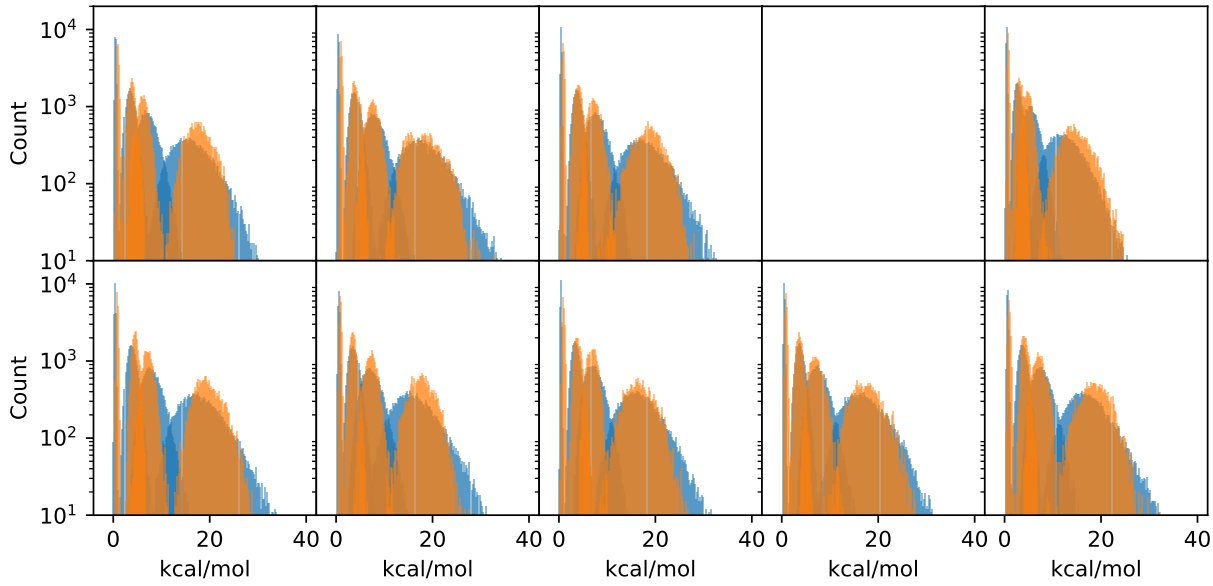

Figure 3.6: Histograms of the energies obtained with MCMC simulations at temperatures 20K, 70K, 120K, and 300K (orange histograms) and via repeated application of a diffusion model at steps 996, 986, 980, and 970 (blue histograms).

ever, because there is sometimes disagreement between GFN2-xTB and the DFT-calculated ground states that the model was trained on, the diffusion chains at low N settle to an energy slightly above zero. To compensate for this, for each molecule we subtract a constant energy ($< 2kcal/mol$) from each chain to equalize the minimum energies achieved by the two chains at the lowest values of T and N.

Histograms of the chain energies for selected values of T and N are shown in Figure 3.6. These values of T and N were chosen to maximize the overlap between the distributions, but they are the same for each of the nine molecules (we discarded the tenth because GFN2-xTB disagreed strongly with DFT about where the local minimum of the PES was), suggesting that there may be a correspondence between N and T that generalizes across molecules. Within each molceule, as T and N increase, we see the same trend of the mean relative energy increasing and the variance of the distribution increasing. As a point of reference, when repeatedly perturbing atomic coordinates with an isotropic normal distribution instead of the diffusion model, the energy diverges, even for extremely small step sizes. Within the diffusion chain, there is no proposal step as in the Metropolis-Hastings algorithm, and there is no form of feedback from a ground-truth energy oracle either. In addition, the model was trained only on geometries, with no energy supervision either on the ground-state geometries or on any non-equilibrium geometries. Despite this, we never observed a diffusion chain diverging, even for high values of N, and there is even reasonable agreement between the

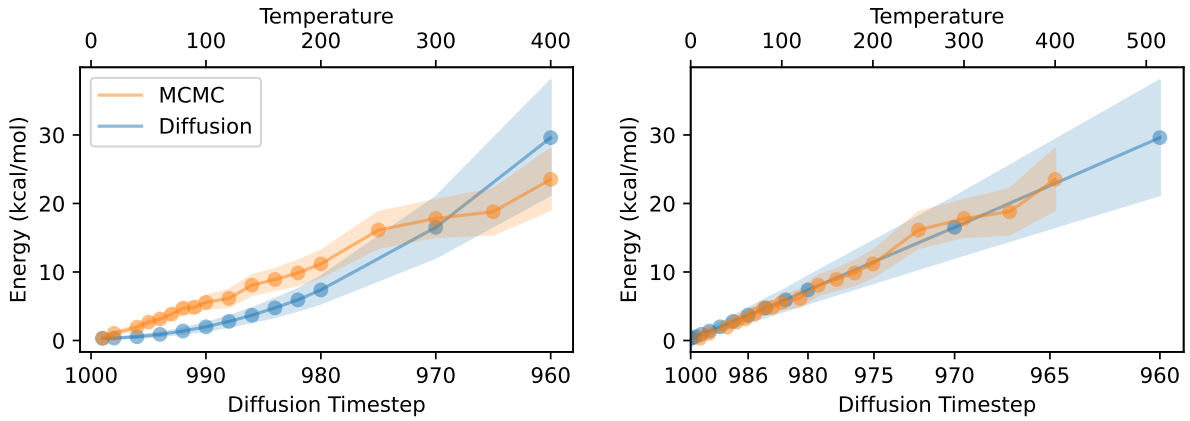

Figure 3.7: Left: average energy of the MCMC and diffusion chains, with shaded regions corresponding to one standard deviation above and below the mean. Values on the x-axis indicate the temperature used in the MCMC simulation (top) and the diffusion timestep N used when making the diffusion chain. Right: same as left, but the lower x-axis is scaled quadratically and stretched linearly to match the slope of the MCMC line.

distributions of energies within the MCMC and diffusion chains.

Next we investigate the relationship between T & N and the average & variance of the resulting energy distributions. Figure 3.7 plots the average energy $\pm$ the standard deviation of the energy for each of the N and T values we considered. As expected, the energy increases linearly with temperature. On the other hand, the energy increases quadratically with decreasing N. This is unsurprising: near the end of inference, the step size decreases roughly linearly, as seen in Figure 3.1, and near a local minimum, we expect the PES to be modeled well as a harmonic oscillator. The left side of Figure 3.7 plots both chains with a linear scale on T and N. The right side of the Figure instead uses a quadratic scale for N, and the x-axis is scaled to equalize the slope between the two chains. Any linear and quadratic functions can be made to line up using this method. Figure 3.8 plots the same quantities, but repeated for each of the nine molecules considered. In this case, we use the same linear scaling for each molecule, so there is no guarantee that the lines will all line up. Even though there is some variation in the heat capacities of the nine molecules (i.e. the slope of the MCMC lines), the diffusion chain consistently generates very similar average energies as the MCMC chain at the corresponding temperature.

The variances of the distributions are also similar, though the diffusion chain consistently results in a wider distribution of energies than the MCMC chain. The energy distributions for these simple molecules from QM9 are unimodal, but it would be interesting to see whether the diffusion chain can reproduce multi-modal energy distributions for more complex molecules.

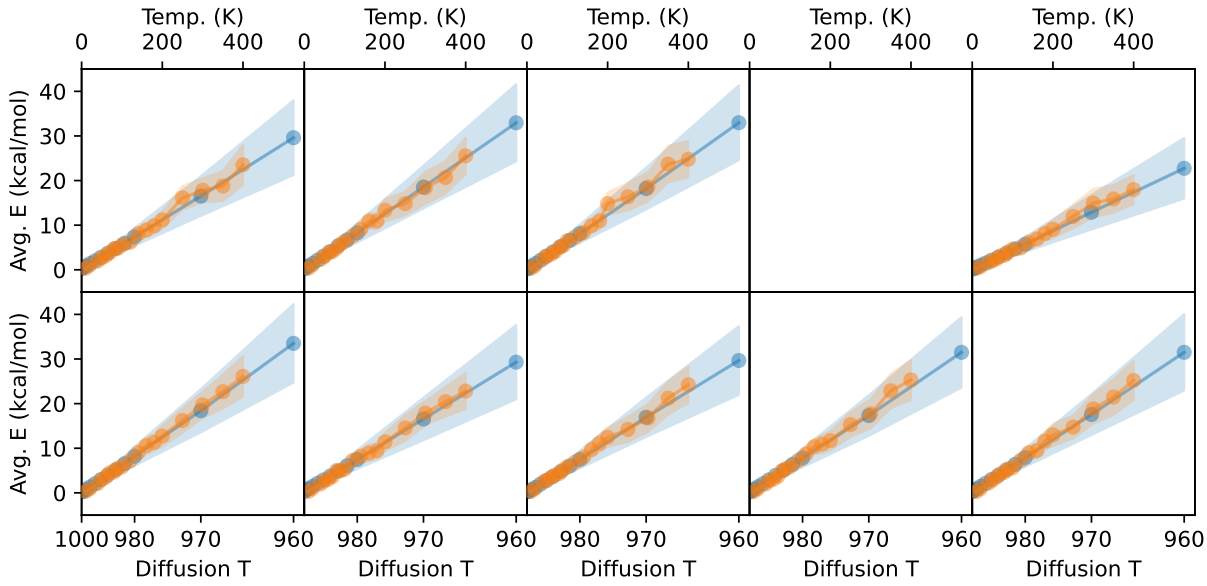

Figure 3.8: Right-hand plot of Figure 3.7 for the nine molecules considered. Only one linear scaling factor is used across all molecules.

3.6 Discussion

Motivated by a need to lessen our reliance on supervised datasets generated via physics-based calculations for learning about the shape of the PES, we investigate in this chapter how much can be learned about the PES by training a diffusion model on only ground state geometries. Our trained model is able to relax structures to 10x lower relative energy than MMFF, and it can produce structures that require 57% fewer DFT steps to relax than MMFF structures. When relaxing structures, the model follows a noisy estimate of the path directly to the ground state rather than taking the path of steepest descent. The model can also sample geometries around a local minimum in the PES from a distribution that resembles the Boltzmann distribution, and it can model the varying heat capacity of different materials.

Although we present some of our findings in terms of capabilities that the model has, we are not proposing that a self-supervised model could outperform state-of-the-art supervised NNIPs on tasks like structure relaxations or MCMC simulations. After all, the training data for the diffusion model used in this chapter consists of only a single point on the PES for each molecule; NNIP training datasets, in contrast, contain many points on the PES for each molecule, all labeled with energies and forces. Rather, we investigate the model's capabilities as a way both to see how far it is possible to get with self-supervision alone, and to gain insight into what information can be learned about the PES from a training set of ground-state geometries.

The molecules investigated here are fairly small and simple. This is an advantage insofar as it allows for easy reasoning about a PES that is well approximated as quadratic, but a disadvantage insofar as it limits our ability to investigate whether diffusion models can learn more complicated potential energy surfaces. For example, following the direct path to the ground state on QM9 molecules suggests that the model has learned about local curvature in addition to simply the local direction of steepest descent. But on a more complex PES, the path that diffusion takes becomes more interesting: does it still follow the direct path to the ground state even if it requires plowing through a large energy barrier? Larger molecules are also more likely to have a Boltzmann distribution with multiple peaks, and it would be interesting to see if diffusion can reproduce this behavior. We leave an investigation of self-supervised learning on larger/more complex molecules to future work.

Chapter 4

Conclusion

In this dissertation, we provided an introduction to the background quantum mechanics knowledge necessary for machine learning researchers to get started working on accelerating electronic structure problems. We surveyed existing work in this area, with an eye towards equivariant neural networks and diffusion-based models. Next, we investigated a particular diffusion model (EDM) more carefully as a way to gain insight into what these sorts of models can learn given only ground state geometries with no energy or force supervision. Although EDM was trained originally to generate new molecules *de novo*, we propose several tasks that EDM can transfer to with no additional training, simply by applying the model in novel ways. In this dissertation we investigate how far EDM can get with no additional training, but future work could investigate the advantage that self-supervised pretraining yields when training on the downstream task as well.

Accelerating electronic structure theory calculations is of critical importance for speeding up the discovery of new materials. This goal is especially important today given ongoing crises in especially energy and healthcare. We hope this dissertation can provide a starting point for future investigations into how to learn about the potential energy surface in a more computationally efficient way than training NNIPs using energy and force supervision.

Bibliography

[1] Simon Axelrod and Rafael Gomez-Bombarelli. "GEOM, energy-annotated molecular conformations for property prediction and molecular generation". In: *Scientific Data* 9.1 (2022), p. 185.

[2] David Balcells and Bastian Bjerkem Skjelstad. "tmQM dataset—quantum geometries and properties of 86k transition metal complexes". In: *Journal of chemical information and modeling* 60.12 (2020), pp. 6135–6146.

[3] Christoph Bannwarth, Sebastian Ehlert, and Stefan Grimme. "GFN2-xTB—An accurate and broadly parametrized self-consistent tight-binding quantum chemical method with multipole electrostatics and density-dependent dispersion contributions". In: *Journal of chemical theory and computation* 15.3 (2019), pp. 1652–1671.

[4] Albert P Bartók, Risi Kondor, and Gábor Csányi. "On representing chemical environments". In: *Physical Review B* 87.18 (2013), p. 184115.

[5] Albert P Bartók et al. "Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons". In: *Physical review letters* 104.13 (2010), p. 136403.

[6] Albert P Bartók et al. "Machine learning unifies the modeling of materials and molecules". In: *Science advances* 3.12 (2017), e1701816.

[7] Simon Batzner et al. "E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials". In: *Nature communications* 13.1 (2022), p. 2453.

[8] Jörg Behler. "Atom-centered symmetry functions for constructing high-dimensional neural network potentials". In: *The Journal of chemical physics* 134.7 (2011), p. 074106.

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

[10] Johannes Brandstetter et al. "Geometric and Physical Quantities improve E(3) Equivariant Message Passing". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=_xwr8gOBeV1.

[11] Lowik Chanussot et al. "Open catalyst 2020 (OC20) dataset and community challenges". In: *Acs Catalysis* 11.10 (2021), pp. 6059–6072.

[12] Sunghwan Choi. "Prediction of transition state structures of gas-phase chemical reactions via machine learning". In: *Nature Communications* 14.1 (2023), p. 1168.

[13] James P Darby, James R Kermode, and Gábor Csányi. "Compressing local atomic neighbourhood descriptors". In: *npj Computational Materials* 8.1 (2022), p. 166.

[14] Volker L Deringer et al. "Gaussian process regression for materials and molecules". In: *Chemical Reviews* 121.16 (2021), pp. 10073–10141.

[15] Ralf Drautz. "Atomic cluster expansion for accurate and transferable interatomic potentials". In: *Physical Review B* 99.1 (2019), p. 014104.

[16] Genevieve Dusson et al. "Atomic cluster expansion: Completeness, efficiency and stability". In: *Journal of Computational Physics* 454 (2022), p. 110946.

[17] David K Duvenaud et al. "Convolutional networks on graphs for learning molecular fingerprints". In: *Advances in neural information processing systems* 28 (2015).

[18] Xiang Fu et al. "Simulate Time-integrated Coarse-grained Molecular Dynamics with Geometric Machine Learning". In: *ICLR Workshop on Deep Generative Models for Highly Structured Data*. 2022.

[19] Victor Garcia Satorras et al. "E (n) equivariant normalizing flows". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 4181–4192.

[20] Johannes Gasteiger, Florian Becker, and Stephan Günnemann. "Gemnet: Universal directional graph neural networks for molecules". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 6790–6802.

[21] Johannes Gasteiger, Janek Groß, and Stephan Günnemann. "Directional Message Passing for Molecular Graphs". In: *International Conference on Learning Representations*. 2020.

[22] Mario Geiger and Tess Smidt. "e3nn: Euclidean neural networks". In: *arXiv preprint arXiv:2207.09453* (2022).

[23] Justin Gilmer et al. "Neural message passing for quantum chemistry". In: *International conference on machine learning*. PMLR. 2017, pp. 1263–1272.

[24] Jonathan Godwin et al. "Simple GNN Regularisation for 3D Molecular Property Prediction and Beyond". In: *International Conference on Learning Representations*. 2021.

[25] Mojtaba Haghighatlari et al. "Newtonnet: A newtonian message passing network for deep learning of interatomic potentials and forces". In: *Digital Discovery* 1.3 (2022), pp. 333–343.

[26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[27] Emiel Hoogeboom et al. "Equivariant diffusion for molecule generation in 3d". In: *International conference on machine learning*. PMLR. 2022, pp. 8867–8887.

[28] Lei Huang et al. "Mdm: Molecular diffusion model for 3d molecule generation". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 4. 2023, pp. 5105–5112.

[29] Bowen Jing et al. "Torsional diffusion for molecular conformer generation". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24240–24253.

[30] James Kirkpatrick et al. "Pushing the frontiers of density functionals by solving the fractional electron problem". In: *Science* 374.6573 (2021), pp. 1385–1389.

[31] Ask Hjorth Larsen et al. "The atomic simulation environment—a Python library for working with atoms". In: *Journal of Physics: Condensed Matter* 29.27 (2017), p. 273002. URL: http://stacks.iop.org/0953-8984/29/i=27/a=273002.

[32] Yibo Li, Jianfeng Pei, and Luhua Lai. "Structure-based de novo drug design using 3D deep generative models". In: *Chemical science* 12.41 (2021), pp. 13664–13675.

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

[34] Shengchao Liu, Hongyu Guo, and Jian Tang. "Molecular Geometry Pretraining with SE (3)-Invariant Denoising Distance Matching". In: *The Eleventh International Conference on Learning Representations*. 2022.

[35] Nicholas Lubbers, Justin S Smith, and Kipton Barros. "Hierarchical modeling of molecular energies using a deep neural network". In: *The Journal of chemical physics* 148.24 (2018), p. 241715.

[36] Yury Lysogorskiy et al. "Performant implementation of the atomic cluster expansion (PACE) and application to copper and silicon". In: *npj computational materials* 7.1 (2021), p. 97.

[37] Albert Musaelian et al. "Learning local equivariant representations for large-scale atomistic dynamics". In: *Nature Communications* 14.1 (2023), p. 579.

[38] Z Qiao et al. "Informing geometric deep learning with electronic interactions to accelerate quantum chemistry." In: *Proceedings of the National Academy of Sciences of the United States of America* 119.31 (2022), e2205221119–e2205221119.

[39] Matthew Ragoza, Tomohide Masuda, and David Ryan Koes. "Learning a Continuous Representation of 3D Molecular Structures with Deep Generative Models". In: ().

[40] Raghunathan Ramakrishnan et al. "Quantum chemistry structures and properties of 134 kilo molecules". In: *Scientific data* 1.1 (2014), pp. 1–7.

[41] Lars Ruddigkeit et al. "Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17". In: *Journal of chemical information and modeling* 52.11 (2012), pp. 2864–2875.

[42] Kristof Schütt, Oliver Unke, and Michael Gastegger. "Equivariant message passing for the prediction of tensorial properties and molecular spectra". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 9377–9388.

[43] Kristof Schütt et al. "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions". In: *Advances in neural information processing systems* 30 (2017).

[44] Kristof T Schütt et al. "Quantum-chemical insights from deep tensor neural networks". In: *Nature communications* 8.1 (2017), p. 13890.

[45] Chence Shi et al. "Learning gradient fields for molecular conformation generation". In: *International conference on machine learning*. PMLR. 2021, pp. 9558–9568.

[46] Gregor Simm, Robert Pinsler, and José Miguel Hernández-Lobato. "Reinforcement learning for molecular design guided by quantum mechanics". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 8959–8969.

[47] Justin S Smith, Olexandr Isayev, and Adrian E Roitberg. "ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost". In: *Chemical science* 8.4 (2017), pp. 3192–3203.

[48] John C Snyder et al. "Finding density functionals with machine learning". In: *Physical review letters* 108.25 (2012), p. 253002.

[49] Philipp Thölke and Gianni De Fabritiis. "Equivariant Transformers for Neural Network based Molecular Potentials". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=zNHzqZ9wrRB.

[50] Nathaniel Thomas et al. "Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds". In: *arXiv preprint arXiv:1802.08219* (2018).

[51] Aidan P Thompson et al. "Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials". In: *Journal of Computational Physics* 285 (2015), pp. 316–330.

[52] Paolo Tosco, Nikolaus Stiefl, and Gregory Landrum. "Bringing the MMFF force field to the RDKit: implementation and validation". In: *Journal of cheminformatics* 6 (2014), pp. 1–4.

[53] Justin M Turney et al. "Psi4: an open-source ab initio electronic structure program". In: *Wiley Interdisciplinary Reviews: Computational Molecular Science* 2.4 (2012), pp. 556–565.

[54] Oliver T Unke and Markus Meuwly. "PhysNet: A neural network for predicting energies, forces, dipole moments, and partial charges". In: *Journal of chemical theory and computation* 15.6 (2019), pp. 3678–3693.

[55] Tian Xie et al. "Crystal Diffusion Variational Autoencoder for Periodic Material Generation". In: *International Conference on Learning Representations*. 2021.

[56] Minkai Xu et al. "Geodiff: A geometric diffusion model for molecular conformation generation". In: *arXiv preprint arXiv:2203.02923* (2022).

[57] Sheheryar Zaidi et al. "Pre-training via Denoising for Molecular Property Prediction". In: *The Eleventh International Conference on Learning Representations*. 2022.

[58] Roman Zubatyuk et al. "Accurate and transferable multitask prediction of chemical properties with an atoms-in-molecules neural network". In: *Science advances* 5.8 (2019), eaav6490.