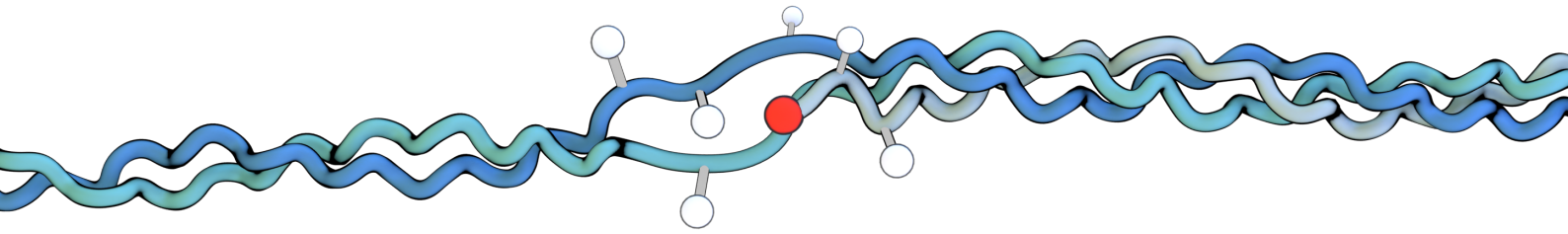
Kai Riedmiller

Dissertation

Predicting Hydrogen Atom Transfer in Collagen



Dissertation

submitted to the combined

Faculty of Mathematics, Engineering and Natural Sciences

of the Ruprecht-Karls-Universität Heidelberg

for the degree of

Doctor of Natural Sciences

Predicting Hydrogen Atom Transfer in Collagen

Put forward by

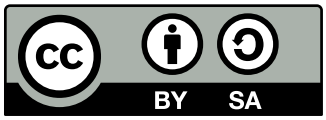
M.Sc. Kai Riedmiller

Referees: Prof. Dr. Frauke Gräter

TT-Prof. Dr. Pascal Friederich

Predicting Hydrogen Atom Transfer in Collagen

Oral examination: 21.06.2024



Abstract

Molecular Dynamics (MD) simulation is an established method for studying biological systems and materials at the molecular level. For example, it can predict the effects of mutations on the folding behavior of proteins, or elucidate the binding mechanisms of drugs to receptors. To achieve this, a molecular system is modeled as beads connected *via* springs or sticks. Goal of this thesis is to develop a method to deepen our understanding of mechanoradicals undergoing hydrogen atom transfer (HAT) reactions. Usually, no new chemical bonds can be formed during an MD simulation, prohibiting the analysis of HAT and other reactions.

In this thesis, a method called KIMMDY is implemented, enabling reactive MD through the use of kinetic Monte Carlo steps. While it has previously been used for homolysis reactions, here its scope is extended to also allow HAT during an MD simulation. First, a data set of HAT reaction barriers is crafted using quantum mechanical (QM) calculations. Then, a graph neural network (GNN) is trained on this data set to predict HAT reaction barriers. The inputs to the GNN are structures sampled from MD simulations, requiring no prior optimization. Thus, the model offers a significant speed-up compared to traditional methods, like reactive force fields, or direct QM calculations.

Furthermore, the KIMMDY 2.0 software package is presented, which can perform kinetic Monte Carlo driven reactive MD simulations in a flexible and user-friendly manner. It supports HAT and homolysis reactions, and can be easily extended to any reaction for which rates are available. For HAT, they are obtained using the aforementioned GNN. However, this approach is not limited to HAT. Through the use of machine learning models, reaction rates for arbitrary reactions become accessible, only requiring a limited amount of QM calculations during training. In KIMMDY, these models enable combining the accuracy and flexibility of QM calculations with the efficiency of MD.

Zusammenfassung

Molekulardynamik (MD) ist eine etablierte Methode zur Untersuchung biologischer Systeme und Materialien auf molekularer Ebene. Mit ihr lassen sich beispielsweise die Auswirkungen von Mutationen auf das Faltungsverhalten von Proteinen vorhersagen oder die Bindungsmechanismen von Wirkstoffen an Rezeptoren aufklären. Hierfür wird ein molekulares System aus Kugeln modelliert, die über Federn bzw. Stäbe verbunden sind. Ziel dieser Arbeit ist es, eine Methode zu entwickeln, die Wasserstoff-Atom-Transfer (HAT) Reaktionen von Radikalen beschreiben kann. Während einer typischen MD Simulation können keine neuen chemischen Bindungen gebildet werden, was die Analyse von HAT und anderen Reaktionen verhindert.

In dieser Arbeit wird eine Methode namens KIMMDY eingesetzt, die reaktive MD durch die Verwendung von kinetischen Monte-Carlo Schritten ermöglicht. Bisher wurde die Methode für Homolysereaktionen verwendet. Hier wird ihr Anwendungsbereich nun erweitert, um auch HAT während einer MD-Simulation zu ermöglichen. Zunächst wird ein Datensatz von HAT-Reaktionsbarrieren durch quantenmechanische (QM) Berechnungen erstellt. Auf diesem Datensatz wird dann ein Graph Neural Network (GNN) trainiert, um HAT-Reaktionsbarrieren vorherzusagen. Die Ausgangsdaten für das GNN sind Strukturen aus MD-Simulationen, welche keine vorherige Optimierung benötigen. Dadurch bietet das Modell eine erhebliche Beschleunigung im Vergleich zu herkömmlichen Methoden wie reaktiven Kraftfeldern oder direkten QM-Berechnungen.

Darüber hinaus wird das Softwarepaket KIMMDY 2.0 vorgestellt, welches auf flexible Weise mittels kinetischer Monte-Carlo Schritten reaktive MD-Simulationen ermöglicht. Es unterstützt HAT- und Homolysereaktionen und kann leicht um beliebige andere Reaktionen erweitert werden, für die Reaktionsraten verfügbar sind. Im Falle der HAT Reaktion werden diese durch das hier trainierte GNN erhalten. Neben HAT Reaktionen ermöglichen Machine-learning Modelle auch die Vorhersage von beliebigen anderen Reaktionen. Das Modell benötigt lediglich eine begrenzte Anzahl an QM-Berechnungen und ermöglicht KIMMDY dann die Genauigkeit und Flexibilität von QM-Berechnungen mit der Effizienz von MD zu kombinieren.

Acknowledgments

First, I would like to thank Prof. Frauke Gräter for her supervision, and her continuous support and guidance throughout this thesis. I am deeply grateful for her advice, for the very pleasant working atmosphere, and for the freedom I could enjoy during my time at HITS. Further, I would like to thank Dr. Ganna Gryn'ova and T.T.-Prof. Pascal Friederich for their support, the discussions, and valuable inputs. Prof. Robert Russell I want to thank for joining the examination committee. Additionally, I would like to acknowledge the Klaus-Tschira-Foundation and HITS for funding my project as part of the HITS Lab.

Most important for the great time I had at HITS were my colleagues from the MBM group, and therefore I would like to thank all of the current and former members. Especially, I want to thank Eric and Jannik, with whom I worked the closest during the last two years. Designing KIMMDY 2.0 together with them was a joy. Also during the last four years, I had the pleasure of supervising Liza, Wojtek, Christoph and Evgeni, and I want to thank them for working with me on this project.

Another main motivation to cycle up the hill to HITS was always the great catering. Besides the kitchen staff, I also want to thank the administration, IT, gardening, and cleaning staff who made HITS this wonderful place.

Last but not least, I would like to thank my friends and family for their support, especially Katharina's during the times I was stressed out over writing this thesis.

Contents

List of Figures

List of Tables

List of Abbreviations

ACE	atomic cluster expansion
ACSF	atom centered symmetry function
BDE	bond dissociation energy
BDFE	bond dissociation free energy
BKL	Bortz-Kalos-Lebowitz
BMK	Boese-Martin for kinetics
BPSF	Behler-Parrinello symmetry function
CBS	complete basis set
CCSD	coupled cluster single double
CNN	convolutional neural network
DFT	density functional theory
DOPA	L-3,4-dihydroxyphenylalanine
EPR	electron paramagnetic resonance
FDA	force distribution analysis
GLY	glycine
GNN	graph neural network
Grappa	graph attentional protein parametrization
HAT	hydrogen atom transfer
HF	Hartree-Fock
HLKNL	hydroxylysino-keto-norleucine
KGCNN	keras graph convolutional neural networks
KIMMDY	kinetic Monte Carlo Molecular Dynamics
kMC	kinetic Monte Carlo
l-MBTR	local many body tensor representation
lr	learning rate
MAE	mean absolute error
MALE	mean absolute logarithmic error
MBTR	many body tensor representation
MC	Monte Carlo

MD	Molecular Dynamics
MLFF	machine learned force field
mlp	multilayer perceptron
MSE	mean squared error
MSLE	mean squared logarithmic error
NMR	nuclear magnetic resonance
PaiNN	polarizable atom interaction neural network
PES	potential energy surface
PYD	pyridinoline
ReLU	rectified linear unit
rfKMC	rejection-free kinetic Monte Carlo
rKMC	rejection kinetic Monte Carlo
ROS	reactive oxygen species
SOAP	smooth overlap of atomic positions
UV/Vis	ultraviolet and visible spectroscopy
ZPVE	zero-point vibrational energy

Chapter 1

Introduction

When our ancestors discovered many thousand years ago, that they could hit together two flint stones to start a fire, they were unknowingly using a mechanochemical reaction to their advantage. By hitting the quartz crystals, they shattered them into small pieces, in turn breaking every covalent bond between the formerly connected surfaces. As a result, they created an extremely reactive surface plasma, rapidly reacting with the atmosphere around, resulting in sparks and smelly nitrogen oxides.[1]

Since then, many other use cases for mechanochemical reactions have been discovered. They can be used to synthesize ceramic particles, like the as a pigment used zinc ferrite $ZnFe_2O_4$, on an industrial scale by milling the reactants.[2] Similarly, silicon nitride Si_3N_4 can be obtained by milling Si in N_2 or NH_3 atmosphere.[3]

Since Staudingers initial works on polymers, it is known that mechanical force can change the chemical structure of a material.[4] In 1940, Kauzmann and Eyring proposed that the shortening of polymers upon mechanical force is caused by homolytic bond rupture, creating radicals in the process. Today, this behavior is well known, and even used to develop self-healing material, or to substitute hazardous radical-generating substances. [5, 6, 7, 8]

However, mechanoradicals are often not beneficial and are known to deteriorate materials. Radicals, unpaired electrons, are in general very short-lived and highly reactive. In biological systems, they are responsible for the generation of reactive oxygen species (ROS), which cause harm classified under the term "oxidative stress". It was shown that applying physical stress on cells leads to an increase in generated ROS.[9] Similarly, we observed radical formation, and subsequently ROS generation, caused by physical stress in collagen.[10] Oxidative stress caused by free radicals is also related to aging processes, although the exact mechanisms are still topic of ongoing research. [11, 12, 13, 14] While ROS can cause significant damage, biology

has also found use for them, as they are involved in signaling cascades.[15]

1.1 Collagen

Collagen is a protein of high relevance, as it is one of the most abundant proteins in humans, and mammals in general.[17, 18] It forms our skin, blood vessels, tendons, and is even integral to bones and teeth. Though, it is not exclusive to mammals either, as it is essential for, *e.g.*, mussels, where it helps them to anchor themselves to the seabed.[19]

Collagen is a protein consisting of chains of amino acids arranged in a regular pattern, GLY-X-Y, where X is often Proline, and Y often is Hydroxyproline. While single chains form a left-handed helix, pairs of three chains assemble into right-handed triple helices of approximately 300 nm length. The remarkable mechanical strength of collagen is achieved by cross-linking triple helices.[20, 21] Several types of cross-links exist, either enzymatically derived, generated by lysyl oxidase, or non-enzymatically.[18] In Figure 1.1, a enzymatic cross-link called Pyridinoline, or PYD for short, is shown connecting two triple helices together.

Figure 1.2 shows the collagen model used in this study. It is obtained from ColBuilder, and spans one overlap and about one gap region.[16] The overlap and gap regions are a result of the offset between triple helices. Within the overlap region, no triple helices are ending, while in the gap region, the chains end in a staggered way. In electron microscopy images, like in Figure 1.3, this leads to the characteristic D-pattern, repeating roughly every 67 nm.

Anchoring mussels against the tides, or in the tendons of a jumping kangaroo,

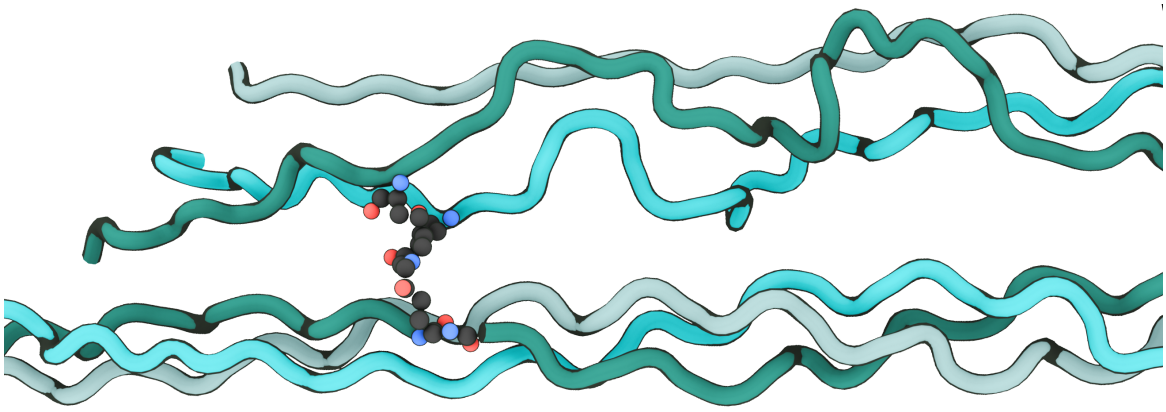


Figure 1.1. Trivalent PYD cross-link connecting two triple helices of collagen in a structure obtained from ColBuilder.[16]

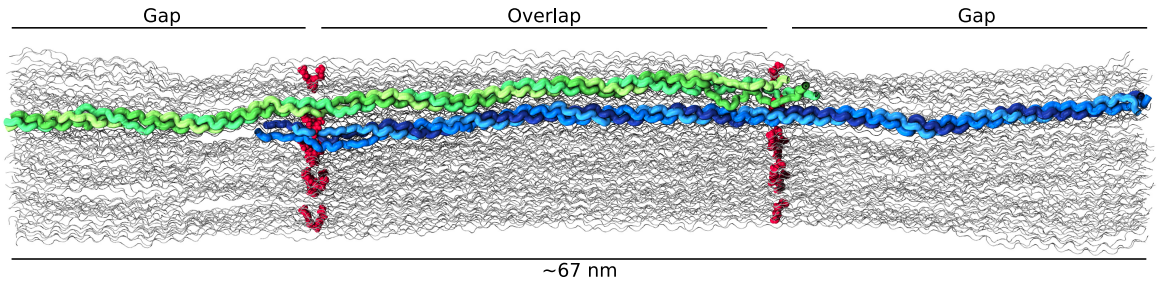


Figure 1.2. Collagen model as obtained from Colbuilder. It spans one gap and one overlap region. Cross-links are shown in red. Two cross-linked triple helices are highlighted. The model contains 41 such triple helices.

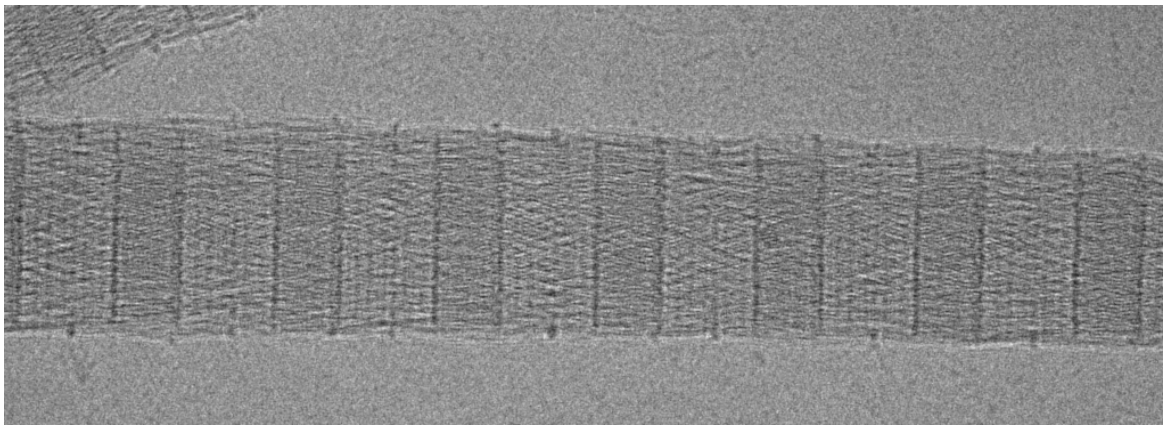


Figure 1.3. Transmission electron microscope image of collagen from a rat Achilles tendon. The repeating pattern of darker overlap, and lighter gap region is well visible. Image kindly provided by Aysecan Ünal.

collagen has to withstand enormous forces. Whenever the forces get too strong, collagen is ruptured, leading to serious injuries. However, even at substantially lower forces, without macroscopic failure of tissue, chemical bonds are cleaved homolytically, yielding mechanoradicals. As initially mentioned, mechanical stress generating radicals is a well known phenomenon from synthetic polymers.[5] In collagen on the other hand, our group was only recently able to measure the generation of radicals through electron paramagnetic resonance (EPR) spectroscopy.[10] The EPR measurements showed an increase of unpaired spins in stressed compared to unstressed rat tail tendon, from which we concluded that bonds must have been cleaved homolytically, creating two radicals in the process. The measured radicals were identified as dihydroxyphenylalanine-anion (DOPA) radicals.

1.2 Mechanochemistry of Collagen in Simulations

The response of collagen to force can be analyzed using Molecular Dynamics (MD) simulations. A molecular model of collagen can be pulled with at a constant force or a constant speed. Then, force distribution analysis (FDA) can be used to analyze in which regions force accumulates, and which parts are stressed less.[22] For collagen, this showed a concentration of force around the cross-links, indicating potential rupture sites in their proximity.

To further analyze where exactly bond rupture occurs, Rennekamp et al. developed KIMMDY (kinetic Monte Carlo Molecular Dynamics), which is discussed in more detail in chapter 5.[23] In short, KIMMDY predicts homolytic bond rupture sites from MD pulling simulations by calculating effective bond dissociation energies from the bond elongation and stiffness. It then uses a Monte Carlo algorithm to choose a bond likely to break, and alters the topology of the system accordingly. KIMMDY simulations of collagen paint a similar picture to the FDA results: Out of 75 rupture events, 60 occurred near cross-links.

However, MD simulations cannot shed light onto the processes following a bond rupture. From experiments, it is known that pulling collagen results in an increase of DOPA radicals. The most likely mechanism by which a radical can traverse through collagen is hydrogen atom transfer (HAT). This reaction is a concurrent movement of a proton and an electron to the position of the radical. The proton leaves one electron behind, resulting in a new radical on its former position.

FDA and KIMMDY suggest, that protein backbone bonds near cross-links break first under stress. Along which ways the radicals move from these breaks to DOPA is unknown, and MD can not answer this question, due to its inability to form new

bonds. To simulate chemical reactions, other methods have been developed, though they lack either speed, accuracy, or resolution to adequately simulate a series of HAT reactions.

1.3 Aim of this Thesis

Simulating chemical reactions on a molecular level is a challenging task for which there is no generally applicable method available yet. KIMMDY enables the simulation of reactions in MD, even ones too slow to occur on the typical MD time scale. However, it requires accurate reaction rates for every possible reaction.

The aim of this thesis is to expand the scope of KIMMDY past homolysis to HAT reactions. To achieve this, we created a machine learning model, capable of predicting HAT reaction rates for single conformations sampled from MD. Furthermore, we rewrote KIMMDY from scratch, establishing the foundation to support arbitrary reactions. The developed machine learning model was integrated into KIMMDY, making it possible to simulate HAT reactions during an MD simulation.

1.4 Outline of this Thesis

This thesis is structured into three main parts. After providing the foundational background knowledge of this work in chapter 2, the first main part is detailing the data generation in chapter 3. This data is then used to train machine learning models in chapter 4. And lastly, KIMMDY 2.0 is introduced in chapter 5, which uses the previously trained models. The results and developed methods are discussed in the final chapter 6.

Chapter 2

Theory and Background

2.1 Molecular Dynamics Simulations

Observing proteins experimentally is a challenge. One can obtain snapshots using methods like X-ray crystallography or NMR spectroscopy, but these are like mere frames of a movie: wanting to conclude the whole plot only from these is ambitious. The task becomes tractable when one builds models of what one has observed in these frames, and thinks of how these models might behave going forward in time. This is where Molecular Modeling and Molecular Dynamics (MD) comes into play. After building models based on high-resolution, but time-independent measurements, MD simulations can propagate the system in time. Many important biological processes can be observed in such simulations, from conformational changes, and protein folding, over ligand binding, to force responses.

The first MD simulation was already performed in the 1950s of simple gas atoms.[24] In the 1970s, the first proteins were simulated.[25] Since the 1990s, popularity of MD simulation is rising constantly, and it has become a frequently used technique.[26] Today, MD often appears in experimental studies as well, either to guide the experiment or to interpret its results. The underlying software tools are mature, and with recent advances in computing power, MD has become a tool routinely used.[27]

At the core of MD is the force field. It describes, what forces particles of certain types exert on each other. All (relevant) forces get summed up per particle $i = 1...N$, to yield the effective force. By solving Newton's equation of motion in small timesteps, one obtains position updates, and the simulation advances.

$$-\frac{\partial V}{\partial r_i} = F_i = m_i \frac{\partial^2 r_i}{\partial t^2} \tag{2.1}$$

To solve Equation 2.1, several different integration algorithms are available. One

often-used algorithm is the "leap-frog" algorithm.[28]

$$v\left(t+\frac{1}{2}\Delta t\right) = v\left(t-\frac{1}{2}\Delta t\right) + \frac{\Delta t}{m}F(t) \tag{2.2}$$

$$r\left(t+\Delta t\right) = r(t) + \Delta t v\left(t+\frac{1}{2}\Delta t\right) \tag{2.3}$$

v are the velocities, r the positions, and Δt the time step. Equation 2.2 describes the update to the velocities. This update happens in between time steps, while the position update in Equation 2.3 happens every time step. From this scheme, the algorithm has its name, as velocities and positions are updated alternately.

There are multiple, essentially equivalent variations of this algorithm, namely the Verlet- and Velocity-Verlet algorithms. Starting from corresponding starting conditions, they produce identical trajectories compared to the leap-frog algorithm.[29, 30]

All these mentioned algorithms conserve energy, total momentum, and angular momentum (without periodic boundaries), and are time reversible, at least for infinitely small time steps. The time step Δt influences speed and accuracy of the simulation, and needs therefore to be chosen carefully. To ensure an efficient simulation it should be chosen as big as possible, while being significantly smaller than the fastest motion in the system. Typically, these are bond vibrations, in particular, stretching vibrations in bonds to hydrogen atoms. As these vibrations would not influence the trajectory in a meaningful way, often hydrogen bonds get constrained in their length, allowing for bigger time steps.

Still, time steps of finite size lead to integration errors, violating energy- and momentum conservation. As a result, the energy of the system can drift over time. To keep the system from building up energy, and ensure one samples the target region of the potential energy surface, one can introduce thermostats and barostats to an MD simulation. These violate energy conservation of the simulation themselves, but compensate errors arising from the integration, the limited system size, or structural issues. Using them, one no longer simulates a microcanonical ensemble (NVE, particle number, volume, energy are constant), but a canonical ensemble (NVT, temperature instead of energy is constant), or a isothermal-isobaric ensemble (NPT, pressure instead of volume is constant).

The choice of force field is very important for any MD simulation, and must fit the simulated system. For simulating proteins, the CHARMM and Amber force fields are established choices.[31, 32] The simulations of collagen mentioned in this work were performed with Amber99sb*-ildnp, a protein force field refined for decades.[33]

A force field consists of parameters of functions describing the interactions of

particles. The following part will lay out the calculation of the potential energy in the MD software GROMACS.[34, 35, 36]

The potential V consists of the following terms:

$$\begin{aligned} V &= V_{bonded} + V_{non-bonded} \\ &= (V_{Bond} + V_{Angle} + V_{Dihedral}) + (V_{Lennard-Jones} + V_{Coulomb}) \end{aligned} \tag{2.4}$$

Bonded interactions are split into 2-, 3-, and 4-body interactions, which correspond to bonds, angles, and dihedral angles, respectively. This separation is arbitrary from a quantum mechanics view, but allows an intuitive understanding of the single terms.

$$V_{bonded} = \frac{1}{2}\sum_{ij} k^r_{ij}(r_{ij} - r^0_{ij})^2 \tag{2.5}$$

$$+ \frac{1}{2}\sum_{ijk} k^\theta_{ijk}(\theta_{ijk} - \theta^0_{ijk})^2 \tag{2.6}$$

$$+ \frac{1}{2}\sum_{ijkl} k^\phi_{ijkl}(1 + cos(n\phi_{ijkl} - \phi^0_{ijkl})) \tag{2.7}$$

k denotes the spring constant of the respective term. r is the distance between atoms, θ the angle, and ξ the dihedral angle. Equation 2.5 describes the bond stretching vibration with a harmonic potential. Similarly, Equation 2.6 uses a harmonic potential for the angles. The dihedral angles are described by the periodic potential in the form of Equation 2.7. For each of these interactions, there are different types of potentials available. In particular for bonds, one could use the morse potential instead, to simulate more realistic bond stretching.

The non-bonded interactions in an MD simulation have three contributions, one based on electrostatic interaction of permanent charges (Coulomb interactions), one based on Van-der-Waals interactions, and one repulsion term based on core-core interactions.

$$V_{non-bonded} = \sum_{ij} 4\epsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] \tag{2.8}$$

$$+ \sum_{ij} \frac{1}{4\pi\epsilon_0\epsilon_r}\frac{q_i q_j}{r_{ij}} \tag{2.9}$$

The repulsion- and Van-der-Waals terms are usually combined into the Lennard-Jones potential, shown in Equation 2.8. ϵ_{ij} determines the depth of the potential, while σ influences its width. Equation 2.9 is the coulomb potential responsible

for electrostatic interaction. Here, ϵ_0 is the vacuum permittivity, ϵ_r the relative permittivity, and q_i and q_j the charges of the respective particles.

As MD is a popular, and well-seasoned technique, there have been many extensions, additional algorithms, and modifications developed. Some, like particle-mesh-Ewald,[37] and neighboring lists allow for the high speed of modern MD, other aim to include further physical phenomena, like polarizable particles[38]. To discuss them all in detail would go beyond the scope of this work. The interested reader is pointed towards "Introduction to Molecular Dynamics Simulation" by Michael P. Allen[30] and the GROMACS reference manual[39].

2.2 Density Functional Theory

MD introduces rather large approximations to the underlying physics, as it categorized interactions between atoms as bonded or non-bonded, making a fixed topology of the system a necessity. Density functional theory (DFT) on the other hand approximates the Schrödinger equation, and therefore models not bonds but quantum mechanical states of electrons around the nuclei. Therefore, bonds are not defined; DFT requires only coordinates of the nuclei and the number of electrons in the system as inputs. Forces acting on the nuclei exerted by electrons are calculated explicitly. During the self-consistent calculation of the electronic state, the positions of the nuclei are constant, justified by the Born-Oppenheimer approximation.

DFT can be used to calculate material and molecular properties rooted in the electronic structure. Macroscopical observables, like UV/Vis spectra can be obtained, just as bond strengths, and reaction enthalpies. Depending on the electronic state, the electrons exert different forces on the nuclei. Optimizing these, one can obtain geometries of the nuclei corresponding to the lowest energy of a particular electronic state, or even the geometry of the transition state of a reaction.

DFT estimates the many-electron time-independent Schrödinger equation:

$$\hat{H}\Psi = \left[\hat{T} + \hat{V} + \hat{U}\right]\Psi = E\Psi \tag{2.10}$$

$\hat{H}$ is the Hamiltonian operator, Ψ the wave function, $\hat{T}$ the kinetic energy operator, $\hat{V}$ the potential energy operator, $\hat{U}$ the electron-electron interaction operator, and E the total energy of the system.

DFT is build-upon the two theorems by Hohenberg and Kohn published in 1964.[40, 41] 1. The electronic wave function is uniquely determined by the ground-state electron density, and hence all ground-state properties of the electronic systems. 2. The energy of an electron distribution is a functional of the electronic

density. Further, the ground state electron density minimizes this functional.

In 1965, Kohn and Sham made the theory usable in practice, by replacing the Hamiltonian by one for a system of non-interacting electrons.[40, 42] This can be expressed as a sum of one-electron operators, which are known. The key property of the fictitious system with non-interacting electrons is, that the electron ground state density is the same as in a real system of interest. The energy can be divided into following terms:[43]

$$E[\rho(r)] = T_{e,non-inter}[\rho(r)] + V_{ne}[\rho(r)] + V_{ee}[\rho(r)] \tag{2.11}$$

$$+\Delta T[\rho(r)] + \Delta V_{ee}[\rho(r)] \tag{2.12}$$

Equation 2.11 consists of the kinetic energy of the non-interacting electrons, the potential energy between the nuclei and the electrons, and the classical electron-electron repulsion. Equation 2.12 is called the exchange-correlation term, and contains corrections to the exact calculated kinetic energy of the fictitious system and to all non-classical electron-electron interactions. While the former term can be solved exactly, the latter terms need to be solved approximately. How these approximations are approached is the major distinction between functionals, and many options have been developed. They determine the accuracy, but also computational cost. In this work, a so-called hybrid functional is used, where the exchange-correlation approximation contains portions of exact exchange obtained from Hartree-Fock calculations.

2.3 Machine Learning for Molecules

For bigger molecular systems beyond a few dozens of atoms, DFT becomes unfeasible, and one needs to reside to less physical-based methods. These are defined by a functional form, and a set of parameters. In the case of MD, the parameters, and to some extent the functional form, is determined by the force field. As mentioned already in section 2.1, the separation of the potential in bond, angle, and dihedral angle terms is somewhat arbitrary. Furthermore, their functional forms are relatively simple, allowing for few parameters to be changed.

But there is no need to stick to the established functional forms: neural networks are universal function approximators, and therefore might be able to model the underlying physics more accurately. In fact, over the past two decades, Machine-learned force fields (MLFFs) have exploded in power, and popularity. Today, they are capable of powering simulations of more than 100 million atoms, while achieving

accuracies closer to DFT than to MD.[44] At the time of writing, they still can not compete speed-wise with highly optimized MD code, though in many aspects the fairer comparison is to DFT. MLFFs do not enforce a fixed topology, making most implementations in principle reactive. Though the quality of the transition state energies, like of the rest of the force field, depends on the training data used, therefore, one must ensure to apply the MLFF only to systems similar to the trained data.

The speed-up of MLFFs compared to DFT is massive: While DFT calculations of medium-sized systems take in the order of hours, MLFFs can make their predictions within milliseconds. A challenge under active development is to close the gap to MD, requiring the MLFF implementations to move close to the hardware.

In the following, a brief history of MLFFs for molecules is presented, using the terminology coined by Jörg Behler.[45] Already in 1995, a neural network was used to model a potential energy surface of CO adsorbing on a Ni(111) surface. The neural network was still highly specialized, requiring a set amount of atoms in a system. Further, the inputs into the model were cartesian coordinates, characterizing the first generation of MLFFs.

The second generation began with the development of local descriptors of the chemical environment around atoms: the atom centered symmetry functions (ACSF), sometimes called Behler-Parrinello symmetry functions (BPSF).[46] They were a big step forward, allowing MLFFs to be applied to much bigger systems, and importantly, to systems of variable sizes. The models also became a lot more powerful, as basic physical laws are included in the descriptor architecture, and therefore do not need to be learned anymore. The descriptors of second generation MLFFs are invariant to translation, rotation, and permutation of the system. This is extremely important, as none of these operations does influence the energy of the system (as long as there is no external field). Popular MLFFs of this generation include the ANI family of models[47, 48, 49], and SchNet[50].

The locality of the descriptors was a big step forward, though, this came with the drawback of being blind for long-ranged interactions. In the third generation of MLFFs, an effort was made to address this issue by tasking the neural network with the prediction of partial charges, or even of the electrostatic interactions directly. Examples of third generation models are TensorMol and PhysNet.[51, 52, 53]

Now, long range electrostatics can be calculated, but still, the underlying partial charges necessary for the calculations are determined locally. The forth-generation model of Behler et al. introduces a non-local charge-equilibration scheme based on CENT.[54, 55] Charges can relocate in a molecule due to changes to the system

outside the typically used cut-off in MLFFs. Using an equilibration scheme, correct partial charges can still be obtained in such a situation. Another model of this generation is AimNet-NSE.[56]

There are at least two other areas of improvement not mentioned in the classification into four generations. First, the descriptors: the jump to the second generation was, as mentioned, a result of moving to invariant descriptors. Besides the mentioned ACSFs, many other invariant descriptors have been developed, *e.g.*, SOAP or MBTR.[57, 58, 59, 60] It has been shown that many descriptors of that era can be unified in the atomic cluster expansion (ACE) framework.[61, 62]

A second major jump for MLFFs was to move from invariant descriptors, to equivariant ones. Equivariant means, that a supported transformation on the input of the descriptor yields the same output as if it is applied to the output directly. In other words, if the input structure is rotated, the output of the descriptor must be a vector rotated in the same way. If invariant descriptors want to include higher order interactions, between three or more atoms, they need to iterate over every possible angle, leading to poor scaling. Equivariant descriptors can iterate only over pairs of atoms, and still convey higher-order information. The first MLFFs using equivariant descriptors were Tensor field networks, Cormorant, and steerable 3D-CNN. [63, 64, 65] Other widely used networks include Nequip, and PaiNN.[66, 67]

The ACE framework unified the invariant descriptors, and can quite easily be expanded to include equivariant descriptors as well (see Multi-ACE [68]). One of the best performing MLFFs today is an implementation of this framework called MACE.[69]

The way, the network "sees" the system through the descriptor is not the only factor contributing to the rapid development of MLFFs. Over the past decade, the underlying neural network architecture has also changed significantly. While the first generation of MLFFs used one single feed-forward neural network, later, different sub-networks were used for different atom types. A significant advancement was the introduction of message passing graph neural networks (GNNs) to the field. Instead of computing the representation of the environment around each atom once at the beginning, GNNs work iteratively. A GNN defines an initial feature state on each atom, and an update function. Each iteration, each atom builds a representation of its environment, *i.e.* it receives messages of its neighboring atoms. Then, the update function updates the atom-wise feature state based on these messages. In the next iteration, the new messages depend on the new feature state of the atoms nearby, and in turn, on the messages those atoms received last iteration. By repeating this scheme, each iteration all atoms experience influence from further-away atoms. The

first time a GNN model was used on molecular data was in 2015 to learn molecular fingerprints.[70] Today, GNNs are widely used in MLFFs, *e.g.*, in SchNet, PaiNN, PhysNet, Nequip, or Mace.[50, 67, 53, 66, 69]

2.4 Kinetic Monte Carlo Simulations

2.4.1 Monte Carlo Methods

Monte Carlo (MC) methods are a highly influential family of algorithms, used in many fields, from nuclear science, over biology, to social sciences.[71] The term describes methods, where one input must be a random value. This value can then be used to, *e.g.*, sample from a unknown distribution, like the potential energy surface (PES) of an molecular adsorption process onto a metal surface. In this case, the random variable would be used to generate different geometries, for which in turn an energy can be computed. MC methods can also be used to estimate certain quantities. For instance, π can be approximated by repeatedly selecting random points in a square that inscribes a circle and comparing the ratios of points inside the circle and square. The accuracy of the approximation improves as more random points are sampled. Modern ray-tracing algorithms used in computer graphics use MC, too. They work by sending out many rays from the camera, which get scattered multiple times on surfaces in a random process. The correct pixel values are determined by averaging many rays together. Both of the last examples determine one non-probabilistic quantity by aggregating multiple values obtained from a random process.

In the past, MC methods were also coupled with MD.[72] In the work of Sadigh et al., a semi-grandcanonical ensemble is simulated, where MC moves are used to transmutate the chemical identity of simulated particles. Structural relaxation processes are often more efficiently simulated by MD rather than MC. In this case, MC and MD steps are performed alternately to relax the simulated system after each MC event in an MD simulation.

2.4.2 Kinetic Monte Carlo

The kinetic energy of particles in MD is usually following the Maxwell-Boltzmann distribution, as shown in Figure 2.1. To overcome a reaction barrier, the reaction partners must contribute enough kinetic energy, sometimes more than most particles have, making it a rare event. In MD simulations, this can be an issue, as the system size is finite, and therefore sometimes not enough particles are simulated for long

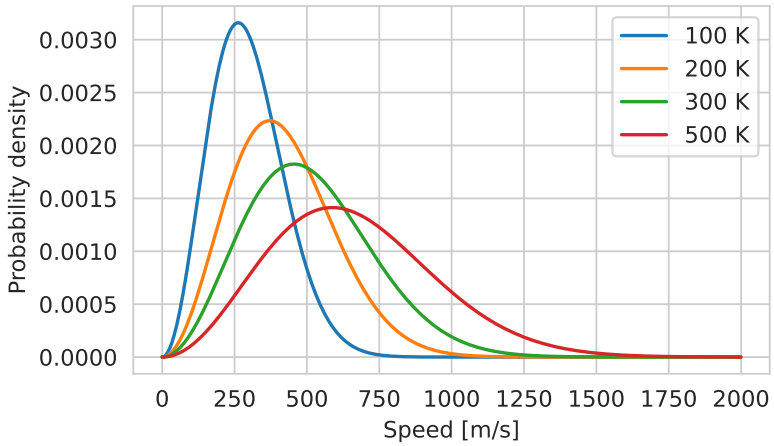


Figure 2.1. Maxwell-Boltzmann distribution of atomic velocities in ideal argon gas at different temperatures.

enough to find those very fast ones. The result is that MD sampling becomes inefficient, spending most computational time on too slow particles.

Kinetic Monte Carlo (kMC) allows the simulation of such reactions from another perspective. It is a type of Markov process, hence it models discrete states, with transition rates between them, which only depend on the current state. Also, it introduces a concept of time to MC. Time does not progress constantly in kMC, but in time jumps of varying sizes. Thus, kMC is capable of modeling slow processes inaccessible to MD. kMC was first used in 1966 to study vacancy migration in different alloys.[73]

There exist two major classes of kMC algorithms: with rejection and rejection free.[74] The rejection-free kMC (rfKMC) method, sometimes called Bortz-Kalos-Lebowitz (BKL) method, selects a random state to switch to, weighted by the transition rates between the current, and next state.[75] Then, it calculates an update to the time based on another random variable, weighted by the sum of all outgoing rates of the current state. The determined next state is then made the current one, and the next iteration of the algorithm begins. Moving to another state connected with a low transition rate is unlikely, as long as there are other, more likely transitions possible. However, if the current state is connected only *via* low transition rates, such an unlikely transition will still occur. In that case, the update to the time will be rather big, indicating a long residence time necessary in the initial state to overcome the high barriers to reach other states. This process makes rfKMC simulations very efficient, as no time is wasted simulating the same state for a long time in the hope, a rare event will occur.

The second class of kMC algorithms is rejection kMC (rKMC), also called the

Metropolis algorithm.[76] Here, the random selection of a state to move to is *not* weighted by the transition rates. This selected move is then either rejected or accepted, based on its transition rate and a random variable. If it is rejected, another random move is attempted. Otherwise, the transition is performed. The time update depends only on the number of possible transitions, a random variable, and an upper bound of the transition rate anywhere in the system. Crucially, this algorithm does not require the calculation of every possible transition rate at each step, but only the rates of a few selected transitions. This is important, as often the evaluation of the transition rates is computationally costly. rfKMC on the other hand is more efficient if transitions rates are easily obtainable, but often low, resulting in many rejected transitions in rKMC.

Other variations of kMC exist, like first-passage kMC, which is tailored towards diffusion of reactive particles.[77, 78] Only the particle with the shortest distance to a reaction site is propagated, saving time by not propagating all particles in the simulation.

2.4.3 kMC and MD

In 2002, kMC has been first combined with MD for simulating soot precursor generation in flames by Violi et al.[79] Nine different reactive sites were defined in small hydrocarbon molecules. In an alternating scheme, the system was propagated in time by MD, and by kMC steps. The role of kMC was to select and perform chemical reactions, while the MD was used to relax the geometry after a reaction, and to sample conformations of the system.

Later, MD/kMC was used by Karl Peter et al. for protein folding.[80] Similarly to Violi et al., they alternated between MD and kMC. However, they used the MD phase to determine similarly behaving sections of the protein, and not to advance the system in time. For those classes, reaction paths and reaction rates are generated. rfKMC is performed to decide on a reaction, then the cycle continues with MD.

MD/kMC can also be used to accelerate processes usually too slow to be observable in MD. Tavenner et al. use MD/kMC to modify atomic diffusion rates in metal alloys.[81] In contrast to established MD/MC methods, this approach allows for the generation of physically meaningful trajectories. Typically, MD/MC would allow arbitrary atoms of different types to be swapped, while when using kMC, atoms can be restricted to only swap with their neighbors. The resulting trajectory can be used to analyze atomistic mechanisms.

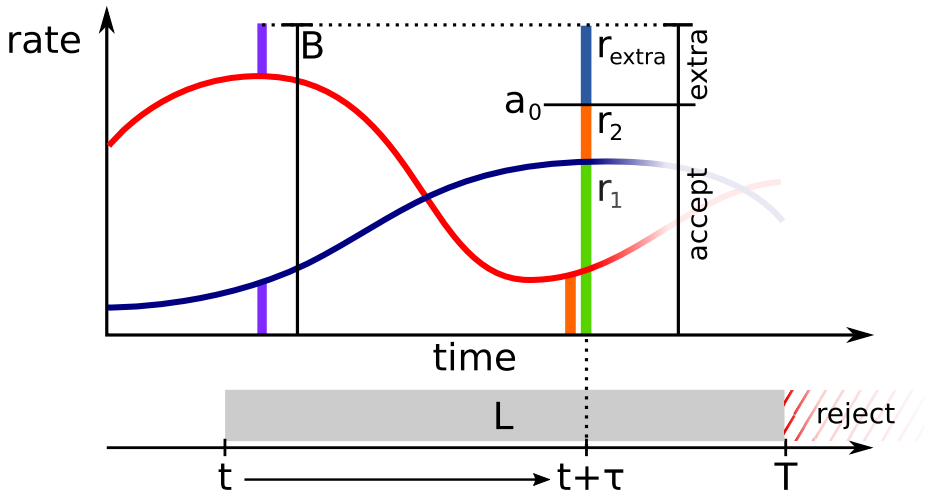


Figure 2.2. Visualization of the Extrande KMC algorithm.[82]

Table 2.1. Symbol descriptions of the Extrande algorithm

Symbol	Description
r_i	rate of reaction i, time dependent
t	current time
T	time of last calculated rate
$L; t < L \leq T$	look-ahead time window
B	upper bound of rate within L
$\tau \sim exp(1/B)$	time step drawn from exponential distribution
$a_0(t) = \sum_i r_i(t)$	sum of rates at time t

Extrande

None of the so far mentioned methods describes properly the relation of the timescales of kMC and MD. How much time passes in MD, and how many kMC attempts are performed are user-defined settings, which can impact the simulation outcome. While not explicitly tailored towards MD/kMC, Extrande formulates a framework to combine the kMC timescale with another timescale.[82] It is one of the algorithms implemented in the KIMMDY software developed in this work. Implementation of, and modifications to Extrande will be discussed in chapter 5.

Extrande is designed to work with time dependent rates. The algorithm starts by computing reaction rates r until a certain time T. Within this look-ahead time window L, the maximum sum of rates, B is calculated. Next, τ, the first random variable, is generated by sampling the exponential distribution $Exp(1/B)$. It is used to advance the current time to $t + \tau$. If $t > T$, t is set to T, and the algorithm is restarted from there, thus rejecting all rates within L. Otherwise, a rfKMC step is

performed at time t to determine the reaction to execute. However, an additional rate $r_{extra} = B - a_0$ is added to the process. If any of the reactions in a_0 is selected, the corresponding reaction is performed. The rest of the calculated rates in L are no longer applicable, therefore the algorithm starts again by computing new rates. If however r_{extra} is selected, no reaction is performed. The algorithm continues from the new t by updating L, B, and τ. Figure 2.2 visualizes the algorithm and the relations of the variables used.

Extrande clearly defines how often, and in what time intervals kMC steps are performed. Also for moderately varying rates, it is an efficient algorithm. If B is low in a given time window, the probability of a reaction happening is low. A large τ is likely, skipping the entire time window. Therefore, little time is wasted trying to perform unlikely kMC events. If however a bigger rate is located within L, τ will be small, to ensure Extrande does not jump over a likely reaction.

Chapter 3

Data generation

Any machine learning model relies on training data to later predict other, unseen data points according to the statistics in the training data. Our goal is to develop a model capable of predicting HAT energy barriers based on snapshots of an MD simulation. Input into such a model must therefore be said snapshot in some form, output the barrier. This chapter is devoted to the creation of input into any model, alongside the matching output to train the model. Much of the content in this chapter is published in *Chem. Sci.*, 2024, **15**, 2518-2527.[83]. Section 3.4 is published in *RSC Adv.*, 2022, **12**, 34557-34564.[84]. Text and figures are adapted or copied from these publications.

Simple, feed forward neural networks need inputs of a fixed size. When working with molecular data, this means a pre-processing step is required to generate so-called descriptors that meet this requirement. In contrast, graph neural networks perform their computations directly on the molecular graph, eliminating the need to generate the descriptors in advance. Instead, these models can process three-dimensional structures as inputs.

Either way, the inputs must properly define the question we want the model to answer: how high is the energy barrier of a particular reaction?

The reaction is defined when it is clear which hydrogen is reacting and where it is going. One could argue that in the organic molecules at hand, the target position is obvious in the presence of a single radical. However, not communicating the target position requires the model to learn in what exact situations an atom is a radical, which is possible, but unnecessarily complicated. To ensure the model can focus on our task of interest, we encode the reacting hydrogen and its target position in the input structure as special, unique elements for all types of models and inputs presented in the following.

3.1 Procedurally Creating Reactive HAT Systems

For the first approach of creating molecular systems in which HAT can occur, we started with the smallest meaningful system: Amino acid pairs. One could craft smaller systems, say HAT within one ethane molecule, but the information gained from such systems would be hardly transferable to proteins. On the other hand, one could build bigger systems, *e.g.*, HAT within real proteins. This would increase the computational cost immensely, as DFT calculations are required to obtain the energy barrier, which we want to predict. Also, as HAT is a local process, it is expected that the benefits of simulating a big protein are outweighed by the more complex sampling process.

To create the reactive systems, we must first define the variables that characterize our system. The first variables are the choice of amino acids, and between which hydrogen positions the reaction will occur. These describe the chemical environment surrounding the reacting hydrogen and the radical site. The rest of the variables describe the geometry of the reactants: the distance the hydrogen has to travel (also called translation distance in this text), the rotation angle of the amino acid of the reacting hydrogen around the hydrogen bond, and finally the tilt angel between the hydrogen bond before and after the reaction. Figure 3.1 A depicts these variables in an example system.

3.1.1 Structure Creation Algorithm

The pool of amino acids we sample from contains all natural amino acids, and the post-translationally modified amino acids dihydroxyphenylalanine (dopa) and hydroxyproline. Additionally, modified versions of these structures are used, in which the free carboxyl group is converted into an amid group to closer mimic the chemical situation in a protein. Water, as well as nine amino acid fragments resulting from breaks in the backbone of a protein, derived from alanine, glutamine, glycine, hydroxyproline, and proline are also added to the pool.

All structures in the pool are considered in every possible charge state. From this pool of structures two, not necessarily the same, structures are selected. Within each of them, a hydrogen atom is randomly picked. For the systems procedurally generated in this work, the translation distances are sampled from a standard exponential distribution with varying offsets from 0.3 Å to 1.5 Å, the tilt angles are sampled from a normal distribution with standard deviation of 45°, and the rotations are sampled in 30° increments. After applying these transformations, the resulting structures are checked for atoms too close to each other. Any system where the min-

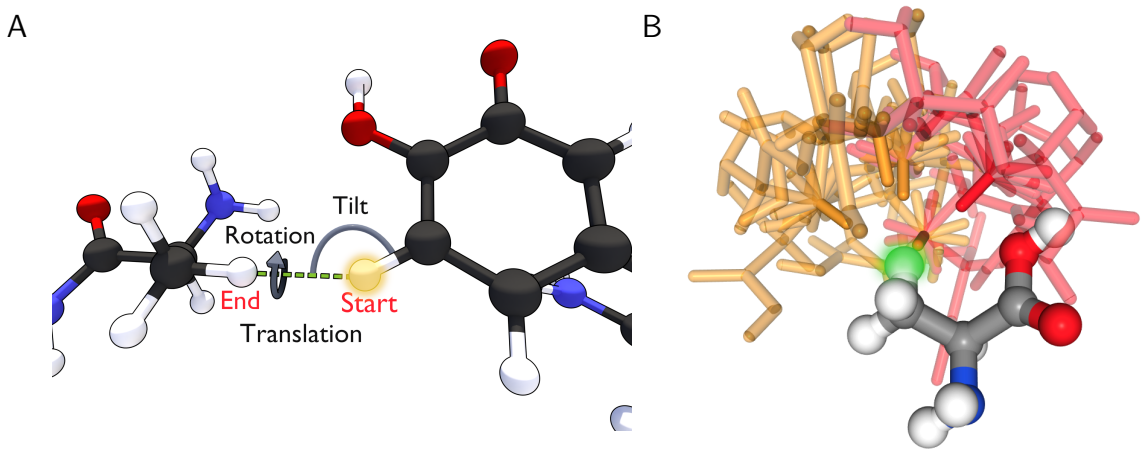


Figure 3.1. **A**: Construction of an exemplary synthetic system. The variables defining the geometry of the reaction are shown: The translation distance is the distance between start and end position of the hydrogen, the rotation angle is the dihedral angle between the reaction partners, and the tilt angle formed by the transferring hydrogen with the donating and accepting heavy atoms. **B**: Multiple generated reaction partners of the green highlighted hydrogen. Orange structures are saved, red structures are too close to the initial molecule and are therefore discarded.

imum distance between the reactants (excluding the reacting hydrogen) is smaller than 2 Å is discarded.

Furthermore, intramolecular reactions are generated from within single amino acids. Any combination of hydrogen atoms with less than 4 Å distance are considered. Systems with atoms closer than 0.8 Å to the transition path are removed (see Figure 3.2). Figure 3.3 A shows the resulting translation distance distribution. The systems are saved in the reactant and product state for further processing.

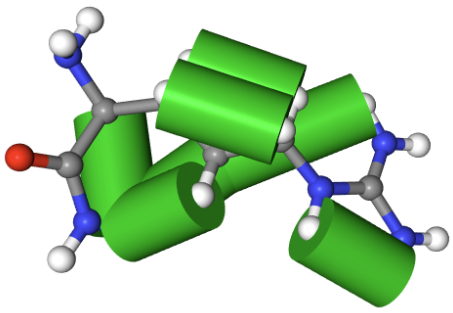


Figure 3.2. Systems with atoms obstructing the transition path are discarded. Highlighted in green are exemplary zones checked for interfering atoms.

In this way, 4393 systems are created, resulting in 8786 energy barriers, as each reaction can be seen in forwards- and backwards direction. After creation, ∼10%, or 441, of the systems are set aside for testing developed models.

3.2 Sampling Reactive HAT Systems from MD Trajectories

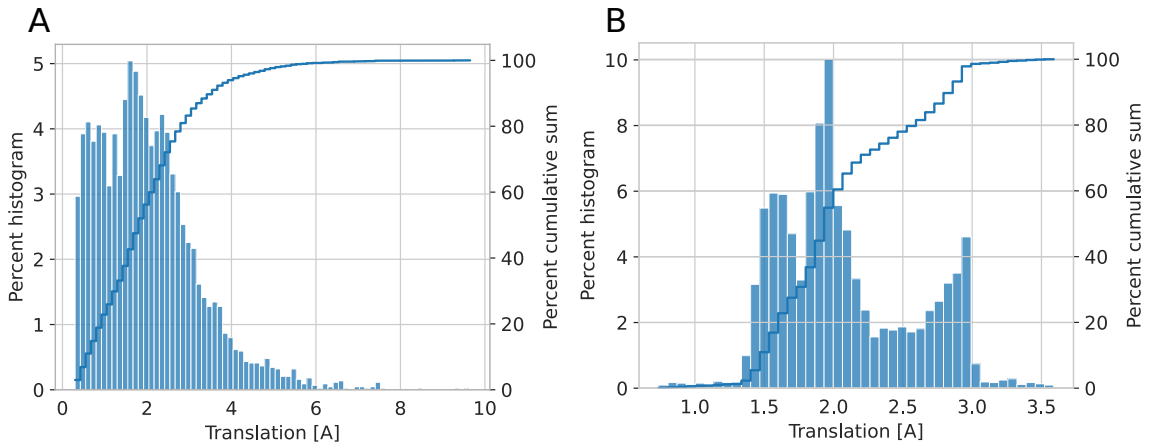


Figure 3.3. Histograms of **A** the synthetic data and **B** the trajectory data over translation distance of the reacting hydrogen.

The goal of this work is to apply the developed model to systems obtained from MD simulations. Therefore, it is natural to sample training data from such simulations. The generation of reactive systems from MD trajectories starts from a collagen model obtained from Colbuilder.[16] It spans one overlap and one gap region. Sequences from *Loxodonta africana*, *Pongo abelii* and *Rattus norvegicus* are used, paired with the divalent HLKNL and the trivalent PYD cross-link at various positions. The simulations are performed with GROMACS 2020.[36, 34] During the simulation tension is applied to each collagen chain. The pulling force per chain is 1 nN,*i.e.*, 3 nN per triple helix. Four different pulling methods are used: I) the triple helices are pulled from both ends, II) and III) they are pulled from one side, while the other is fixed in place, and IV) on different triple helices different forces are exerted, drawn from a Gaussian distribution with $F_{av} = 1nN$ mean force and width $= F_{av}/3$. In this last scheme, the outer ring of triple helices is still pulled at the average force to prevent triple helices from sliding. The simulations were performed by Bennedikt Rennekamp; the simulations and conclusions are published jointly in *Nat Commun* **14**, 2075 (2023).[85]

The simulations can further by divided in two groups, one containing two radicals as a result of a backbone break, the other consisting of intact collagen system without radicals. For the latter, HAT reactive systems are sampled by H-H distance around every hydrogen atom. As energy barriers, E_a, heavily depend on the translation distance, an emphasis is put on shorter translations when sampling. >98% of

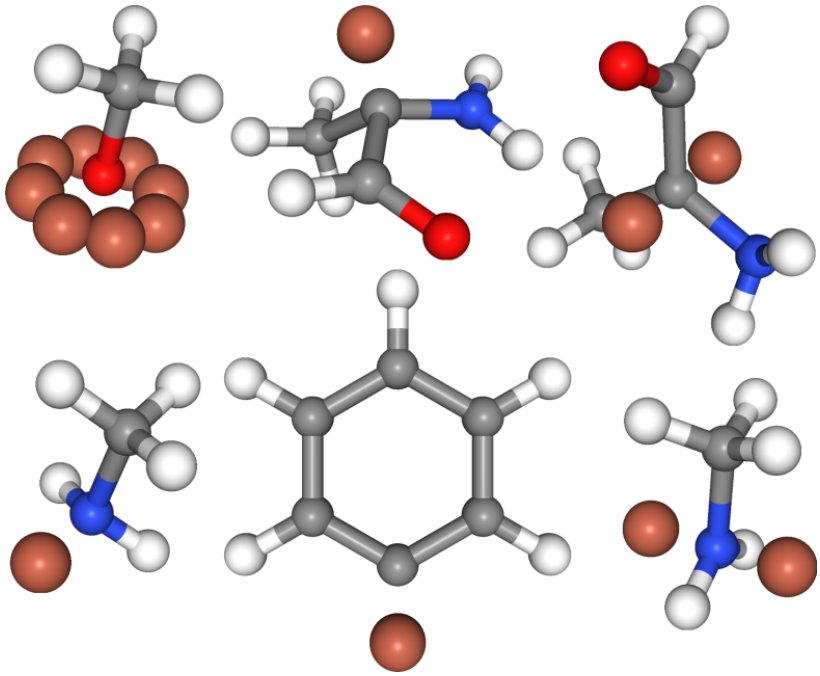


Figure 3.4. By the algorithm suggested hydrogen end positions shown in red around radical atoms.

the samples have translation <3 Å, >50% <2 Å (see Figure 3.3 B). Like for the intramolecular procedurally generated reactions, systems with atoms closer than 0.8 Å to the transition path are removed. Each system is saved in the reactant and product state. The radicals in the other type of simulation are a result of homolytic bond breakages performed using KIMMDY.[23] In these trajectories, only potential HAT reactions involving the existing radicals are considered. Again, systems with atoms intersecting the transition path of the hydrogen where excluded. We again want to save the the system in the reactant and product state like in the fully saturated systems. Though, the exact end position of the reacting hydrogen is not known, and must therefore be guessed from the geometry to generate the product. In case of an ambiguous endpoint, like the hydrogen in an alcohol group, the smallest distance on a 109.5° cone around the oxygen is sampled (See Figure 3.4).

Whether there was a radical present in the system already or not, the region around the reaction is cut-out from the big simulation box to reduce the number of atoms that will be fed into the DFT calculations later. The borders of the cut-out system need to be designed in a way that does not interfere with these DFT calculations, therefore bonds can not just get cleaved. Instead, capping groups need to be added, which mimic the continuation of the protein outside of the DFT region. For the N terminus acetyl groups, and for the C terminus NH-CH3 groups are added. The cut-out region is determined by the atoms in alpha position to the

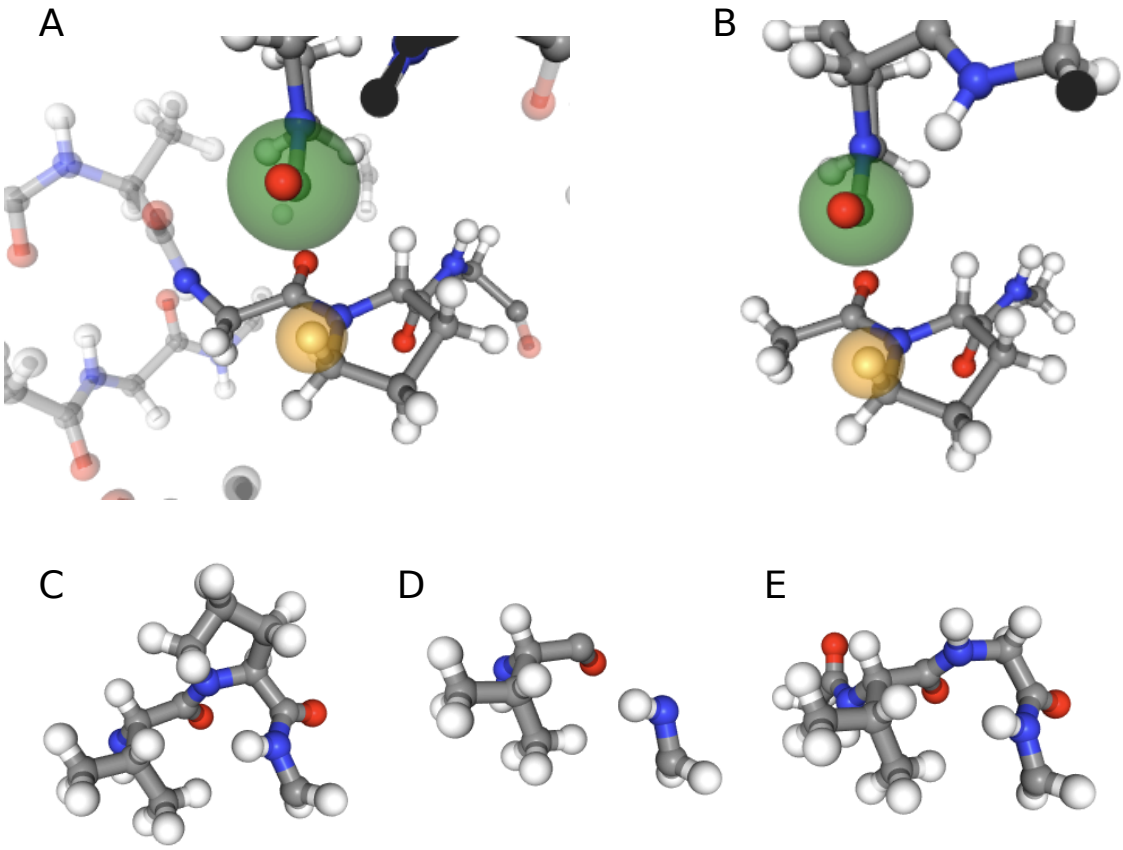


Figure 3.5. Capping of trajectory systems. The radical is highlighted green, the reacting hydrogen orange. **A** A trajectory system with its context around shown translucently. **B** Same system as in A now isolated with the capping groups added. **C** A intramolecular HAT reaction between the radical (bottom right) and the adjacent CH_3 group. **D** Same system as in C. The original bridging amino acid between the reacting groups is removed. **E** Same system as in C and D, with capping groups added. A glycine replaces the removed amino acid, in the background an acetyl group caps the N terminus.

radical atoms in the reactant and product state: all amino acids with atoms in this group are included fully in the cut-out region. The next atoms from adjacent amino acids are then used to construct the capping groups (See Figure 3.5 A and B). If the selected amino acids are part of the same protein chain, but with one other amino acid in between, this amino acid is replaced by glycine to decrease the atom amount (Figure 3.5 C-E). For a given set of selected atoms in a trajectory, only the system with the smallest translation distance is kept, as otherwise a large amount of highly correlated systems would be generated.

Overall, 5261 systems are created from trajectories, *i.e.* 10522 barriers. Again, ∼10%, or 528, of the systems are reserved for testing.

3.3 Obtaining the Transition Path

We now have the geometries of reactive systems in their start and end state. To calculate the energy barrier for these systems, one needs to obtain the energy of the transition state, which is the highest point on the transition path connecting the reactant to the product state. The most accurate way to obtain the transition path requires an DFT optimization towards a saddle point on the potential energy surface. Though, such calculations are expensive, and take for the systems generated in the last two sections from hours up to days, making it unfeasible to generate a big data set of many thousand systems in this way. Instead, the transition path is guessed by interpolating the position of the reacting hydrogen from its coordinates in the reactant state to the ones in the product state. The resulting transition path is only a rough representation of the true transition path, but the goal at this point is to capture the general concept of HAT for a machine learning model to learn. It does not, for example, contain other rearrangements of atoms besides the reacting hydrogen, although a change of the hybridization state upon donating or accepting a hydrogen would be expected. In the next section, more accurate transition states will be obtained, which the final model will use. For now, the rest of this section will detail the creation of transition paths by interpolation.

The interpolation is performed in ten steps, resulting in eleven points including the start and end structure. In the first iterations of the data set, it was found that the transition state is always localized in one of the middle five steps, or right at the beginning or the end. Therefore, in all calculations thereafter, steps 2,3,9, and 10 are omitted to increase efficiency.

To calculate the barrier, the energies of the remaining seven structures need to be calculated using single point energy DFT calculations, which were conducted with Gaussian 09 (rev. D.01).[86] The barrier is then calculated as the difference of the first- to the highest energy. The functional used is BMK with the basis set 6-31+G(2df,p).[87] It was chosen due to its good performance at an acceptable computational cost. Unlike many functionals, BMK is specifically optimized to accurately determine transition- and ground state energies. Its performance was tested in several benchmark studies,[88, 89] even some specifically on proton transfer reactions.[90]

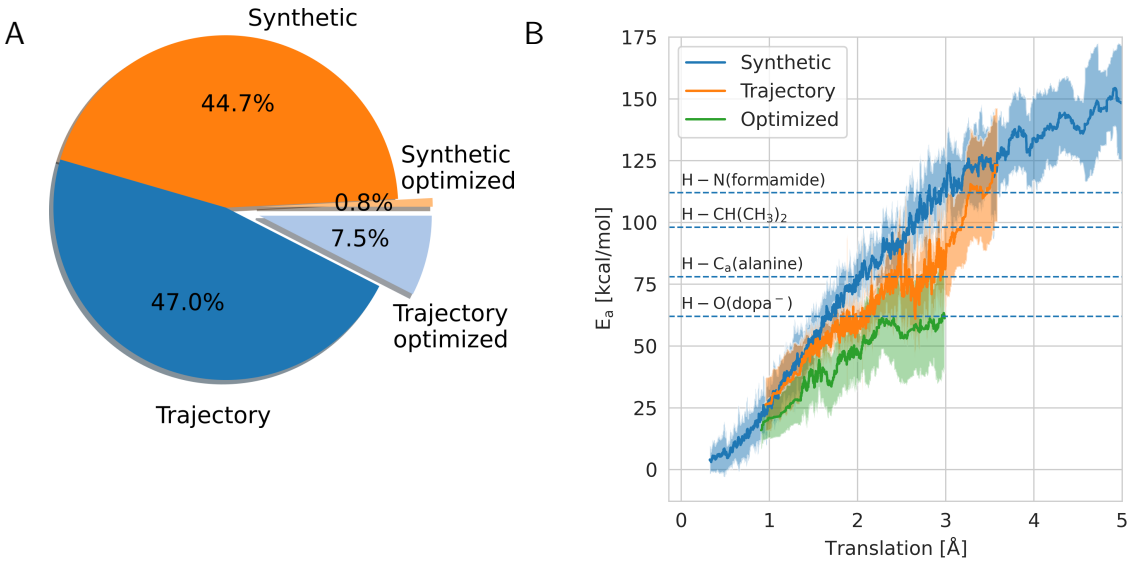


Figure 3.6. **A**: Data distribution of the synthetic and trajectory data sets. **B**: Rolling average of the calculated energy barriers of HAT reactions in the data set vs. the distance the hydrogen has to move during the reaction. The shaded area corresponds to ± standard deviation, dashed lines indicate bond dissociation energies for context.

3.3.1 DFT Optimizations

As outlined in the previous section, the interpolated transition paths are cheap to calculate, but do not reach the desired accuracy to allow good reaction rate calculations. This is mainly due to the inability of the described process to model an open-shell system. The geometries used up to this point all are results of traditional MD force fields, not incorporating proper radical parameters. Therefore, the radical atom exhibits the same close-shell geometry as if it would not be a radical. This artificially increases the energy of reactant- and product state, but especially the transition state, leading often to overestimated reaction barriers (see Figure 3.6 B). Additionally, the transition state energy would be lowered as the reactants typically move towards each other, exchange the hydrogen, and then move apart again. This process is not replicated using the interpolated transition paths.

Starting from structures of the synthetic- and trajectory data set we obtain more accurate energy barriers through the optimization of the ground- and transition state prior to the energy calculation. Three optimizations are required per system: one for the reactant-, the product-, and the transition state structure, respectively, to obtain the barriers in forward and backward direction. The optimizations of the

reactant- and product state start from the same MD structures as the interpolation process. For the transition state, the middle structure produced by interpolation serves as the starting structure. The optimization itself is performed using the same DFT level as for single point calculations, BMK/6-31+G(2df,p).

All optimizations of transition states are followed by a frequency calculation to ensure the existence of a single imaginary frequency, corresponding to the correct reaction. While time-intensive, this step is required, as in some cases the optimization from the interpolated transition state to the saddle point corresponding to HAT leads to a wrong saddle point. This can be seen in the hydrogen moving away, and, *e.g.*, attempting to bind to an oxygen, or nitrogen nearby. In these cases, either an attempt to restart the optimization from a manually modified starting structure was performed, or the system was discarded.

To reflect the embedding of the reactants into the structure of the material, here the protein backbone, the atom positions other than the reactive core are frozen throughout the optimization. Core of the reaction is defined as the reacting hydrogen, the donor and acceptor atoms, as well as every hydrogen atom attached to those. Another reason to freeze most of the structure is that the final model is intended to be applied on MD trajectories. We want to restrict the DFT optimization to degrees of freedom that cannot be sampled in MD, namely the geometry changes due to the closed-shell to open-shell electronic structure transition. Degrees of freedom such as side chain rearrangement should *not* be sampled during DFT optimization, but rather in the preceding MD simulation.

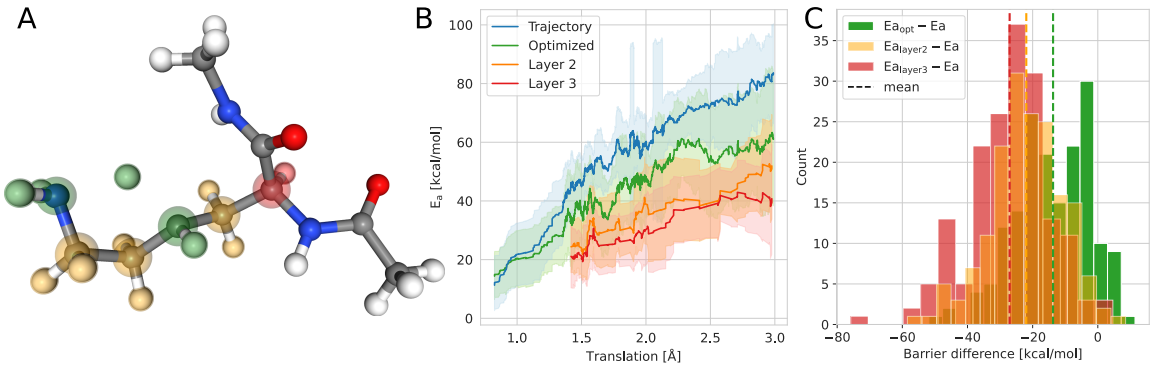


Figure 3.7. Analysis of differently sized optimization regions. **A** Illustration of different possible optimization regions. The ultimately used region is highlighted green, the next bigger region layer 2 yellow, and layer 3 red. **B** Comparison of the non-optimized trajectory systems, the same systems optimized as in the main text, and these systems optimized at layer 2 and layer 3. **C** Histogram of the barriers optimized at all different layers in relation to the non-optimized ones.

To achieve this, the impact of differently sized optimization regions was analyzed, shown in Figure 3.7. In the example system shown in subfigure **A**, the finally used optimization region is highlighted in green. Unfreezing layer 2 (yellow) already allows unrestricted movement of the complete side chain. This leads to the situation, that in some optimization steps (of start, transition state, end) a hydrogen bond between the NH_3, and an oxygen in the backbone is formed, and in others not, resulting in a disturbed HAT barrier. Furthermore, this rearrangement could be sampled in MD, therefore we want the DFT optimization to stick to other degrees of freedom not covered by MD. Unsurprisingly, bigger optimized regions do lead to on average lower barriers, as can be seen in panel B and C of Figure 3.7.

As the optimization procedure is computationally expensive, only a subset of the non-optimized synthetic and trajectory systems could be optimized. The final data set consists of 803 optimized systems (1606 barriers), 725 of them stemming from trajectory systems, 78 from synthetic systems, which corresponds to 7.5%, and 0.8%, respectively (see Figure 3.6 A). The membership in the test- or training set is inherited from the non-optimized systems to prevent training data leaking into the test set. Of the 803 optimized systems, 12% are set aside for testing.

3.3.2 Results

At this point, we generated a data set of structures where HAT reactions can occur, along with the associated energy barriers. The data set spans the relevant conformational space and provides valuable insight into the behavior of HAT reactions in collagen, on top of enabling the creation of predictive models.

Unsurprisingly, the calculated barriers show a strong dependence on the distance the hydrogen atom has to travel during the reaction, as can be seen in Figure 3.6 B. However, barriers vary significantly for a given translation, substantiating the need to use more complex system descriptions than just translation.

The synthetic data, by construction, includes data at translations down to 0.3 Å, and with barriers smaller than 20 kcal/mol, cases not covered in the trajectories, where small interatomic distances are disfavored and thus rare. Translations found in trajectories start from 0.7 Å, but the vast majority are larger than 1.2 Å (Figure 3.3 B). Out of the 10 522 reactions from trajectory systems, only 24 reactions were found with barriers below 20 kcal/mol. In 21 of these systems, the reaction involves at least one hetero atom. Also, the translations are comparatively short, with an average distance of 1.2 Å. Examples of the lowest and highest barriers are shown in Figure 3.8. It can be concluded that quite specific structural arrangements are required for HAT to occur spontaneously in collagen. Further MD studies can help

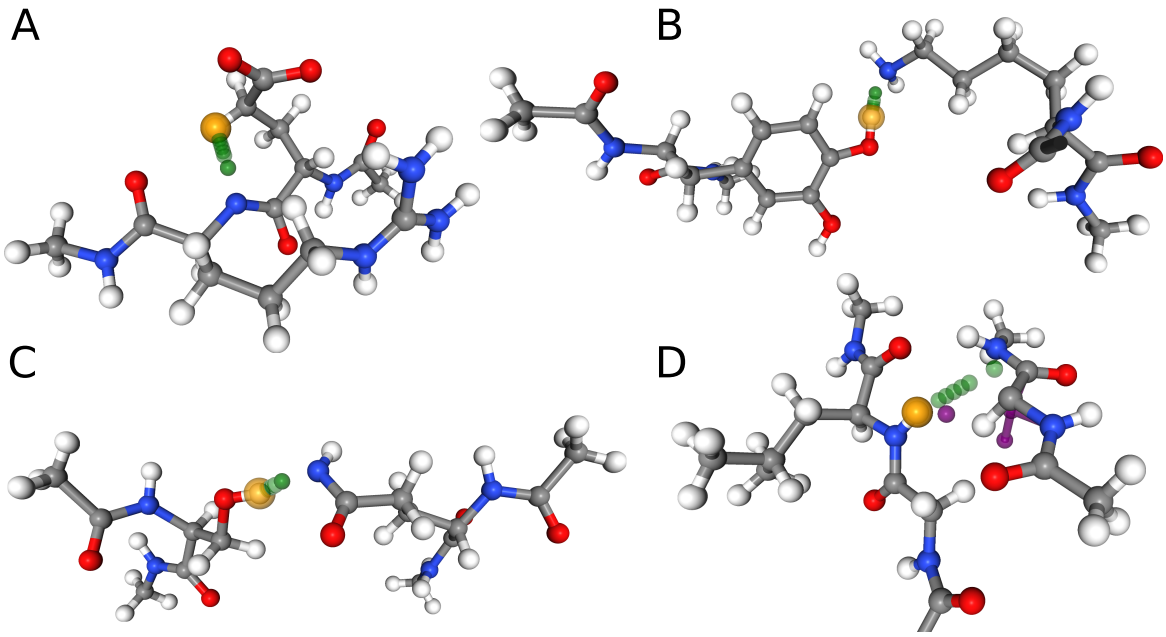


Figure 3.8. Example HAT reactions with low barriers in A-C, high barrier in D. All barriers in kcal/mol. The start position of the hydrogen is highlighted orange, the interpolated transition path green. **A** $Ea = 19.8; Ea_{opt} = 3.6$ The strong decrease of the barrier during optimization is due to the donating CH_2 group adapting sp^2 conformation. **B** $Ea = 0.3; Ea_{opt} = 0.4$ The barrier in the reverse direction has also a low barrier of $Ea = 8.7$ **C** $Ea = 14.8; Ea_{opt} = 13.0$ Very little rearrangement necessary during the reaction. **D** $Ea = 112.0; Ea_{opt} = 108.8$ The high barrier is caused by another hydrogen interfering with the reaction path. In purple, the optimized transition state is shown, where the hydrogen is pushed out of the way. This pushing opens the transition path, but comes with an energy penalty, leading to the high barrier. Additionally, the reacting hydrogen has to travel 2.9 Å.

to estimate how often such situations arise.

As mentioned above, realistic energy barriers are expected to be closer to those computed for at least partially optimized systems. But, since optimizations at the DFT level of theory for the entire data set are prohibitively expensive, only a subset of reaction paths was optimized on the BMK/6-31+G(2df,p) level of theory. The mean absolute deviation between the barriers computed based on geometries from MD, and based on DFT optimized geometries is 13.6 ± 11.6 kcal/mol (Figure 3.9). This deviation highlights the importance of the geometry optimization for obtaining accurate barrier predictions. As expected, optimized barriers are lower than the bond dissociation energy (BDE) of the dissociating bond of the reaction, in contrast to the non-optimized data set with unfavorable paths in case of large reaction distances (see Figure 3.8D for an example).

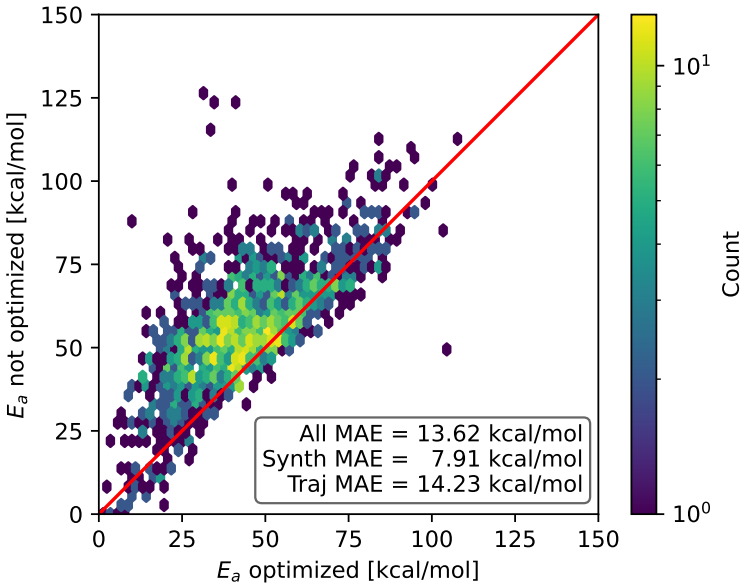


Figure 3.9. Histogram of optimized vs. not optimized energy barriers.

Notably, the energy barriers of synthetic systems change less when optimized compared to barriers from trajectory systems. (Figure 3.6 B) This analysis validates the procedural structure building process. If the synthetic systems were unreasonable, they would change more drastically compared to structures produced by MD simulation.

3.4 Bond Dissociation Energies and HAT

3.4.1 Introduction

Within reactions of the same family, the activation energy is proportional to the reaction enthalpy. While not always accurate, this relation manages to describe many HAT reactions between, *e.g.*, a metal complex and similar organic reactants.[91]

The relation was qualitatively discovered by Brønsted in 1928, formulated in 1936 and 1938 by Bell, Evans, and Polanyi, and can be written as: [92, 93, 94]

$$E_a = k_1 + k_2 \Delta G \tag{3.1}$$

where k_1 and k_2 are constants within a given family of reactions.[95]

For HAT reactions, ΔG can be calculated as the difference in bond dissociation free energy (BDFE), showing a direct connection between the activation energy and BDFE.[96] The bond dissociation energy (BDE) is defined as the enthalpy required to homolytically cleave a bond X-Y:

$$\begin{aligned} \mathrm{X{-}Y} &\longrightarrow \mathrm{X}\cdot + \mathrm{Y}\cdot \\ BDE =& H(X\cdot) + H(Y\cdot) - H(\mathrm{XY}) \end{aligned} \tag{3.2}$$

In HAT reactions, there is no change in particle number, and little in particle size, also no change of charges. Therefore, $\Delta S \cong 0$ for many HAT reactions, which is why instead of the BDFE, often the BDE is used instead. If, however, the entropy does change during the HAT reaction, it is important to use BDFE.[91]

Marcus theory describes a similar relation for outer sphere electron transfer: [97]

$$\Delta G^{\ddagger} = \frac{(\Delta G + \lambda)^2}{4\lambda} \tag{3.3}$$

Here, λ is the intrinsic barrier of rearranging reactants and solvent molecules after the change in electronic state.

Marcus theory is a powerful framework applicable to many types of systems.[98] While not covered by a good theoretical basis, it also manages to describe a wide range of HAT reactions.[91]

Both, the Bell-Evans-Polanyi principle and Marcus theory create a connection between the energy of activation, and the reaction enthalpy. For HAT, the reaction enthalpy can be calculated from the difference in BDE of the reacting hydrogen in the reactant and the product. Therefore, the X-H BDEs could be a potentially valuable input into any predictive model of HAT barriers, or more general, kinetics and mechanisms of radical reactions in proteins.

Given some experimental measurements of a similar reference bond, one can correct a calculation of a BDE using the isodesmic reaction method. An isodesmic reaction is a reaction where the number of bonds of each type stays constant. The underlying assumption is, that in a chemically similar system very similar systematic errors occur in the calculation, which then can be corrected:

$$\begin{aligned} \mathrm{X{-}H + Y\cdot} &\longrightarrow \mathrm{X\cdot + Y{-}H} \\ \Delta \mathrm{H_{DFT}} &= \mathrm{H(Y{-}H) + H(X\cdot) - H(X{-}H) - H(Y\cdot)} \\ &= \mathrm{BDE_{DFT}(X{-}H) - BDE_{DFT}(Y{-}H)} \end{aligned} \tag{3.4}$$

$$\mathrm{BDE(X{-}H)_{isod}} = \Delta \mathrm{H_{DFT}} + \mathrm{BDE(Y{-}H)_{exp}} \tag{3.5}$$

In 2012, Moor et al. calculated BDEs of all hydrogen atoms in the natural amino acids using the DFT functional B3LYP/6-31G(d), correcting the result using experimental references in an isodesmic reaction scheme. While an important step forward at that time, B3LYP is one of the less well performing functional for radical reactions.[99] Also, the set of references used lacks proper reference compounds for benzylic C–H, carboxylic O–H, and aromatic N–H bonds. Instead, tert-butyl amine was used as a reference compound for all side chain N–H bonds.

In the rest of this section, our improvements over the state of literature is described, and our produced data is analyzed. These results are also published in *RSC Adv.*, 2022, **12**, 34557-34564.[84]

3.4.2 BDE Calculation of X-H in Amino Acids

We report X–H BDEs of amino acids in proteins computed using the isodesmic reaction method at the BMK/6-31+G(2df,p) level of theory.[87] We validate our BMK results by comparing the BDEs of smaller amino acids to results obtained at the G4(MP2)-6X level of theory.[100]

BMK is a functional developed explicitly for transition states, while maintaining accuracy on equilibrium properties. It has been shown to perform well for both, thermodynamics and kinetics of radical reactions.[101, 102, 89] G4(MP2)-6X is a composite method that uses BMK/6-31+G(2df,p) geometries and performs almost as well as G4 at lower computational cost. The G4(MP2)6X energy is obtained as follows:

$$\begin{aligned} E_{\mathrm{G4(MP2)-6X}} = {} & \mathrm{HF/CBS} + E^{\mathrm{corr}}_{\mathrm{SCS-MP2}}/\mathrm{G3MP2LargeXP} \\ & + \Delta E^{\mathrm{corr}}_{\mathrm{S-CCSD}}/6-31\mathrm{G(d)} + E^{\mathrm{corr}}_{\mathrm{S-(T)}}/6-31\mathrm{G(d)} \\ & + \mathrm{HLC} + \mathrm{ZPVE} + E_{\mathrm{SO}} \end{aligned} \tag{3.6}$$

HF/CBS is a complete basis set (CBS) extrapolation of the Hartree-Fock (HF) energy. E^{corr} are frozen-core correlation energies. $E^{\mathrm{corr}}_{\mathrm{S-CCSD}}$ and $E^{\mathrm{corr}}_{\mathrm{S-(T)}}$ are scaled CCSD and perturbative triples correlation energies beyond $E^{\mathrm{corr}}_{\mathrm{SCS-MP2}}/6-31\mathrm{G(d)}$. SCS-MP2 is the spin-component-scaled MP2 method.[103] HLC is a higher-level-correction term that depends on the number of α and β valence electrons, and E_{SO} is a spin-orbit correction from experiments or accurate calculations. The zero-point vibrational energy (ZPVE) is obtained using scaled BMK/6-31+G(2df,p) frequencies. In a benchmark study of reaction energies for C_a−H abstraction in amino acids by HO$\cdot$, G4(MP2)-6X showed the best performance out of the tested methods.

As outlined before, we are using reference BDE values for the isodesmic reaction method. All used reference compounds are depicted in Figure 3.10, experimental and calculated BDEs are listed in Table 3.1. Where possible, the experimental BDEs reported by Luo,[104] were used. Solely the BDE of C_a−H in alanine is not reported, therefore the G4(MP2)-6X BDE is used instead. Including alanine as a reference compound seemed preferable over the use of another compound because C_a radicals are captodatively stabilized, and a reference compound should include this special electronic stabilization.[105] Overall, BDEs of reference bonds computed at the G4(MP2)-6X and BMK/6-31+G(2df,p) levels of theory agree well with experimental values. The mean absolute errors (MAEs) are 6.2 kJ/mol and 8.6 kJ/mol, respectively.

The electronic structure of the amino acids and thereby the BDEs are sensitive to their chemical surrounding. Hence, for proteins, adequate capping groups at the N and C termini that model the influence of the protein backbone must be used.

To assess to what extent the way the protein backbone is modeled influences the computed BDEs, we compared BDEs of the C_α atom with varying capping groups. The way the protein backbone is modelled can be expected to have a particularly strong influence on the C_α−H BDEs, not only because of the close proximity of the C_α−H bond to the capping groups, but also because its radical is captodatively stabilized by the backbone amide groups. C_α−H BDEs of glycine and alanine were calculated with three different types of capping groups: without, with smaller capping groups corresponding to the next backbone atom, and with larger capping groups corresponding to the next two backbone atoms. Calculations were performed

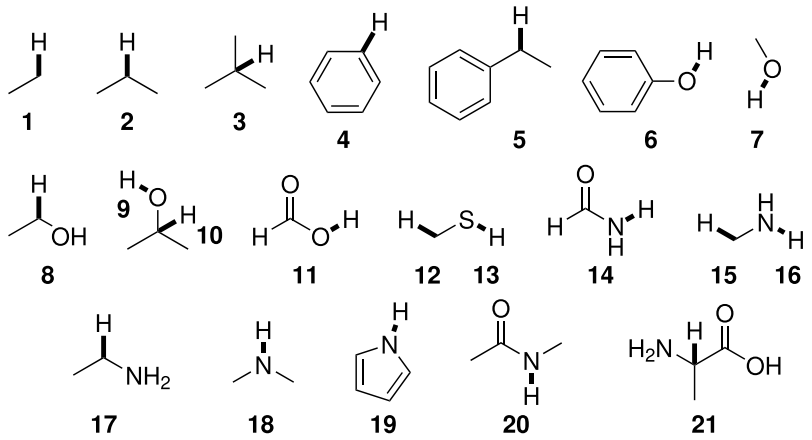


Figure 3.10. Reference compounds used to compute BDEs using the isodesmic reaction method. Reference bonds are highlighted and numbered.

using BMK/6-31+G(2df,p) and G4(MP2)-6X, as well as using B3LYP/6-31G(d) for comparison to previous literature results.[106] G4(MP2)-6X BDEs were computed directly, DFT BDEs were obtained directly and using the isodesmic reaction method. The results are shown in Figure 3.11.

Capping alanine with a formyl and an amino group raises the $C_\alpha-H$ BDE by around 30 kJ/mol compared to the free amino acid by lowering the captodative stabilization by the electron donating and withdrawing groups. This strong dependency on capping indicates a significant influence of the protein backbone on BDEs. Calculations at the G4(MP2)-6X level of theory further show that, although the $C_\alpha-H$ BDE of alanine as a free amino acid is lower than that of glycine as a free amino acid, already when considering the next atoms along the peptide backbone the relative order is reversed. The difference between the two BDEs becomes larger if one includes the next two atoms. This is in agreement with previous calculations at the G3(MP2)-RAD level of theory.[107] The trend might be due to increased steric interactions for the C_α alanine dipeptide radical that outcompete the inductive stabilization (a trend that appears reversed for leucine, see Table 3.2).[108] Therefore, both composite methods come to the conclusion that the $C_\alpha-H$ BDE of glycine in a protein should be lower than that of alanine. At the BMK/6-31+G(2df,p) level of theory, this is reproduced for the acetyl and N-methyl capping groups. Hence, all X−H BDEs in this work were computed using these capping groups. The resulting peptide models are shown in Figure 3.12. At the B3LYP/6-31G(d) level of theory, the BDE of the alanine dipeptide remains higher than that of the glycine dipeptide even if acetyl and N-methyl capping groups are used, although the exact values are closer to the G4(MP2)-6X results than at the BMK/6-31+G(2df,p) level of theory.

Table 3.1. Theoretical and experimental BDEs of reference bonds used in the isodesmic reaction method in kJ/mol. Experimental values are taken from Luo [104]

Bond	BMK/6-31+G(2df,p)	G4(MP2)-6X	Experimental
1	422.7	423.0	420.5 ± 1.3
2	410.1	412.2	410.5 ± 2.9
3	399.3	403.2	400.4 ± 2.9
4	469.4	/[a]	472.2 ± 2.2
5	374.1	383.4	357.3 ± 6.3
6	356.5	372.2	362.8 ± 2.9
7	423.4	437.6	440.2 ± 3.0
8	391.3	396.0	401.2 ± 4.2
9	427.2	442.1	442.3 ± 2.8
10	386.2	392.0	396.5
11	462.6	471.2	468.6 ± 12.6
12	401.3	397.3	392.9 ± 8.4
13	359.5	366.5	365.7 ± 2.1
14	477.7	471.3	454.0
15	383.6	388.6	392.9 ± 8.4
16	410.7	415.1	425.1 ± 8.4
17	378.5	384.2	377.0 ± 8.4
18	384.5	391.5	395.8 ± 8.4
19	399.6	396.7	405.8 ± 8.4
20	442.1	442.0	445.6
21	316.3	328.5	/

[a]Convergence failure in CCSD(T) calculation.

Consequently, if one were to use B3LYP, at least the next three atoms along the protein backbone would have to be included.

BMK BDEs are given in Table 3.2, G4(MP2)-6X BDEs can be found in Table 3.3. In general, BDEs at the BMK/6-31+G(2df,p) level of theory agree well with BDEs at G4(MP2)-6X level of theory (mean absolute deviation of 4.8 kJ/mol, Figure 3.13).

The peptides were modeled in a fully extended *trans* geometry ($\omega \approx 180°$), except for proline and hydroxyproline, where both *cis* and *trans* conformers were modeled by setting ω, ϕ, ψ, and ω' going from the N to the C terminus to approximately 0°, −75°, 160°, 180°, and 180°, −75°, 150°, 180°, respectively.[109] Where applicable, amino acids were considered in their charged and neutral forms. For histidine, the neutral N_1-H and N_3-H tautomers were considered. In the gas phase, the N_3-H

Table 3.2. BDEs and BDFEs in kJ/mol, computed at the BMK/6-31+G(2df,p) level of theory. BDEs were computed using the isodesmic reaction method and the reference bond given in the last column. BDFEs were computed directly. AA, amino acid. Pos., position

AA	Pos.	BDE	BDFE	Bond	AA	Pos.	BDE	BDFE	Bond
Ala	α	366.6	319.5	21	Hyl	α*	352.5	313.9	21
	β	429.2	396.5	1		β*	395.7	366.7	2
Arg^+	α*	326.9	288.5	21		γ	411.6	371.8	2
	β	360.0	333.2	2		δ	402.3	356.2	10
	γ	352.9	322.8	2		δ'	438.7	390.6	9
	δ	391.0	358.4	17		ϵ	372.4	339.4	17
	ϵ	470.1	425.1	18		ζ	428.5	380.0	16
	η	493.2	443.2	16	*cis* Hyp	α	371.2	324.7	21
Arg	α	361.3	312.6	21		1	418.5	379.6	2
	β	411.7	367.9	2		2	402.2	355.4	10
	γ	410.1	375.4	2		2'	443.6	393.0	9
	δ	410.2	374.0	17		3	384.1	342.8	2
	ϵ	370.0	319.3	18	*trans* Hyp	α	389.8	338.7	21
	η	412.8	363.0	16		1	420.6	380.5	2
Asn	α	340.2	294.3	21		2	401.4	352.2	10
	β*	409.8	363.7	2		2'	436.2	385.6	9
	δ*	459.4	440.1	14		3	375.3	337.7	2
Asp^-	α^{*a}	355.7	302.8	21	Ile	α	375.9	325.6	21
	β^{*a}	387.3	348.5	2		β	396.2	355.8	3
Asp	α*	358.6	315.0	21		γ'	421.1	386.1	1
	β	398.1	360.6	2		γ	402.3	368.0	2
	δ	469.1	424.0	11		δ	419.3	385.7	1
Backbone	1^{*b}	461.5	415.1	18	Leu	α	355.9	312.5	21
C term.$^-$	α	374.8	328.9	21		β	412.4	375.3	2
C term.	α	360.0	313.8	21		γ	398.1	354.6	3
Cys	α	354.0	306.8	21		δ	419.7	385.7	1
	β	382.4	354.7	12	Lys^+	α*	289.1	247.9	21
	γ	346.7	310.7	12		β	332.5	302.2	2
DOPA	α	368.0	320.1	21		γ	393.2	356.5	2
	β	354.3	331.8	5		δ*	337.3	306.1	2
	1	474.9	432.4	4		ϵ	430.6	394.5	17
	2	321.8	282.3	6		ζ	427.9	376.4	16
	3	354.9	310.4	6					
	4	482.2	440.8	4					
	5	477.0	435.7	4					

Continuation of Table 3.2

AA	**Pos.**	**BDE**	**BDFE**	**Bond**	**AA**	**Pos.**	**BDE**	**BDFE**	**Bond**
$DOPA^-$ (1)	α	368.6	322.4	21	Lys	α	360.9	313.1	21
	β	347.7	326.0	5		β	413.1	372.3	2
	1	472.4	431.9	4		γ	408.8	367.5	2
	3	316.5	272.9	6		δ	411.9	370.0	2
	4	465.1	426.2	4		ϵ	377.6	342.6	17
	5	471.8	430.1	4		ζ	423.1	371.6	16
$DOPA^-$ (2)	α	373.0	323.9	21	Met	α	362.4	315.8	21
	β	318.4	297.8	5		β	410.5	373.9	2
	1	454.6	416.3	4		γ	375.1	347.6	12
	2	260.4	218.9	6		ϵ	387.6	360.4	12
	4	472.6	434.0	4	N term.$^+$	α	412.1	362.1	21
	5	455.4	416.5	4	N term.	α	338.9	292.3	21
Gln	α*	341.4	303.6	21		1	415.0	364.4	16
	β*	403.9	374.5	2	Phe	α	368.1	318.9	21
	γ*	379.2	353.5	2		β	358.4	337.0	5
	ϵ	448.0	438.5	14		1	473.5	437.7	4
Glu^-	α	355.7	305.5	21		2	472.1	432.1	4
	β	393.2	353.6	2		3	473.8	433.1	4
	γ	389.8	353.7	2	*cis* Pro	α	369.7	325.0	21
Glu	α	356.2	313.5	21		1	412.5	375.6	2
	β	404.5	374.9	2		2	412.5	372.0	2
	γ	388.1	356.2	2		3	410.6	375.6	17
	ϵ	459.9	418.2	11	*trans* Pro	α	389.7	339.2	21
Gly	α	359.0	315.1	21		1	420.0	379.5	2
His^+	α	356.4	309.2	21		2	414.5	371.9	2
	β*	375.7	355.3	5		3	377.1	343.6	17
	1*	449.3	406.8	19	Ser	α*	357.8	313.6	21
	2	533.9	493.9	4		β	436.4	384.4	7
	3*[c]	452.1	412.0	19		γ	398.2	353.8	8
	4*	540.2	498.9	4					

Continuation of Table 3.2

AA	**Pos.**	**BDE**	**BDFE**	**Bond**	**AA**	**Pos.**	**BDE**	**BDFE**	**Bond**
His (N_1−H)	α*	339.5	297.7	21	Thr	α*	357.2	313.4	21
	β^d	348.0	331.0	5		β'	441.1	391.2	9
	1	394.3	349.9	19		β	390.4	345.2	10
	2	492.1	455.0	4		γ	420.1	384.6	1
	4	483.7	446.1	4	Trp	α	372.6	331.9	21
His (N_3−H)	α*	339.9	298.9	21		β	361.4	344.4	5
	β	351.9	332.4	5		1	500.1	465.8	4
	2	491.8	456.0	4		2	382.8	344.3	19
	3	399.2	355.5	19		4	475.1	438.2	4
	4	501.6	466.3	4		5	474.0	440.3	4
Hyl^+	α	355.3	307.7	21		6	474.8	442.3	4
	β*	391.3	350.8	2		7	477.2	442.5	4
	γ	410.6	372.2	2	Tyr	α	367.5	322.5	21
	δ	390.0	342.0	10		β	354.2	329.7	5
	δ'	490.3	433.0	9		1	471.8	435.6	4
	ϵ	431.7	397.2	17		2	475.7	440.3	4
	ζ	423.0	369.2	16		3	359.6	318.8	6
Ace-Ala-NMe	1	409.2	376.8	1	Val	α	375.4	318.7	21
	2	448.4	408.3	20		β	399.8	354.3	3
	3	450.1	412.4	20		γ	413.9	377.8	1
	4	398.7	355.6	15					

*Significant change in hydrogen bonding strength or pattern upon H abstraction. [a]Dipeptide exhibits extensive hydrogen bonding that distorts it from a fully extended geometry ($\phi \approx 82°$,$\psi \approx 66°$). [b]Hydrogen bonding between the N−H bond of the C terminal N-methyl amino capping group and the central amide group. Dihedrals going from N to C terminus change from $\phi \approx -159°$, $\psi \approx 163°$, $\omega \approx 176°$, $\phi' \approx -161°$, $\psi' \approx 162°$, $\omega' \approx 178°$ to $\phi \approx -158°$, $\psi \approx 168°$, $\omega \approx 148°$, $\phi' \approx -101°$, $\psi' \approx 65°$, $\omega' \approx 178°$. [c]Deviates from a fully extended geometry ($\phi \approx -164°$,$\psi \approx 91°$). [d]Formyl capping group on N terminus.

Table 3.3. BDEs computed at the G4(MP2)-6X level of theory. C_α–H BDEs were computed directly, all other BDEs were computed using the isodesmic reaction method.

Amino acid	**Position**	**BDE [kJ/mol]**	**Amino acid**	**Position**	**BDE [kJ/mol]**
Ala	α	360.8	Ser	α	353.4
	β	424.7		β	395.2
Asn	α	334.0		γ	434.1
	β	404.7	Thr	α	351.8
Asp^-	α	352.2		β'	437.1
	β	385.9		β	387.3
Asp	α	351.9		γ	415.0
	β	393.3	Val	α	369.5
	δ	458.5		β	395.6
Cys	α	349.7		γ	409.0
	β	380.9	Ace-Ala-NMe	1	404.2
	γ	343.8		2	445.5
Gly	α	350.3		3	447.9
Leu	β	406.3		4	393.4
	γ	392.4			
	δ	413.6			

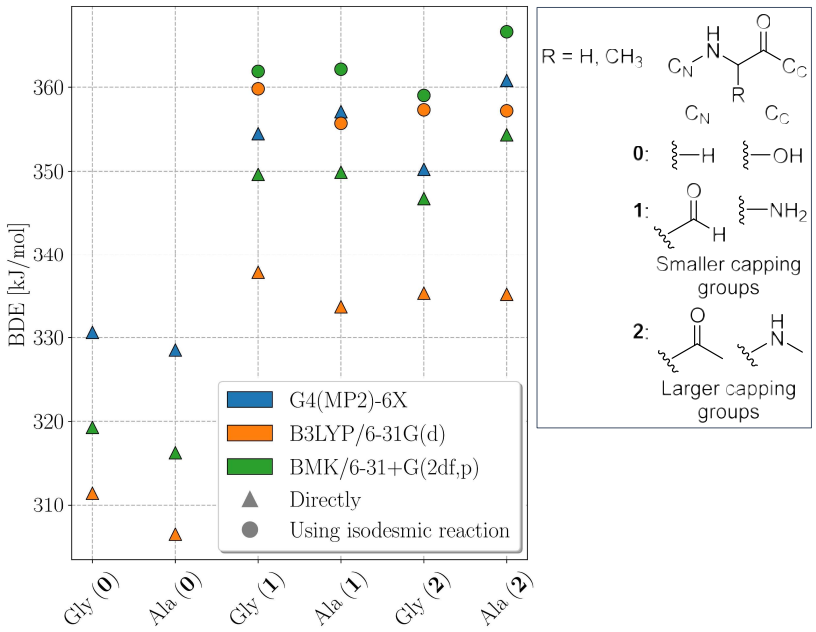


Figure 3.11. C_α–H BDEs of glycine (Gly) and alanine (Ala) with increasingly larger capping groups as described in the inset at the BMK/6-31+G(2df,p), B3LYP/6-31G(d), and G4(MP2)-6X levels of theory. Triangles indicate values that were computed directly, circles indicate values computed using the isodesmic reaction method.

tautomer is more stable by 4.9 kJ/mol. For arginine, in agreement with previous investigations,[110] out of the five possible neutral tautomers, the one in which the two terminal N atoms are doubly protonated was found to be most stable, and so this tautomer was considered.

In some cases, initial geometry optimizations resulted in hydrogen bonding between the amino acid side chain and backbone to different extent in radical and non-radical structures. This intramolecular hydrogen bonding is well known, and has also been found in a comprehensive study of the conformational landscape of amino acid dipeptides.[111] For the computation of BDEs, this would lead to energetic artifacts. Intramolecular hydrogen bonding is unlikely to occur in proteins, since the backbone amide groups are involved in secondary structure formation. In aqueous solution, polar side chain groups will also be hydrogen-bonded to water molecules or coordinated to metal ions. Therefore, we aimed at minimizing intramolecular hydrogen bonding and changes in hydrogen bonding pattern. To this end, the same initial structure was used for geometry optimizations of both dipeptides and dipeptide radicals to reflect the potential structural constraints imposed by the protein that would make large conformational changes unlikely. Yet, in some cases, significant changes in hydrogen bonding strength or pattern were observed,

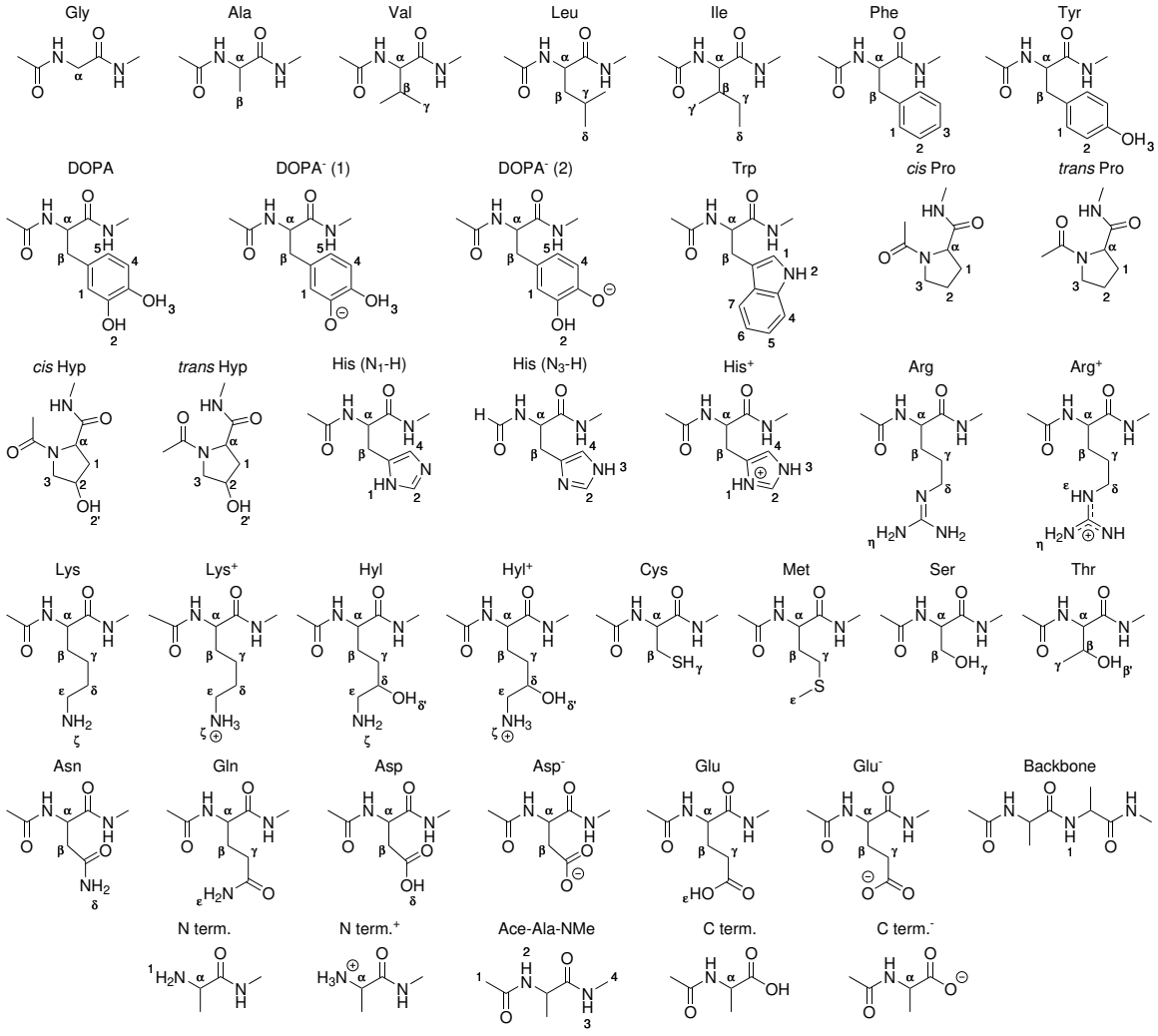


Figure 3.12. Structures of canonical amino acids, termini, and modified amino acids for which BDEs were computed. Labels denote the X-H bond, and correspond to the position in Table 3.2

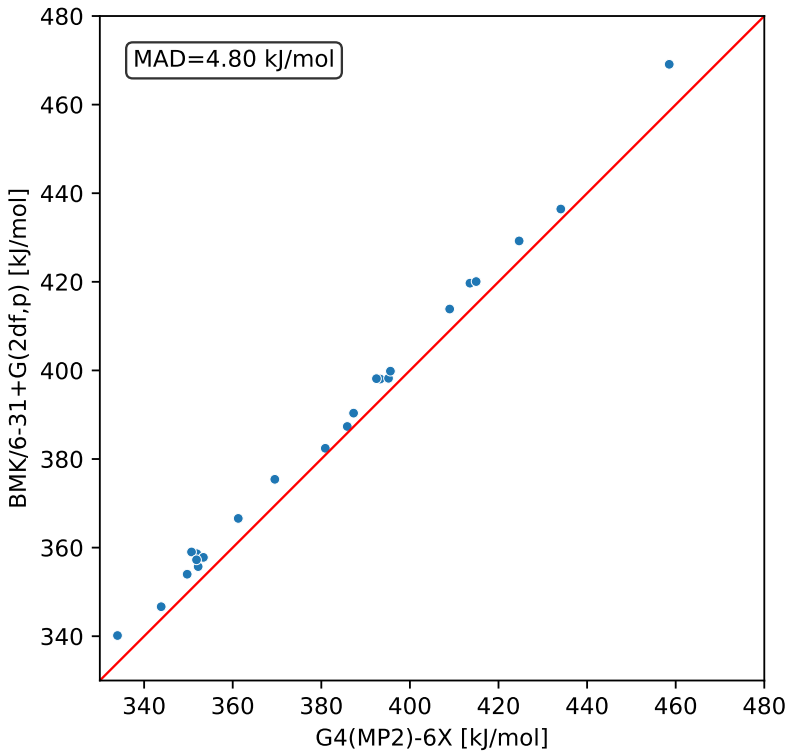


Figure 3.13. BDEs calculated using BMK/6-31+G(2df,p) vs. G4(MP2)-6X

as specified in the footnotes of Table 3.2.

BMK and G4(MP2)-6X BDEs overall agree well with other literature results for the same model peptides (Table 3.4). Deviations from the BDEs reported by Zipse and colleagues[113, 108] can be attributed to the fact that they use Boltzmann averages. The resulting difference is small for the BDEs in tyrosine, proline, and phenylalanine, since these amino acids have less conformationally flexible side chains, but becomes apparent for the BDEs in cysteine, where the side chain is more flexible. Deviations from the BDEs reported by Chan et al.[112] for amino acids with polar side chains come from the fact that they have used the lowest energy conformations of the peptides, which involve hydrogen bonding for those amino acids. In this work, hydrogen bonding in general and in particular changes in hydrogen bonding pattern between peptides and peptide radicals were avoided, and by design of our protocol the resulting conformers do not necessarily represent the most stable conformer.

Having a comprehensive data set of BDEs at hand, we analyzed trends in the BDEs across the chemical space covered by the amino acids in Figure 3.14. We grouped the X–H BDEs according to the reference compound used in the isodesmic reaction and show the distribution of the ten largest groups as well as an example radical for each group. Since the reference compounds were chosen to correspond to the local chemical environment of the radical, the histogram highlights the influence

Table 3.4. Comparison of BDEs at the BMK/6-31+G(2df,p) level of theory with literature results

AA	**Pos.**	**BMK**	**G4(MP2)-6X**	**Other**
Asn	α	340.2	334	374.9[a], 375.2[b]
Asp	α	358.6	351.9	367.5[a], 366.9[b]
Cys	α	354.0	349.7	355.3[a], 355.9[b], 374.5[c], 373.0[d], 374.6[c,x], 375.2[e,x]
Cys	γ	346.7	343.8	367.5[c], 367.4[d], 369.5[e] 366.8[f], 367.9[g], 367.3[h]
Gln	α	341.4	/	354.7[a], 354.2 [b]
Glu	α	356.2	/	382.2[a], 382.1[b]
His (N_1–H)	α	339.5	/	384.4[a], 383.5[b]
Ile	α	375.9	/	378.8[a], 378.7[b]
Leu	α	355.9	/	376.4[a], 376.6[b]
Lys	α	360.9	/	380.1[a], 379.5[b]
Met	α	362.4	/	382.1[a], 382.0[b]
Phe	α	368.1	/	360.3[a], 370.2[c], 370.7[d], 371.0[c,x], 372.9[e,x]
Pro	α	389.7	/	381.9[a], 382.6[b], 393.0[c], 391.5[d], 396.7[e,x]
Ser	α	357.8	353.4	392.3[a], 393.8[b]
Thr	α	357.2	351.8	366.7[a], 368.7[b]
Trp	α	372.6	/	364.3[a]
Tyr	α	367.5	/	360.2[a], 369.9[c], 368.7[d] 371.2[c,x], 372.7[e,x]
Tyr	3	359.6	/	346.1[c,x], 365.6[d], 349.5[e,x]
Val	α	375.4	369.5	381.2[a], 381.9[b]

[a]DSD-PBE-P86/aug'-cc-pVTZ+d, [b]G4(MP2)-6X, computed directly using the lowest energy conformations from ref.[111], taken from ref.[112] [c]G3(MP2)-RAD, [d]IMOMO(G3B3,G3(MP2)-RAD), [e]ROMP2/6-311+G(3df,2p)//UB3LYP/6-31G(d), [f]G3B3, taken from ref.[113] [g]G3(MP2)//B3LYP, [h]G3, computed directly, taken from ref.[114]. Zipse and colleagues [113, 108] use SH_2, H_2O, and CH_4 as reference compounds and Boltzmann averages. [x]Best conformer only.

of the local chemical environment on the BDEs. The observed stability trends agree with known chemical principles: captodative and resonance stabilization leads to low BDEs, as can be seen for C_a-H, as well as benzylic C−H and O−H bonds. Secondary C−H bonds and C−H bonds adjacent to heteroatoms have medium high BDEs. Primary C−H BDEs are higher. N−H, O−H, and aromatic C−H bonds are associated with the highest BDEs.

3.4.3 Method Details

Gas-phase BDEs and BDFEs at 298.15 K were computed using Gaussian09.[86] Frequency analyses were carried out to verify the structures were local minima. Spin contamination was found to be minimal for all systems (at maximum $\langle \hat{S}^2 \rangle$ = 0.7509 after correction). All side chain X-H bonds in all canonical amino acids, hydroxyproline, hydroxylysine, DOPA, as well as X-H bonds in the C and N termini, the backbone, and the acetylated and N-methyl amidated N and C termini were considered. To model the local environment in a protein, C termini were extended by an N-methyl amino group and N termini by an acetyl group. The model peptides were built using Avogadro[115] and GaussView[86]. To minimize intramolecular hydrogen bonding and obtain more reliable BDEs, for amino acids with polar side chains, side chain dihedral angles were set to $\chi_1 \approx 60°$ and $\chi_i \approx 180°$, i > 1, before geometry optimization, and the maximum size for the initial optimization step was set to $0.1\,a_0$. If the geometry optimization still led to hydrogen bonding, the maximum size for the optimization step was set to $0.05\,a_0$ and then to $0.01\,a_0$, and updating the step size during optimization was suppressed. To avoid hydrogen bonding, conformer sampling was not performed. For dipeptide radicals, however, starting structure resulting from the deletion of each chemically equivalent H atom were considered. If the geometry optimization converged to a saddle point, the structure was displaced along the vibration with the imaginary frequency in both directions and the geometry optimization was restarted. If more than one chemically equivalent structure converged to a local minimum, the lower energy conformer or the conformer that did not exhibit hydrogen bonding between the side chain and the backbone was selected.

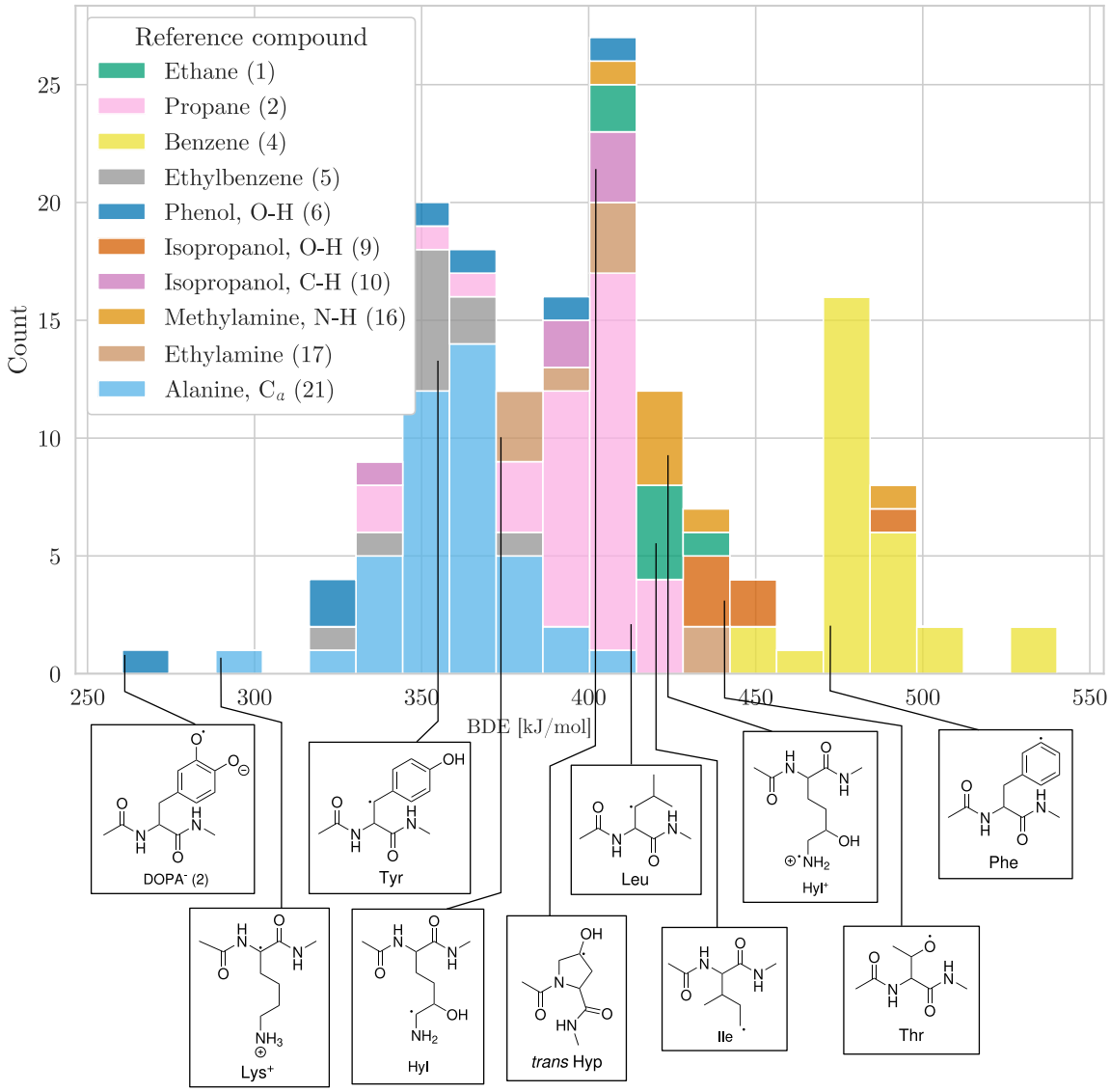


Figure 3.14. Distribution of BDEs calculated at the BMK/6-31+G(2df,p) level of theory. BDEs are grouped and colored according to the reference compound used in the isodesmic reaction method. In the legend, the numbers in parentheses refer to the numbering of reference bonds in Figure 3.12. For each group, we show an arbitrary example molecule. For visual clarity, only the ten biggest groups are shown.

3.5 Many Body Tensor Representation

Basic neural networks have a predetermined number of nodes in each layer, including the input layer. As a result, they can only accept inputs of dimensionalities supported by the architecture. This works well for, *e.g.*, images, where each input image can be scaled or cropped to the same number of pixels. However, for molecules, this would limit a model to process only molecules of a specific size. To circumvent this issue, one can calculate specific molecular properties independent of the number of atoms. Using such properties, one can construct so-called descriptors.

The many body tensor representation (MBTR) is based on pairwise measurements of distances, angles and dihedral angles.[57] Single angles are not dependent on the number of atoms, though the number of angles still is. MBTR therefore combines all angles (and other measurements) into histograms with a fixed number of bins. It also separates the measurements per combination of elements, *i.e.*, all C-C distances are grouped in one histogram, O-H distances in another. All these histograms then get concatenated into one big vector, the MBTR descriptor.

As outlined above, MBTR is a global descriptor, which means it describes the entire system rather than the chemical environment surrounding a specific point. This is useful if one is interested in the energy of a system, its solubility, etc. A reaction barrier, on the other hand, does not depend on the complete system, but only on the immediate environment of the reacting atoms. Local problems like this can be tackled with a variation called local-MBTR (l-MBTR). Here, the calculation of distances and angles is restricted to pairs/triplets containing a specified point. For HAT, we choose the target position of the reacting hydrogen as the center point of l-MBTR. This position is encoded as a special element 'X'. In the same way, the reacting hydrogen is encoded differently to all other hydrogens as 'Y' to properly define the reaction.

In this work, the implementation of l-MBTR in the DScribe package is used.[116] The l-MBTR data set is based on the same structures as used for the GNN. Section 3.1 and section 3.2 describes their generation process.

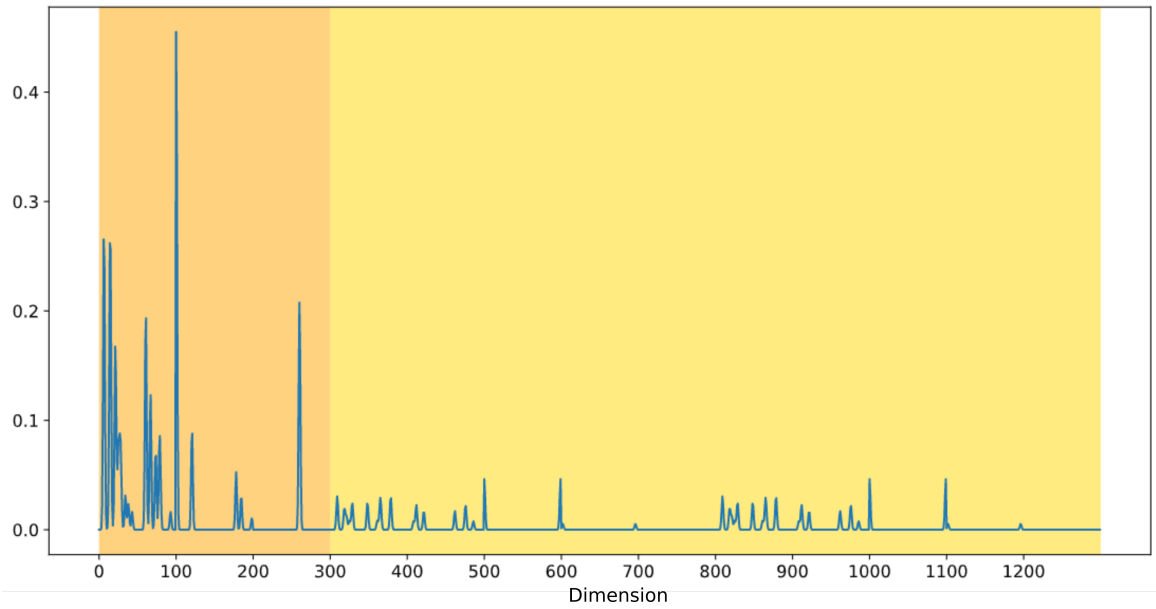


Figure 3.15. l-MBTR around a radical in a reaction between two alanine molecules. The left orange part contains all bonds, the yellow one all angles.

Chapter 4

Predicting Hydrogen Atom Transfer Barriers

The performance of different kinds of statistical models was tested in this work. Most of this section will focus on the graph neural network (GNN) approach, as the performance of this was found to be the best at that time. Later sections in this chapter will detail other approaches.

The GNN architecture and its results presented in this chapter are published in *Chem. Sci.*, 2024, **15**, 2518-2527.[83] Text and figures are adapted or copied from there.

4.1 Graph Neural Network Model

The neural network PaiNN, short for polarizable atom interaction neural network, is used to predict the energy barrier of a given reaction.[67] Specifically, it is an equivariant message-passing neural network, and thus at the time of this writing, a member of the best-performing type of network for tasks involving molecular structures. The PaiNN architecture is an evolution of the invariant SchNet architecture.[50] In this work, the implementation of PaiNN in the package `KGCNN` was adapted.[117]

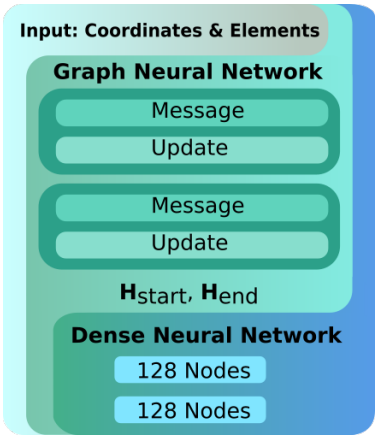


Figure 4.1. Architecture of the graph neural network for HAT barrier prediction based on PaiNN.[67]

The developed input pipeline starts from the in the previous chapter produced reactant- and product structures. From the product structure,

Table 4.1. Hyperparameters optimized for the PaiNN model using bayesian optimization. `lr` stands for learning rate. `out embedding` references which parts of the graph gets fed into the dense neural network part. `mlp` is the dense neural network. `style` is the shape of the neural network. `norm equiv` whether to apply graph layer normalization. `norm node` whether to apply batch normalization. `pooling` how the features selected in `out embedding` are pooled prior to the dense neural network.

Hyperparameter	**Best**	**Options**
Loss	MAE	MAE, MSE, MALE, MSLE
lr start	8e-4	1e-5 to 5e-3
lr scheduler	cos	None, cos, exp
lr fraction	0.01	1e-3 to 1
out embedding	H only	H only, H and distance
mlp style	static	static, shrinking
mlp layers	2	1 to 4
mlp layer size	128	50 to 1000
mlp repetitions	1	1 to 3
message iterations	2	1 to 3
norm equiv	false	true, false
norm node	false	true, false
pooling	sum	mean, sum

the end position of the moving hydrogen is extracted, and as a special element added to the reactant structure. In there, the reacting hydrogen is encoded as a special element as well to distinguish the reacting hydrogen from all others, and define the reaction. Two rounds of message passing are performed, after which the features of the reacting hydrogen in its start- and end position are extracted, concatenated, and fed into a dense neural network, consisting of two layers with 128 nodes each, using the swish activation function.[118] A single node with a linear activation function follows. The GNN is trained to optimize the mean absolute error using the adam optimizer with learning rate decay and early stopping.[119]

The performance characteristics of the model are defined by multiple hyperparameters (see Table 4.1). As optimizing these is a computationally expensive process, Bayesian optimization was used, as implemented in the `Keras Tuner` package.[120]

To increase accuracy and obtain a measure of uncertainty, an ensemble of ten models with random initializations is trained. The models are validated using 10% of the training data, each model using a random training/validation split.

4.1.1 Model Evaluation

The energy barriers are strongly correlated with the translation distance, as can be seen in Figure 3.6 B. While the former is not known prior to any calculations, we want to restrict the evaluation to physically meaningful systems, and exclude extremely high barrier of over 100 kcal/mol. For this reason, we evaluate the performance of our trained model on two data sub sets, one only with systems exhibiting translation distances < 2 Å, and one < 3 Å. In the following, performance metrics for both cutoffs are presented. $\pm$ denotes the standard deviation.

Performance metrics are summarized in Figure 4.2. Panels A and B show the performance for trajectory systems with translations below 2 Å, *i.e.*, focus on the most feasible HAT reactions, while panels D to F show measurements using all available trajectory data.

As can be seen from Figure 4.2 A, we achieve MAEs of 2.4 ± 2.5 kcal/mol using the ensemble model. Individual models only achieve 2.7 ± 2.7 kcal/mol on average on the trajectory data with the same translation cutoff in place (Figure 4.2 B). The prediction quality heavily depends on the amount of training data available, as shown in Figure 4.2 C. Adding training data improves the model up to around 90% of the available data, which corresponds to 14 068 individual barriers. Thus, the amount of training data generated at the BMK/6-31+G(2df,p) level is approximately required to reach this accuracy, but also appears to suffice, as the learning curve plateaus towards the end.

The accuracy decreases for systems with bigger translations and energy barriers, as shown in Figure 4.2 D and E: increasing the translation cutoff from 2 Å to 3 Å introduces more high-barrier systems to the test set, which seem to be harder to predict exactly. For the prediction of the propagation pathway of a radical in a complex environment, this might be acceptable though, as reactions with high energy barriers are unlikely to occur under ambient conditions. As mentioned, the use of an ensemble model also brings the advantage of an uncertainty measure: the standard deviation between the models. In Figure 4.2 F, the absolute ensemble error is plotted against the ensemble standard deviation together with a rolling average. For a low standard deviation (smaller than 1.7 kcal/mol), one can assume a low prediction error (<3 kcal/mol) quite confidently. On the other hand, higher standard deviations no longer scale reliably with the error.

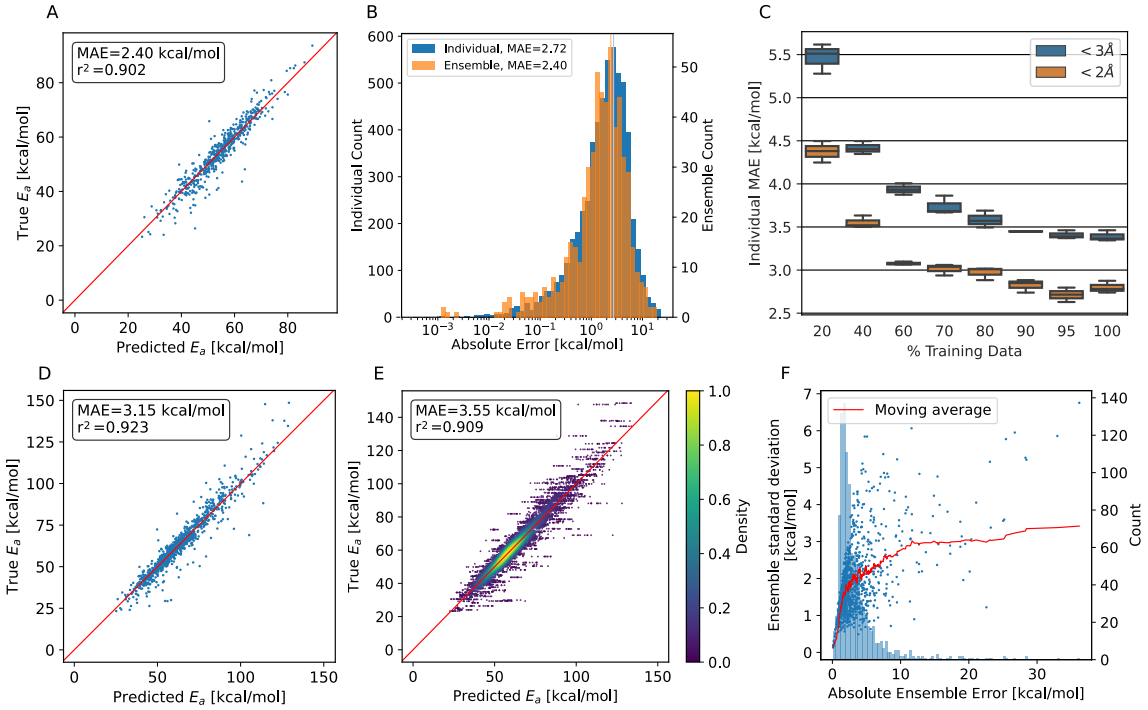


Figure 4.2. Model performance predicting HAT barriers on test data. **A** Predicted energy barriers vs. ground truth using the PaiNN ensemble model on trajectory test data with translation <2 Å. **B** Histogram of the prediction errors of individual PaiNN models and of the ensemble model. The mean of both distributions is shown as a vertical line. **C** Performance of three individual models trained on fractions of the complete training set. 100% corresponds to 15 633 training points. **D** Predicted energy barriers vs. ground truth using the PaiNN ensemble model on trajectory test data with translation <3 Å. **E** Predicted energy barriers vs. ground truth using ten individual PaiNN models on trajectory test data with translation <3 Å. **F** The absolute error of the PaiNN ensemble model on all trajectory data vs. the standard deviation of the predictions of individual models within the ensemble. In red, the mean ensemble standard deviation is plotted, and light blue in the background a frequency plot of occurring errors.

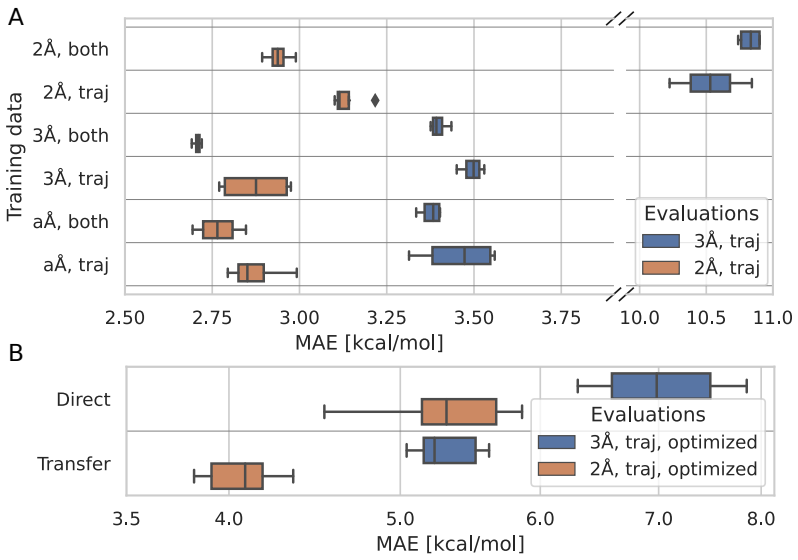


Figure 4.3. Differently trained models evaluated on comparable subsets of the test data. **A** Comparison between models trained on different data sets as indicated on the y-axis. 2A, 3A, and aA correspond to data with translations below 2 Å, 3 Å, or all translations, respectively. 'traj' refers to data only from trajectories, 'both' includes in addition the synthetic data. Four models were trained per data set. **B** Comparison between transfer learning and training directly on the optimized data only. Again, both models were evaluated on trajectory data <3 Å and <2 Å. All ten original models are used in transfer learning, ten new models were trained for the direct learning approach.

4.1.2 Training Data Impact

To understand to what degree a given part of the data improves the model, multiple models were trained on different parts of the training data and evaluated on a subset of the test data. The models were trained on trajectory and synthetic data, or on trajectory data only. Additionally, several translation cutoffs were used, either at 2 Å or 3 Å or without a cutoff. In all cases, the models were evaluated on trajectory data below 2 Å or 3 Å. This setup shows that the model benefits from being trained on synthetic systems alongside trajectory systems, even if it is evaluated only on trajectory systems (Figure 4.3 A).
Similarly, the model performance for systems with translations smaller than 2 Å improves when it is trained on larger translations. Data that is rather distant from the target prediction with regard to its chemistry and geometry still adds to the predictive power of the model. Therefore, we trained the final model on synthetic and trajectory data without a translation cutoff.

4.1.3 Transfer Learning DFT Optimized Barriers

So far, we showed that the model can well reproduce energy barriers close to DFT accuracy on structures from MD trajectories and synthetic ones. However, as mentioned previously in subsection 3.3.1, realistic energy barriers are expected to be closer to those computed for at least partially optimized systems.

To correct for the deviation between the barriers computed for non-optimized and optimized reaction paths, models already trained on non-optimized systems were retrained in a transfer learning scheme to be as data-efficient as possible. When using transfer learning, one often freezes most of the network and retrains only parts of it. Here, however, we found the best results when not freezing any part of the model. Still, models trained with transfer learning substantially outperform models trained on optimized data alone (Figure 4.3 B). Note that only the training target, *i.e.*, the barrier, was changed for transfer learning, and not the input to the model. The same non-optimized structures are fed into the model, as it is intended to be used on non-optimized structures from MD simulations. In other words, the model learns a mapping between non-optimized MD structures and DFT barriers of the optimized reaction paths. The ensemble model for predicting optimized barriers achieves an mean absolute error (MAE) of 3.6 ± 3.2 kcal/mol on trajectory data with translations of less than 2 Å and 4.9 ± 4.0 kcal/mol on translations less than 3 Å (Figure 4.4 A and B, respectively). The learning curve of the transfer learning procedure in Figure 4.4 C suggests that the model is data limited, as the accuracy

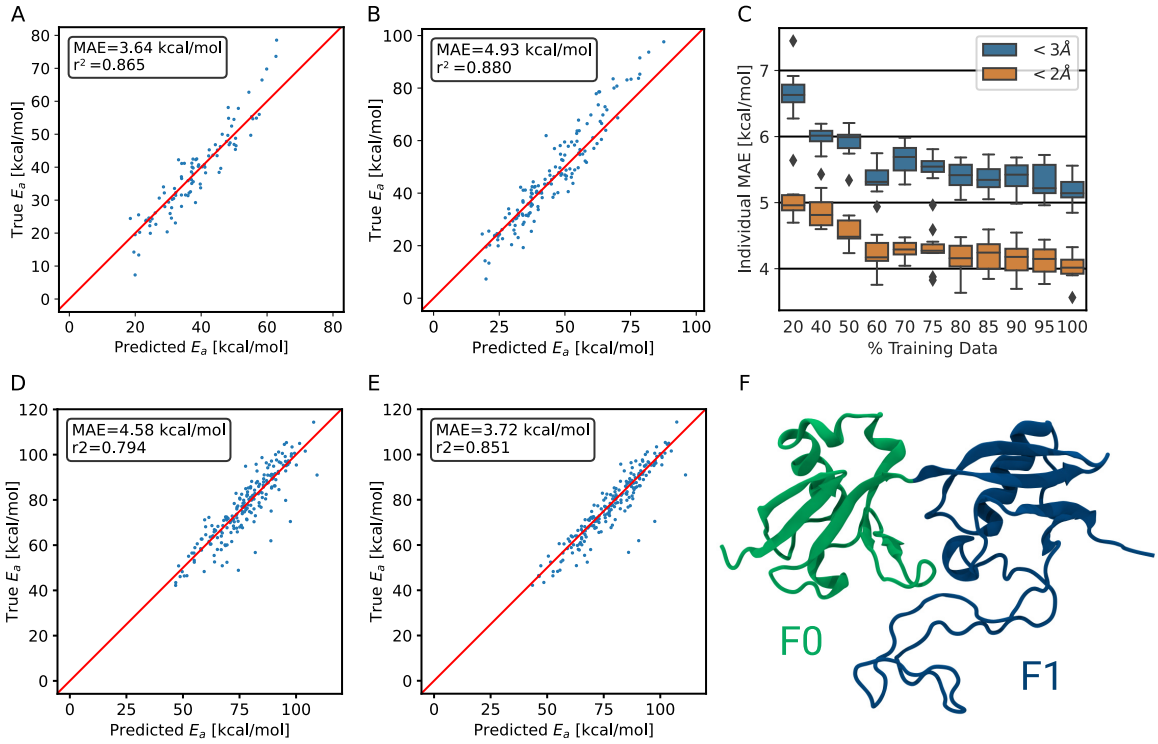


Figure 4.4. Performance of the transfer-learned ensemble model on **A** trajectory test data <2 Å and on **B** <3 Å trajectory test data. **C** Learning curve of the transfer learning process. The test MAE of the individual models is shown. **D** Performance of the ensemble model trained on synthetic and collagen-trajectory data on the F0F1 domain of FERM. **E** Ensemble model performance after transfer learning on as little as 500 structures of FERM on same evaluation set as in D. **F** The F0 and F1 domain of FERM used in out-of-domain predictions. It includes α-helices, β-sheets, and an intrinsically disordered region.

increases in particular from 90% over 95% to 100% of the optimized test data.

4.1.4 Out-of-Domain Predictions

To demonstrate that the method is not restricted to collagen-like systems only but shows transferability, we validated the performance on a different protein, the F0F1 domain of FERM. It exhibits a vastly different composition of amino acids, *i.e.*, of chemical environments. It also is structurally diverse, consisting of two folded parts and one intrinsically disordered region (Figure 4.4 F). The simulations were kindly provided by Buhr et al.[121] System preparation was done analogous to training data generation, as detailed in section 3.2. For predicting the barriers, the ensemble model trained on collagen trajectories and synthetic systems was used. As shown in Figure 4.4 D, the model does not perform as good as on its training target col-

lagen, but still achieves an MAE of 4.6 ± 4.8 kcal/mol. We note that collagen only consists of triple helices and the model has not seen other secondary structures, let alone intrinsically disordered proteins. On top, the amino acid distribution varies significantly between the training data set and the testing data used here. Therefore, this is close to the worst case scenario, and yet the model still delivers useful results. More importantly, starting from the pretrained model, the performance can be improved at a low cost in a transfer learning scheme as outlined earlier. Using only 500 new training structures from the FERM F0F1 domain, the MAE decreases already to 3.7 ± 4.2 kcal/mol (Figure 4.4 E).

4.2 Feed-Forward Neural Networks

The PaiNN GNN is a quite complex machine learning model. To justify this complexity, the performance of simpler models was tested as well. In section 3.5 the l-MBTR descriptor was introduced. Together with a basic feed-forward neural network, this descriptor is put to the test in the current section.

The used network consists of three hidden layers of shrinking sizes (1000, 500, and 100 neurons), utilizing the ReLU activation function, followed by a single output node with linear activation. The hyperparameters were determined by a non-exhaustive manual grid search. Similar to the graph neural network, an ensemble of ten models is trained. For the training, like for the GNN, data from the synthetic, and from the trajectory data set is used. Figure 4.5 shows the achieved performance. In A, the ensemble model is shown, which reaches an MAE of 4.8 kcal/mol B shows the ten individual models. Their average MAE is 5.1 kcal/mol. Although the model does not reach the accuracy of GNN approaches, it is evident that the model is able to learn some behavior of the system.

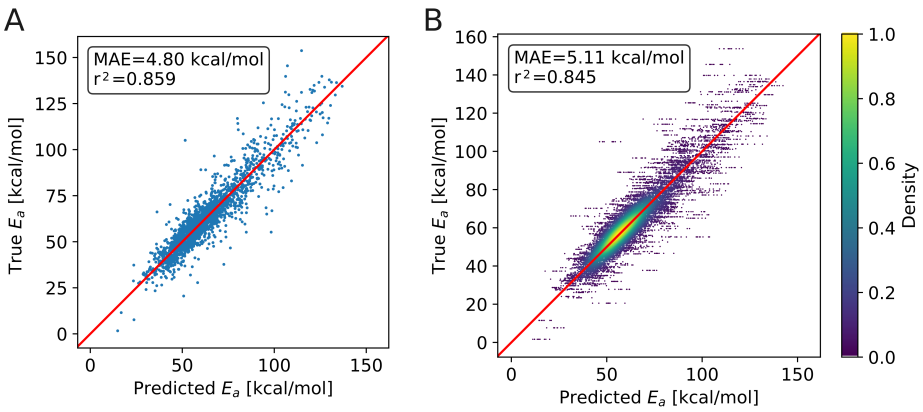


Figure 4.5. HAT barrier prediction performance using the l-MBTR descriptor. The model was trained on synthetic, and trajectory data. Only trajectory data predictions are shown. **A** Ensemble model. **B** Individual models.

Chapter 5

KIMMDY 2.0

Our aim for developing KIMMDY 2.0 is to provide a framework for reactive MD. KIMMDY stands for kinetic Monte Carlo Molecular Dynamics, and was first developed by Benedikt Rennekamp et al.[23] This new version is a complete rewrite of the code basis, and broadens the scope of the software. KIMMDY 1.0 was focused on predicting bond rupture sites in stretched (bio-)polymer. It could modify the topology of the system accordingly, and the MD simulation could be resumed from there.

With the 2.0 version, we build an extensible framework for reactive MD powered by kMC algorithms to evaluate possible reactions. Where previously user actions were required between the steps, KIMMDY 2.0 automatizes the complete workflow, while still allowing for flexibility in the execution order.

KIMMDY 2.0 is developed by Eric Hartmann, Jannik Buhr, and myself. My main contribution is a plugin for HAT prediction, based on the work described in section 4.1. Other large contributions include the implementation of the Extrande algorithm (section 2.4.3), the reaction recipe implementation, the (initial) configuration file parser, and the task list implementation. At the time of writing, I have contributed to 1,230,201 lines of code, and deleted 629,170. However, we design and develop KIMMDY 2.0 in close cooperation. The software architecture, user interface, features, and implementation details were frequently discussed, reviewed, and decided upon as a group.

5.1 Workflow and Capabilities

A run of KIMMDY is composed of multiple tasks, which can be combined in various ways. One core task of KIMMDY is to run MD simulations. For this, GROMACS is used as the simulation engine. While the user does not need to call GROMACS themselves, they need to provide the configuration files for the run (mdp files). Multiple different types of simulation can be defined and executed in arbitrary order. Also, `plumed`, a tool to analyze and modify simulations on the fly, is supported in any MD simulation.[122, 123, 124] We use `plumed` in the homolysis reaction plugin to determine bond distances without frequently needing to write the full structure to disk.

Once a simulation has run, KIMMDY can perform a reaction task next. This can represent any chemical process that modifies the binding situations or the positions of some atoms. A reaction task processes the output of a preceding simulation and outputs a set of reaction recipes and their corresponding reaction rates. Recipes are composed of basic reaction steps. Those include breaking or forming a bond, changing the position of an atom, or requesting an MD simulation, typically to equilibrate after some topology changes.

In the next phase, KIMMDY determines which reaction to execute in a kMC process based on the provided rates. Multiple kMC algorithms are implemented, *e.g.*, rejection-free kMC, or Extrande (see section 2.4).[82] Once the decision is made for one reaction, KIMMDY updates the topology and structure. It also features an interface to allow for automatic reparametrization of the topology on the fly, powered by novel machine learning approaches like Grappa.[125] Afterwards, it progresses to the next simulation task.

KIMMDY 2.0 is aimed to be easy to use with the provided reaction types, while being extensible by the user. The software is distributed through pypi for easy installation, and features a plugin system for reactions, and reparametrizing the topology.

Currently, we provide a reaction plugin for homolysis, based on measurements of bond elongations. This reproduces the KIMMDY 1.0 features. Furthermore, the GNN model predicting energy barriers for HAT, discussed in the previous chapters, is packaged as another plugin to enable HAT in KIMMDY simulations.

Tools to analyze and visualize the output of KIMMDY are provided. To prevent user errors when writing KIMMDY input files, its syntax is recognized by modern text editors to provide auto completion and helpful descriptions of each option.

5.1.1 User Experience

To facilitate the use of KIMMDY, we publish extensive documentation about the use and functionality of this project.[126] The documentation website is built using `quarto`.[127] Parts of the documentation are built directly from the input file definition, as well as the Python documentation strings. This helps to avoid redundancy, which could lead to a mismatch between functionality and documentation.

The main input to KIMMDY is a yaml formatted configuration file. yaml is a well-defined file format, that allows to be validated against published definitions. These can include which fields are required, their expected types, possible options, and even help texts. For KIMMDY, we take advantage of these features. Users of modern code editors, like neovim or vs code, receive hints and help texts while writing KIMMDY input files, avoiding typos and formatting errors.

KIMMDY structures its output into single directories per task it performs. Tasks are the way, KIMMDY internally organizes its execution order. Each of these directories is numbered, and contains the log of the respective task, and all eventually generated output files. This enables the user to quickly retrace every step of a KIMMDY run. A joint version of all logs is also made available, containing additional logging entries for general, not task specific events. Different log levels are available to generate concise logs by default, and offer more information when needed.

The main outputs of KIMMDY are MD trajectories, which can be analyzed using established tools. However, to round-off the user experience, we provide command line tools for analyzing more KIMMDY specific aspects of a run.

5.1.2 User Interface

KIMMDY 2.0 is mainly designed to be used from a terminal. A KIMMDY run is evoked by the `kimmdy` command, all available options can be listed by adding the `--help` argument. The most important argument is `--input`, which defines the path to the input file for a KIMMDY run. This input is a yaml formatted file, allowing the various options to be structured for clarity. Listing 1 shows an example of a KIMMDY input file. Each KIMMDY run has a `name`, which determines the directory name created for all generated output files. In case the name already exists, it is automatically incremented. Next, a few general parameters for GROMACS are defined, including the force field to be used. In line 5 to 7, the system is defined, by supplying a topology, structure, and index file. As the example run will use `plumed`, line 8 defines the plumed input file. The next section from lines 9 to 16 defines all settings and configuration files for three different MD simulations: one equilibration, one pulling simulation with plumed activated, and one relaxation. This is followed by a section describing the way the system should be modified (line 17 to 22). Coordinates will be changed by performing the previously defined relaxation MD. This MD switches smoothly from the topology prior, to the topology after the reaction, using the "slow-growth" feature of GROMACS. Besides the coordinates, also the topology is changed. Line 22 defines the use of Grappa for reparametrizing the system after each reaction. The possible `reactions` are defined right-after, from line 23 to 26. Here, only homolysis will be performed, but KIMMDY allows the use of different reactions within the same run, or even the same step. The last `sequence` section is crucial to every KIMMDY run. It offers a high degree of control over the run by defining which actions are performed, and in what order. A step in the sequence may consist of previously defined MD runs, or a reaction step. The sequence can be arbitrarily long, and can contain repetition blocks, as shown in line 30 to 32.

KIMMDY includes additional parts that can also be accessed through the command line interface. `kimmdy-analysis` bundles various tools for post-run analysis. `kimmdy-modify-top` allows the modification of systems outside a KIMMDY run. Specific hydrogen atoms can be deleted, and the system can be parametrized using Grappa. This can be useful for the generation of training data for a custom HAT model, fine tuned to a specific system.

```yaml
name: 'kimmdy_001'
gromacs_alias: 'gmx'
gmx_mdrun_flags: '-maxh 24 -dlb yes -nt 8 -npme 0 -ntmpi 1'
ff: 'amber99sb-star-ildnp.ff'
top: 'topol.top'
gro: 'npt.gro'
ndx: 'index_backbone.ndx'
plumed: 'plumed.dat'
mds:
  equilibrium:
    mdp: 'pullf1500_equil.mdp'
  pull:
    mdp: 'pullf1500.mdp'
    use_plumed: true
  relax:
    mdp: 'pullf1500_slow_growth.mdp'
changer:
  coordinates:
    md: 'relax'
    slow_growth: true
  topology:
    parameterization: 'grappa'
reactions:
  homolysis:
    edis: 'edissoc.dat'
    itp: 'ffbonded.itp'
sequence:
  - equilibrium
  - mult: 2
    tasks:
      - pull
      - reactions
  - equilibrium
```

Listing 1: Example KIMMDY input yaml file. This run would perform six MD simulations and two homolysis reactions.

5.1.3 Analysis Tools

Through `kimmdy-analysis`, several tools are provided to analyze various aspects of KIMMDY runs. The most basic, but highly useful tool is `trjcat`, which concatenates the generated trajectories. As the reactions do not necessarily occur in the last frame of an MD simulation, the frames after a reaction need to be excluded from the final composite trajectory. KIMMDY already splits the recorded trajectories at the correct positions. This tool is aware of these splits, and concatenates the trajectories accordingly.

`kimmdy-analysis energy` sets the usual energy analysis by `gmx energy` into the context of KIMMDY, by showing the KIMMDY MD steps next to the energy plot. An example KIMMDY run of three HAT reactions is shown in Figure 5.1. Here, an alanin-dipeptide system is simulated, and after each HAT reaction, it is reparametrized by Grappa. The HAT reaction places the moving hydrogen as close as possible to its destination before the simulation continues, but it does not change any other atom positions. Therefore, the previously sp^2 hybridized radical now has the parameters for a sp^3 structure, and the additional hydrogen close to it, but has not yet adopted this new geometry. As a result, the energy of the angle potential exhibits a spike just after the reaction. It is important to note that a short equilibration MD would be typically performed before continuing with the production MD to exclude those artifacts from further analysis.

`kimmdy-analysis runtime` analyzes the runtime of kimmdy. Besides showing the runtime of all tasks performed it can group tasks, and types of tasks together, as shown in Figure 5.2. The example analyzes the runtime of KIMMDY performing 20 consecutive HAT reactions after an initial homolysis reaction. For the HAT reactions, the GNN described in section 4.1 is used. As the simulated system is quite small (same as in Figure 5.3), the MD simulations are comparatively fast. For bigger systems, the MD runtime will increase, while the time required for HAT reaction barrier predictions will not, as it only atoms close to a radical are taken into account. Therefore, the computational overhead of KIMMDY should be negligible in most use cases.

`kimmdy-analysis radical_population` analyzes where, and for how long radicals are in the system. The resulting statistic can be visualized in two ways. First, as values on a three-dimensional snapshot of the system, as shown in Figure 5.3 A. For this, a pdb file of the structure is generated, containing the calculated radical occupancy as its beta value for visualization with established software, like `vmd`. And second, as a bar plot of the fractional radical occupancy per atom (Figure 5.3 C). Depending on the number of involved atoms, the structure, or the bar plot might

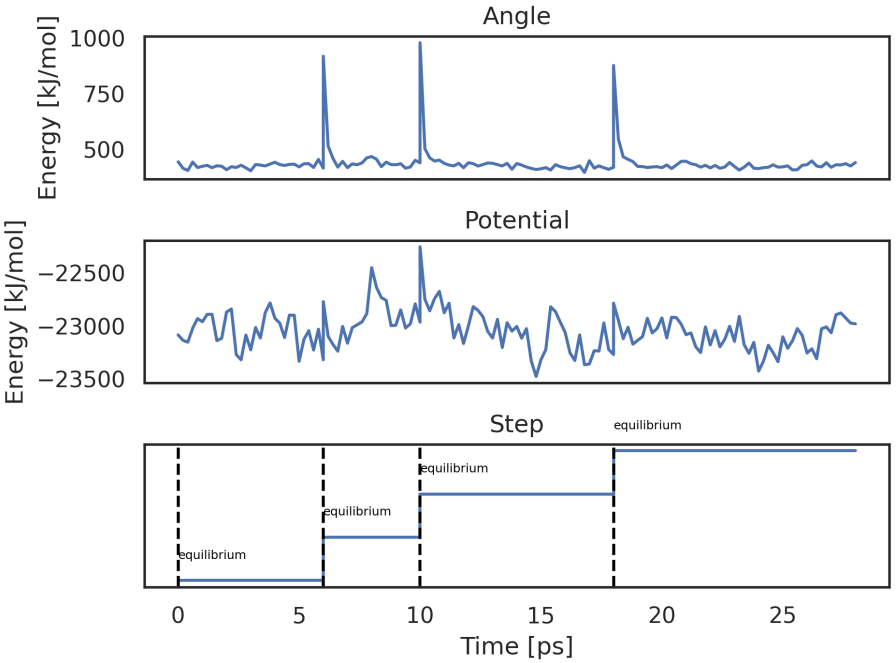


Figure 5.1. Analysis produced by `kimmdy-analysis energy`. All energy terms supported in the GROMACS tool `gmx energy` can be plotted. The lowest plot shows which KIMMDY MD run is running at a given time. Here, three HAT reactions are performed, each in between two equilibrium MD simulations.

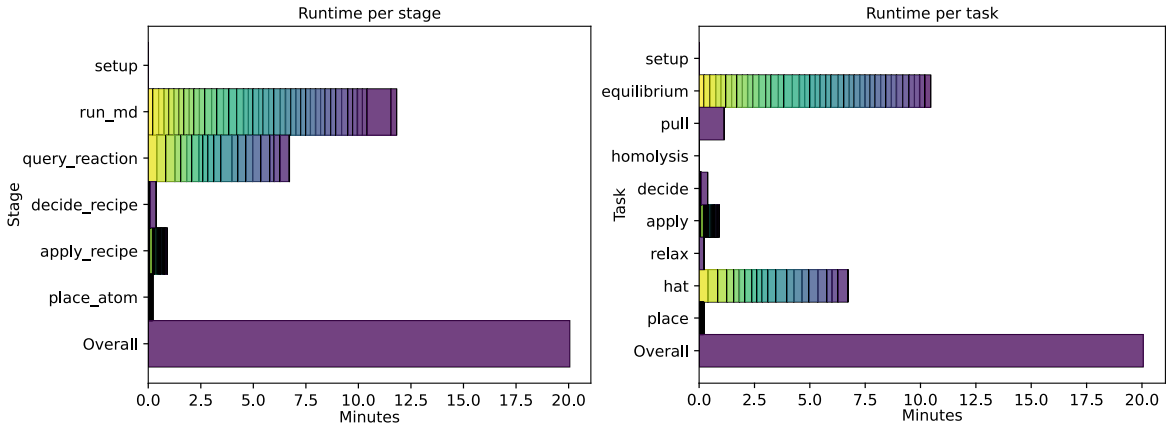


Figure 5.2. Runtime analysis of the different steps during a KIMMDY run. One homolysis reaction was performed in this run, followed by 20 HAT reactions. The bar plots are colored according to the start time of each step.

be more useful.

The `kimmdy-analysis radical_migration` tool can generate visualizations like in Figure 5.3 B, depicting between which atoms HAT reactions occur. The arrows can be colored according to the number of HAT reactions, or the maximum rate for the respective transition.

By default, KIMMDY generates in the kMC step of every reaction a plot, showing calculated rates over the course of the latest simulation. An example can be seen in Figure 5.3 D. If in the kMC step a reaction is chosen, the corresponding rates are highlighted bold and dashed. Each reaction recipe has a string representation, consisting of the involved atom numbers and symbols for the actions. Bonds breaking are depicted as ϟ. A new bond, and a movement are both represented with →. If the automatic figure generation is turned off, they can be generated at a later point using `kimmdy-analysis rates`.

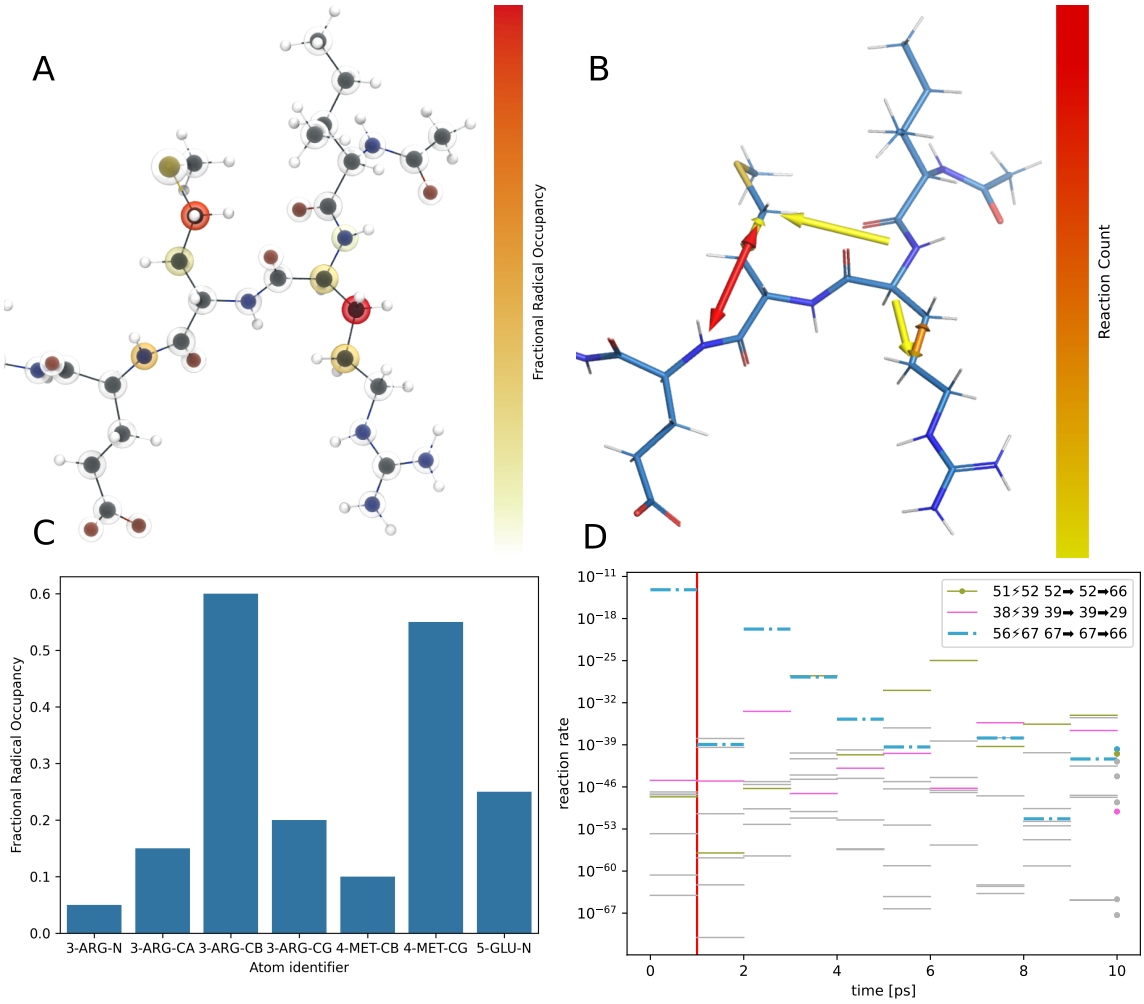


Figure 5.3. Results of KIMMDY included analysis tools of several HAT reactions after a bond rupture. After an initial pulling simulation, the backbone bond between C_{alpha} and N of the arginine in the bottom right of the structure is broken, followed by 20 HAT reaction steps. Both analyses take the complete KIMMDY run into account, but map the measurements on the initial structure. **A**: Radical occupancy of all atoms. Most atoms are never a radical, and therefore white. **B**: Arrows indicate between which atoms HAT occurs, the color from yellow to red indicates the number of transitions. **C**: Bar graph of radical occupancy shown in A. **D**: Rates of possible HAT events in a single HAT reaction step. The chosen HAT reaction is plotted dashed and bold. The point in time, from which the simulation is continued afterwards is highlighted with a red vertical line. For the most likely reactions, the recipe is shown in the legend of the plot. Bond breaks are indicated as ⚡, movements, and newly created bonds as ➔.

5.2 Software Architecture of KIMMDY 2.0

The major components of KIMMDY are depicted in Figure 5.4. In the following, implementation details of KIMMDY will be discussed.

5.2.1 Call Structure

A KIMMDY run is orchestrated from a central class called `RunManager`. After the user input from the command line interface is handled in the `cmd` module, a `RunManger` is instantiated with a `config` built from the input yaml file. During the initialization, the topology (see below) is loaded, and the plugins for reactions and parametrizing are instantiated. The run then starts with a call to `runmanager.run`. This triggers building of the initial task queue, which then gets processed. In addition to this main task queue, there is also a priority queue available, for tasks to submit other tasks to. These get executed as soon as the current task has finished, but before any eventual other tasks in the main queue.

During the task queue building process, reactions are expanded into three tasks. The first one calls the plugin itself to obtain reaction recipes. Then, one recipe must be chosen from all possible ones in a kMC step. Finally, the recipe must be applied at the correct time of the last simulation.

5.2.2 Input Configuration

As discussed in subsection 5.1.2 from a user's perspective, KIMMDY expects a configuration file for every run, formatted as yaml. This file can contain key-value pairs, where the values themselves can be dictionaries, lists, or other simple types. From the input yaml, a `Config` object is build recursively. Whenever a value is a dictionary, it becomes a new sub-`Config`. This allows KIMMDY to pass specific sub-`Config` objects to certain tasks. The tasks therefore do not need to worry about finding the appropriate settings for them, instead they receive exactly the information they need.

The allowed fields within the input yaml are defined in so-called schemes, which are json files. Besides the in subsection 5.1.1 mentioned use, to guide the user in creating input yaml files, the schemes also are used to verify the input to KIMMDY at runtime. Furthermore, the types defined in the scheme are used to parse the input appropriately, *e.g.*, to resolve paths.

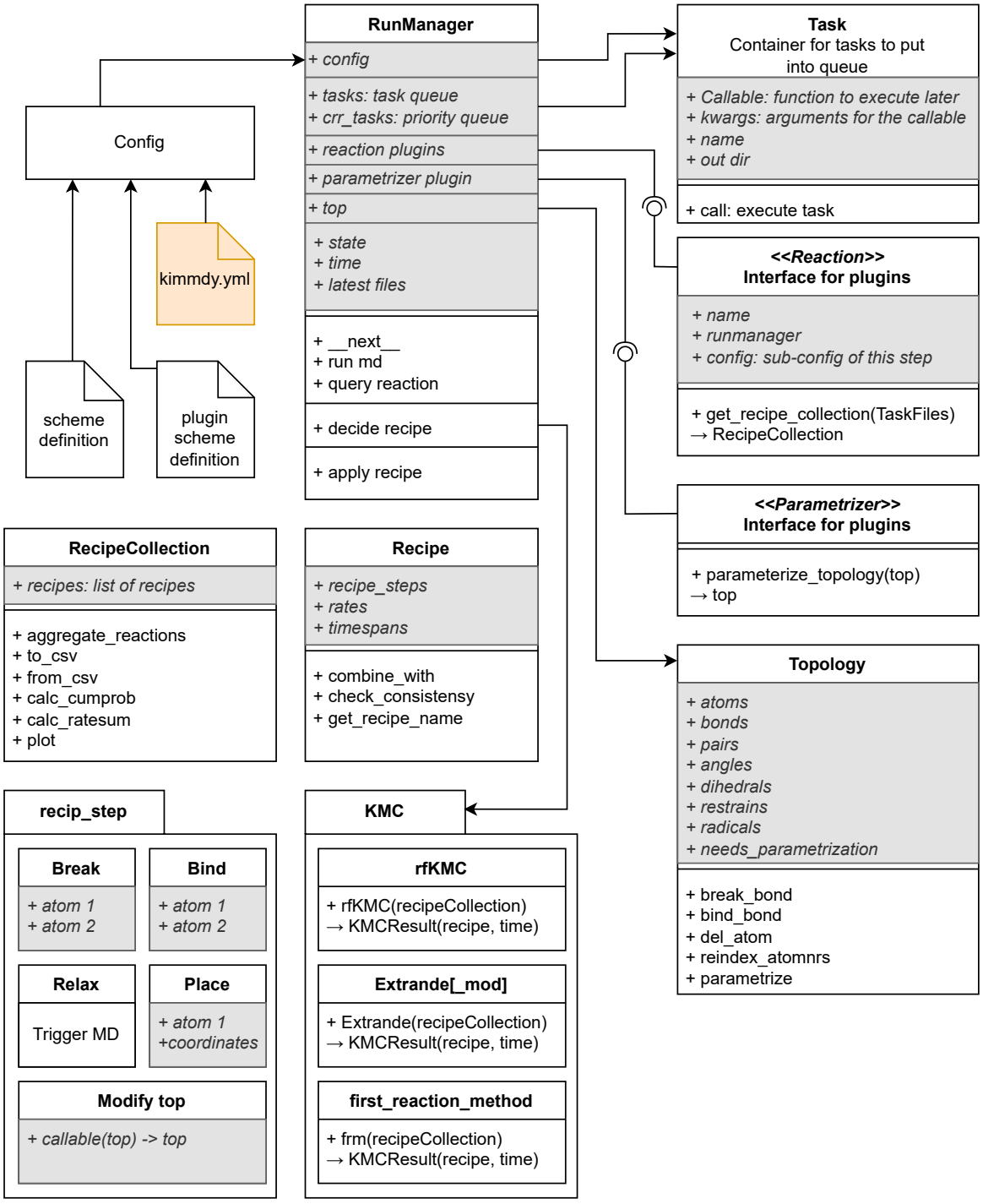


Figure 5.4. Major components of KIMMDY 2.0. Attributes have a gray, the user-supplied input a orange backdrop.

5.2.3 Internal Topology

From the initially user-supplied topology, an internal topology object is build. First, the `top` file is parsed into a dictionary, and all includes are resolved. With this dictionary, a `Topology` object is instantiated. The force field is explicitly resolved, meaning every interaction has individual parameters. This enables the use of Grappa for reparametrizing on-the-fly, but is still compatible with established force fields, like Amber or Charmm.[33, 31]

The `Topology` supports breaking bonds, creating bonds, and deleting hydrogen atoms. For all of these actions, the interaction terms, like pair exclusions, bonds dihedrals, etc., are updated accordingly. Furthermore, it identifies, and keeps track of possible radicals, and whether it needs parametrization due to some modifications. Creating bonds, deleting bonds, and moving atoms all do not change the index of the atoms stored in the topology. This ensures that each atom is trackable using standard analysis techniques. The downside is, that residues are not necessarily continuous in the resulting topology anymore. However, this should not cause issues in most cases. If continuous residues are needed, the `Topology` provides a function to reindex itself. After a atom is deleted, the topology is reindexed by default. To finally use the `Topology` in an MD simulation, a new `top` file is written.

5.2.4 Tasks

Tasks are containers for functions and their arguments to be called at a later point. They can be stored in a queue until it is their turn to be executed. When a task is finally executed, it creates itself a `TaskFiles` object. This object is the interface for a task to read and write files. The newest versions of a particular type of file, *e.g.*, a structure can be obtained, and the task can check where it should write new files. Then, the stored function gets executed, and afterwards returns the `TaskFiles` object. It is used again to discover the newly created files, and to build-up a record of which task used or created which files.

5.2.5 Reactions

Reactions are not directly build into KIMMDY, but through a plugin system. Whenever a plugin is installed, it registers itself, so KIMMDY can discover all available reactions during the `Config`-building process.

The interface for reactions is rather simple. KIMMDY defines an abstract `ReactionPlugin` class, which the plugin must inherit from. Within, there is a single function, which must be implemented by the plugin, called `get_recipe_collection`.

This is treated as a task, therefore receives access to the recent files through `TaskFiles`. Based on the latest simulation results, the plugin then generates reaction rates alongside reaction `recipe`s. These `recipe`s describe all modifications to the topology and structure required to perform the reaction corresponding to the calculated rate. A `recipe` is composed of `recipe_steps`, which can be breaking a bond, creating a bond, placing an atom at some coordinates, modifying the topology, or requesting a specific MD simulation.

How the reaction rates are determined is up to the plugin. They can be calculated based on simple or complex physical models or predicted by a machine-learned model. Currently, we provide a set of reaction plugins, utilizing different methods to generate rates. The `homolysis` plugin calculates rates based on bond elongations and bond strengths. The `hat_naive` plugin pickes random hydrogen atoms to react, and is used for development purposes. And lastly, the `HAT_reaction` plugin utilizes the GNN detailed in section 4.1 to generate rates.

5.2.6 Kinetic Monte Carlo

After a reaction plugin has generated some rates and recipes, a kMC algorithm is used to determine which reaction to execute. KIMMDY implements several kMC algorithms to choose from, depending on the use case and type of reaction. Although a reaction defines a preferred kMC algorithm, the user can override it if desired.

In section 2.4, rfKMC and Extrande are introduced. Both of these algorithms are implemented in KIMMDY. Furthermore, a modified version of Extrande is available as well.

rfKMC is the default for homolysis reactions. In those reactions, the preceding simulation is used to sample bond distances, which then are averaged. Therefore, it makes sense to use a time independent algorithm, like rfKMC. Note that time-independent does not refer to the ability to handle time-dependent rates. A Recipe is picked by the algorithm based on the integral over all provided rates per recipe. Therefore, it is not able to pinpoint an exact point in time when the reaction should occur. The algorithm is designed to move between independent states, therefore, it does not matter from where *inside* a state a transition is happening. The complete last simulation is treated as one state. KIMMDY still needs a specific time at which the structure modifications should be applied. Therefore, it selects the point in time with the highest rate. This should result in the most physically reasonable trajectory possible.

Both rfKMC and Extrande are suitable algorithms to use for HAT, although the default is Extrande (with the modifications detailed below). For situations where a

large set of possible radical positions must be evaluated for their stability, rfKMC can be a good choice. It allows for improbable reactions to occur, reducing the risk of getting stuck as easily. Additionally, it is an efficient algorithm, as in every step a HAT reaction is performed. Yet, this can also be a disadvantage if the user intends to obtain a physically plausible trajectory. In a system experiencing pulling forces, the environment of a radical can rapidly change. In some situations, no reasonable reaction may be possible, and continuing the MD simulation may be the better option than attempting a reaction. Extrande could produce the desired behavior, as in this algorithm reactions and continuing the MD compete with each other.

However, the implementation as described in section 2.4.3 has some issues in the outlined use case. In theory, Extrande can deal with varying rates by adapting its time step τ according to the maximum expected rate. This method is effective for minor fluctuations in the rate, but if the fluctuations are significant, τ may become very small, resulting in a slow progression through the trajectory.

To address this issue, a simple modification would be to select shorter look-ahead time windows L, that contain less fluctuating rates. In fact, in the KIMMDY implementation of `extrande_mod`, L is chosen so short, that the rates within are constant. The additionally required evaluations of more time windows do not cause issues, as the algorithm is efficient within windows of small rate fluctuations. This modification has the side effect of disabling the eponymous extra reaction channel, as B is now always equal to a_0.

Every kMC function must return a `KMCResult` object, containing the chosen recipe, the probability of it, the point in time when the recipe should be applied, and a time delta, specifying the possible jump in time.

5.2.7 Code Quality and Maintainability

Maintainers of academic software often change rapidly due to the nature of short-term projects, such as master's and doctoral degrees. Hence, writing robust and maintainable code was made a top priority in the development of KIMMDY.

To minimize the friction when working with multiple people on the same code base, we established clear conventions throughout the developing process. The documentation string style adheres to the standards published by `numpy`.[128] Commit messages are formatted according to `conventional commits`.[129] Merge conflicts are minimized through the use of unambiguous code formatting provided by `black`.[130] To avoid bugs and regressions, KIMMDY uses automatized tests. At the time of writing, 102 tests are implemented, covering 75% of the code. Most of the code not covered is part of the standalone tools and analysis scripts. Writing

tests also for these is planned prior to the main release. Tox is used to run the tests against the supported versions of Python, 3.10 and 3.11.[131]

For managing the code base, we use git. The project is hosted on GitHub, and uses the GitHub Flow branching strategy. This means, we have one main branch, and several feature branches. Whenever a feature is finished, a pull request is made into the main branch, requiring a review of another core maintainer. Stable releases of KIMMDY are marked using tags in the main branch.

The process of managing the code base becomes less error-prone when less manual work is required. We therefore use GitHub actions to automate many aspects of our release pipeline. Running the tests can be triggered for every pull request by adding a label to it. Likewise, the building of the documentation can be triggered with another label.

All commits to the main branch of KIMMDY are collected automatically into a separate release pull request. Whenever this is merged, workflows are triggered to increment the semantic version number, create a corresponding tag, publish a change log, build a KIMMDY package for a release on pypi, and update the online documentation.

Chapter 6

Discussion and Outlook

This thesis presents the development of a predictive model for hydrogen atom transfer (HAT) reaction barriers, from the data generation, over model building, to its implementation in the wider-scope software KIMMDY. As a result, KIMMDY is now able to simulate HAT reactions in an MD setting. However, as is common in science, there is still progress to be made on several aspects of this project. In the following, the state and perspective of the training data, the model, and KIMMDY will be discussed.

The data used to train and evaluate the machine learning model determine its quality and capability. In chapter 3, a two-step training procedure proved to be an expedient approach, yielding a large lower-quality data set, and a small higher-quality data set. This was necessary, due to the required amount of training data of our model of choice, PaiNN. Figure 4.2 and Figure 4.4 show that for this model the point of diminishing returns in accuracy with increasing data set size is reached. Hence, the increase in prediction quality would not justify the cost of generating more training data.

The models approach already the quality of the underlying DFT method. The authors of the functional of our choice, BMK, originally targeted an accuracy of 2 kcal/mol on energy barriers.[87] Depending on the benchmarks, BMK achieves an MAE of 0.8 kcal/mol to 5 kcal/mol relative to CCSD(T) calculations.[90, 88] Therefore, with the achieved accuracy of 3.6 kcal/mol on optimized data, we are close to DFT accuracy.

The decision to use PaiNN for predicting HAT barriers was made after an extensive screening of alternative machine learning models performed by Elizaveta Bobkova during her master thesis, co-supervised by me. Her work included testing the performance of a random forest regressor[132], a Gaussian process regressor[133], feed-forward neural networks, and the graph neural networks MEGNet[134],

DimeNet++[135], SchNet[50], and PaiNN[67]. While not directly comparable with the results presented in this work due to changes of the data set, her results show a clear performance trend between the analyzed models. Random forest, Gaussian process regression, and feed-forward NNs achieve similar results, with MAEs around 5 kcal/mol. MAGNet performs slightly better, with approximately 4.5 kcal/mol. SchNet and DimeNet++ reach a similar accuracy of 3.5 kcal/mol, while PaiNN beats the tested competitors with a MAE slightly below 3.0 kcal/mol. These results are inline with the results presented in section 4.1 and section 4.2, where the feed-forward neural network using l-MBTR achieved 4.8 kcal/mol.

Preliminary experiments show that newer architectures like the Atomic Cluster Expansion based MACE GNN[62, 69] might be more data efficient, rendering the less accurate data set obsolete. Going forward with this project, reevaluating new machine learning architectures could be worthwhile, especially since KIMMDY is built flexibly enough to allow such changes without requiring major modifications.

The inclusion of BDEs into the input for the machine learning models was expected to improve the predictive performance, due to the connection of reaction barriers to BDEs as described in section 3.4. However, this was not the observed result, and instead the performance increase was not significant in any of the tested models. One reason could be that the 196 calculated values were already encoded in the barrier data set. In this case, though, the learning process should at least be accelerated, which was not the case. Alternatively, the influence of the BDEs on the barrier could just be too weak in comparison to the influence of the geometry.

As described in subsection 3.3.1, the developed model is predicting HAT barriers of DFT optimized geometries, given MD structures as inputs. This rather unusual task has some implications on the data generation process, and the use of the model in KIMMDY. Often reaction rates are calculated between the optimized start conformer and the transition state conformer. This approach is suited to draw conclusions for a bulk system in solution, but not for single conformers, as it is our goal here. Sampling conformers according to the Boltzmann statistic is task of the MD. The optimization should therefore preserve this conformers, while correcting errors in the structures due to the limited capability of the force field. Especially to obtain proper transition state structures the DFT optimization is highly important, as the reactants typically move towards each other in the transition state. The impact of the optimization is shown in Figure 3.9. During a KIMMDY run, the rate predictions need to be frequent enough to ensure a rate prediction for every conformational change not included in the DFT optimization.

A significant drawback of the data set, as it is presented here, is that the MD structures used as input for the model do not contain proper radical parameters. This is not an issue for the predicted rates, as they are based on DFT optimized structures. However, the MD simulation is flawed. sp^3 hybridized atoms donating a hydrogen in a HAT reaction will continue to be sp^3 hybridized when the simulation continues, as traditional protein force fields like Amber and Charmm do not contain parameters for radicals in all the different possible positions.[33, 31] This is where Grappa proves invaluable, as it can provide these missing parameters, and therefore can ensure physically accurate behavior of radicals in the MD simulations.

However, the presented HAT prediction model is still expecting radicals with the same geometry as their non-radical counterparts. Just correcting the MD parameters with proper parameters does not suffice. Hence, a new training set must be generated to allow the use of Grappa together with the HAT prediction model. So far, each reaction could be viewed from both sides, resulting in two barriers to train on per three DFT optimization calculations. This is possible, as the reactant structures are known for both reaction directions. In the forward direction, it is directly sampled from MD, in the backward direction, it differs only in the position of the reacting hydrogen, and can therefore be generated algorithmically. When using Grappa parameters, this is no longer the case, as now the radical state affects the geometry at the MD level. Although each reaction now can only be used in forward direction, the computational cost is not doubled, since the product state no longer needs to be calculated. Only the reactant and transition state do still need to be optimized using DFT, which results in an increase of the computational cost by one third. At the time of writing, this new data set is under construction, and the model trained on it will be released alongside the KIMMDY 2.0 publication. User will have the option to use it in conjunction with Grappa-generated MD parameters or to stick with classical force fields and obtain rates from the GNN introduced in this work.

It is of utmost importance to keep the task in mind the model is solving. As it is exclusively trained on HAT reactions, it is unaware of the existence of other possible reaction mechanisms. To use Bayes' vocabulary, the posterior is the predicted rate, given the prior assumption that the reaction is a HAT. Therefore, the user must take care to interpret the predicted rates correctly, and only apply the model to appropriate situations. However, additional reaction mechanisms could be included in two ways. First, the model could be trained to predict rates of alternative mechanisms as well, either implicit, by training always on the lowest barrier, or explicitly, by building multiple outputs into the model, one for each mechanism. Second, one network per mechanism could be trained. This would be very similar from KIM-

MDY's perspective to a multi-output model, as rates of different mechanisms enter into the kMC process. Depending on the system, such other mechanisms could be proton-coupled electron transfer (PCET) other than HAT, like consecutive proton transfer, followed by electron transfer (PT/ET) or *vice versa* (ET/PT).[136, 137] Furthermore, pure proton movement can occur in case of acid-base reactions. As shown in Figure 4.4, the model is to some degree transferable to other systems than collagen. However, at this point it is only trained to predict HAT in collagen, and some accuracy decrease should be expected when applying it to other systems without retraining.

The aim of developing KIMMDY is to model reactions within MD simulations. However, KIMMDY is not the only available solution to do so. Reactive force fields like AIREBO, COMB, or ReaxFF were developed to perform MD without a fixed topology.[138, 139, 140, 141, 141] Like usual MD, ReaxFF defines a fixed-form potential to act on the atoms, but with more terms to allow a flexible bond order. These additional terms, together with the need for a charge rearrangement method, account for the increased computational cost compared to standard MD.[142] Therefore, reactive force fields can not match the system size achievable with non-reactive force fields.

Recently, another method was developed, allowing reactive MD simulations. Many machine learning force fields are in principle capable of modeling reactions, as they usually do not require a fixed topology of the simulated system. Instead of working with chemical bonds, they often build up some sort of environmental description around each atom, based on the relative positions of other atoms in the vicinity. If their training data includes transition state energies, they can be used as a reactive force field. Currently, their computational cost is still higher than usual MD, although this gap is closing.[44]

Classical MD can also be paired with quantum mechanics simulations (QM/MM). Here, a reactive region needs to be predefined, which then gets calculated using QM methods. On the MD level, all atoms are still simulated, but the atoms in the QM region obtain their forces from the QM calculation. Not only are these often more accurate, as the QM calculation is not based on a force field, it also allows molecules to react within the QM zone. Compared to pure QM calculations, bigger systems can be simulated with QM/MM, although QM/MM is even more costly than reactive force fields.

Still, any reactive force field or QM/MM method can only simulate reactions fast enough to occur within the simulated time. MD can today achieve milliseconds of simulation time.[143] Yet, many reactions only occur on much longer timescales, and

can therefore not be simulated using techniques relying on a continuous simulation time. Kinetic Monte Carlo simulations solve this problem, as they jump between low energy states, without the need to simulate every step along the way.

KIMMDY manages to incorporate reactions into MD without major speed overhead like the mentioned methods. Furthermore, they all are restricted to reactions occurring on the timescale of MD. The rfKMC approach enables the modeling of slower reactions, as lower rates result in longer time jumps. Extrande does not jump further in time than the MD simulation allows, but it can accelerate reactions using a scaling factor to bring slow reactions on the MD timescale. Using kMC, reactions are treated explicitly, unlike in traditional or machine learned reactive force fields, where a reaction is a smooth movement on the potential energy surface.

Our approach to reactive MD makes KIMMDY very flexible. Besides the simulation of HAT and homolysis reactions in all-atom models of proteins, KIMMDY can be applied to other reactions in other types of systems. It could enable the analysis of hydrolysis reactions in DNA or RNA systems. Furthermore, KIMMDY is not restricted to all-atom simulations. Coarse-grained systems, where one bead represents multiple atoms, are compatible with KIMMDY and could be used to simulate polymerization reactions on a large scale. Depending on the desired simulation accuracy, reaction rates could be obtained from complex machine learning models trained on QM data, or simple physical models.

6.1 Summary

In this thesis, a new graph neural network model is trained for the prediction of hydrogen atom transfer reaction barriers. The generation of the training data is presented, and the performance of the resulting model evaluated. With an accuracy of 2.4 kcal/mol on non-optimized structures, and 3.6 kcal/mol on DFT-optimized ones, it achieves accuracies close to DFT, and suitable for reaction rate predictions. The software package KIMMDY 2.0 enables reactive simulations by coupling MD with kinetic Monte Carlo algorithms. KIMMDY utilizes the aforementioned machine learning model to perform hydrogen atom transfer reactions, but is not restricted to this application. It is designed as a platform for reactive MD, currently also capable of performing hydrolysis reactions. The analysis of further types of reactions in an MD setting will become feasible through the development of additional reaction plugins in the future. This thesis paves the way towards efficient simulation of biological and synthetic soft matter systems including reactions they undergo during their dynamics.

Bibliography

[1] Gerd Kaupp. Mechanochemistry: The varied applications of mechanical bond-breaking. *CrystEngComm*, 11(3):388–403, February 2009. https://pubs.rsc.org/en/content/articlelanding/2009/ce/b810822f.

[2] M. Mozaffari and H. Masoudi. Zinc Ferrite Nanoparticles: New Preparation Method and Magnetic Properties. *Journal of Superconductivity and Novel Magnetism*, 27(11):2563–2567, November 2014. https://doi.org/10.1007/s10948-014-2625-x.

[3] Qiang Gu, Jing Zhang, Hongxia Li, Guoqi Liu, Honggang Sun, and Bingbing Fan. Synthesis of α-Si3N4 powder by high energy ball milling assisting molten salt nitridation method at low temperature. *Ceramics International*, 45(15):18445–18451, October 2019. https://www.sciencedirect.com/science/article/pii/S0272884219315585.

[4] H. Staudinger and W. Heuer. Über hochpolymere Verbindungen, 93. Mitteil.: Über das Zerreißen der Faden-Moleküle des Poly-styrols. *Berichte der deutschen chemischen Gesellschaft (A and B Series)*, 67(7):1159–1164, 1934. https://onlinelibrary.wiley.com/doi/abs/10.1002/cber.19340670708.

[5] Mary M. Caruso, Douglas A. Davis, Qilong Shen, Susan A. Odom, Nancy R. Sottos, Scott R. White, and Jeffrey S. Moore. Mechanically-Induced Chemical Changes in Polymeric Materials. *Chemical Reviews*, 109(11):5755–5798, November 2009. https://doi.org/10.1021/cr9001353.

[6] Martin K. Beyer and Hauke Clausen-Schaumann. Mechanochemistry: The Mechanical Activation of Covalent Bonds. *Chemical Reviews*, 105(8):2921–2948, August 2005. https://pubs.acs.org/doi/10.1021/cr030697h.

[7] Takahiro Matsuda, Runa Kawakami, Ryo Namba, Tasuku Nakajima, and Jian Ping Gong. Mechanoresponsive self-growing hydrogels inspired by muscle training. *Science*, 363(6426):504–508, February 2019. https://www.science.org/doi/10.1126/science.aau9533.

[8] Koji Kubota, Julong Jiang, Yuri Kamakura, Reon Hisazumi, Tsubura Endo, Daiyo Miura, Shotaro Kubo, Satoshi Maeda, and Hajime Ito. Using Mechanochemistry to Activate Commodity Plastics as Initiators for Radical Chain Reactions of Small Organic Molecules. *Journal of the American Chemical Society*, 146(1):1062–1070, 2024.

[9] Kenneth E. Chapman, Scott E. Sinclair, Daming Zhuang, Aviv Hassid, Leena P. Desai, and Christopher M. Waters. Cyclic mechanical strain increases reactive oxygen species production in pulmonary epithelial cells. *American Journal of Physiology-Lung Cellular and Molecular Physiology*, 289(5):L834–L841, November 2005. https://journals.physiology.org/doi/full/10.1152/ajplung.00069.2005.

[10] Christopher Zapp, Agnieszka Obarska-Kosinska, Benedikt Rennekamp, Markus Kurth, David M Hudson, Davide Mercadante, Uladzimir Barayeu, Tobias P Dick, Vasyl Denysenkov, Thomas Prisner, et al. Mechanoradicals in tensed tendon collagen as a source of oxidative stress. *Nature communications*, 11(1):1–8, 2020.

[11] Lizette Gil, Werner Siems, Birgit Mazurek, Johann Gross, Peter Schroeder, Peter Voss, and Tilman Grune. Age-associated analysis of oxidative stress parameters in human plasma and erythrocytes. *Free Radical Research*, 40(5):495–505, January 2006. https://doi.org/10.1080/10715760600592962.

[12] Kenneth B. Beckman and Bruce N. Ames. The Free Radical Theory of Aging Matures. *Physiological Reviews*, 78(2):547–581, April 1998. https://www.physiology.org/doi/10.1152/physrev.1998.78.2.547.

[13] D. Harman. Aging: A Theory Based on Free Radical and Radiation Chemistry. *Journal of Gerontology*, 11(3):298–300, July 1956. https://academic.oup.com/geronj/article-lookup/doi/10.1093/geronj/11.3.298.

[14] Umm-e-Ammara Warraich, Fatma Hussain, and Haroon Ur Rashid Kayani. Aging - Oxidative stress, antioxidants and computational modeling. *Heliyon*, 6(5):e04107, May 2020. https://linkinghub.elsevier.com/retrieve/pii/S2405844020309518.

[15] Fan Jiang, Kunlun Yin, Kun Wu, Mingmin Zhang, Shiqiang Wang, Heping Cheng, Zhou Zhou, and Bailong Xiao. The mechanosensitive Piezo1

channel mediates heart mechano-chemo transduction. *Nature Communications*, 12(1):869, February 2021. https://www.nature.com/articles/s41467-021-21178-4.

[16] Agnieszka Obarska-Kosinska, Benedikt Rennekamp, Aysecan Ünal, and Frauke Gräter. ColBuilder: A server to build collagen fibril models. *Biophysical Journal*, 120(17):3544–3549, 2021.

[17] Gloria A. Di Lullo, Shawn M. Sweeney, Jarmo Körkkö, Leena Ala-Kokko, and James D. San Antonio. Mapping the Ligand-binding Sites and Disease-associated Mutations on the Most Abundant Protein in the Human, Type I Collagen *. *Journal of Biological Chemistry*, 277(6):4223–4231, February 2002. https://www.jbc.org/article/S0021-9258(20)87534-6/abstract.

[18] P. Fratzl. Collagen: Structure and Mechanics, an Introduction. In Peter Fratzl, editor, *Collagen: Structure and Mechanics*, pages 1–13. Springer US, Boston, MA, 2008. https://doi.org/10.1007/978-0-387-73906-9_1.

[19] Peter R. Shewry, Arthur S. Tatham, and Allen J. Bailey. *Elastomeric Proteins: Structures, Biomechanical Properties, and Biological Roles*. Cambridge University Press, February 2010.

[20] David R Eyre, Mercedes A Paz, and Paul M Gallop. CROSS-LINKING IN COLLAGEN AND ELASTIN. *Annual Review of Biochemistry*, 53:717–748, July 1984.

[21] J. I. Thompson and J. T. Czernuszka. The Effect of Two Types of Cross-Linking on Some Mechanical Properties of Collagen. *Bio-Medical Materials and Engineering*, 5(1):37–48, January 1995. https://content.iospress.com/articles/bio-medical-materials-and-engineering/bme5-1-05.

[22] Bogdan I. Costescu and Frauke Gräter. Time-resolved force distribution analysis. *BMC Biophysics*, 6(1):5, May 2013. https://doi.org/10.1186/2046-1682-6-5.

[23] Benedikt Rennekamp, Fabian Kutzki, Agnieszka Obarska-Kosinska, Christopher Zapp, and Frauke Gräter. Hybrid Kinetic Monte Carlo/Molecular Dynamics Simulations of Bond Scissions in Proteins. *Journal of chemical theory and computation*, 16(1):553–563, 2020.

[24] B. J. Alder and T. E. Wainwright. Phase Transition for a Hard Sphere System. *The Journal of Chemical Physics*, 27(5):1208–1209, November 1957. https://doi.org/10.1063/1.1743957.

[25] J. Andrew McCammon, Bruce R. Gelin, and Martin Karplus. Dynamics of folded proteins. *Nature*, 267(5612):585–590, June 1977. https://www.nature.com/articles/267585a0.

[26] Scott A. Hollingsworth and Ron O. Dror. Molecular Dynamics Simulation for All. *Neuron*, 99(6):1129–1143, September 2018. https://linkinghub.elsevier.com/retrieve/pii/S0896627318306846.

[27] Romelia Salomon-Ferrer, Andreas W. Götz, Duncan Poole, Scott Le Grand, and Ross C. Walker. Routine Microsecond Molecular Dynamics Simulations with AMBER on GPUs. 2. Explicit Solvent Particle Mesh Ewald. *Journal of Chemical Theory and Computation*, 9(9):3878–3888, September 2013. https://doi.org/10.1021/ct400314y.

[28] R. W Hockney, S. P Goel, and J. W Eastwood. Quiet high-resolution computer models of a plasma. *Journal of Computational Physics*, 14(2):148–158, February 1974. https://www.sciencedirect.com/science/article/pii/0021999174900102.

[29] William C. Swope, Hans C. Andersen, Peter H. Berens, and Kent R. Wilson. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *The Journal of Chemical Physics*, 76(1):637–649, January 1982. https://doi.org/10.1063/1.442716.

[30] Norbert Attig, Kurt Binder, Helmut Grubmüller, and Kurt Kremer. *Computational Soft Matter: From Synthetic Polymers to Proteins. Lect: Lecture Notes*. Number 23 in NIC Series. NIC, Jülich, 2004.

[31] Jing Huang, Sarah Rauscher, Grzegorz Nawrocki, Ting Ran, Michael Feig, Bert L. de Groot, Helmut Grubmüller, and Alexander D. MacKerell. CHARMM36m: An improved force field for folded and intrinsically disordered proteins. *Nature Methods*, 14(1):71–73, January 2017. https://www.nature.com/articles/nmeth.4067.

[32] Robert B. Best and Gerhard Hummer. Optimized Molecular Dynamics Force Fields Applied to the Helix-Coil Transition of Polypeptides. *The Journal of*

Physical Chemistry B, 113(26):9004–9015, July 2009. https://doi.org/10.1021/jp901540t.

[33] Abil E. Aliev, Martin Kulke, Harmeet S. Khaneja, Vijay Chudasama, Tom D. Sheppard, and Rachel M. Lanigan. Motional timescale predictions by molecular dynamics simulations: Case study using proline and hydroxyproline sidechain dynamics. *Proteins: Structure, Function, and Bioinformatics*, 82(2):195–215, 2014. https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.24350.

[34] H. J. C. Berendsen, D. van der Spoel, and R. van Drunen. GROMACS: A message-passing parallel molecular dynamics implementation. *Computer Physics Communications*, 91(1):43–56, September 1995. https://www.sciencedirect.com/science/article/pii/001046559500042E.

[35] David Van Der Spoel, Erik Lindahl, Berk Hess, Gerrit Groenhof, Alan E. Mark, and Herman J. C. Berendsen. GROMACS: Fast, flexible, and free. *Journal of Computational Chemistry*, 26(16):1701–1718, December 2005. https://onlinelibrary.wiley.com/doi/10.1002/jcc.20291.

[36] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindahl. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 1–2:19–25, September 2015. https://www.sciencedirect.com/science/article/pii/S2352711015000059.

[37] Tom Darden, Darrin York, and Lee Pedersen. Particle mesh Ewald: An N·log(N) method for Ewald sums in large systems. *The Journal of Chemical Physics*, 98(12):10089–10092, June 1993. https://doi.org/10.1063/1.464397.

[38] Pedro E. M. Lopes, Jing Huang, Jihyun Shim, Yun Luo, Hui Li, Benoît Roux, and Alexander D. Jr. MacKerell. Polarizable Force Field for Peptides and Proteins Based on the Classical Drude Oscillator. *Journal of Chemical Theory and Computation*, 9(12):5430–5449, December 2013. https://doi.org/10.1021/ct400781b.

[39] Mark Abraham, Andrey Alekseenko, Vladimir Basov, Cathrine Bergh, Eliane Briand, Ania Brown, Mahesh Doijade, Giacomo Fiorin, Stefan Fleischmann, Sergey Gorelov, Gilles Gouaillardet, Alan Grey, M. Eric Irrgang, Farzaneh

Jalalypour, Joe Jordan, Carsten Kutzner, Justin A. Lemkul, Magnus Lundborg, Pascal Merz, Vedran Miletic, Dmitry Morozov, Julien Nabet, Szilard Pall, Andrea Pasquadibisceglie, Michele Pellegrino, Hubert Santuz, Roland Schulz, Tatiana Shugaeva, Alexey Shvetsov, Alessandra Villa, Sebastian Wingbermuehle, Berk Hess, and Erik Lindahl. GROMACS 2024.1 Manual. February 2024. https://zenodo.org/records/10721192.

[40] P. Hohenberg and W. Kohn. Inhomogeneous Electron Gas. *Physical Review*, 136(3B):B864–B871, November 1964. https://link.aps.org/doi/10.1103/PhysRev.136.B864.

[41] Maylis Orio, Dimitrios A. Pantazis, and Frank Neese. Density functional theory. *Photosynthesis Research*, 102(2):443–453, December 2009. https://doi.org/10.1007/s11120-009-9404-8.

[42] W. Kohn and L. J. Sham. Self-Consistent Equations Including Exchange and Correlation Effects. *Physical Review*, 140(4A):A1133–A1138, November 1965. https://link.aps.org/doi/10.1103/PhysRev.140.A1133.

[43] Christopher J. Cramer. *Essentials of Computational Chemistry: Theories and Models*. J. Wiley &sons, Chichester, 2nd ed edition, 2004.

[44] Albert Musaelian, Anders Johansson, Simon Batzner, and Boris Kozinsky. Scaling the leading accuracy of deep equivariant models to biomolecular simulations of realistic size. http://arxiv.org/abs/2304.10061, April 2023.

[45] Tsz Wai Ko, Jonas A Finkler, Stefan Goedecker, and Jörg Behler. A fourth-generation high-dimensional neural network potential with accurate electrostatics including non-local charge transfer. *Nature communications*, 12(1):1–11, 2021.

[46] Jörg Behler and Michele Parrinello. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Physical Review Letters*, 98(14):146401, April 2007. https://link.aps.org/doi/10.1103/PhysRevLett.98.146401.

[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. *Chemical science*, 8(4):3192–3203, 2017.

[48] Justin S Smith, Benjamin T Nebgen, Roman Zubatyuk, Nicholas Lubbers, Christian Devereux, Kipton Barros, Sergei Tretiak, Olexandr Isayev, and

Adrian E Roitberg. Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning. *Nature communications*, 10(1):1–8, 2019.

[49] Christian Devereux, Justin S. Smith, Kate K. Davis, Kipton Barros, Roman Zubatyuk, Olexandr Isayev, and Adrian E. Roitberg. Extending the Applicability of the ANI Deep Learning Molecular Potential to Sulfur and Halogens. *Journal of Chemical Theory and Computation*, 16(7):4192–4202, June 2020.

[50] Kristof T Schütt, Huziel E Sauceda, P-J Kindermans, Alexandre Tkatchenko, and K-R Müller. SchNet–A deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24):241722, 2018.

[51] Tristan Bereau, Denis Andrienko, and O. Anatole von Lilienfeld. Transferable Atomic Multipole Machine Learning Models for Small Organic Molecules. *Journal of Chemical Theory and Computation*, 11(7):3225–3233, July 2015. `https://doi.org/10.1021/acs.jctc.5b00301`.

[52] Kun Yao, John E. Herr, David W. Toth, Ryker Mckintyre, and John Parkhill. The TensorMol-0.1 model chemistry: A neural network augmented with long-range physics. *Chemical Science*, 9(8):2261–2269, 2018. `https://pubs.rsc.org/en/content/articlelanding/2018/sc/c7sc04934j`.

[53] Oliver T. Unke and Markus Meuwly. PhysNet: A Neural Network for Predicting Energies, Forces, Dipole Moments, and Partial Charges. *Journal of Chemical Theory and Computation*, 15(6):3678–3693, June 2019. `https://doi.org/10.1021/acs.jctc.9b00181`.

[54] Somayeh Faraji, S. Alireza Ghasemi, Samare Rostami, Robabe Rasoulkhani, Bastian Schaefer, Stefan Goedecker, and Maximilian Amsler. High accuracy and transferability of a neural network potential through charge equilibration for calcium fluoride. *Physical Review B*, 95(10):104105, March 2017. `https://link.aps.org/doi/10.1103/PhysRevB.95.104105`.

[55] S. Alireza Ghasemi, Albert Hofstetter, Santanu Saha, and Stefan Goedecker. Interatomic potentials for ionic systems with density functional accuracy based on charge densities obtained by a neural network. *Physical Review B*, 92(4):045131, July 2015. `https://link.aps.org/doi/10.1103/PhysRevB.92.045131`.

[56] Roman Zubatyuk, Justin S. Smith, Benjamin T. Nebgen, Sergei Tretiak, and Olexandr Isayev. Teaching a neural network to attach and detach electrons from molecules. *Nature Communications*, 12(1), August 2021.

[57] Haoyan Huo and Matthias Rupp. Unified representation of molecules and crystals for machine learning. *arXiv preprint arXiv:1704.06439*, 2017.

[58] Albert P Bartók, Risi Kondor, and Gábor Csányi. On representing chemical environments. *Physical Review B*, 87(18):184115, 2013.

[59] Jörg Behler. Perspective: Machine learning potentials for atomistic simulations. *The Journal of chemical physics*, 145(17):170901, 2016.

[60] Michael J. Willatt, Felix Musil, and Michele Ceriotti. Atom-density representations for machine learning. *arXiv prepring arXiv:1807.00408v3*, July 2018.

[61] Ralf Drautz. Atomic cluster expansion for accurate and transferable interatomic potentials. *Physical Review B*, 99(1):014104, January 2019.

[62] Geneviève Dusson, Markus Bachmayr, Gábor Csányi, Ralf Drautz, Simon Etter, Cas van der Oord, and Christoph Ortner. Atomic cluster expansion: Completeness, efficiency and stability. *Journal of Computational Physics*, 454:110946, April 2022. `https://www.sciencedirect.com/science/article/pii/S0021999122000080`.

[63] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds. `http://arxiv.org/abs/1802.08219`, May 2018.

[64] Brandon Anderson, Truong Son Hy, and Risi Kondor. Cormorant: Covariant Molecular Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. `https://proceedings.neurips.cc/paper/2019/hash/03573b32b2746e6e8ca98b9123f2249b-Abstract.html`.

[65] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco S Cohen. 3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. `https://proceedings.neurips.cc/paper_files/paper/2018/hash/488e4104520c6aab692863cc1dba45af-Abstract.html`.

[66] Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P. Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E. Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13(1), May 2022.

[67] Kristof T. Schütt, Oliver T. Unke, and Michael Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. `https://arxiv.org/abs/2102.03150v4`, February 2021.

[68] Ilyes Batatia, Simon Batzner, Dávid Péter Kovács, Albert Musaelian, Gregor N. C. Simm, Ralf Drautz, Christoph Ortner, Boris Kozinsky, and Gábor Csányi. The Design Space of E(3)-Equivariant Atom-Centered Interatomic Potentials. `http://arxiv.org/abs/2205.06643`, November 2022.

[69] Ilyes Batatia, David P Kovacs, Gregor Simm, Christoph Ortner, and Gabor Csanyi. MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 11423–11436. Curran Associates, Inc., 2022. `https://proceedings.neurips.cc/paper_files/paper/2022/file/4a36c3c51af11ed9f34615b81edb5bbc-Paper-Conference.pdf`.

[70] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. `https://proceedings.neurips.cc/paper_files/paper/2015/hash/f9be311e65d81a9ad8150a60844bb94c-Abstract.html`.

[71] Dirk P. Kroese, Tim Brereton, Thomas Taimre, and Zdravko I. Botev. Why the Monte Carlo method is so important today. *WIREs Computational Statistics*, 6(6):386–392, November 2014. `https://wires.onlinelibrary.wiley.com/doi/10.1002/wics.1314`.

[72] Babak Sadigh, Paul Erhart, Alexander Stukowski, Alfredo Caro, Enrique Martinez, and Luis Zepeda-Ruiz. Scalable parallel Monte Carlo algorithm for atomistic simulations of precipitation in alloys. *Physical Review B*, 85(18):184203, May 2012. `https://link.aps.org/doi/10.1103/PhysRevB.85.184203`.

[73] W. M. Young and E. W. Elcock. Monte Carlo studies of vacancy migration in binary ordered alloys: I. *Proceedings of the Physical Society*, 89(3):735, November 1966. https://dx.doi.org/10.1088/0370-1328/89/3/329.

[74] A. Prados, J. J. Brey, and B. Sánchez-Rey. A dynamical monte carlo algorithm for master equations with time-dependent transition rates. *Journal of Statistical Physics*, 89(3):709–734, November 1997. https://doi.org/10.1007/BF02765541.

[75] A. B. Bortz, M. H. Kalos, and J. L. Lebowitz. A new algorithm for Monte Carlo simulation of Ising spin systems. *Journal of Computational Physics*, 17(1):10–18, January 1975. https://www.sciencedirect.com/science/article/pii/0021999175900601.

[76] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, June 1953. https://doi.org/10.1063/1.1699114.

[77] Tomas Opplestrup, Vasily V. Bulatov, George H. Gilmer, Malvin H. Kalos, and Babak Sadigh. First-Passage Monte Carlo Algorithm: Diffusion without All the Hops. *Physical Review Letters*, 97(23):230602, December 2006. https://link.aps.org/doi/10.1103/PhysRevLett.97.230602.

[78] Aleksandar Donev, Vasily V. Bulatov, Tomas Oppelstrup, George H. Gilmer, Babak Sadigh, and Malvin H. Kalos. A First-Passage Kinetic Monte Carlo algorithm for complex diffusion–reaction systems. *Journal of Computational Physics*, 229(9):3214–3236, May 2010. https://linkinghub.elsevier.com/retrieve/pii/S0021999110000057.

[79] A. Violi, A. Kubota, T. N. Truong, W. J. Pitz, C. K. Westbrook, and A. F. Sarofim. A fully integrated kinetic monte carlo/molecular dynamics approach for the simulation of soot precursor growth. *Proceedings of the Combustion Institute*, 29(2):2343–2349, January 2002. https://www.sciencedirect.com/science/article/pii/S1540748902802851.

[80] Emanuel Karl Peter and Joan-Emma Shea. A hybrid MD-kMC algorithm for folding proteins in explicit solvent. *Physical Chemistry Chemical Physics*, 16(14):6430–6440, March 2014. https://pubs.rsc.org/en/content/articlelanding/2014/cp/c3cp55251a.

[81] Jacob P. Tavenner, Mikhail I. Mendelev, and John W. Lawson. Molecular dynamics based kinetic Monte Carlo simulation for accelerated diffusion. *Computational Materials Science*, 218:111929, February 2023. https://www.sciencedirect.com/science/article/pii/S0927025622006401.

[82] Margaritis Voliotis, Philipp Thomas, Ramon Grima, and Clive G. Bowsher. Stochastic Simulation of Biomolecular Networks in Dynamic Environments. *PLOS Computational Biology*, 12(6):e1004923, January 2016. https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1004923.

[83] Kai Riedmiller, Patrick Reiser, Elizaveta Bobkova, Kiril Maltsev, Ganna Gryn'ova, Pascal Friederich, and Frauke Gräter. Substituting density functional theory in reaction barrier calculations for hydrogen atom transfer in proteins. *Chemical Science*, 15:2518–2527, 2024. http://dx.doi.org/10.1039/D3SC03922F.

[84] Wojtek Treyde, Kai Riedmiller, and Frauke Gräter. Bond dissociation energies of X–H bonds in proteins. *RSC Advances*, 12(53):34557–34564, 2022.

[85] Benedikt Rennekamp, Christoph Karfusehr, Markus Kurth, Aysecan Ünal, Debora Monego, Kai Riedmiller, Ganna Gryn'ova, David M. Hudson, and Frauke Gräter. Collagen breaks at weak sacrificial bonds taming its mechanoradicals. *Nature Communications*, 14(1):2075, April 2023. https://www.nature.com/articles/s41467-023-37726-z.

[86] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, B. Mennucci, G. A. Petersson, H. Nakatsuji, M. Caricato, X. Li, H. P. Hratchian, A. F. Izmaylov, J. Bloino, G. Zheng, J. L. Sonnenberg, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. Bearpark, J. J. Heyd, E. Brothers, K. N. Kudin, V. N. Staroverov, R. Kobayashi, J. Normand, K. Raghavachari, A. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, N. Rega, J. M. Millam, M. Klene, J. E. Knox, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, R. L. Martin, K. Morokuma, V. G. Zakrzewski, G. A. Voth, P. Salvador, J. J. Dannenberg, S. Dapprich, A. D. Daniels, Ö. Farkas, J. B. Foresman, J. V. Ortiz, J. Cioslowski, and D. J. Fox. Gaussian 09 Revision D.01. GaussianInc.WallingfordCT2013.

[87] A Daniel Boese and Jan ML Martin. Development of density functionals for thermochemical kinetics. *The Journal of chemical physics*, 121(8):3405–3416, 2004.

[88] Martin Korth and Stefan Grimme. "Mindless" DFT benchmarking. *Journal of chemical theory and computation*, 5(4):993–1003, 2009.

[89] Yan Zhao and Donald G. Truhlar. How Well Can New-Generation Density Functionals Describe the Energetics of Bond-Dissociation Reactions Producing Radicals? *The Journal of Physical Chemistry A*, 112(6):1095–1099, February 2008. https://doi.org/10.1021/jp7109127.

[90] Giuseppe Felice Mangiatordi, Eric Brémond, and Carlo Adamo. DFT and Proton Transfer Reactions: A Benchmark Study on Structure and Kinetics. *Journal of Chemical Theory and Computation*, 8(9):3082–3088, August 2012.

[91] James M Mayer. Understanding hydrogen atom transfer: From bond strengths to Marcus theory. *Accounts of Chemical Research*, 44(1):36–46, 2011.

[92] J. N. Bronsted. Acid and Basic Catalysis. *Chemical Reviews*, 5(3):231–338, October 1928. https://pubs.acs.org/doi/abs/10.1021/cr60019a001.

[93] Ronald Percy Bell. The theory of reactions involving proton transfers. *Proceedings of the Royal Society of London. Series A - Mathematical and Physical Sciences*, 154(882):414–429, April 1936. https://royalsocietypublishing.org/doi/10.1098/rspa.1936.0060.

[94] M. G. Evans and M. Polanyi. Inertia and driving force of chemical reactions. *Transactions of the Faraday Society*, 34(0):11–24, January 1938. https://pubs.rsc.org/en/content/articlelanding/1938/tf/tf9383400011.

[95] Shantanu Roy, Stefan Goedecker, and Vladimir Hellmann. Bell-Evans-Polanyi principle for molecular dynamics trajectories and its implications for global optimization. *Physical Review E*, 77(5):056707, May 2008. https://link.aps.org/doi/10.1103/PhysRevE.77.056707.

[96] Jeffrey J Warren and James M Mayer. Predicting organic hydrogen atom transfer rate constants using the Marcus cross relation. *Proceedings of the National Academy of Sciences*, 107(12):5282–5287, 2010.

[97] Rudolph A. Marcus. Theoretical relations among rate constants, barriers, and Broensted slopes of chemical reactions. *The Journal of Physical Chem-*

istry, 72(3):891–899, March 1968. https://pubs.acs.org/doi/abs/10.1021/j100849a019.

[98] Ra A. Marcus and Norman Sutin. Electron transfer in chemistry and biology. *Biochimica et Biophysica Acta*, 811(3):265–322, August 1985. https://zenodo.org/records/1258475.

[99] Johnny Hioe and Hendrik Zipse. Radical Stability—Thermochemical Aspects. In Chryssostomos Chatgilialoglu and Armido Studer, editors, *Encyclopedia of Radicals in Chemistry, Biology and Materials*. Wiley, 1 edition, January 2012. https://onlinelibrary.wiley.com/doi/10.1002/9781119953678.rad012.

[100] Bun Chan, Jia Deng, and Leo Radom. G4(MP2)-6X: A Cost-Effective Improvement to G4(MP2). *Journal of Chemical Theory and Computation*, 7(1):112–120, January 2011. https://doi.org/10.1021/ct100542x.

[101] Ekaterina I. Izgorodina, David R. B. Brittain, Jennifer L. Hodgson, Elizabeth H. Krenske, Ching Yeh Lin, Mansoor Namazian, and Michelle L. Coote. Should Contemporary Density Functional Theory Methods Be Used to Study the Thermodynamics of Radical Reactions? *The Journal of Physical Chemistry A*, 111(42):10754–10768, October 2007. https://doi.org/10.1021/jp075837w.

[102] Yan Zhao and Donald G. Truhlar. Density Functionals with Broad Applicability in Chemistry. *Accounts of Chemical Research*, 41(2):157–167, February 2008. https://doi.org/10.1021/ar700111a.

[103] Stefan Grimme. Improved second-order Møller–Plesset perturbation theory by separate scaling of parallel- and antiparallel-spin pair correlation energies. *The Journal of Chemical Physics*, 118(20):9095–9102, May 2003. https://doi.org/10.1063/1.1569242.

[104] Yu-Ran Luo and Yu-Ran Luo. *Comprehensive Handbook of Chemical Bond Energies*. CRC Press, Boca Raton, 2007.

[105] Heinz G. Viehe, Zdenek Janousek, Robert Merenyi, and Lucien Stella. The captodative effect. *Accounts of Chemical Research*, 18(5):148–154, May 1985. https://doi.org/10.1021/ar00113a004.

[106] Benjamin N Moore and Ryan R Julian. Dissociation energies of X–H bonds in amino acids. *Physical Chemistry Chemical Physics*, 14(9):3148–3154, 2012.

[107] Johnny Hioe, Marianne Mosch, David M. Smith, and Hendrik Zipse. Dissociation energies of Cα–H bonds in amino acids – a re-examination. *RSC Advances*, 3(30):12403–12408, July 2013. https://pubs.rsc.org/en/content/articlelanding/2013/ra/c3ra42115e.

[108] Johnny Hioe, Gökcen Savasci, Harald Brand, and Hendrik Zipse. The Stability of Cα Peptide Radicals: Why Glycyl Radical Enzymes? *Chemistry – A European Journal*, 17(13):3781–3789, 2011. https://onlinelibrary.wiley.com/doi/abs/10.1002/chem.201002620.

[109] Alexander A. Morgan and Edward Rubenstein. Proline: The Distribution, Frequency, Positioning, and Common Functional Roles of Proline and Polyproline Sequences in the Human Proteome. *PLOS ONE*, 8(1):e53785, January 2013. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0053785.

[110] Jan Norberg, Nicolas Foloppe, and Lennart Nilsson. Intrinsic Relative Stabilities of the Neutral Tautomers of Arginine Side-Chain Models. *Journal of Chemical Theory and Computation*, 1(5):986–993, September 2005. https://doi.org/10.1021/ct049849m.

[111] Yongna Yuan, Matthew J. L. Mills, Paul L. A. Popelier, and Frank Jensen. Comprehensive Analysis of Energy Minima of the 20 Natural Amino Acids. *The Journal of Physical Chemistry A*, 118(36):7876–7891, September 2014. https://doi.org/10.1021/jp503460m.

[112] Bun Chan, Amir Karton, Christopher J. Easton, and Leo Radom. α-Hydrogen Abstraction by •OH and •SH Radicals from Amino Acids and Their Peptide Derivatives. *Journal of Chemical Theory and Computation*, 12(4):1606–1613, April 2016. https://doi.org/10.1021/acs.jctc.6b00007.

[113] Johnny Hioe and Hendrik Zipse. Radicals in enzymatic catalysis—a thermodynamic perspective. *Faraday Discussions*, 145(0):301–313, January 2010. https://pubs.rsc.org/en/content/articlelanding/2010/fd/b907121k.

[114] Maria Victoria Roux, Concepción Foces-Foces, Rafael Notario, Manuel A. V. Ribeiro da Silva, Maria das Dores M. C. Ribeiro da Silva, Ana Filipa L. O. M. Santos, and Eusebio Juaristi. Experimental and Computational Thermochemical Study of Sulfur-Containing Amino Acids: L-Cysteine, l-Cystine,

and l-Cysteine-Derived Radicals. S-S, S-H, and C-S Bond Dissociation Enthalpies. *The Journal of Physical Chemistry B*, 114(32):10530–10540, August 2010. https://doi.org/10.1021/jp1025637.

[115] Marcus D. Hanwell, Donald E. Curtis, David C. Lonie, Tim Vandermeersch, Eva Zurek, and Geoffrey R. Hutchison. Avogadro: An advanced semantic chemical editor, visualization, and analysis platform. *Journal of Cheminformatics*, 4(1):17, August 2012. https://doi.org/10.1186/1758-2946-4-17.

[116] Lauri Himanen, Marc O. J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Yashasvi S. Ranawat, David Z. Gao, Patrick Rinke, and Adam S. Foster. DScribe: Library of descriptors for machine learning in materials science. *Computer Physics Communications*, 247:106949, 2020. https://doi.org/10.1016/j.cpc.2019.106949.

[117] Patrick Reiser, Andre Eberhard, and Pascal Friederich. Graph neural networks in TensorFlow-Keras with RaggedTensor representation (kgcnn). *Software Impacts*, page 100095, 2021. https://www.sciencedirect.com/science/article/pii/S266596382100035X.

[118] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for Activation Functions. *arXiv prepring arXiv:1710.05941v2*, October 2017.

[119] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv prepring arXiv:1412.6980v9*, December 2014.

[120] Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. KerasTuner. https://github.com/keras-team/keras-tuner, 2019.

[121] Jannik Buhr, Florian Franz, and Frauke Gräter. Intrinsically disordered region of talin's FERM domain functions as an initial PIP2 recognition site. *Biophysical Journal*, 122(7):1277–1286, April 2023. https://linkinghub.elsevier.com/retrieve/pii/S0006349523001261.

[122] Massimiliano Bonomi, Davide Branduardi, Giovanni Bussi, Carlo Camilloni, Davide Provasi, Paolo Raiteri, Davide Donadio, Fabrizio Marinelli, Fabio Pietrucci, Ricardo A. Broglia, and Michele Parrinello. PLUMED: A portable plugin for free-energy calculations with molecular dynamics. *Computer Physics Communications*, 180(10):1961–1972, October 2009. https://www.sciencedirect.com/science/article/pii/S001046550900157X.

[123] Massimiliano Bonomi, Giovanni Bussi, Carlo Camilloni, Gareth A. Tribello, Pavel Banáš, Alessandro Barducci, Mattia Bernetti, Peter G. Bolhuis, Sandro Bottaro, Davide Branduardi, Riccardo Capelli, Paolo Carloni, Michele Ceriotti, Andrea Cesari, Haochuan Chen, Wei Chen, Francesco Colizzi, Sandip De, Marco De La Pierre, Davide Donadio, Viktor Drobot, Bernd Ensing, Andrew L. Ferguson, Marta Filizola, James S. Fraser, Haohao Fu, Piero Gasparotto, Francesco Luigi Gervasio, Federico Giberti, Alejandro Gil-Ley, Toni Giorgino, Gabriella T. Heller, Glen M. Hocky, Marcella Iannuzzi, Michele Invernizzi, Kim E. Jelfs, Alexander Jussupow, Evgeny Kirilin, Alessandro Laio, Vittorio Limongelli, Kresten Lindorff-Larsen, Thomas Löhr, Fabrizio Marinelli, Layla Martin-Samos, Matteo Masetti, Ralf Meyer, Angelos Michaelides, Carla Molteni, Tetsuya Morishita, Marco Nava, Cristina Paissoni, Elena Papaleo, Michele Parrinello, Jim Pfaendtner, Pablo Piaggi, GiovanniMaria Piccini, Adriana Pietropaolo, Fabio Pietrucci, Silvio Pipolo, Davide Provasi, David Quigley, Paolo Raiteri, Stefano Raniolo, Jakub Rydzewski, Matteo Salvalaglio, Gabriele Cesare Sosso, Vojtěch Spiwok, Jiří Šponer, David W. H. Swenson, Pratyush Tiwary, Omar Valsson, Michele Vendruscolo, Gregory A. Voth, Andrew White, and The PLUMED consortium. Promoting transparency and reproducibility in enhanced molecular simulations. *Nature Methods*, 16(8):670–673, August 2019. `https://www.nature.com/articles/s41592-019-0506-8`.

[124] Gareth A. Tribello, Massimiliano Bonomi, Davide Branduardi, Carlo Camilloni, and Giovanni Bussi. PLUMED 2: New feathers for an old bird. *Computer Physics Communications*, 185(2):604–613, February 2014. `https://www.sciencedirect.com/science/article/pii/S0010465513003196`.

[125] Leif Seute, Eric Hartmann, Jan Stühmer, and Frauke Gräter. Grappa – A Machine Learned Molecular Mechanics Force Field. `http://arxiv.org/abs/2404.00050`, March 2024.

[126] Kai Riedmiller, Jannik Buhr, and Eric Hartmann. KIMMDY. `https://hits-mbm-dev.github.io/kimmdy/`.

[127] J.J. Allaire, Charles Teague, Carlos Scheidegger, Yihui Xie, and Christophe Dervieux. Quarto. `https://github.com/quarto-dev/quarto-cli`, February 2024.

[128] Style guide - numpydoc v1.7.0rc0.dev0 Manual. `https://numpydoc.readthedocs.io/en/latest/format.html#common-rest-concepts`.

[129] Conventional-commits/conventionalcommits.org. https://github.com/conventional-commits/conventionalcommits.org, March 2024.

[130] Łukasz Langa and contributors to Black. Black: The uncompromising Python code formatter. https://github.com/psf/black, March 2024.

[131] tox development team. Tox. https://github.com/tox-dev/tox, March 2024.

[132] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, and David Cournapeau. Scikit-learn: Machine Learning in Python. *MACHINE LEARNING IN PYTHON*.

[133] Mark van der Wilk, Vincent Dutordoir, S. T. John, Artem Artemev, Vincent Adam, and James Hensman. A Framework for Interdomain and Multioutput Gaussian Processes. http://arxiv.org/abs/2003.01115, March 2020.

[134] Chi Chen, Weike Ye, Yunxing Zuo, Chen Zheng, and Shyue Ping Ong. Graph Networks as a Universal Machine Learning Framework for Molecules and Crystals. *Chemistry of Materials*, 31(9):3564–3572, May 2019. https://doi.org/10.1021/acs.chemmater.9b01294.

[135] Johannes Gasteiger, Shankari Giri, Johannes T. Margraf, and Stephan Günnemann. Fast and Uncertainty-Aware Directional Message Passing for Non-Equilibrium Molecules. http://arxiv.org/abs/2011.14115, April 2022.

[136] Hélène Decornez and Sharon Hammes-Schiffer. Model Proton-Coupled Electron Transfer Reactions in Solution: Predictions of Rates, Mechanisms, and Kinetic Isotope Effects. *The Journal of Physical Chemistry A*, 104(41):9370–9384, October 2000. https://doi.org/10.1021/jp001967s.

[137] R. I. Cukier. Theory and simulation of proton-coupled electron transfer, hydrogen-atom transfer, and proton translocation in proteins. *Biochimica et Biophysica Acta (BBA) - Bioenergetics*, 1655:37–44, April 2004. https://www.sciencedirect.com/science/article/pii/S000527280300197X.

[138] Thomas C. O'Connor, Jan Andzelm, and Mark O. Robbins. AIREBO-M: A reactive model for hydrocarbons at extreme pressures. *The Journal of Chemical Physics*, 142(2):024903, January 2015.

[139] Jianguo Yu, Susan B. Sinnott, and Simon R. Phillpot. Charge optimized many-body potential for the $\mathrm{Si}/{\mathrm{SiO}}_{2}$ system. *Physical Review B*, 75(8):085311, February 2007. https://link.aps.org/doi/10.1103/PhysRevB.75.085311.

[140] Thomas P Senftle, Sungwook Hong, Md Mahbubul Islam, Sudhir B Kylasa, Yuanxia Zheng, Yun Kyung Shin, Chad Junkermeier, Roman Engel-Herbert, Michael J Janik, Hasan Metin Aktulga, et al. The ReaxFF reactive force-field: Development, applications and future directions. *npj Computational Materials*, 2(1):1–14, 2016.

[141] Adri CT Van Duin, Siddharth Dasgupta, Francois Lorant, and William A Goddard. ReaxFF: A reactive force field for hydrocarbons. *The Journal of Physical Chemistry A*, 105(41):9396–9409, 2001.

[142] Itai Leven, Hongxia Hao, Songchen Tan, Xingyi Guan, Katheryn A. Penrod, Dooman Akbarian, Benjamin Evangelisti, Md Jamil Hossain, Md Mahbubul Islam, Jason P. Koski, Stan Moore, Hasan Metin Aktulga, Adri C. T. Van Duin, and Teresa Head-Gordon. Recent Advances for Improving the Accuracy, Transferability, and Efficiency of Reactive Force Fields. *Journal of Chemical Theory and Computation*, 17(6):3237–3251, June 2021. https://pubs.acs.org/doi/10.1021/acs.jctc.1c00118.

[143] Pablo Herrera-Nieto, Adrià Pérez, and Gianni De Fabritiis. Characterization of partially ordered states in the intrinsically disordered N-terminal domain of p53 using millisecond molecular dynamics simulations. *Scientific Reports*, 10(1):12402, July 2020. https://www.nature.com/articles/s41598-020-69322-2.