Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Chemistry

Kristine Inga Qing Laurila

A multiscale modelling of 2-aminoethanol (MEA) within the Python-ecosystem

Master's thesis in Applied Theoretical Chemistry
Supervisor: Per-Olof Åstrand
May 2024

NTNU
Norwegian University of
Science and Technology

Kristine Inga Qing Laurila

A multiscale modelling of 2-aminoethanol (MEA) within the Python-ecosystem

Master's thesis in Applied Theoretical Chemistry
Supervisor: Per-Olof Åstrand
May 2024

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Chemistry

NTNU
Norwegian University of
Science and Technology

Abstract

A multiscale modelling approach has been employed to investigate the benchmark molecule in post-combustion carbon dioxide capture (PCC), 2-aminoethanol (MEA), with the hope to increase the insight in the conformer stability, interconnectivity and corresponding reactivity. All calculations have been carried out within a Python-ecosystem, utilising the quantum chemistry software *Python Simulations of Chemistry Framework* (PySCF) for *Density Functional Theory* (DFT) calculations using the *Unrestricted Kohn-Sham* approach. 25 conformers of MEA have been identified at COSMO/B3LYP/aug-cc-pVDZ level of theory, with five conformers exhibiting intramolecular hydrogen bonding (HB), resulting in increased stability. However, two conformers that has not been determined in resent work, could not be fully ascertained. Transition state searches has been performed for 23 conformers, revealing interconversion barrier ranging from 0.66 to 5.15 kcal mol^{-1}. Notably, interconversions between states exhibiting HB, were found to have the lowest barrier, while interconversion of the most stable state, were found to exhibit the highest barrier of 5.15 and 4.77 kcal mol^{-1}. Rate constants have been calculated using transition state theory (TST), and the result yielded rate constants ranging from 4.69×10^{8} to 9.46×10^{14} s^{-1}. These has further been used in kinetic Monte Carlo (kMC) simulations, indicating that the states with HB were most populated. However, not all state were possible to consider, so the results from the simulations does not reflect the entire conformational diversity to its full extent. Lastly, three conformers have been examined at COSMO/B3LYP/cc-pVDZ level of theory, to determine whether the reacting conformer affects the kinetics of the reaction. While the results could point in a direction towards this, computational issues prevent conclusive findings, and further exploration is required to fully understand the impact of conformers on reaction kinetics.

Sammendrag

En multiskala modelleringsmetode har blitt brukt for å studere referansemolekylet, 2-aminoetanol (MEA), i etter-forbrenning fangst av karbondioksid (PCC), med håp om å øke innsikten i konformasjonsstabilitet, konformasjonskonnektivitet og tilsvarende reaktivitet. Samtlige beregninger har blitt utført innenfor et Python-økosystem, der kvantekjemi-programvaren *Python Simulations of Chemistry Framework* (PySCF) har vært benyttet for å gjennomføre tetthetsfunksjonalteori (DFT) beregninger ved hjelp av den ikke-restriktive Kohn-Sham (UKS) tilnærmingen. 25 konformasjoner av MEA har blitt identifisert ved COSMO/B3LYP/aug-cc-pVDZ nivået av teori, hvor fem konformasjoner innehar intramolekylære hydrogenbindinger (HB), hvilket resulterer i økt stabilitet. To av konformasjonene, som imidlertid ikke har blitt påvist i nyere litteratur, kunne ikke fult ut bekreftes. Overgangstilstand-søk (TS) har blitt utført for 23 konformasjoner, og avslørte at interkonverteringsbarrierer som spenner fra 0.66 til 5.15 kcal mol^{-1}. Det fremtrer at interkonverteringer mellom tilstander som innehar HB, hadde den laveste barrieren, mens interkonvertering av den mest stabile tilstanden, hadde den høyeste barrieren på 5.15 og 4.77 kcal mol^{-1}. Hastighetskonstanter har blitt beregnet ved hjelp av overgangstilstandteori (TST), og resultatene ga hastighetskonstanter som spenner fra 4.69×10^{8} til 9.46×10^{14} s^{-1}. Disse har videre blitt brukt i kinetisk Monte Carlo (kMC) simuleringer, som indikerer at tilstandene med HB var preget av en høyere andel av den totale populasjonen. Det var imidlertid ikke mulig å betrakte samtlige tilstander, dermed slik gjenspeiler ikke resultatene fra simuleringene hele konformasjonsmangfoldet til det fulle. Til slutt har tre konformasjoner blitt undersøkt for å avgjøre om hvorvidt den reagerende konformasjonen påvirker reaksjonskinetikken ved COSMO/B3LYP/cc-pVDZ nivået av teori. Til tross for at resultatene peker i retning av dette, forhindrer beregningsproblemer faktiske resultater, og ytterligere undersøkelse er nødvendig for å fult ut forstå konformasjonens innvirkning på kinetikken.

Preface

This thesis was written as a final fulfilment of the two year master program, Master of Science in Chemistry (MSCHEM), with specialisation in Applied Theoretical Chemistry at Norwegian University of Science and Technology. The work in this thesis was carried out during the final academic year from the autumn of 2023 to the spring of 2024.

Acknowledgements

I wish to thank my supervisor Per-Olof Åstrand for our weekly meetings where he has challenged me and guided me throughout this work. I would also like to thank Eirik Falck da Silva from SINTEF Industy, for providing insight pertaining to the current status of research within the field of solvent carbon capture.

I would further like thank my partner Erik, for being amusing and caring as always, but also for the extra cores. Lastly, I wish to express my gratitude to the rest of my family and friends for their infinite encouragement and support, which I consider invaluable.

Enjoy reading!

Kristine Inga Qing Laurila

Table of Contents

List of Figures

List of Listings

List of Tables

1 Introduction

Ever since 1930s has alkanolamines played a central role, serving as an absorbent in studies concerning carbon dioxide (CO_2) capturing[20]. 2-aminoethanol (MEA) is the most extensively studied molecule in post-combustion carbon dioxide capture (PCC) by amine absorption and is considered as a benchmark solvent owing to its lower costs, high absorption capability and fast absorption rate. The molecule is bifunctional, containing a primary alcohol and a primary amine and a pKa = 9.50 at 25℃[11]. Moreover, MEA is corrosive and may cause irritation when in direct contact with the skin depending on the concentration and duration of exposure[1]. Furthermore, it is not considered volatile[16, 67].

Another area in which MEA is seen is in phospholipids *phosphatidylethanolamines* (cephalin[28]), known to be the second most abundant glycerophospholipid in eucaryotic cell membranes[21, 54]. Moreover, they are widely used as a complexing agents, where they act as mono- or bidentate agents[52]. Due to its widespread application, understanding the conformational interconvertions of MEA is of considerable interest. Apart form previous work, all conformers has been desirable to map. Despite having isomers, its structural importance may have an important impact pertaining

Facilities that utilise PCC-technologies are typically power plants where CO_2 is a flute gas. A great advantage is that these technologies can be implemented in existing infrastructures, know as *retrofitting* without requiring additional reconstructions[30, 89]. The mechanism of capture CO_2 involves a reversible chemical reaction. A solvent serves as a medium to capture and release CO_2, which can then be collected. In the absorber at relative low temperature (~ 50°C), a chemical bond between CO_2 and the amine is formed. By increasing the temperature (~ 120°C) in the desorber, the CO_2 can be released[67]. However, a major limitation is that the need to increase the temperature in order to regenerate the amines leads to an increased operational cost[2].

All amines undergo degradation to some extent, and some degrade more rapidly than others. Primary amines tend to degrade faster than secondary and tertiary due to higher degree of free rotation. Secondary and tertiary amines that has bulky groups adjacent to the amine group, introduce steric effects. These have been shown to adversely affect the reaction kinetics of the the amine-CO_2 reaction[3, 63]. Furthermore, alkalinity has been shown to play a role in reaction kinetics by increasing the affinity for CO_2. Particularly for secondary amines[3, 63, 70].

Studies over the past years have further shown that MEA (among many other amines) degrade into undesirable compounds, and can be further catalysed by dissolved metals[14, 25]. Subsequent degradation products will further contribute to corrosion[61]. The chemical reaction are irreversible chemical processes leading to amine loss. Solvent degradation is a serious issue, as it results in economic losses and operating problems. Despite other solvent and solvent mixtures has taken its place in research questions on finding the optimal solvent, MEA remains of interest, because its degradation path resembles other of primary amines in terms of degradation products. Also, conducting experiments with it is advantageous due to its relative fast kinetics. The transferability makes MEA studies useful in general. However, kinetic studies seems to be scares pertaining to the conformational point of view. Only one work has explicitly stated which conformation of MEA that undergoes a reacction[83]. For that reason this thesis seeks to gain insight in conformational stability of MEA.

Density Functional Theory (DFT) has been used to study the stability of MEA in the quantum chemistry software, *Python Simulations of Chemistry Framework*, *PySCF*[72]. All conformers reported are has been obtained using the *Unrestricted Kohn-Sham* (UKS) approach in PySCF. Furthermore, the B3LYP functional with aug-cc-pVDZ basis set were used to determine the *reference* structures in gas phase. Subsequently, solvent effects were accounted for by using the continuum model COSMO, which for that matter is the is the only implemented solvent model in *PySCF* as to this date. The solvent parameter for water that was used is the dielectric constant. All conformer calculations are further reported at standard condition, i.e. 298.15K and 101 325 Pa.

Transition State Theory has been used to obtain the rate constants needed to carry out kinetic Monte Carlo (kMC) simulations. Studying the conformer interconversion in a kinetic Monte Carlo simulation has not been preformed so far. Based on the findings form the simulation, the formation of HEGly will be tested with different conformers.

In this work, a significant amount of time that was not spent on the chemistry of MEA was dedicated to developing efficient, versatile and user-friendly cods for newcomers to PySCF. Working within the Python ecosystem offers numerous advantages, and among the greatest is how conveniently other packages can very easily be integrated. By focusing on clarity and accessibility, the code can hopefully become useful for those new to PySCF, or others that might be curios.

This thesis has been constructed as follows: Chapter 2 entails theory on DFT with related aspects, transition state theory and how to derive the rate constants and lastly a short introduction to kinetic Monte Carlo. Chapter 3, aims to describe the approach used to obtain the results, chapter 4 contains the results and discussion and chapter 5 concludes the work and gives some thought on the outlook. In addition, the appendices can be found at the end.

2 | Theoretical background

The following chapter entails the appropriate theory needed to grasp the subject matter. An introduction of *electron structure theory* that underlie DFT will be provided in the beginning, followed by DFT theory and related aspects in molecular modelling. The next section presents transition state theory (TST) and describes how it can be used to obtain the rate constants. Finally, a briefly introduction on *kMC* will be given.

2.1 The Many-Body Schrödinger Equation

A fundamental equation in quantum mechanics is the *Schrödinger equation*. It provides means for solving a *wave function*, which is a description of a system's quantum state, and from that, molecular properties of chemical systems can be calculated. The wave function itself is a probability amplitude, however, its physical meaning is difficult to interpret. Nevertheless, there is a proportional relation between the *square modulus* of the wave function, $|\psi(\vec{r})|^2 = \psi(\vec{r})^* \psi(\vec{r})$ and the probability, of finding a single particle at point $\vec{r} = (x, y, z)$ in an infinitesimal volume element $d\vec{r} = dxdydz$ according to *Born's interpretation*. From that it follows that the product of the wave function and its complex conjugate $|\psi(\vec{r})|^2$ is a probability density. It is advantageous to consider a normalised function, because then Born's interpretation can be regarded as an equality rather than a proportionality[7]. The form of the Schrödinger equation depends on the physical system being described and how it is described. In computational chemistry, it is typically the *time-independent*, *non-relativistic*, *electronic* Schrödinger equation that is being solved. The emphasis on *electronic* implies that the *Born-Oppenheimer* approximation (BOA)[17] has been adopted - one of quantum chemistry's cornerstones, allowing the wave function to be separated into an electronic part and a nuclear part. A justification for adopting it lies in recognising the significant mass difference between nuclei and electrons ($M_I \sim 1800 m_i$), which results in electrons responding almost instantaneously to displacement of the nuclei[7, 40, 45]. Moreover, the electronic wave function is a function of spatial position of electrons, whereby the nuclear spatial position only has parametric dependence, hence the semicolon $\psi(\vec{r}; \vec{R})$[7, 26]. In the context of the BOA, it follows a Hamiltonian operator, where the kinetic energy term for the nuclei is omitted because it is considered fixed in space[8]. Additionally, since the potential energy operator for the nuclei-nuclei repulsion term is constant, and

the wave function is invariant to constant terms, it is conventionally excluded in the Hamiltonian. Instead, it gets included as a classical term at the end of the calculation[7, 26]. The resulting electronic Schrödinger equation can then be written more descriptively as follows

$$\hat{H}\psi_i(\vec{r_i};\vec{R}) = \left\{ -\frac{\hbar^2}{2m}\sum_{i=1}^{n}\nabla_i^2 - j_0\sum_{i=1}^{n}\sum_{I=1}^{N}\frac{Z_I}{r_{iI}} + \frac{1}{2}j_0\sum_{\substack{\mathrm{i}\,=\,1,\\ \mathrm{j}\,=\,\mathrm{i}\,+\,1}}^{n}\frac{1}{r_{ij}} \right\}\psi_i(\vec{r_i};\vec{R}) = E_i\psi_i(\vec{r_i};\vec{R}) \quad (2.1)$$

where $\hat{H}$ is the *Hamiltonian* operator $\hat{H}$, that operates on the eigenfunction $\psi_i(\vec{r};\vec{R})$ and returns the same eigenfunction $\psi_i(\vec{r};\vec{R})$ scaled by its corresponding eigenvalue E, which is the electronic energy[8]. j_0 is introduced as a condensed expression $j_0 = \frac{e^2}{4\pi\varepsilon_0}$, where e is the elementary charge and ε_0 is the vacuum permittivity[7]. In most computational programs, these are converted to atomic units, leading to $j_0 = 1$.

The first term in the Hamiltonian is the electronic kinetic energy operator, where $\hbar$ is the *reduced Planck's constant*, m is the electron mass and ∇^2 is the *Laplacian* operator. When Cartesian coordinates are evaluated, it consists of the sum of three second partial derivatives with respect to the three axis. The second term describes the electron-nucleus attraction and is a potential energy contribution where Z_I is the number of nuclei and r_{iI} is the distance between electron i and nucleus I. The last term is also a potential energy operator, and it captures the electron-electron repulsion between electron i and electron j, and r_{ij} denotes the distance between them[7].

Although, considering the Schrödinger equation within the BOA, only the hydrogen molecule ion $H_2{}^+$ can be solved exact. The electron-electron repulsion term with the r_{ij} dependence makes an exact solution impossible to obtain for systems containing more than one electron. It is possible to approximate the the true many-electron wave function as a product of n one-electron wave functions $\psi(\vec{r};\vec{R}) = \prod_{i=1}^{n}\psi_i(\vec{r};\vec{R})$, according to the *orbital approximation*[7]. However, each of these one-electron wave function are affected by the presence of all other electrons in the system. Consequently, the wave function for one electron i cannot be determined without having considering all other electrons simultaneously, which makes the Schrödinger equation a many-body problem[64].

2.2 Density Functional Theory

Being a problem of $3n$ variables, where n denotes the number of electrons in the system, the Schrödinger equation quickly becomes computational expensive as the number of electrons increases with the system size. This is especially problematic for larger molecules. Developing other methods for solving the Schrödinger equation therefore becomes of great desire. One of these is density functional theory (DFT), which started to emerge during the 1920s. In 1927 the Thomas-Fermi model was presented[7, 75], which tackled the problem completely differently. The approach sought to use the electron density as a mean to express the energy, as opposed to the electronic wave function. Ultimately, reducing the dimensions of the problem from $3n$ to 3. The Thomas-Fermi model can be viewed as the predecessor to DFT, but unfortunately, there was no theoretical hold

in that the energy could directly related to the electron density[7].

2.2.1 The Hohenberg-Kohn Theorems

In 1964 two decisive theorems for the developemnt of DFT were presented, namely the Hohenberg-Kohn (HK) theorems[38]. The first theorem is called the *Hohenberg-Kohn existence theorem* (proved in appendix A.1), which states that *The ground state energy and all other ground state electronic properties are uniquely determined by the electron density*[7]. The electron density function, $\rho(\vec{r})$, is decisive for any ground state properties. Essentially, this means that any ground state property, take the energy (E_0) for instance, can be represented as a functional of the electron density function. A functional can be thought of as a rule for transforming a function into a number, while a function can be seen as a rule to transform a number into the same or a different number[45].

Pertaining to DFT-theory, the ground state energy can mathematically be represented as functional of the ground state electron density function, denoted as[1]

$$E_0 = E[\rho_0(\vec{r})] \tag{2.2}$$

where the hard brackets, imply functional, embracing the electron density function $\rho(\vec{r})$[45].

The implication of the existence theorem is that there exists an alternative method for obtaining the energy, namely by deriving it from the basis of the ground state electron density

$$E[\rho] = \overbrace{T[\rho] + V_{ee}[\rho]}^{E_{HK}[\rho]} + \int \rho(\vec{r})\nu(\vec{r})d\vec{r} \tag{2.3}$$

where $T[\rho]$ is the electronic kinetic energy contribution of and $V_{ee}[\rho]$ is the electron-electron potential energy contribution. The integral involves summation (integration) of the products of the electron density $\rho(\vec{r})$ and the *external potential* $\nu(\vec{r})$, over all points in the volume element $d\vec{r}$. The external potential (eq. (2.4)) is a potential energy term and corresponds to the electrostatic interaction between negatively charged electrons and the positively charged nuclei[40, 45, 87].

$$\nu(\vec{r}) = -j_0 \sum_{I=1}^{N} \frac{Z_I}{|\vec{r} - \vec{R}_I|} \tag{2.4}$$

Also, the external potential is analogous to the second term in eq. (2.1). In all, the existence theorem forms the basis in which the the Hamiltonian can be obtained by deriving it from the electron density[7, 26]

$$\hat{H} = -\frac{\hbar^2}{2m}\sum_{i=1}^{n}\nabla_i^2 + \sum_{i=1}^{n}\nu(\vec{r}_i) + \frac{1}{2}j_0 \sum_{\substack{\mathrm{i}\,=\,1,\\ \mathrm{j}\,=\,\mathrm{i}\,+\,1}}^{n} \frac{1}{r_{ij}} \tag{2.5}$$

Moving on to the second theorem, the *Hohenberg-Kohn variational theorem*, which is an analogue

[1] which further on will be referred to as the electron density unless explicitly stating anything else. Also the subscript 0 will be omitted for simplicity

to the variational theorem seen in the ad inito methods stating that the *Rayleigh ratio*[2] is greater or equal to the exact ground state energy, $\mathcal{E} \geq E_0$. Similarly, DFT also make use of a trail function, but rather using a trail wave function, a trail electron density function (ρ_t) is presented. The theorem states that this trail function yields an energy greater or equal to the exact energy obtained by the true electron density function (ref. appendix A.2)[7]. Mathematically it can be stated as

$$E[\rho_t] \geq E_0 \tag{2.6}$$

For a method to be variational is an advantageous feature, as it assures that any calculated energy is the upper bound to the true energy. By applying the variational principle to the energy functional, the energy gets minimised with respect to the ground state electron density for a fixed number of electrons

$$\frac{\delta}{\delta\rho(\vec{r})}\left\{E[\rho(\vec{r})] - \mu\int\rho(\vec{r})d\vec{r})\right\} = 0 \tag{2.7}$$

and is subjected to the constraint

$$\int\rho(\vec{r})d\vec{r} - n = 0 \tag{2.8}$$

where $\rho(\vec{r}) \geq 0 \; \forall \; \vec{r}$. The ground state electron density, must then satisfy

$$\mu = \nu(\vec{r}) + \frac{\delta E_{HK}[\rho]}{\delta\rho(\vec{r})} \tag{2.9}$$

where δ signifies the functional derivative The theorem also ensures that the number of electrons n is conserved and always positive[26, 40, 45].

2.2.2 The Kohn-Sham Equations

Up until this point the Hohenberg and Kohn proved the energy could be determined from the electron density - the premise of DFT. However, it was just that, a proof, and there was no guidelines on how $E[\rho]$ should be constructed. It was not until 1965 that a solution emerged. Kohn and Sham developed an approach with a set of solvable equations, refereed to as the *Kohn-Sham approach*[43]. They formulated one-electron equations that offer a potential pathway to acquire the electron density. The approach presents a fictitious reference system containing n non-interaction electrons in an external potential $\nu_{ref}(\vec{r})$, where the electron density $\rho_{ref}(\vec{r})$ is equivalent to the true electron density $\rho(\vec{r})$[7, 26, 45]. The reference Kohn-Sham hamiltonian is expressed as a sum of one-electron equations

$$\hat{h}_{ref} = \sum_{i=1}^{n}\hat{h}_i^{KS} \tag{2.10}$$

$$\hat{h}_i^{KS} = -\frac{\hbar^2}{2m}\nabla_i^2 + \nu_{ref}(\vec{r}_i) \tag{2.11}$$

[2]The well known variational theorem, whereby ψ_t is the trail wave function

$$\mathcal{E} = \underbrace{\frac{\langle\psi_t|\hat{H}|\psi_t\rangle}{\langle\psi_t|\psi_t\rangle}}_{\text{Rayleigh ratio}} \geq E_0$$

The one-electron Kohn-Sham orbitals φ_m^{KS} are eigenfunctions of the one-electron Kohn-Sham hamiltonians $\hat{h}_i^{KS}$. A similar eigenvalue equation as given in equation (2.1), can also be expressed in context of the Kohn-Sham approach

$$\hat{h}_i^{KS}\varphi_m^{KS}(i) = \varepsilon_m^{KS}\varphi_m^{KS}(i) \tag{2.12}$$

where the eigenvalue ε_m^{KS} is the orbital energy corresponding to the Kohn-Sham orbital $\varphi_m^{KS}(i)$. Furthermore, in KS-DFT the ground state wave function can be expressed as a single Slater determinant Φ_0^{KS}. The Slater determinant is comprised of Kohn-Sham spin orbitals $\phi_n^{KS}(\vec{x_n})$

$$\begin{aligned}\Phi_0^{KS}(\vec{x}_n) &= \frac{1}{\sqrt{n!}}\begin{vmatrix} \phi_1^{KS}(\vec{x}_1) & \phi_2^{KS}(\vec{x}_1) & \dots & \phi_n^{KS}(\vec{x}_1) \\ \phi_1^{KS}(\vec{x}_2) & \phi_2^{KS}(\vec{x}_2) & \dots & \phi_n^{KS}(\vec{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1^{KS}(\vec{x}_n) & \phi_2^{KS}(\vec{x}_n) & \dots & \phi_n^{KS}(\vec{x}_n) \end{vmatrix} \\ &= \det|\phi_1^{KS}(\vec{x_1})\phi_2^{KS}(\vec{x_2})\dots\phi_n^{KS}(\vec{x_n})|\end{aligned} \tag{2.13}$$

where $\vec{x_n}$ in $\phi_n^{KS}(\vec{x_n})$ denotes that the Kohn-Sham spin orbitals is a product of the Kohn-Sham spatial orbitals φ_n^{KS} and spin state (α or β), when only closed-shell systems are considered. The motivation in using a Slater determinants with spin orbitals originate from the fundamental quantum properties of electrons. Particularly them being fermions and therefore must obey the *Pauli principle*, stating that the wave function must be anti-symmetric with respect to interchange of any pair of electrons. A direct implication is the *Pauli exclusion principle*, ensuring that two electrons are prohibited from occupying the same quantum state[7]. The first principle translates to interchanging a pair of rows in the determinant, while the second principle corresponds to a situation where the determinant has a pair of spin orbitals in common, i.e. two columns are identical, ultimately collapsing the determinant.

The total ground state energy functional for a many electron system can be expressed in terms of the reference system accompanied with a correction term

$$\begin{aligned}E[\rho] &= T[\rho] + V_{ee}[\rho] + \int \rho(\vec{r})\nu(\vec{r})d\vec{r} \\ &= T_{ref}[\rho_{ref}] + J_{ref}[\rho_{ref}] + \int \rho(\vec{r})\nu(\vec{r})d\vec{r} + T[\rho] + V_{ee}[\rho] - (T_{ref}[\rho_{ref}] + J_{ref}[\rho_{ref}])\end{aligned} \tag{2.14}$$

Since the reference system is a fictitious one presented by Kohn-Sham, it is by definition set to have the same electron density as the real system as already mentioned

$$\rho(\vec{r}) = \rho_{ref}(\vec{r}) \tag{2.15}$$

By demanding this, the energy expression given in eq. (2.14) can be rewritten

$$E[\rho] = T_{ref}[\rho] + J[\rho] + \int \rho(\vec{r})\nu(\vec{r})d\vec{r} + E_{xc}[\rho] \tag{2.16}$$

The V_{ee} contribution has now been split into a classical component $J_{ref}[\rho]$ and a non-classical component E_{xc}. The first contribution accounts for the Coulomb electron-electron interaction[3]

$$J[\rho] = \frac{1}{2}j_0 \int\int \frac{\rho(\vec{r_1})\rho(\vec{r_2})}{r_{12}} d\vec{r_1}d\vec{r_2} \tag{2.17}$$

[3]which sometimes in literature is referred to as the *Hartree energy* $E_H[\rho]$

in terms of *one-point* electron density $\rho(\vec{r})$

$$\rho(\vec{r}) = \sum_m \varphi_m^{KS*}(\vec{r})\varphi_m^{KS}(\vec{r}) \tag{2.18}$$

The last term in eq. (2.16), E_{xc} is called the exchange-correlation energy. It accounts for the residual part of the true kinetic energy. That being the difference between the true kinetic energy and the fictitious kinetic energy($T[\rho] - T_{ref}[\rho]$) and the non-classical electrostatic interaction ($V_{ee}[\rho] - J[\rho]$)

$$E_{xc} = T[\rho] + V_{ee}[\rho] - (T_{ref}[\rho] + J[\rho]) \tag{2.19}$$

Now eq. (2.9) can be redefined as

$$\mu = \nu_{eff}(\vec{r}) + \frac{\delta T_{ref}[\rho]}{\delta\rho(\vec{r})} \tag{2.20}$$

where the effective potential $\nu_{eff}(\vec{r})$ is defined as

$$\nu_{eff}(\vec{r}) = \nu(\vec{r}) + \frac{\delta J[\rho]}{\delta\rho(\vec{r})} + \frac{\delta E_{xc}[\rho]}{\delta\ \rho(\vec{r})} \tag{2.21}$$

and must be equal to the reference potential in order to solve the Kohn-Sham equations. The functional derivative of the Coulomb functional is

$$\frac{\delta J[\rho]}{\delta\rho(\vec{r})} = j_0 \int \frac{\rho(\vec{r'})}{|\vec{r} - \vec{r'}|} d\vec{r'} \tag{2.22}$$

The exchange-correlation potential for the functional derivative of E_{xc} is

$$\nu_{xc}(\vec{r}) = \frac{\delta E_{xc}[\rho]}{\delta\rho(\vec{r})} \tag{2.23}$$

Consequently, the effective potential (eq. (2.21)) can be rewritten into a function of position and not in terms of functionals as before

$$\nu_{eff}(\vec{r}) = \nu(\vec{r}) + j_0 \int \frac{\rho(\vec{r'})}{|\vec{r} - \vec{r'}|} d\vec{r'} + \nu_{xc}(\vec{r}) \tag{2.24}$$

$\nu(\vec{r})$ corresponds to the total external potential acting on all n electrons collectively. By subjecting a reference system to a particular external potential and solving the Kohn-Sham equation ((2.25)) with the same external potential, but for a real system, the electron density which minimises the energy functional can be found[7, 40, 87].

$$\left\{ \hat{h}_1 + j_0 \int \frac{\rho(\vec{r'})}{|\vec{r} - \vec{r'}|} d\vec{r'} + \nu_{xc}(\vec{r}) \right\} \varphi_m^{KS}(i) = \varepsilon_m^{KS} \varphi_m^{KS}(i) \tag{2.25}$$

where the one-electron hamiltonian presented in eq. (2.11) has been redefined since ν_{ref} has been substituted by ν_{eff} seen in eq. (2.21)[7]

$$\hat{h}_1 = -\frac{\hbar^2}{2m}\nabla_1^2 + \nu(\vec{r}_1) \tag{2.26}$$

The KS equations are solved through a self-consistent method. This self-consistency arises because the initial guess for the electron density serves as the input for the KS equations and is aimed at converging towards the actual solutions. The procedure can be summarised in the following points[64]

1. **Initial guess** of the electron density $\rho(\vec{r})$ ref. eq. (2.18)
2. **Solve the Kohn-Sham equations** (2.25) using the initial guess to obtain φ_m^{KS}
3. **A new electron density is calculated** $\rho_{KS}(\vec{r})$ based on the findings in step 2
4. **Compare** $\rho_{KS}(\vec{r})$ with $\rho(\vec{r})$. If the current density $\rho_{KS}(\vec{r})$ is sufficiently close to $\rho(\vec{r})$, then the calculation is considered converged towards a self consistent solution. If not, repeat the steps 2-4 until convergence is achieved

The initial guess is based on the *superposition of atomic densities*. This essentially means that all the electron densities of all the individual atoms in the system gets added together in a sense. The resulting electron density is an initial approximation for how electrons are distributed in the entire system[7].

What decides whether a calculation can be considered converged, is often pre-determined in the program code as so called *convergence parameters* or *convergence criteria*. In general, the stricter the convergence parameters, the more computationally expensive they become.

2.2.3 Exchange-Correlation Functional

The main shortcoming in DFT lies in the approximation of the unknown exchange-correlation functional $E_{XC}[\rho]$. The accuracy of the DFT-calculations relies on the chosen exchange-correlation functional, often just referred to as *the functional* in the context of DFT. Numerous approximate methods have been proposed and they can be organised in a hierarchical classification system known as Perdew's *Jacob's Ladder*[55] - an analogy to the biblical story of a ladder leading to *heaven*. In this context, heaven corresponds the realm of *chemical heaven*, where an universal functional exists and its computational accuracy is *divinely* accurate. The base of the ladder symbolises to the *Hartree world*, which neglects electron correlation effects beyond the mean-field level.

The exchange-correlation functional is often decomposed into exchange and correlation components

$$E_{XC}[\rho] = E_X[\rho] + E_C[\rho] \tag{2.27}$$

Most work on DFT in the recent years has been focused on development of a better functional. The choice of what functional to use is often based on the problem at hand, experience, the availability of functionals in the program and the computational cost and efficiency. Additionally, it is also desirable for the functional to be free of self-interaction, that being for a one-electron system, the exchange energy cancel both the Coulomb energy and the correlation energy. This appears to be an obvious requirement for a functional, yet there are actually no functional that fulfils it. As a consequence, the system seems more stable than it actually is. The error corresponding is called *self-interaction error* (SIE)[40]. While some functionals perform well for very specific systems, there is unfortunately no universal functional, like the one referred to in the analogy. This is probably the main limitations in the accuracy of DFT predictions for some systems. There is a vast number of different functional out there and choosing the best one.

The most popular functional, owing to the great success is DFT, is the *B3LYP* - Becke three-parameter Lee-Yang-Parr functional and is a favourite among many DFT practitioners[6]. It belongs to the *Hybrid Generalised Gradient Approximation* (Hybrid GGA) family of functional and aims to improve the exchange-correlation functional by introducing a density gradient in addition to the local density. As the name states it is a hybrid functional of exact exchange (from Hartree-Fock theory) and empirically determined parameters, fitted to improve the accuracy[7, 40].

$$E_X^{B3LYP} = (1-\alpha)E_X^{LSDA} + aE_X^{HF} + b\Delta E_X^B + (1-c)E_c^{LSDA} + cE_c^{LYP} \tag{2.28}$$

where $a = 0.20$, $b = 0.72$ and $c = 0.81$ are parameters from experimental data. The first three exchange-terms are the Local Spin Density Approximation (LSDA) E_X^{LSDA}, Hartree-Fock exchange E_X^{HF} and Becke (B or B88) exchange E_X^B. The remaining two are the LSDA correlation-term E_c^{LSDA} and the *LYP* correlation term E_c^{LYP}[40]

2.2.4 Basis Set Selection

To give an exact representation of the molecular orbitals φ^{KS}, a complete set of basis functions $\{\chi_\mu\}$ have to be used[7]. The completeness implies an infinite basis set and plays a decisive role in the accuracy of the computational results. However, using an infinite set of basis functions is not computational feasible, thus a finite m basis set must be used $\{\chi_\mu\}_{\mu=1,\dots,n}$. The corresponding error is called *basis set truncation error*, and translates to the difference between the *HF limit*[4] and the computed lowest energy in HF-SCF calculation[7].

Because the molecular orbitals are unknown, they are approximated[5] through the Linear Combination of Atomic Orbitals (LCAO), in which the molecular orbitals are comprised by linear combination of atomic orbitals $\chi(\vec{r})$

$$\varphi^{KS}(\vec{r}) \approx \sum_{\mu}^{m} C_{\mu i}\chi_\mu(\vec{r}) \tag{2.29}$$

where $C_{\mu i}$ are the unknown orbital coefficients[7].

The most common basis function in computational chemistry is the Gaussian-type orbitals (GTO). A GTO, centred on the nucleus of the atom, is described by

$$g_{ijk}(\vec{r}) = Nx^i y^j z^k e^{-\alpha r^2} \tag{2.30}$$

where N is a normalisation constant, α is a positive exponent and r is the distance between the electron and the nucleus[7, 40]. i, j, k are all non-negative integers corresponding to all the permitted quantum numbers n, l and m_l. E.g. $i = j = k = 0$ correspond to s (one possibility, $(0,0,0)$), $i+j+k = 1$ correspond to p-type Gaussian (3 possibilities $(1,0,0)$, $(0,1,0)$, $(0,0,1)$) and $i+j+k = 2$ correspond to d-type Gaussian (6 possibilities) and so on.

[4] a theoretical concept that represents a point of convergence (regarding the energy limit) in the Hartree-Fock framework. It represents the lowest electronic energy that can be obtained within the framework of the Hartree-Fock method as various computational parameters are systematically improved

[5] only the case if the basis set is incomplete[40]

Similar consideration as those for selecting the functional apply for the basis set. Just as for the functional, there is also a vast number of basis set to chose from. Some additional ideal features for the *divine* basis set could include: **closeness to the basis set limit**: ability to produce results that approach the basis set limit, **fast convergence:** capable to achieve basis set convergence fast, **versatile** in the sense that it should be applicable for various methods (HF, DFT, Coupled Cluster) and **available** for the entire periodic table. Unfortunately, there exist no such heavenly basis function either. Also, pertaining to point 3, most basis sets that are developed these days are designed to improve specific features[40].

Among the most popular basis sets is the *Dunning correlation consistent basis sets*, and are specifically designed for post-HF methods. Ideally, they tend to systematically increase accuracy with increasing basis set size[45]. Thir acronyms are *cc-pVXZ* where *cc* stands for *correlation consistent*, *p* denotes polarisation functions and V for valence, $X \in \{D = double, T = triple, Q = quadruple, 5 = quintuple\}$ and stand for the number of shells the valence functions are split into and Z for zeta. E.g. cc-pVTZ then translates to *correlation-consistent polarised valence triply split zeta*. These basis sets are typically used to describe as the name states, polarisation of the electron density. Essentially this refers to forces describing the geometry and bonding such as covalent bonds. When searching for ways to describe forces over a longer distance from the nuclei, like molecular polarisability, *diffuse* functions are of great importance, and thus incorporated. These are denoted with the prefix *aug* for augmented. Ultimately this provides more flexibility.

Since the purpose of basis set is to describe and represent the electrons in a system and are chosen by the operator *a priori*, there are of course some errors related to the choice of a finite basis set. One of these errors are referred to as the *basis set superposition error* (BSSE) and emerge when using a finite basis set. It arises when a too small basis set is used, causing atoms and fragments that are close in space to *borrow* basis functions from each other, resulting in a spurious attractive energy contribution, artificially lowering the energy[19, 35].

Another basis set related error is the *basis set incompleteness error* (BSIE) and arises due to the limited flexibility of the basis set to accurately to accurately capture the electron density within the LCAO approximation. For atoms, BSIE is quantified as the difference between the results calculated at a given basis set and the corresponding result obtained at a complete basis set. However, for a molecule the corresponding difference contain both BSIE and BSSE, making it difficult to unravel the exact contribution from each component[40].

2.2.5 Concepts and Applications of the Potential Energy Surface

A *potential energy surface* (PES) of a polyatomic specie provides a perception of the energy landscape of the corresponding system. It describes the relationship between the potential energy and the corresponding geometry. Both minima and first-order saddle points (corresponding to *transition states*) are of interest. Conceptually, the PES can be visualised by plotting the energy

against the spatial coordinates, by providing a set of solutions where the electronic Hamiltonian has been solved, within the BOA, at various geometries. All while the system is at ground state[26].

Mapping out the entire PES is not only impossible but also impractical, as the most interesting phenomena typically occurs along a specific path - the *reaction coordinate*. By considering only a single reaction coordinate, the PES can be simplified to a one dimensional curve. When studying a chemical reaction the energy profile changes. Stationary points correspond to locations where the gradient is zero. These can be minima or maxima corresponding to an equilibrium geometry and a transition state respectively. The equilibrium geometry represents the most favourable conformation in terms of bond angles, bond lengths and energy. For increasing displacement from the equilibrium geometry, the molecular potential energy increases. A transition state is an energy maximum in one direction, and a minimum all other directions. Moreover, this transition state is associated with one and only one negative eigenvalue. Subsequently, it gives origin to the single imaginary frequency, which characterises a first order transition state[7, 40]

As previously mentioned, stationary points on the PES are characterised by a zero gradient, i.e. zero first-order partial derivatives, $\frac{\partial E}{\partial q} = 0$, where E is the energy and q is a geometric parameter. However, there is no way telling whether this is a minima, maxima or a saddle point for that sake. In order to do so, chemists turn to mathematicians for tools. They often utilise multi-variable calculus to solve optimisation problems[7]. The *Hessian matrix* $\mathcal{H}$ can be used as a tool to distinguish the stationary points[34].

$$\mathcal{H}_E = \begin{vmatrix} \frac{\partial^2 E}{\partial q_1^2} & \frac{\partial^2 E}{\partial q_1 \partial q_2} & \cdots & \frac{\partial^2 E}{\partial q_1 \partial q_n} \\ \frac{\partial^2 E}{\partial q_2 \partial q_1} & \frac{\partial^2 E}{\partial q_2^2} & \cdots & \frac{\partial^2 E}{\partial q_2 \partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 E}{\partial q_n \partial q_1} & \frac{\partial^2 E}{\partial q_n \partial q_2} & \cdots & \frac{\partial^2 E}{\partial q_n^2} \end{vmatrix} \tag{2.31}$$

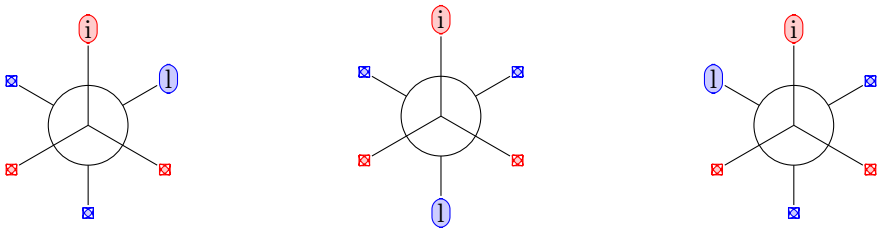
The matrix is comprised of second-order partial derivatives of the energy with respect to the spatial coordinates[6].

The distinction between global and local minima, pertaining to conformation analysis is interesting as one seeks to find the most stable conformation, where there might be several equally stable isomers. It can be illustrated with considering the potential energy versus the torsion/dihedral angle in a butane molecule, i-j-k-l. The global minimum occurs when i and l are in an *anti periplanar/trans* arrangement, while a local minimum occurs when the i and l are in *syn-clinal/gauche* arrangement. The term *anti* refers to a dihedral angle exceeding $\pm 90°$, while a dihedral angle $< \pm 90°$ is designated as *syn*. *Periplanar* describes when i and l are roughly in the same plane at a dihedral angle of $\pm 30°$ or $\pm 150°$. When the dihedral angle falls within $+30°$ to $+150°$ or $-30°$ to $-150°$, it is termed *clinal*[6, 58]. Figure 2.1 illustrates the three most common *staggered* conformations, where none of

[6]There is also a related matrix, the *Jacobian matrix that is comprised of first-order partial derivatives,*

$$\mathcal{J}_E = \begin{vmatrix} j_x & j_y \\ j_x & j_y \end{vmatrix}$$

the atoms/substituents overlap. This way of visualising a molecule is called the *Newman projection*, the perspective is looking along the j-k axis, which has fixed coordinates, while i and l vary in a circular manner.



(a) gauche $\sim 60°$ **(b)** trans $\sim 180°$ **(c)** gauche $\sim -60°$

Figure 2.1: Red means that the atoms/substituents are in front, while blue means the atoms/substituents are in the back. figure **(a)**, **(b)** and **(c)** illustrates how the $\pm$ gauche and trans conformers look in a Newman projection

2.3 Solvation Effects with COSMO

In genera, for condensed phase systems, a solvent model ought to be applied. Pertaining to DFT a *continuum solvation model*, such as the COnductor-like Screeening MOdel (COSMO) is the most common method to include solvent effects. The approach does not consider actual solvent molecules in the calculation[19]. Instead, a molecular-shaped cavity is employed, where the molecule under study get encapsulated. The cavity may be defined by the van der Waals radii of the atoms in the corresponding molecule scaled by some empirical factor. Moreover, in continuum models, solvents are considered as a uniform polarisable medium with a characteristic *dielectric constant* which is specific to the solvent[40].

2.4 Transition State Theory

This section entails the theory on how quantum mechanics can be used to derive the rate constant using TST, and also how it can be related to thermochemical data. However, the nuclei are assumed to obey classical mechanics. Although, cases do occur when classical mechanics do not apply such as quantum mechanical *tunneling* and rare events. However, these aspects are neglected in TST[29]. Further, one of the most important approximations to note, is that all equations assume non-interacting particles. Hence, the accuracy is optimal for *ideal gas* behaviour. This subsequently introduces some errors, since most systems under study are not under such conditions.

The time perspective a chemical process has been allowed to proceed is decisive for the final product. A shorter time period may often result in a *kinetically* stable product, which in that time range is the most viable product. However, it might not be the most stable product overall. If the reaction is allowed to run for a longer period of time the *thermodynamically* stable product may eventually

be the product.

2.4.1 Principle of Detailed Balance and Equilibrium

First, some important concept in TST ought to be put out. In that context, it is best to begin with a simple chemical reaction. Take an arbitrary reaction going from state A to state B, and reverse

$$A + B \underset{\mathrm{k}_r}{\overset{\mathrm{k}_f}{\rightleftharpoons}} C \tag{2.32}$$

where k_f and k_r are the forward and reverse rate constants respectively. The rate of formation of the product C can be described by chemical kinetics using the equation

$$\frac{d[C]}{dt} = k[A][B] \tag{2.33}$$

where k is the observable and measurable rate constant and $[A]$, $[B]$ and $[C]$ are the concentrations. The *principle of detailed balance* states: "for an elementary reaction, the overall forward rate and the overall reverse rate are identical at equilibrium"[27]. Mathematically, it can be expressed as:

$$k_f[A]_{eq}[B]_{eq} = k_r[C]_{eq} \tag{2.34}$$

where $[A]_{eq}$, $[B]_{eq}$ and $[C]_{eq}$ are the concentrations of the species at equilibrium. To illustrate that eq. (2.34) is a condition for equilibrium it could be useful to write out the rate expressions for the reaction presented in eq. (2.32)[27].

$$\frac{d[A]}{dt} = \frac{d[B]}{dt} = -k_f[A][B] + k_r[C] \tag{2.35}$$

$$\frac{d[C]}{dt} = k_f[A][B] - k_r[C] \tag{2.36}$$

Applying the equilibrium condition in eq. (2.34) and substitute it into the equations (2.35) and (2.36) gives

$$\frac{d[A]}{dt} = \frac{d[B]}{dt} = 0 \tag{2.37}$$

$$\frac{d[C]}{dt} = 0 \tag{2.38}$$

which implies that the rates forward and reverse are equal at equilibrium. Furthermore, the equilibrium condition in eq. (2.34) can be shown to relate k_f and k_r to the equilibrium constant K[27]. This is a very important relation.

$$K = \frac{[C]_{eq}}{[A]_{eq}[B]_{eq}} = \frac{k_f}{k_r} \tag{2.39}$$

2.4.2 Derivation of the Rate Constant from Transition State Theory

To calculate the rate constant k using transition state theory, the same equation as in eq. (2.32) will be considered, but with a slightly different expression

$$A + B \overset{K^{\ddagger}}{\rightleftharpoons} (AB)^{\ddagger} \xrightarrow{\mathrm{k}^{\ddagger}} C \tag{2.40}$$

TST assumes a *quasi-equilibrium* in which the first step is an pre-equilibrium between the reactants and the *activated complex* denoted by the double dagger (‡)[6, 9, 27, 29]

$$K^{\ddagger} = \frac{[(AB)^{\ddagger}]}{[A][B]} \tag{2.41}$$

The overall reaction rate converting reactants to products, can be expressed with the number (or concentration) of the respective reactants by rearranging and inserting eq. (2.41) into the rate expression[27]

$$\frac{d[C]}{dt} = k^{\ddagger}[(AB)^{\ddagger}] = k^{\ddagger}K^{\ddagger}[A][B] = k[A][B] \tag{2.42}$$

Substituting eq. (2.33) and is comprised of [6, 27]

$$k = k^{\ddagger}K^{\ddagger} \tag{2.43}$$

At the transition state, there is no Boltzmann distribution of states, because the activated complex is not *persistent*. For that reason, TST turns to statistical mechanics to solve for the rate constant $k^{\ddagger}$ and the equilibrium constant $K^{\ddagger}$[6].

Statistical mechanics tries to connect the behaviour of individual particles to thermodynamic *observables*, i.e. measurable quantities. A *partition function* q gives information on how particles are *partitioned* throughout the states and relates the macroscopic thermodynamic properties and microscopic models. It is a sum of *Boltzmann factors* $e^{-\beta E_j}$, where $\beta = \frac{1}{k_B T}$ and k_B is the Boltzmann constant[27].

As already mentioned, $K^{\ddagger}$ is an equilibrium constant and can be constructed using *molecular partition functions*[27]. These account for contributions from translation q_t, rotational q_r, vibrational q_ν and electronic q_e degrees of freedom[40].

The translational partition function can be expressed as

$$q_t = \left(\frac{2\pi m k_B T}{h^2}\right)^{\frac{3}{2}} V \tag{2.44}$$

where m is the mass (in kg) of the molecule, h is *Planck's constant* and V is the volume[27]. Using the volume is quite tricky in this case, since it is not known. However, because ideal gas is assumed, $pV = nRT = \left(\frac{N}{N_A}\right) N_A k_B T$, and $V = \frac{N k_B T}{p}$, where N is the number of gas molecules[50]. Hence, the implemented translational partition function becomes

$$q_t = \left(\frac{2\pi m k_B T}{h^2}\right)^{\frac{3}{2}} \frac{N k_B T}{p} \tag{2.45}$$

It should be mentioned that in some cases the, the software is accustomed to use molar scale. If that is the case, V is the volume of 1 mol of ideal gas[40]. The rotational partition function depends on the geometry of the molecule. For linear molecules with a single moment of inertia I, the partition function is defined as

$$q_r = \frac{T}{\sigma \theta_r} = \frac{8\pi^2 I k_B T}{h^2} \tag{2.46}$$

For the general case of non-linear polyatomic molecules, the rotational partition function includes the three principal moment of inertial I_A, I_B and I_C,

$$q_r = \frac{(\pi I_A I_B I_C)^{\frac{1}{2}}}{\sigma} \left(\frac{8\pi^2 k_B T}{h^2}\right)^{\frac{3}{2}} \tag{2.47}$$

where σ is the *symmetry factor*, θ_r is the rotational temperature $\theta_r = \frac{h^2}{8\pi^2 I k_B}$, where high temperature is assumed ($T \gg \theta_r$) and I is the moment of inertia[27, 29, 49, 50]. The symmetry factor or *symmetry number* is the number of orientations that render indistinguishable molecules[8].

The moments of inertia can moreover be related to rotation constants as they are reciprocally connected. When rotational constants are in in frequency units the relation is

$$I_A = \frac{h}{8\pi^2 A}, \qquad I_B = \frac{h}{8\pi^2 B}, \qquad I_C = \frac{h}{8\pi^2 C} \tag{2.48}$$

where A, B and C are rotation constants[29, 36, 49, 50]. Subsequently the rotational partition function can be expressed in terms of rotational constants

$$q_{\text{r, linear}} = \frac{T}{\sigma\theta_r} = \frac{k_B T}{\sigma h B} \tag{2.49}$$

$$q_r = \left(\frac{k_B T}{h}\right)^{\frac{3}{2}} \frac{\pi^{\frac{1}{2}}}{\sigma\,(ABC)^{\frac{1}{2}}} = \left(\frac{k_B T}{h}\right)^{\frac{3}{2}} \frac{1}{\sigma}\sqrt{\frac{\pi}{ABC}} \tag{2.50}$$

The vibrational partition function only consider real vibrational modes[22, 41]. Meaning modes with imaginary frequencies, corresponding to transition states are ignored. For that reason, the partition function for a transition state, generally has one less vibration to consider[40]. The overall vibrational partition function is expressed as

$$q_v = \prod_{i=1}^{K} \frac{1}{1 - e^{\frac{-h\nu_i}{k_B T}}} \tag{2.51}$$

where ν_i is vibration i, and K is the number of vibrational modes. For linear molecules $K = 3N - 5$, but for non-linear molecules $K = 3N - 6$ and N is the number of atoms in the molecule. Each of the K modes, has their characteristic vibrational temperature[27, 29]

$$\theta_{v_i} = \frac{h\nu_i}{k_B} \tag{2.52}$$

The equilibrium constant $K^{\ddagger}$ can be expressed in terms of the molecular *partition functions*.

$$K^{\ddagger} = \left(\frac{q_{(AB)^{\ddagger}}}{q_A q_B}\right) e^{\frac{-\Delta E^{\ddagger}}{k_B T}} \tag{2.53}$$

where $q_{(AB)^{\ddagger}}$ is the partition function for the transition state and q_A and q_B are the partition functions for the reactant A and B respectively. $\Delta E^{\ddagger}$ denotes the activation energy, whereby $\Delta E^{\ddagger}$ can be expressed as the transition state energy subtracted by the energy of the reactants[27]. At the transition state the activated complex exhibit high energy due to its unfavourable geometry, rendering it unstable. Whether the reaction progresses in reverse, breaking the bond, or forward, towards bond formation, the energy is either way lowered. This is the outcome because the strained geometry at the transition state gets alleviated, resulting in a more stable geometry. Presuming

these behave like any other equilibrium systems, the partition function for the transition state can be factorised[27]

$$q_{(AB)^{\ddagger}} = \overline{q^{\ddagger}} q_{\xi} \tag{2.54}$$

where $\overline{q^{\ddagger}}$ denotes the partition function for the transition state with the ordinary q_t, q_r and q_ν, while q_ξ is the partition function for the single non-equilibrium vibrational degree of freedom pertaining to bond forming event. Since the reacting bond along ξ is only partially formed, its corresponding vibrational frequency ν_ξ is small[27].

$$\nu_{\xi} = \frac{1}{2\pi} \left(\frac{k_{\xi}}{\mu} \right)^{\frac{1}{2}} \tag{2.55}$$

where k_ξ is the spring constant and $\mu_{AB} = \frac{m_A m_B}{m_A + m_B}$ is the *reduced mass*, a measure of the effective inertial mass. It allows a two-body system to be represented as a one-body system[27, 40].

Because the frequency ν_ξ and the spring constant k_ξ of the reacting bond is small, the partition function for vibration can be simplified, since $e^{-x} \approx 1 - x$ for small x[27].

$$q_{\xi} = \frac{1}{1 - e^{\frac{-h\nu_{\xi}}{k_B T}}} \approx \frac{1}{1 - (1 - \frac{h\nu_{\xi}}{k_B T})} = \frac{1}{\frac{h\nu_{\xi}}{k_B T}} = \frac{k_B T}{h\nu_{\xi}} \tag{2.56}$$

Immediately after the system reaches transition state, TST expect the system to continue towards the product state at frequency ν_ξ. The rate constant for this declining step is

$$k^{\ddagger} = \nu_{\xi} \tag{2.57}$$

However, not every oscillation with a corresponding ν will lead to the activated complex proceeding to product state. To account for deviation from theory, a *transmission coefficient* κ may be included, which in most cases is close to unity. Consequently, the rate constant pertaining to the downhill step becomes $k^{\ddagger} = \kappa\nu_\xi$[27]. Nevertheless, there are special cases where κ is larger than unity, such as in quantum mechanical tunneling, as briefly touched upon in the introduction to the theory. These scenarios occur when the reaction traverses through PES-barriers, causing κ to deviate from unity[27, 29].

2.4.3 The Eyring Equation and Connections to Arrhenius Equiation

The rate constant k can finally be obtained by substituting eq. (2.53), (2.54), (2.56) and (2.57) into eq. (2.43). This is known as the *Eyring equation*

$$\begin{aligned} k &= k^{\ddagger} K^{\ddagger} \\ &= k^{\ddagger} \left(\frac{q_{(AB)^{\ddagger}}}{q_A q_B} \right) e^{\frac{-\Delta E^{\ddagger}}{k_B T}} \\ &= k^{\ddagger} \left(\frac{\overline{q^{\ddagger}} q_{\xi}}{q_A q_B} \right) e^{\frac{-\Delta E^{\ddagger}}{k_B T}} \\ &= k^{\ddagger} \left(\frac{\overline{q^{\ddagger}} \left(\frac{k_B T}{h \nu_{\xi}} \right)}{q_A q_B} \right) e^{\frac{-\Delta E^{\ddagger}}{k_B T}} \\ &= \left(\frac{k_B T}{h \nu_{\xi}} \right) \nu_{\xi} \left(\frac{\overline{q^{\ddagger}}}{q_A q_B} \right) e^{\frac{-\Delta E^{\ddagger}}{k_B T}} \\ &= \left(\frac{k_B T}{h} \right) \left(\frac{\overline{q^{\ddagger}}}{q_A q_B} \right) e^{\frac{-\Delta E^{\ddagger}}{k_B T}} \\ &= \left(\frac{k_B T}{h} \right) \overline{K^{\ddagger}} \end{aligned} \tag{2.58}$$

where $\overline{K^{\ddagger}}$ represents the equilibrium constant for the stable degrees of freedom, since the unstable degrees of freedom (ξ) has been factored into $\frac{k_B T}{h}$[27]. This enables another famous expression where the equilibrium constant relates to activation parameters.

$$-k_B T \ln \overline{K^{\ddagger}} = \Delta G^{\ddagger} = \Delta H^{\ddagger} - T \Delta S^{\ddagger} \tag{2.59}$$

where $\Delta G^{\ddagger}$ is the *activation free energy*, $\Delta H^{\ddagger}$ is the *activation enthalpy* and $\Delta S^{\ddagger}$ is the *activation entropy*. Consequently, the rate constant can be expressed in terms of thermodynamic properties

$$k = \left(\frac{k_B T}{h} \right) e^{\frac{-\Delta G^{\ddagger}}{k_B T}} = \left(\frac{k_B T}{h} \right) e^{\frac{-\Delta H^{\ddagger}}{k_B T}} e^{\frac{-\Delta S^{\ddagger}}{k_B}} \tag{2.60}$$

Opinions seems to be divided whether a direct relation can be drawn between the Eyring equation and the Arrheius equation (2.61)

$$k = A e^{\frac{-E_a}{RT}} \tag{2.61}$$

where A is a pre-exponential factor and E_a denotes the activation energy. According to collision theory, these parameters translate to the rate constant, given that all collisions result in reaction, and the lowest energy required for a collision to result in a reaction respectively[9]. Some claim the $\Delta H^{\ddagger}$ term relates to the activation energy E_a seen in *Arrhenius equitation* (2.61) and $(k_B T h) \exp \Delta S^{\ddagger} k_B$ is the pre-exponential factor A[27]. Others disagree and argue that this connection is incorrect, because Arrhenius equation originates from empirical observations and the corresponding rate constant is the *macroscopic rate constant*. On the other hand, the rate constant in the Eyring equation accounts for mechanistic considerations and yields a *microscopic rate constant*. However, they do agree that some connections hold, particularly for single-step unimolecular or bimolecular reactions[6].

2.5 Kinetic Monte Carlo Simulations

The theoretical review has up until this point discussed reaction barriers at a microscopic level, where events (e.g. adsorption/desorption, bond forming/breaking, diffusion) in general are governed by electronic structure. When zooming out just a little, dynamics and thermodynamics comes into play at a mesoscopic scale. The focus was tuned into chemical reactions and deriving their corresponding rate constants with statistical mechanics. Lastly, at the macroscopic scale, where industrial reactors operates, phenomena such as transport as well as mass- and temperature-gradients in a reactor geometry needs to be taken into account[5].

kMC serves as a mesoscopic/macroscopic tool for simulating chemical kinetics within the field of chemistry. The advantage of using kMC is to study the time evolution of coupled processes using a stochastic approach, which arguably has a firmer physical basis from a kinetic theoretic point of view. The total number of molecules is considered to be large and the system has a temperature. Chemical reactions can be assessed by solving partial differential equations (PDE) in terms of concentrations, but it may lead to exponential decay. kMC offers an advantage by expressing the quantity of chemical species in terms of discrete integer numbers. Also, kMC handles oscillations and fluctuating behaviours in concentrations over time, as opposed to PDE[31]

The problem to be solved can (for consistency) be described by considering the same bi-molecular reaction discussed in TST, eq. (2.32). The rate equation is expressed a bit differently

$$r = \frac{dN_C}{dt} = -\frac{dN_A}{dt} = -\frac{dN_B}{dt} = k_f N_A N_B - k_b N_B \tag{2.62}$$

where N_A, N_B and N_C denotes number instead of concentrations[39]. There are several algorithm that can be used, each with its own characteristics and advantages, but the main traits of them can more or less be summarised as follows[5]:

1. Make a directory of all possible paths and their corresponding rate constants, k_p, and set some initial conditions
2. Compute $k_{tot} = \sum_p k_p$
3. Generate two random numbers $n_1, n_2 \in (0, 1]$
4. Determine the reaction paths q, that satisfies the condition $\sum_{p=1}^{q} k_p \geq n_1 k_{tot} \geq \sum_{p=1}^{q-1} k_p$
5. Execute the randomly chosen process q
6. Update the simulation time: $t + \Delta t$, where $\Delta t = -\frac{\ln(n_2)}{k_{tot}}$

kMC continues until stop condition is fulfilled. The motivation in having randomness in the algorithm, lies is why this is an stochastic approach and can be elaborated by considering the same analogy as (Andersen et al., 2019)[5]. Imagine a stack of segments, where each segment represents a process. The thickness of the segments are proportional to the rate constants. When drawing a random number $n_1 \in (0, 1]$ and multiplying it by k_{tot}, a new number is obtained q. This new

number is then compared to a condition and if it is fulfilled, it can be executed. Processes with a larger rate constant will consequently be chosen more often[5, 39].

3 | Computational details

This chapter entails information about the computational method used to obtain the results. It begins with a brief introduction outlining the software utilised, followed by the object behind the chosen methods. Details on how the code functions will also be given in detail.

3.1 Overview

The choice of which program to work with will in many cases be predetermined by the work environment. While prioritising familiarity with a particular software may be the natural choice, the value of exploring alternative options should not be underestimated as this might lead to discovering better suited tools for a specific task.

In this work, initial molecular geometries were generated using the molecular editor and visualiser *Avogadro*[33] version 1.2.0. All programming tasks were carried out within the *Python* ecosystem using the version 3.11.4. This was motivated by the growing popularity of Python in the field of chemistry, as well as other scientific disciplines, owing to its versatility, simplicity and abundant ecosystem of libraries and tools[62].

DFT-calculations were carried out using the *Python Simulations of Chemistry Framework*[72] (*PySCF*) version 2.3.0. *PySCF* offers a *Python* environment and is free and open-source[56]. One of the greatest advantages is that there is no need to learn a new input language from scratch, unlike almost every other software out there. Also, the *PySCF* packages is structured as any other python module, making it easy to integrate with other modules and customise into personalised scripts.

The primary object behind every code line was to develop a script capable of executing as many of the necessary tasks to obtain parameters for later use in determining the rate coefficient. The aim was to create scripts that are as versatile as possible. Additionally, efforts were made to automate the code for maximum efficiency. Next, the rate coefficients were derived from transition state theory and with the results from the DFT-calculations, they could be calculated. Finally, the rate coefficient were utilised in a kinetic Monte Carlo code, from which the reaction rates could be obtained. The code used in this work is the same that was used in the work by Hestad et al., 2016[37], where thermal decomposition of cyclohexane was investigated.

The calculations were primarily performed on a MacBook Pro equipped with a 1.4 GHz Quad-Core Intel Core i5 processor and 8 GB memory. This applies to the conformational analysis and kMC simulations. While the remaining calculations were carried on a MacBook Pro featuring the M2 chip, 16 GB memory and 10 Cores. Additionally, GitHub was utilised to enhance file management throughout this thesis.

3.2 DFT Calculations in PySCF

The *unrestricted Kohn-Sham* (UKS) approach was utilised in all calculations. The notation of the MEA conformations has been adapted from previous work, originally by Radom et al., 1973[59] and later carried on by Räsänen et al., 1983[60], Kelterer et al., 1994[42], Buemi et al., 1996[18], Silva et al., 1999[65], Vorobyov et al., 2002[78], da Silva et al., 2006[66] and Wang et al., 2009[80]. A general example of how a conformer is denoted is xYz, where x refers to the C−C−N−lp dihedral angle and *lp* denotes the lone pair on the nitrogen. The middle dihedral angle O−C−C−N is denoted by an upper-case letter, and z is the C−C−O−H dihedral angle. The one-letter abbreviations are: g or G for positive *gauche* (or *syn-clinal*), g' of G' for negative *gauche* (or *syn-clinal*) and t or T for *trans* (or *anti-periplanar*).

Initially, all conformers were calculated at a lower level of theory (B3LYP/cc-pVDZ) in gas phase to obtain a *reference* geometry. Later the were re-optimised using the continuum solvent model *ddCOSMO*, which is implemented in*PySCF*. Subsequently, they were re-optimised again, but at a higher level of theory (B3LYP/aug-cc-pVDZ) first in gas phase and then with a solvent. All minima were confirmed by the absence of imaginary frequencies, while all transition states were confirmed by the presence of one, and only one imaginary frequency, which corresponds to a saddle point on the PES. Vibration frequencies, rotation constants and symmetry numbers were calculated for the optimised structures using the `pyscf.hessian` module. The *geomeTRIC*-interface[81] in PySCF was employed to conduct the energy minimisation calculations. For the transition state searches, the *qsdopt*[71] module was utilised, with the method for the numerical hessian calculation set to`numhess_method = 'central'` (default is 'forward'). The results from running the script were written into various output files namely a *result.out*-file which contains all parameters required for later calculations, a *new.xyz*-file containing the optimised coordinated and a *error.log*-file containing the redirected terminal log.

3.3 Automatising Concurrent Calculations

All reference geometries was initiated manually, by executing the command:
`nohup script.py >& error.log &` in the terminal. A major drawback is that this approach necessitates continuous presence and interrupts a stable workflow, as the operator must monitor the each calculation from initiation to completion, before initiating the the next one to ensure the use of the computers maximum capacity. Automatising the code alleviates this constraint by allowing

the individual to work concurrently on other tasks while the calculations proceed iteratively. The entire scripts for geometry minimisation and transition state search can be found in the appendix B.1 and B.2.

3.3.1 Function to Find xyz-files

Calling the function `find_and_read_files` in listing 3.1 will locate and store information about the file name, file content (xyz-coordinates) and the file path of each molecule with the specified file extension `file_extension_to_find` within the specified directory `directory_to_search`.

```
1  def find_and_read_files(directory, file_extension):
2      name_list = []
3      atoms_init_list = []
4      file_path_list = []
5
6      for root, dirs, files in os.walk(directory):
7          for file in files:
8              if file.endswith(file_extension):
9                  file_path = os.path.join(root, file)
10                 file_path_list.append(file_path)
11                 with open(file_path, 'r') as f:
12                     atoms_init = f.read()[2:]
13
14                 atoms_init_list.append(atoms_init)
15                 filename = os.path.basename(file)
16                 modified_name = filename.replace(file_extension, '')
17                 name_list.append(modified_name)
18
19     return  file_path_list, atoms_init_list, name_list
```

Listing 3.1: Function to find, read and store file information and file paths

The function uses `os.walk` to traverse the entire `directory_to_search`. It iterates over all files, searching for the `file_extension`. If the file matches the `file_extension`, the code first stores the `file_path` and appends it to the `file_path_list`. Secondly, it opens the file and retrieves the file content starting from line 3, because the two former lines contain information about the number of atoms in the system and an empty line before the specific atoms with xyz-coordinates appear. The coordinates are stores in the `atoms_init` variable and appended it to the `atoms_init_list`. Third, the function then modifies the file name by isolating the *basename* and removing the `file_extension` which is common for all files. The `modified_name` is appended to the `name_list`. Finally, the function returns the three lists: `name_list`, `atoms_init_list` and `file_path_list`.

3.3.2 Function to Write and Read a xyz-file

To visualise the optimised geometry, a xyz-file is generated after the geometry optimisation.

```
1  def write_xyz(mol_opt, out_xyz):
```

```
    with open(out_xyz, 'w+') as f:
        f.write(f'{mol_opt.natm}\n\n')
        for i in range(mol_opt.natm):
            b2a = mol_opt.atom_coord(i, unit = 'ANG')
            symbol = mol_opt.atom_symbol(i)
            f.write(f'{symbol} \t {b2a[0]} \t {b2a[1]} \t {b2a[2]}\n')
        f.seek(3)   # Move the pointer to the beginning
        opt_coords = f.read()
    return opt_coords
```

Listing 3.2: Function write and read file content

The `write_xyz` function takes the arguments `mol_opt` and `out_xyz`, which is the mol-object of the optimised system and the file name to be generated, respectively. It opens the file in write mode, denoted 'w+'. The plus sign indicates that the file is opened for both writing and reading purposes. The function then writes the number of atoms in the system, followed by two new lines. Furthermore, it iterates over all atoms and writes the atom symbol with the corresponding coordinates on the same line, followed by a line break. This is the standard format for xyz-files. The variable `b2a` ensures that the coordinates are written in angstrom instead of bohrs, which is the default value. After the xyz-file has been written, the pointer is located at the bottom of the fill. `f.seek(2)` moves it to line 3 where the information about the atoms and the coordinates begin. Finally `f.read()` stores all information in the `opt_coords` variable, which will be used in the result file for comparison purposes.

3.3.3 Function for Geometry Optimisation

The function in listing 3.3 is responsible for conducting the following tasks: Geometry optimisation: it optimises the geometry to obtain the optimised energy and coordinates. Frequency analysis: It obtains the vibration frequencies. Thermochemistry analysis: It calculates the necessary parameters such as rotation constants and symmetry number, which will be utilised later to derive the rate constants.

```
def geometry_opt(basis, charge, diel, functional, pressure, temperature, verbose,
    ↪atoms_init, result_out, out_xyz, error_log):
    # Open the error log file to redirect the output
    with open(error_log, 'w') as error_file:
        # Redirect stdout and stderr to the error log file
        sys.stdout = sys.stderr = error_file

        mol = gto.M(atom = atoms_init, basis = basis, charge = charge, verbose =
            ↪verbose)
        mf = mol.UKS(xc = functional).ddCOSMO()
        mf.with_solvent.eps = diel
        mf.run()
        mol_opt = optimize(mf)

```

```
13          mf_opt = mol_opt.UKS(xc=functional).ddCOSMO()
14          mf_opt.with_solvent.eps = diel
15          mf_opt.run()
16          opt_energy = mf_opt.e_tot
17          energies = mf_opt.scf_summary
18
19          hess = mf_opt.Hessian().kernel()
20          freq_info = thermo.harmonic_analysis(mol_opt, hess)
21          thermo_info = thermo.thermo(mf_opt, freq_info['freq_au'], temperature,
                ↪pressure)
22          # thermo_info = thermo_info[0] # Neccessary for PySCF 2.4.0
23          vibration_frequencies = np.array(freq_info['freq_wavenumber'])
24          rotation_constants = thermo_info['rot_const'][0]
25          symmetry_number = thermo_info['sym_number'][0]
26
27      # Restore stdout and stderr to their original state
28      sys.stdout = sys.__stdout__
29      sys.stderr = sys.__stderr__
30
31      opt_coords = write_xyz(mol_opt, out_xyz)
32      write_result(result_out, atoms_init, opt_energy, energies, opt_coords,
            ↪freq_info, thermo_info, vibration_frequencies, rotation_constants,
            ↪symmetry_number)
33
34      return opt_energy, energies, opt_coords, freq_info, thermo_info,
            ↪vibration_frequencies, rotation_constants, symmetry_number
```

Listing 3.3: Function to do geometry optimisation, vibration analysis

The `geometry_opt` function is consequently responsible for the most expensive calculations. Further more, to avoid the error log of all molecules ending up in one single log file the script redirects standard output and standard error to a separate error log file for the corresponding calculation. It is to ensure that any error message generated, ends up in the file corresponding to the associated calculation. This is achieved by setting `sys.stdout` and `sys.stderr` equal to the corresponding `error_file`. Since the optimisation method is called in the line `mol_opt = optimize(mf)`, the error log might as well be opened in the very beginning and the subsequent code related to the calculation should then be placed within this code block.

Since the stdout and stderr were redirected from the terminal to an error log file, `sys.stdout = sys.__stdout__` and `sys.stderr = sys.__stderr__` are implemented to restore the standard error and the standard output to its original state as the comment states. Essentially, this means that any subsequent output or error will be written to the terminal (unless `>&` is used of course). Excluding these lines may cause an error, but regardless, the calculation will still proceed nonetheless, and result files will be generated.

3.3.4 Configuration and Concurrent Processing

The last section contains the initial set up configuration, but most importantly, the script uses the `ProcessPoolExecutor` from the `concurrent.futures` module to achieve concurrent processing of multiple molecules.

```
    directory_to_search = f'/Users/kristine/pyscf/mea/minima/gasphase/augccpvdz'
    file_extension_to_find = f'_minimum_b3lyp_augccpvdz.xyz'
    result_path = f'/Users/kristine/pyscf/mea/water/minima'

    name_list, atoms_init_list, file_path_list = find_and_read_files(
        ↪directory_to_search, file_extension_to_find)

    # Use ProcessPoolExecutor to run tasks concurrently
    with ProcessPoolExecutor(max_workers = workers) as executor:
        tasks = []
        for name, atoms_init, file_path in zip(name_list, atoms_init_list,
            ↪file_path_list):
            mol_dir = os.path.join(result_path, basis, name)
            if not os.path.exists(mol_dir):
                os.makedirs(mol_dir)

                out_xyz = f'{mol_dir}/{name}_{solvent}_{state}_{functional}_{basis
                    ↪}.xyz'
                result_out = f'{mol_dir}/{name}_{solvent}_{state}_{functional}_{
                    ↪basis}.out'
                error_log = f'{mol_dir}/{name}_{solvent}_error.log'

                # Start a new task for each molecule
                task = executor.submit(geometry_opt, basis, charge, diel,
                    ↪functional, pressure, temperature, verbose, atoms_init,
                    ↪result_out, out_xyz, error_log)
                tasks.append(task)

            else:
                print(f'The {mol_dir} directory already exists')

        for task in tasks:
            task.result()
```

Listing 3.4: Parameter setting and function calling

The block `if __name__ == '__main__':` sets up several parameters of the calculation, including the name of directory to search through and the file extension to look for. These last two parameters help narrow down the search area and specify the desired files. It is also important to note that these parameters can be modified to target other directories or file types based on what type of calculation is desired and under which conditions.

The for loop iterates through the lists `names`, `atoms_init_list`, and `file_paths` simultaneously

using `zip()` to group corresponding elements together. This is crucial to ensure that the files are processed correctly.

Inside the loop, filenames for output files (`out_xyz`, `result_out` and `error_log`) are generated based on the specified parameters and name variable. Before running the `geometry_opt` function, the code checks if the `result_path` file already exists. If it does exist, the calculation is skipped for that particular file, and a message will be printed to the error log file to indicate the skip.

The `ProcessPoolExecutor` class contains the argument `max_workers`, specifying the maximum number of processes the pool can contain. Here the number of maximum tasks is determined by the parameter `workers`. The function `geometry_opt` is the function that will be executed asynchronously and gets submitted under the `task` variable to the process pool executor. `task_list` is a list containing all the `geometry_opt` tasks submitted to process pool executor. Each iteration of the loop involves processing a different molecule with its unique set of name, file containing input coordinates and the corresponding file path. The tasks runs concurrently and enhance the efficiency through parallelisation of the tasks.

3.4 Connecting Two Minima to a Maximum

In order to gain an overview of how to connect all conformations of MEA, the use of a common measurement comes in handy. A natural choice has been to considered the change in dihedral angles, as that is a common measurement that changes going from one conformation to another. Since PySCF does not calculate it, it has in this work been derived. The derivation can be found in appendix A.3 and the code can be found in appendix B.3.

The optimised geometries obtained from the DFT-calculations has been extracted from the generated xyz-files. This involved pasting entire xyz-files as multi strings in a *Jupyter Notebook* (JN) environment. Within the JN-environment, the code from the appendix could be employed to calculate the dihedral angles for all conformations of MEA. Furthermore, the python package *py3Dmol* version 2.0.4 has been utilised to visualise the conformers while calculating the dihedral angles. This step was essential due to the disordering of atoms observed when the input xyz-files were generated individually in Avogadro. As a second verification the dihedral angles (except those involving lone pairs), they were checked in Avogadro using the *Click to Measure* tool.

3.5 Calculating the Rate Constants from TST

The rate constants were calculated based on the theoretical framework provided in the theory section 2.4, which described the transition state theory approach for determining rate constants. The code that has been used for these calculation can be found in the appendix B.6. Furthermore, the functions used to calculate the rotational and translational partition functions, along with the vibrational temperature, were based on PySCF's *thermo* function. Some modifications were made

due to discrepancies in the units between the theoretical framework and PySCF's default values. Finally, the rate constant could be implemented in the kMC-code to run simulations.

4 | Results and Discussion

This chapter will first present results obtained from the conformers at minima. These results will also be seen in light of the gauche-gauche-interactions, which for this situation results in favourable hydrogen bonding. Next the results pertaining to the transition state will be presented. Issues related to calculations will be discussed and attempts to rationalise them. Additionally, some reasoning concerning the choices made will be addressed. Following this, some more technical aspects of this work will be explained. Finally, the results from the calculations of the rate and rate constant will be presented. A few calculations concerning secondary degradation paths will also be given at the end, and these will be based on results from the former calculations.

4.1 MEA Conformational Exploration in PySCF

25 conformations of MEA conformers has been calculated at both minima and transition state (TS), connecting two minima to obtain the interconversion barrier of the conformers. Relative energy minima and dihedral angles for all conformers determined in this work can be found in Table 4.1. From these results, 1g'Gg'/15gG'g was found to have the lowest absolute energy, consequently all other conformers are reported relative to it. The conformers with the lowest energies, all exhibit intramolecular hydrogen bonding: 1g'Gg', 2gGg'[1], 3gGt, 4tGt, 5tGg, 6gGg. The first two are stabilised by NH $\cdots$ O hydrogen bonds (HB), while the remaining four are stabilised by OH $\cdots$ N HB. Overall, conformers where the central dihedral O–C–C–N is in the *gauche* form tend to exhibit lower energies compared to *trans* from. This is in consonance with previous theoretical studies[18, 24, 65, 78, 80]. Ideally, equivalent isomers should exhibit the same energy. However, they do not. This can be explained by slight differences in their xyz-inputs. These small discrepancies might have propagated through the optimisation processes and are reflected in variations of energy, bond length and angles. This applies throughout the discussion of conformers.

The minima of the conformers 14g'Gg/27gG'g' could not be determined in gas phase. However they converged into structures with expected dihedral angles of the conformers. Unfortunately, the minima could not be confirmed by the absence of imaginary frequencies.

[1] Not determined in this work

Table 4.1: Conformers (including their isomers) with dihedral angles and relative electronic energy[a] of the minima, calculated at B3LYP/aug-cc-pVDZ level of theory including effects using COSMO

Conformer	**C−C−N−lp**[b]	**O−C−C−N**[b]	**C−C−O−H**[b]	**Energy**[c]
1g'Gg'	−42.4	53.4	−38.3	0.00
15gG'g	42.8	−54.0	38.7	0.00
2gGg'				[d]
16g'G'g				
3gGt	57.5	65.4	−174.6	1.36
17g'G't	−57.3	−65.0	174.9	1.36
4tGt	−177.0	63.5	−175.0	1.30
18tG't	175.9	−63.4	173.9	1.29
5tGg	−175.6	60.4	67.3	1.25
19tG'g'	175.6	−60.5	−67.2	1.19
6gGg	59.3	62.7	67.6	1.25
20g'G'g'	−59.3	−62.3	−67.6	1.24
7tGg'	174.4	63.3	−71.5	1.38
21tG'g	−174.8	−62.7	71.9	1.40
8tTg	179.2	179.1	66.2	1.67
22tTg'	179.6	−179.0	−66.3	1.67
9tTt	−180.0	−179.8	179.5	1.80
10gTt	49.5	178.1	178.6	2.11
23g'Tt	−49.7	−178.4	−179.3	2.11
11gTg	49.8	177.7	67.3	2.04
24g'Tg'	−49.7	−177.9	−66.1	2.02
12g'Tg	−50.0	179.2	65.3	2.00
25gTg'	49.4	−179.6	−66.0	1.99
13g'Gt	−51.8	68.7	−179.3	2.36
26gG't	52.5	−67.9	178.0	2.32
14g'Gg	−53.8	64.4	63.3	2.20[e]
27gG'g'	53.4	−64.3	−63.4	2.17[e]

[a] Sum of electronic and zero-point energies at the B3LYP/aug-cc-pVDZ level

[b] Ångstrom

[c] kcal mol^{-1}

[d] No local minimum found

[e] Local minimum not confirmed

The length of the HB are presented in Table 4.2. From these results, the shorter HB tend to correlate with increased stabilising effect, giving insight into why the 1g'Gg'/15gG'g conformers exhibits the lowest energy. Additionally, a natural bond orbital (NBO) analysis conducted by Wang et al., 20009[80] suggested that this lowering is attributed by delocalisation interaction, which results in lowering the energy levels of the occupied orbital. Results from other studies[65, 78, 80] are also presented in Table 4.2. All of them in gas phase. Among these studies, the one by Wang et al., 2009[80] is the one that resembles this work the most, using similar level of theory.

The five conformer (including their isomers) exhibiting HB are shown in Figure 4.1. The conformer are displayed with atom label and number according to their ordering in the xyz-file. The very mellow yellow dashed lines represent the hydrogen bonds ranging from the oxygen/nitrogen atom to either of the hydrogen atoms.

Table 4.2: Intramolecular hydrogen bond (HB) length[a] of conformers (including their isomers) and relative electronic energy (same as in table 4.1).

Conformer		**HB**[a]	**HB**[c]	**HB**[d]	**HB**[e]	**Energy**[b]
1g'Gg'	OH ⋯ N	2.210	2.293	2.289	2.32	0.00
15gG'g		2.218				0.00
2gGg'	OH ⋯ N		2.580	2.587	2.55	[c]
16g'G'g						
3gGt	NH ⋯ O	2.589	2.533	2.538	2.49	1.36
17g'G't		2.589				1.36
4tGt	NH ⋯ O	2.621	2.600	2.599	2.55	1.30
18tG't		2.607				1.29
5tGg	NH ⋯ O	2.673	2.636	2.641	2.58	1.25
19tG'g'		2.671				1.19
6gGg	NH ⋯ O	2.637	2.568	2.575	2.53	1.25
20g'G'g'		2.632				1.24

[a] Å, this work

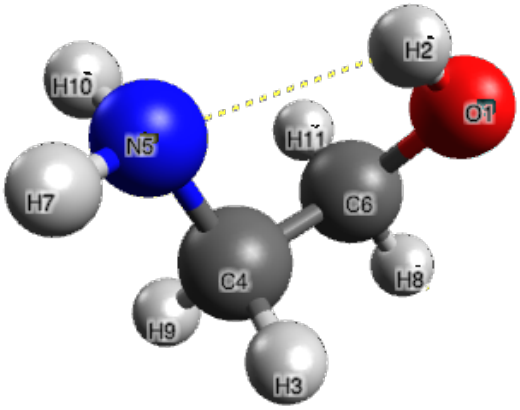
[b] kcal mol^{-1}, this work

[c] No local minimum found

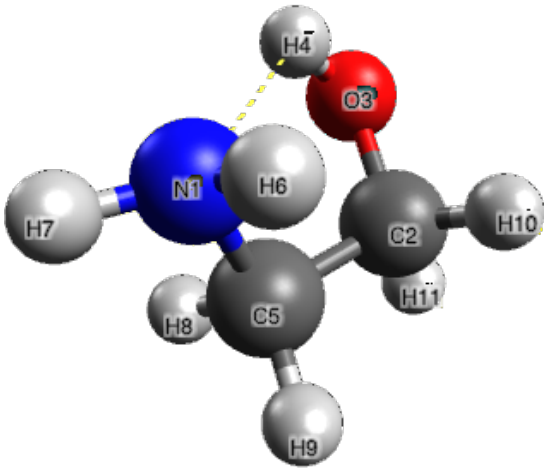
[d] Å, B3LYP/6-311G(2d,2p) level from Vorobyov et al., 2002[78]

[e] Å, B3LYP/aug-cc-pVDZ level from Wang et al., 2009[80]

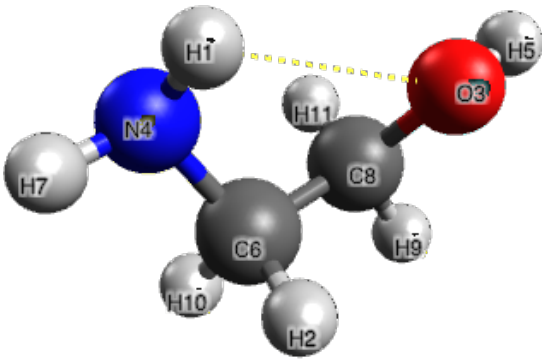
[f] Å, HF/6-31G* level from Silva et al., 1999[65]



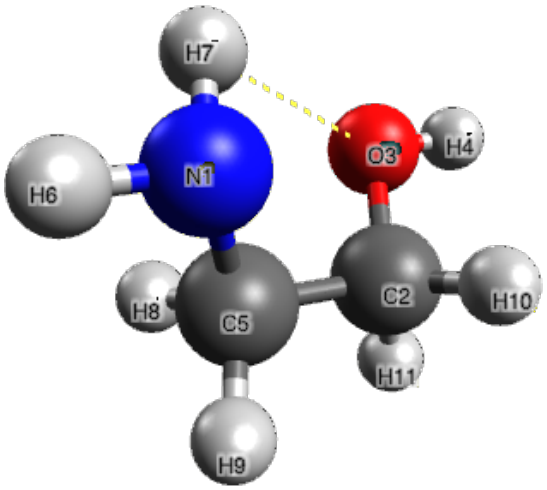
(a) 1g'Gg'



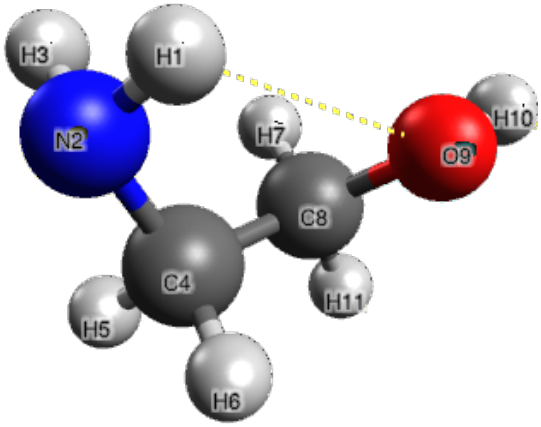
(b) 15gG'g



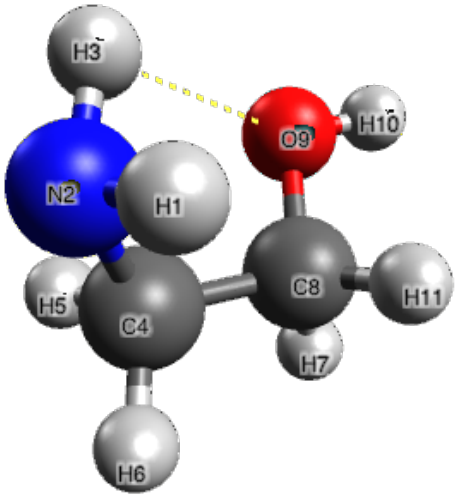
(c) 3gGt



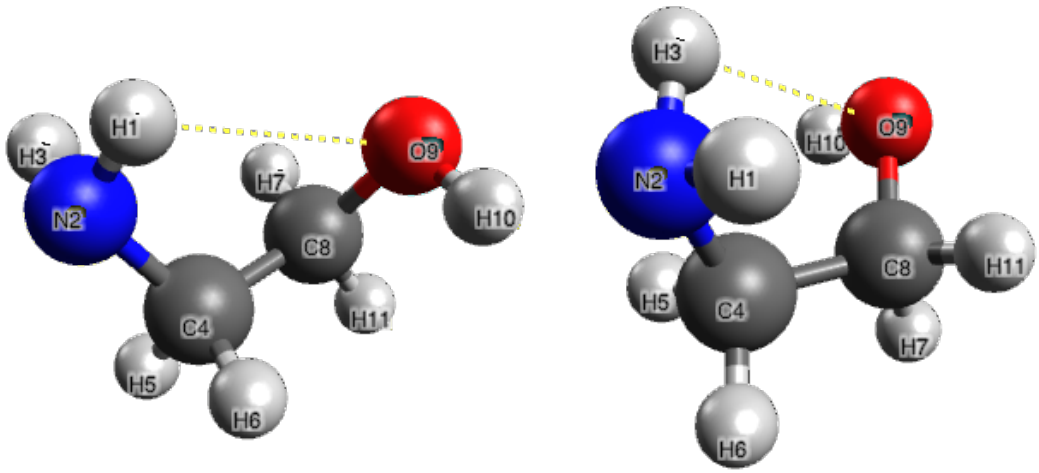
(d) 17g'G't



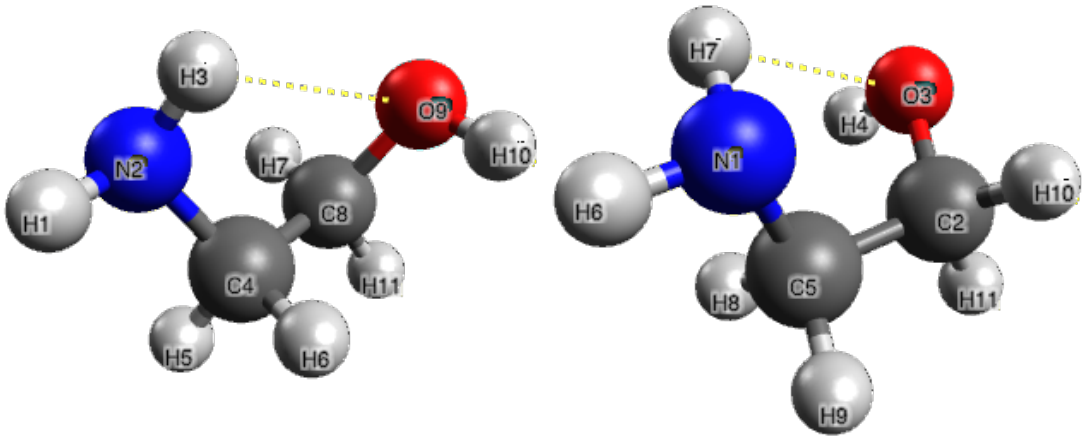
(e) 4tGt'



(f) 18tG't



(g) 5tGt **(h)** 19tG'g'



(i) 6gGg **(j)** 20g'G'g'

Figure 4.1: The conformers with hydrogen bonds

Interconversion barriers, relative its corresponding minimum are presented in Table 4.3. The TS energies typically span between 0.66-5.15 kcal mol^{-1}, but for interconversion between two states where both exhibit intramolecular hydrogen bonds, the barrier are even lower. This special situation occurs between 17g'G't and 16g'G'g and also between 4tGt and 5tGg. The values for both 2gGg'/16g'G'g and 14g'Gg/27gG'g' are missing because their minima could not be determined or confirmed. The angles at the TS are *eclipsed*, meaning the angels tend to take a value between the two staggered conformations, as expected.

Table 4.3: Conformers (including their isomers) with dihedral angles and interconversion barrier[a], calculated at B3LYP/aug-cc-pVDZ level of theory including effects using COSMO

Conformer	**C−C−N−lp**[b]	**O−C−C−N**[b]	**C−C−O−H**[b]	**Energy**[c]
1g'Gg'	−54.1	121.0	−64.5	5.15
15gG'g	51.9	−110.8	63.0	4.77
2gGg'				[d]
16g'G'g				
3gGt	54.1	121.6	178.6	3.71
17g'G't	−58.1	−65.4	120.4	0.66
4tGt	177.0	59.1	124.7	0.96
18tG't	178.9	−121.1	−178.7	3.64
5tGg	−177.3	119.6	67.0	3.81
19tG'g'	177.7	−119.0	−67.8	3.88
6gGg	53.7	119.9	68.2	3.84
20g'G'g'	−55.3	−105.8	−69.7	3.34
7tGg'	−172.2	−0.5	−68.4	5.12
21tG'g	173.5	−0.2	70.1	5.09
8tTg	−177.7	118.5	68.7	3.41
22tTg'	−177.4	121.6	−64.7	3.34
9tTt	178.2	−121.1	−178.4	3.16
10gTt	49.5	178.1	178.6	3.05
23g'Tt	−53.1	120.8	179.2	3.06
11gTg	53.8	120.3	67.8	3.06
24g'Tg'	−53.7	−179.6	−66.0	3.12
12g'Tg	−55.6	118.9	68.7	3.40
25gTg'	54.5	−118.3	−69.4	3.41
13g'Gt	−53.5	116.3	178.6	2.76
26gG't	109.7	−60.3	178.6	1.76
14g'Gg				[d]
27gG'g'				

[a] TS energy with respect to the corresponding minima at the B3LYP/aug-cc-pVDZ level

[b] Degrees

[c] kcal mol^{-1}

[d] No local maximum found

To provide an overview of the presented information, Figure 4.2, illustrates how the states connect. The blue-coloured states signifies minima, while the red represent TSs. Additionally, states with intramolecular hydrogen bonding, are highlighted with a blue background. Arrows with arrowhead indicate that a calculation has been carried out where that state has been the input and resulted in the state the arrowhead points to. The lines without arrowheads represent how a state is implicitly accessed. By noting what dihedral angle is changed, the next minimum can be *predicted*. Lastly, the dashed line with an arrowhead signifies that the connection was drawn based on a different approach than the others. Unlike the latter, the arrowhead points away from a TS, representing that a energy minimisation has been carried out to draw the connection.

Unfortunately, some of the states were missing due to the consequential failure to determine the conformers at their respective minima and then conduct a TS search on those structures. What is particularly problematic is that these three states are not accessible. One of them exhibits intramolecular hydrogen bonding, which has been shown in another study by Silva et at., 1999[65] to be substantially populated, constituting 6.9 % of the entire population. That being the third most stable conformer in that work. In another study by Wang et al., 2009[80] that same conformer were found to amount for 9.52 % of the population at 296 K.

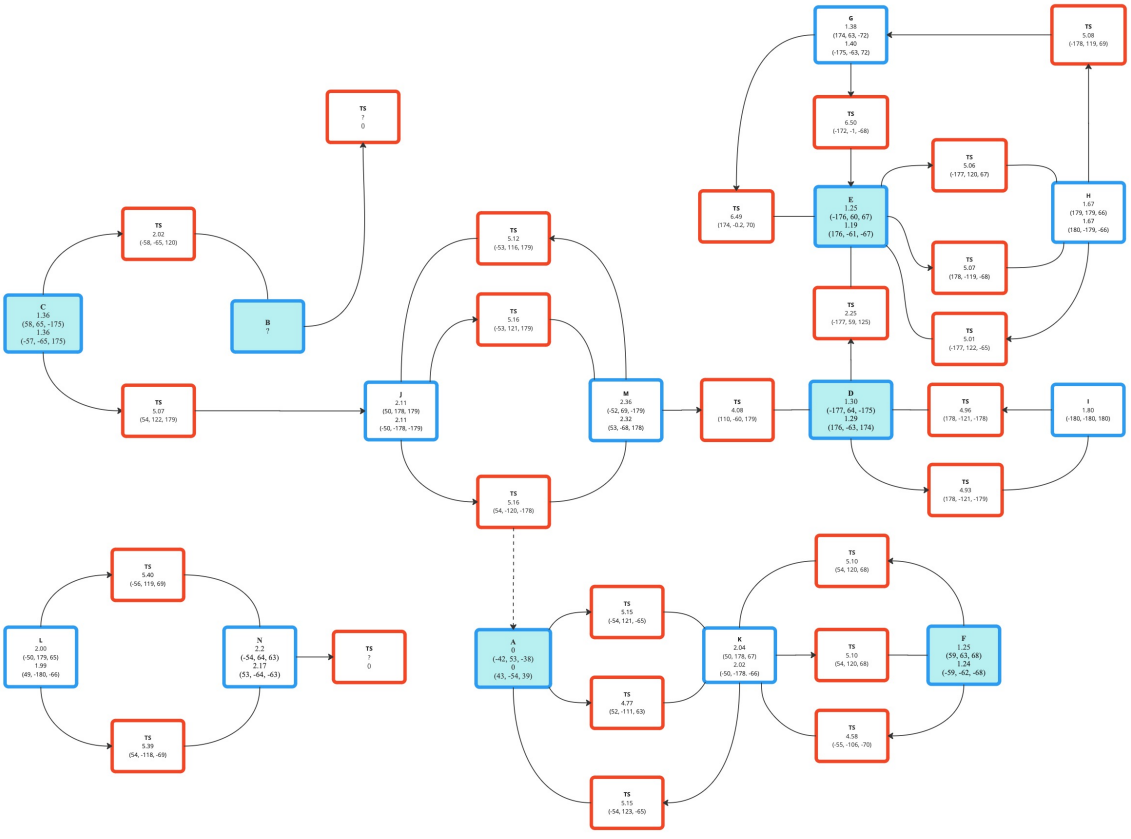


Figure 4.2: Map of the conformer interconversions

4.1.1 Conformer Stability

As previously stated, the equilibrium geometries of the conformers 2gGg' and its isomer 16g'G'g could not be determined either in gas phase or aqueous solution. When these were tried to be determined, the optimisation process converged to the 1g'Gg' and 15gG'g respectively. Despite several attempts to perturb the geometry slightly, the optimisation process consistently converged to the 1g'Gg' and 15gG'g conformers. This suggests that they are not stable in either the gas phase or solvent.

Furthermore, the 14g'Gg/27gG'g' conformers were not possible to determine in gas phase as the geometry optimisation of them resulted in the conformers 1g'Gg'/15gG'g. However, in aqueous solution they were identified. This suggests that they are more stable in a solvent than in gas phase. In a molecular dynamics study, by da Silva et al., 2007[66], MEA was studied both as a pure liquid and aqueous solution at 298 K and 333 K. Although the 14g'Gg conformer was identified, with a relative energy of 4.83 kcal mol^{-1} at B3LYP/6-311++G(2d,2p)-level it is worth noting that this did not correspond to a local minimum as it was obtained from a restrained geometry optimisation.

More interestingly, a theoretical study of MEA carried out by Wang et al., 2009[80] manged to identify the 2gGg' conformer using the same functional and basis set as in this work, with a relative energy of 1.59 kcal mol^{-1}. Other studies have supported this findings using different theoretical approaches. Buemi, 1996[18] identified the conformer at the MP2/6-31G** level, with a relative energy 2.05 kcal mol^{-1}. Additionally, Silva et al., 1998 identified it at the HF/6-31G* level with a relative energy 1.57 kcal mol^{-1}. Vorobyov et al., 2002[78] carried out a computational study exploring various equilibrium structures of the MEA conformations at the B3LYP/6-311++G(2d,2p) level in gas phase. The 2g'Gg conformer was identified in this study, with a relative energy of 1.31 kcal mol^{-1} at 298 K in gas phase.

The study by Wang et al., 2009[80] moreover suggested that increasing the temperature might reveal less stable conformations in the temperature dependent PES of MEA. However, no specific temperature increase was recommended. Attempts to increase the temperature to 393.15 K (desorber condition), both in gas phase and aqueous solution, resulted in the conformer 2gGg' reverting to 1g'Gg', indicating its independence from temperature.

Kelterer et al., 1993[42] conducted an ab-initio RHF study, exploring the basis set dependence of MEA's conformations. They found that conformers, including 2gGg' and 14g'Gg, where sensitive to changes in basis set. Most of the basis set tried out was Pople basis set and the minimal basis sets STO-nG. Notably, this sensitivity was observed for RHF and MP2 calculations. Although efforts were made to change the basis set, the conformer reverted back to 1g'Gg'.

It should emphasised that none the above studies[18, 66, 78, 80] were not able to identify the 14g'Gg/27gG'g' conformers, as in this work. However, they did manage to identify the 2gGg' conformer which may have a grater importance due to its intramolecular hydrogen bonding.

Additionally, only two previous studies has been able to determine the conformers 14g'Gg/27gG'g', namely Random et al., 1973[59] and Räsänen et al. 1983[60].

4.2 Conformer Rate Constants via TST

The value of having the gas phase structures became apparent at this point. As previously discussed, the numerous emerging imaginary frequencies posed a significant obstacle for further progress when COSMO was considered. Fortunately, the frequencies from the gas phase structures could be utilised as a workaround, allowing for the calculation of the vibrational partition function, that enters the pre-factor in the rate constant.

The calculated rate constants from TST theory are presented in Table 4.4. A new notation is introduced, designating each equivalent conformer as a common state, ranging from A to N. Despite the inability to determine state B and N, they are still included in the table for completeness. Moreover, the magnitude of the calculates rate constants are inverse proportional to the relative energies of the states. This can be rationalised by the fact that the rate constants characterises the probability of a the system transitioning from one state to another[79].

For example, in the first interconversion, the system is in state A and jumps to state K, where state A has exhibits the lowest energy. Here, the forward rate constant is less than the reverse, reflecting the higher likelihood of the system transitioning from state K to A, rather than from A to K.

It is hard to miss the six marks in Table 4.4, denoting that the rate constant could not be determined. This is becase the equilibrium geometry could not be determined, and consequently, the TS search could not be conducted either. This applies to the states B and N. However for the states C and L, some rate constants are calculated. Due to the fact that calculating the rate constant only consider the minimum and the corresponding TS, these rate constants could be determined. Unfortunately, the missing conformers propagates into the simulation. Since the states do not have a rate constant for both forward and reverse reaction, they cannot be included in the kMC simulation as that would violate the principle of detailed balance. Essentially, this means that for all pairs i and j, the amount of escaping from i to j, must equal the amount of escaping from j to i. This ensures that the system reaches equilibrium.

Table 4.4: Rate constants derived from TS theory, utilising the code in appendix B.6 with rates obtained from kMC by[37]

State	Interconversion	k_f [a]	k_r [a]
A	1g'Gg' $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 24g'Tg'	6.63e+08	1.23e+10
	15gG'g $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 11gTg	1.39e+09	2.67e+10
B	2gGg'	[b]	[b]
	16g'G'g		
C	3gGt $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 10gTt	6.16e+09	1.74e+10
	17g'G't $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 16g'G'g	1.41e+12	[b]
D	4tGt $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 5tGg	8.52e+11	7.74e+11
	18tG't $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 9tTt	6.09e+09	1.29e+10
E	5tGg $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 8tTg	4.78e+09	9.13e+09
	19tG'g' $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 22tTg-	4.45e+09	8.72e+09
F	6gGg $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 11gTg	5.48e+09	1.48e+10
	20g'G'g' $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 24g'Tg'	1.28e+10	3.48e+10
G	7tGg' $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 19tG'g'	6.13e+08	4.69e+08
	21tG'g $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 5tGg	6.22e+08	4.77e+08
H	8tTg $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 5tGg	8.93e+09	4.68e+09
	22tTg' $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 7tGg'	1.02e+10	6.78e+09
I	9tTt $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 18tG't	1.23e+10	5.83e+09
J	10gTt $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 26gG't	1.71e+10	1.86e+10
	23g'Tt $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 13g'Gt	1.57e+10	2.22e+10
	10gTt $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 15gG'g	1.71e+10	7.85e+08
K	11gTg $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 6gGg	1.48e+10	5.45e+09
	24g'Tg' $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 1g'Gg'	1.32e+10	7.07e+08
L	12g'Tg $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 14g'Gg	9.46e+14	[b]
	25gTg' $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 27gG'g'	9.70e+09	
M	13g'Gt $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 23g'Tt	2.07e+10	1.73e+10
	26gG't $\underset{k_r}{\overset{k_f}{\rightleftharpoons}}$ 18tG't	1.47e+11	3.92e+10
N	14g'Gg	[b]	[b]
	27gG'g'		

[a] s^{-1}

[b] not possible to determine

4.2.1 Challenges and Assumptions in TST

In the theory section on TST, some assumption were made regarding the determination of the rate constants. Initially, it was assumed that the reaction coordinate was separable, allowing the chemical reaction to be described by a single reaction coordinate. Next, it was assumed that once reached the TS, the structure must proceed to the product state. Lastly, it was assumed that the energy of the particles follows a Boltzmann distribution. These assumptions will consequently have an impact on the results.

TST assumes that the reaction can be described by a single reaction coordinate. While this may be sufficient for some reactions at low temperatures, it can oversimplify more complex reactions at high temperatures. TST also assumes that the reaction coordinate corresponds to the lowest barrier on the PES. While this might apply for low temperatures, it is not necessary applicable for reactions at higher temperatures, where the higher energy states are more populated and more complex collisions give rise to TSs that are not necessarily the lowest[27]. Improved alternatives exist, such as *Variational TS Theory* (VTST). Allison and Truhlar 1998[4] conducted a study where they compared TST and VTST, discovering that VTST is more accurate than TST at high temperatures. However, at lower temperatures TST seemed to be more accurate, artificially, because it usually overestimates the rates and does not account for *tunneling*, which is more prominent at lower temperatures[26].

This further leads to the assumption that every trajectory crosses the TS once, which translates to the transmission coefficient κ being equivalent to unity. As already implied, TST treats the nuclei according to classical mechanics, and does not take into account quantum mechanical effects, such as *tunnelling* or *rare events*, causing κ to deviate from unity. Eyring[29] himself argue that these aspects can be neglected, justifying it by the fact that the barriers are flat around the TS, which makes tunneling less feasible. However, at low temperatures, tunneling contributions dominates causing $\kappa > 1$, while at higher temperatures re-crossing is important resulting in a $\kappa < 1$[40]. These effect can be approximated by according to several available methods, such as the *Bell* correction[13], the *Wigner* correction or the *Wentzel–Kramer–Brillouin* correction, which all aim to include semi-classical contributions[40].

The last assumption, concerning that the energy of the particles follow a Boltzmann distribution, implies that each reactive intermediate is persistent enough for a Boltzmann distribution to be established. This is particularly problematic in multi-step reactions, where they might be short-lived and follow a non-statistical distribution of intermediates[6, 23].

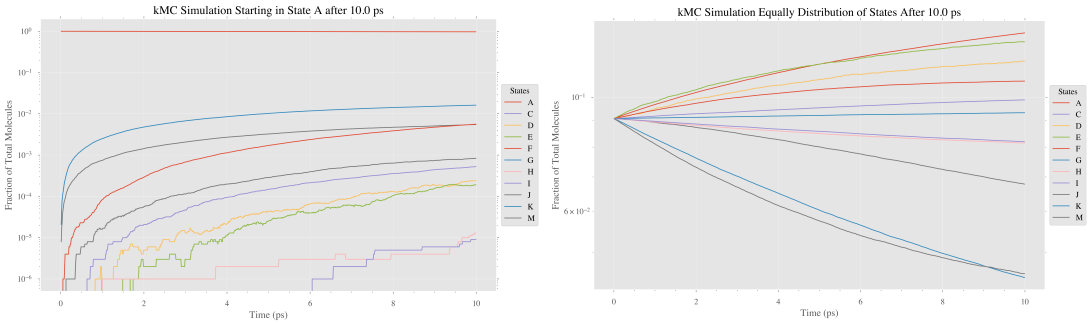
4.2.2 kMC Simulation on MEA Conformers

Six kMC simulation has been carried with six different initial conditions, each representing a slightly different distribution of conformers among the states. In all simulations, the number of conformers was 10^6, with each conformer either being distributed almost equally among the states or all starting

in state A. The number was chosen to give a statistically representative sample of the population. After 10 ps, 100 ps and 5 ns, the distribution of conformers can be assessed using Figures 4.3, 4.4 and 4.5. These intervals were the most interesting to consider in addition to being the most relevant according to literature[52, 66]

Overall, results indicate that state A is the most populated, likely owing to its favourable intramolecular hydrogen bonding, which stabilises it. Since it is in abundance, it will not be discussed in detail in the following discussion.

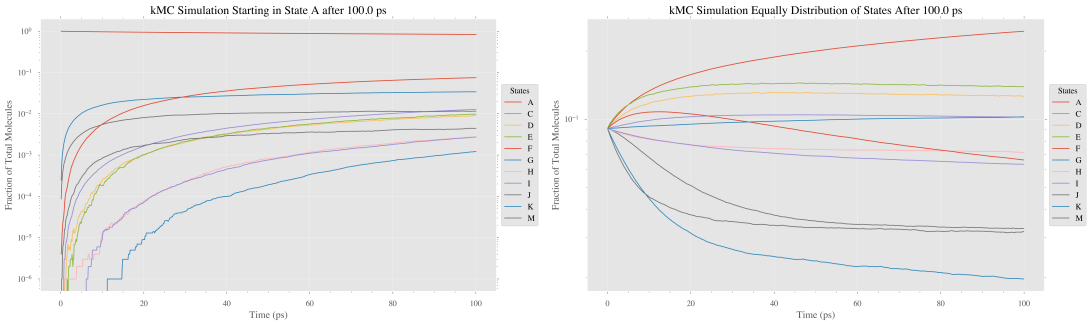
The evolution after 10 ps is presented in Figure 4.3. In Subfigure 4.3a, the fraction of state K and J increases rapidly within the first picosecond (ps). After about 6 ps, they seem to increase at a slower rate. Additionally, their increase seem to stabilise. Additionally, they appeared to be more stable than the other states. When considering the situation depicted in Subfigure 4.3b, both simulations show that state F is among the more abundant conformers after 10 ps. More interestingly, it shows that the state E has a greater fraction of conformers than A within the first 4.5 ps. However, the difference becomes increasingly smaller as time approaches 4.5 ps. State D also seems to increase quite stable. Overall, simulation where all conformers start equally seems to evolve most stable.



(a) The states K, F and J are the most dominant **(b)** The states E, D and F are most dominant

Figure 4.3: Conformer distribution within 10 ps

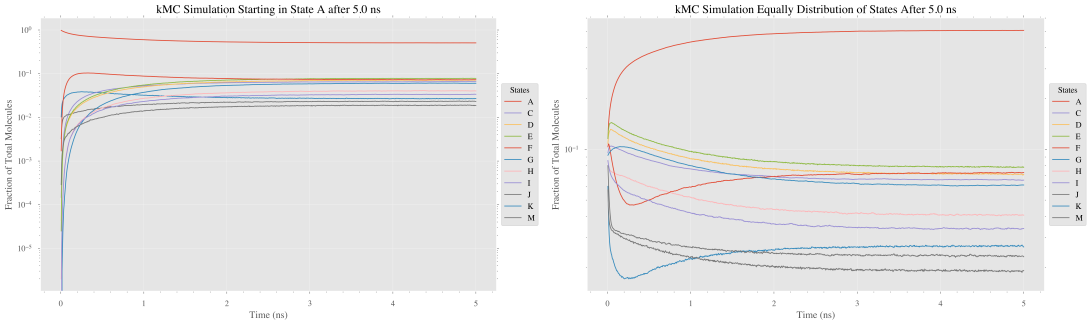
Moving on to Figure 4.4, which has ten times as long simulation time as the previous, the same trends can be observed here. What is interesting is that state F bypasses the states J, around 10 ps, as seen in Subfigure 4.4a. In Subfigure 4.4b, state E and D remains the most dominant. However, after the first 25 and 30 ps, state C decreases below the states C and G, respectively. Although the populations in C and G are about the same (10.21% and 10.25%) at 100 ps. More notably, there is a noticeable decline in the population of state F at a higher rate than the remaining states. This is perhaps coupled to the increase of state A, since state F is directly connected to state, which further leads to state A.



(a) The states F, K and C are the most dominant **(b)** The states E, D and G are most dominant

Figure 4.4: Conformer distribution within 100 ps

Finally, the last simulation has gives insight in the distribution of states during 5 ns. The results is depicted in Figure 4.5. Essentially, they give the same information, where the states E, F and D are the most populated. Around 4 ns the systems appears to reach an equilibrium, as the further simulation time does not lead to any additional changes. In Subfigure 4.5b, it can more easily be seen that the states F, J and K are the main contributors to the increase of state A, when the initial condition are started considering all molecules equally distributed. This can be seen in connection with their rate constant for the forward (towards state A) reaction being larger than the reverse, reflecting the likelihood of crossing that barrier.



(a) The states E, F and D are the most dominant **(b)** The states E, F and D are most dominant

Figure 4.5: Conformer distribution within 5 ns

The final conformer distribution can be found in Table 4.5. It is clear that, state A dominates, which is expected, followed by states E, F, D and C. Common for all is they exhibit intramolecular hydrogen bonds. State G seem to stand out from the the latter states as it is quite populated compared to the remaining conformers. The forward and reverse rate constants are of similar magnitude, suggesting that neither state, from which they originate or move to, appears to be significantly preferred over the other. From literature, state G has furthermore been considered the most populated state both in pure MEA and in infinite dilution in water[66, 83].

Table 4.5: Conformer distribution at 5 ns

States	A	C	D	E	F	G	H	I	J	K	M
Distribution[a]	50.74	6.57	7.15	7.84	7.29	6.15	4.10	3.38	2.31	2.65	1.9

[a] Percentage of total number of conformers (10^6), started equally distributed

It must be emphasised once again that these result does not reflect the complete picture if all conformers were present. Perhaps, if all rate constant were possible to determine the results would have been different. With that being said, results from other studies indicate that the missing conformers in this work does not constitute a large population in pure solution or in infinite dilution in water.

4.3 Conformer Selection in Reaction

Based on the findings regarding the population of conformers, selecting specific conformers can now be reasoned with some basis. Although conformers 1g'Gg'/15g'Gg' were the most populated of the states due to their intramolecular hydrogen bonds of the character (OH $\cdots$ N), it is not inconceivable to believe that these are less reactive for some of the reactions considered, and perhaps that another conformers are more appropriate. For instance, the conformers in state D (4tGt), E(5tGg) or G(7tGg') may perhaps be suitable candidates, because they are of higher population and more importantly the reaction centre is more accessible for nucleophelic attacks.

In literature, the choice of conformers seem to be rather arbitrary, except from one study by Xie et al., 2010[83] that explicitly selected a conformer (7tGg') to react with CO_2 based on highest population, and discovered that all conformers have similar reactivity except for 1g'Gg'. It would be interesting to gain some insights into whether this actually has an influence or not when considering MEA reacting with larger structures, where factors such as steric hindrance comes into play. Based on literature findings[65, 66] and kMC simulation results, the conformers 1g'Gg', 4tGt and 7tGg' conformers will be considered.

4.3.1 Formation of HEGly

In this section the TSs corresponding to the formation of HEGly has been under study. The formation of HEGly is one of the biggest mysteries, as it seen in abundance in pilot plant[67]. The information of the conformer distribution from the kMC simulation will be considered in this context, but also what seems to be the generally accepted opinion will be taken into account. It must be emphasised, every calculation up to this point have considered temperature at 298.15 K. Meaning that the conformer distribution from the kMC simulation might take a different turn, as the energy corresponding to the higher energy states likely are more populated. For that reason, literature findings on the conformation population in that specific situation is accounted for.

There is one mechanism in literature for rationalising the formation of HEGly. The mechanism was suggested by Vevelstad 2013[77] and is presented in Figure 4.6.

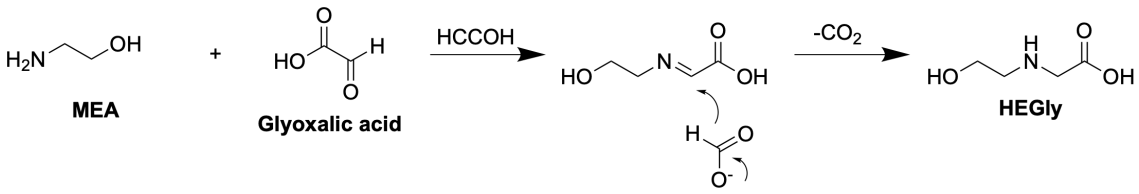
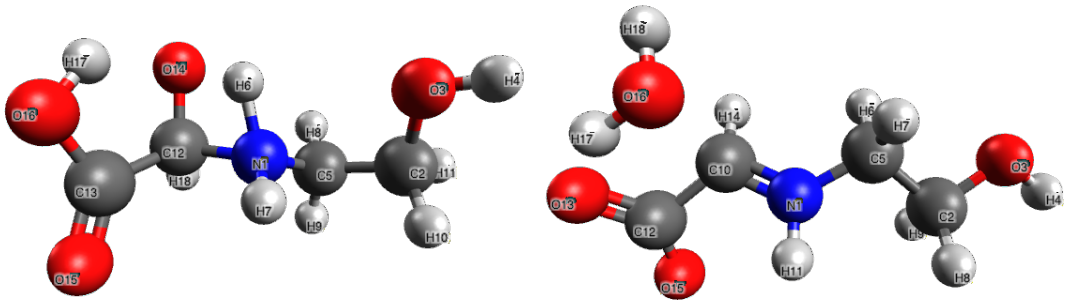


Figure 4.6: Mechanism for HEGly suggested by Vevelstad 2013[77]

A computational study of these exact mechanism was first carried out by Gupta et al. 2014[32], and will be used as a comparison. They used the combination PCM(water)/B3LYP/6-31+G(d,p). The mechanism involves a nucleophilic attack, where the amine functionality in MEA acts as the nucleophile on the aldehyde functionality of glyoxalic acid. This step also involves a simultaneous protonation (H6 to O14, separated by 1.378 Å), forming an intermediate which is a hemiamial (hydroxyl and amine on the same carbon). The TS for this step was calculated using three different conformation: g'Gg' (state A), tGg' (state G) and tTt (state I) and their corresponding barriers are presented in Table 4.6. Subfigure 4.7a illustrates the lowest TS1(tTt) geometry and the same structure is used further.

The subsequent step involves dehydration, with a barrier of 17 kcal mol^{-1}, where one of the alcohol gets protonated and leaves as water. The intermediate formed can either be a zwitterion or an amine. Whether it is an imine or zwitterion depends on which of the alcohols gets protonated. If the secondary alcohol gets protonated, then a zwitterion is formed. If the primary alcohol gets protonated, then the imine is formed. However, since the zwitterion formation corresponded to a lower reaction barrier, as opposed to the imine (55.23 kcal mol^{-1} from the work by Gupta et al., 2014[32]), the zwitterion pathway is thought to be more feasible. The corresponding TS TS2, is depicted in Subfigure 4.7b.



(a) Transition state 1 **(b)** Transition state 2

Figure 4.7: TS1 for the nucleophelic attack, and TS2 for the zwitterion formation

The next step is believed to be an acid-base reaction, which is considered to be instantaneous and

therefore does not have a stable TS. The final step is a hydrogenation resulting in the formation of HEGly and formic acid. The geometry of the corresponding TS is presented in Figure 4.8 with a reaction barrier of 35.6 kcal mol^{-1}. It involves four fragments: the zwitterion, the glyoxalic acid from the previous step, formaldehyde and water. The reaction involves hydrogenation at C10, where formaldehyde acts as the hydride donor aided by water, resulting in HEGly and formic acid.

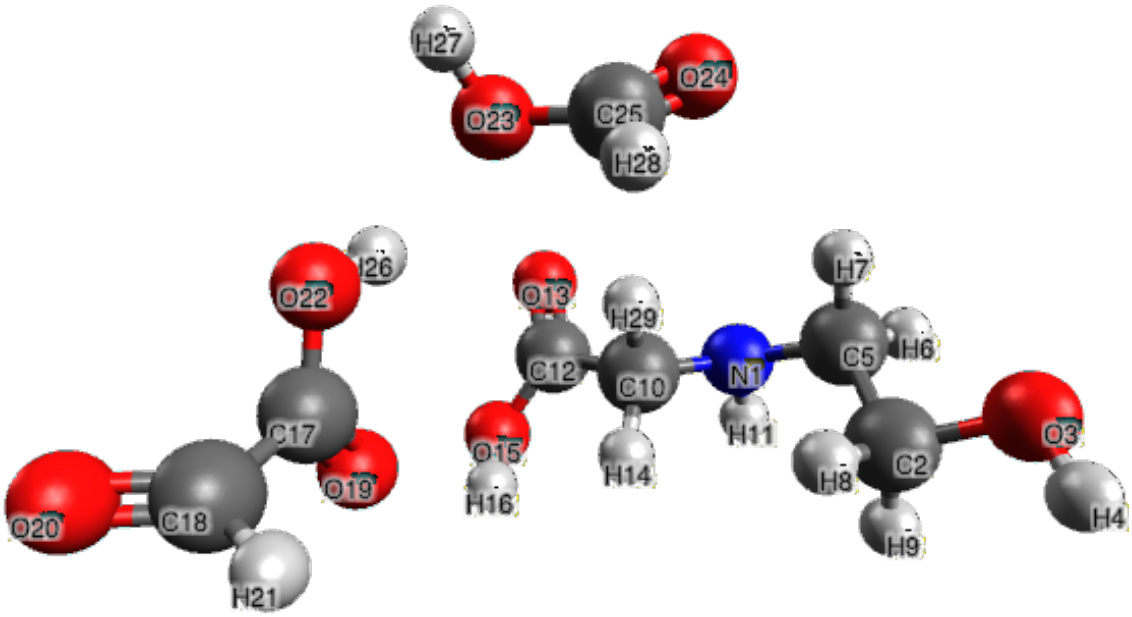


Figure 4.8: Transition state 4

The results obtained in this work is also compared to the original work, as seen in Table 4.6, and they seem to mostly align.

The study by Gupta et al., 2014[32], further concludes that the last step is the rate determining step (rds). Based on the findings in this work, it seems also depend on the starting conformer. If the most stable conformer constituting state A, which also is in abundance, is the starting reacting conformer, the rds might be the first step. However, it must be emphasised that the basis set used in this work is low and that will consequently affect the accuracy of the results. Particularly, pertaining to comparing TS1(1g'Gg') and TS4, which are very close in energy.

Additionally, there were some convergence issues pertaining to the TSs, which led to neither of them converging. This can perhaps be explained by the fact that several structures exhibit resonance. A weakness to DFT, CC, and other theories is that they aim for a single structure, but in situations of resonance, the optimisation procedure might alternate between the two (or more), resulting in it never converging. Another important issue, was that some of the TS also exhibited more than one imaginary frequencies. This applied to both the gas phase structures and those considered with solvent effects. However, these extra imaginary frequencies were small $< i100cm^{-1}$ and it is not

uncommon for larger and more flexible structures to yield several small imaginary frequencies as the may resemble free rotation. This will be addressed more in detail later on.

Table 4.6: Activation energy (TS with respect to isolated reactants) for TS1, TS2 and TS3 at COSMO/B3LYP/cc-pVDZ level of theory at 120°C with solvent effects considered with COSMO

Transition State	**Activation Energy**[a]	E_a[b]
TS1(1g'Gg')	37.1	
TS1(4tGt)	29.0	24.8[c]
TS1(7tGg')	24.4	
TS2	17.1	17.1
TS4	35.6	38.8

[a] kcal mol^{-1}

[b] From Gupta et al. 2014[32] at PCM/B3LYP/6-31+G(d,p) level of theory

[c] Conformer not specified

4.4 Computational challenges

Obtaining the underlying DFT-results proved to has far from smooth sailing. At every turn, unexpected trouble arised, making the progress slow. Debugging have required a substantial amount of effort, alongside attempts to make the process as efficient as possible with the available resources.

4.4.1 Challenges in Remote SSH-connection

Initially, test calculations were carried out on a remote Linux computer through a SSH-connection. Each and all calculations was executed in the background, using the `nohup` command followed by the script to run, redirection of the terminal output and an ampersand (&) at the end to ensure that any terminal log would be redirected. Furthermore, a VPN-connection was required to maintain the connection if not connected to the same network. However, as the calculations became more expensive, this approach began to break down. The calculations would unexpectedly stop updating. No message was printed either in the terminal window or redirected to the log file. From the log-file it appeared to have just paused the calculation. However, checking the process status with the `top` command, revealed that the task had indeed terminated despite no indication of any errors.

Exactly why this problem arose is difficult to ascertain. A possibility could be that the `nohup` command did not work properly for this situation. As a solution some other commands were attempted instead. One of the alternatives attempted was `screen`, which opens a *screen* within

the same session. Here, the same command as given in the previous paragraph could be executed in the background and the terminal can be used by *detaching* it using 'Ctrl + A', followed by a 'd'. However, this did not seem to work. The `tmxu` command was also tried, which also aims to run Linux commands in background using similar commands as screen, but with different key commands ('Ctrl + B' followed by 'D'). Unfortunately, this attempt proved unsuccessful as well.

A possible explanation for these issues could be that using the operating system module `os` to extract the information from the xyz-files might have required a continuous connection to the remote SSH-connection. This is just a speculation, but it is possible that the the terminal window had to be open to maintain a stable connection, or perhaps the VPN-connection played an important role.

Due to the relative large set of molecules and the computation time, it was decided to conduct the calculations on the Mac Books. Interestingly, none of these issues arose, even after exiting the terminal. Additionally, this decision also led to a decrees in the overall computation time.

4.4.2 Determining the Symmetry

One of the initial challenges encountered was determining the correct symmetry number σ. Although the MEA molecule has a symmetry number of unity, it was of importance to model other molecules with a symmetry number larger than one. The significance of this value can be seen in the context of the partition function for rotation, which subsequently contributed to the prefactor for the rate constant. Ensuring accuracy in this value thus becomes of great importance. One could argue that it is not essential for the program to compute it, as it can be determined from group theory, using the method in PySCF offers efficiency. Additionally, since it enters the partition function, calculating it concurrently seemed logical.

Some test molecules H_2O, CH_4, CH_3F and CH_3CH_3 were used to verify the accuracy if the symmetry number. However, the obtained values were 2, 3, 1 and 1, respectively. In other words, the values deviated significantly, whereas the values ought have been 2, 12, 3, and 6. In other words, the values were far from the the real ones. Attempts were made to specify the symmetry when building the molecule object, but this resulted in an error message. However, this did not seem to be accepted as it printed an error message. Because the symmetry values were printed out, although inaccurate, the problem was likely not related to the function itself, but perhaps the associated detection tolerance. Studying the source code, it was discovered that the default tolerance was set to $1e-5$ in the geom.py file. This essentially translates to the maximum allowed deviation for identifying symmetry. Adjusting this threshold to $1e-3$ resulted in more reliable symmetry determination.

4.4.3 Convergence Issues for Transition States

Conducting TS searches require that the input structure resembles the TS structure for it to converge. This has been decisive in order to obtain a structure with one imaginary frequency, characterising a TS. PySCF offers two approaches to conduct TS optimisations either using the *geomeTRIC*

algorithm or the *Quadratic Steepest Descent* (QSD) method by Sun and Ruedenberg, 1994[71]. The approach by geomeTRIC, converged much faster than the QSD. However, no imaginary frequencies for any structures could be obtained. So the latter was selected.

During QSD optimisation, the calculation conducts numerical hessian calculations. The method involves numerically approximating the second derivatives of the energy by perturbing the atomic positions slightly, and observing the resulting changes in the energy gradient. PySCF offers two methods for how the numerical hessian can be calculated using finite differences; *forward* and *central*. The default value is *forward*. In this work, the numerical hessian was calculated using the central finite difference method. As opposed to the forward method perturbing each atomic positions in a single direction, the central method perturbs each atomic position in two opposite directions around the equilibrium. An average is taken of the resulting changes in the gradient to approximate the second derivative[45]. The forward difference in generally not as accurate as the central difference[26]. On the other hand, utilising central difference is more expensive. As the latter proved better, it was chosen.

4.4.4 Addressing Imaginary Frequencies

A major challenge encountered in this work has been both the absence of no imaginary frequencies and presence of numerous imaginary frequencies during the optimisation of structures while considering COSMO. Before employing QSD with central differences, the optimisations were difficult to converge with imaginary frequencies, without perturbing the equilibrium geometries to a great extent.

Given the importance of characterising these structures at stationary points, addressing this issue was paramount, but they were also important for calculating the vibrational partition function. Interestingly enough, the issue with several imaginary frequencies was not an issue at all when considering the gas phase structures. However, upon introducing solvent effects, multiple imaginary frequencies emerged. It is possible that introducing a solvent effects, have an impact on how the hessian is being calculated, since the imaginary frequencies are the square root of the eigenvalues of the hessian. Moreover, it is worth noting that it is not uncommon to observe imaginary frequencies of low magnitude, especially for flexible molecules. Indeed, conducting regular hessian calculation on non-equilibrium structures frequently yields up to several small imaginary frequencies[69].

Regarding the situation from a more technical point of view, encountering imaginary frequencies during energy minimisation, may resemble that depicted in Figure 4.9. It illustrates a phenomenon known as a textitdouble well potential, in which the optimisation process may oscillate between the two the two basins and reach a maxima. Although the figure is a bit exaggerated, it explains the conceptual idea.

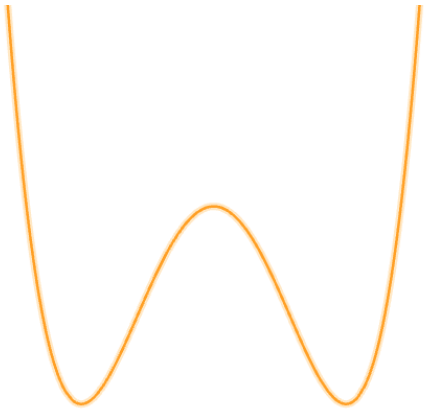


Figure 4.9: An illustration of how optimising a minimum may results in a imaginary frequency

Despite using pre-optimised structures before re-optimisation with solvent effects, the imaginary frequencies still remained. They were not small either. They ranged from i50 cm^{-1} to i200 cm^{-1} for cc-pVDZ basis set. Moreover, when using a larger basis set, imaginary frequencies could readily exceed i1000cm^{-1}

Sure et Grimme, 2013[74], further argue that the small imaginary frequencies resemble free rotation. Jensen 2015 reasons that flat PES are more susceptible for imaginary frequencies, and an approach to circumvent them can be to make the convergence criteria more stringent, e.g. fine-tuning the grid size. The same work also advises using the *central difference* method, if the hessian is calculated using finite differences. However, it is worth noting that using central difference is a more expensive approach. Another piece of advise has been to use the vibrational contributions for the gas phase structures. This is also supported by Sure et al., 2014[73]. They further stated that analytical frequencies cannot be calculated when COSMO has been utilised, and therefore advise using the frequencies from he gas phase structures. Finally, this rationalise the choice of using the vibrations frequencies of the gas phase structures when calculating the vibrational partition function.

4.4.5 Navigating PySCF - Challenges and Benefits

A general drawback of most computational software is to learn a new language. However, as many students nowadays are familiar with type of programming language the transition is often smoother to other. One could argue that since Python might be one of the most popular, PySCF is a good choice.

The output from conducting calculations in PySCF can be adjusted by tuning the print level `verbose`, ranging from 0 to 9. The printed output display information pertaining to the optimisation procedure and coordinates of the optimised structure with corresponding energy. However, the format can be a bit impractical as accessing and viewing the optimised structures is not an automatic accessed. As a consequence, PySCF encourages the user to utilise Pythons tools to the fully.

Navigating the PySCF documentation can at times be challenging. While the existing documentation

is comprehensive, there are sporadic typos and inconsistencies preventing a smooth execution of the scrips. Moreover, updates to PySCF occasionally results in relocation of certain modules, and these are not reflected in the example files. One notable inconsistencies pertains to conducting a harmonic analysis, where the function takes two arguments, the molecule object and the hessian.

During geometry optimisations with *geomeTRIC*, both the optimised molecule object (`mol_opt`) along with the hessian `hess`, which are parsed in the source code. Alternatively, the molecular structure associated with the mean field object `mf_opt.mol` can be parsed. However, for transition state optimisation, only one of these works, `mf_ts.mol`. Hence, the it is named `optimizer`, rather than `mol_ts`, to avoid confusion since it does not contain the molecule object. Strangely enough, `mol` also yields the same results, despite being created before the optimisation process was initiated (but note that the same does not apply when using geomeTRIC). It appears to store the information about the optimisation process, which also explains why that same mol-object must be used to print out the new coordinates.

Accessing the data and figuring out how it may be computed are by no means straight forward nor self-explanatory. Additionally, debugging errors and interpreting unexpected results can be challenging and tedious, as the resources and community support available for PySCF-specific issues are limited. Nevertheless, with dedication and perseverance, mastering PySCF can be a rewarding, enabling users to perform advances calculation in a relative simple and accessible software. Particularly, when other Python-modules are integrated. Despite these drawbacks, the benefits of using PySCF outweigh the challenges.

4.4.6 Strategies for Deriving Molecular Metrics

Unfortunately, no open source code was found for calculating the bond lengths or dihedral angles from xyz-files, and sadly PySCF print it in the log file either. As a result, the codes to obtain this has been developed in this work. The metrics were derives using vector mathematics and trigonometry. Then most challenging aspect of this, proved to be determining the correct sign and magnitude for the dihedral angles. Natural choices such as `np.arcsine()` and `np.arcsine()`, however, yielded the correct sign, but incorrect magnitude, and correct magnitude, but incorrect sign respectively. `np.arctan()` was also tried out, but it deteriorated the situation. This can be rationalised by looking into their domain and range, which is explained in detail in appendix A.3.

Using numpy's `sign` function in combination with `np.arcsine()` was also attempted, as `np.arcsine()` managed to yield the correct magnitude. The idea behind integrating the sign function was to provide information on the sign of the angle. Although this approach could potentially work, it would be very cumbersome to determine this for each of the three dihedrals in every single conformer ($2x25$, for equilibrium and transition state optimisations) when there are so many. Efforts were therefore made to discover even better solutions.

In the search for better solutions, the function `np.arctan2()` emerged. The idea of using the it was

inspired by the same principle used to calculate the *solar azimuth angle* by Zhang et al., 2021[88] according to the *east-counterclockwise convention*. Essentially, `np.arctan2()` is utilised to calculate an angle in the range $[-\pi, \pi]$ or equivalently $[-180°, 180°]$, and more usefully, similar issues to those encountered in this work arise when using `np.arcsine()` or `np.arcsine()` are used.

The calculation of the bonds and dihedrals was as previously mentioned in the method chapter, performed in *Jupyter Notebook*. The main reason for using this approach was to ensure that the correct atoms were considered when the metrics were calculated. It was discovered that, for some of the structures, the ordering of the atoms within the xyz-file were mixed, rendering an all-around function with predetermined atoms unreliable in a terminal environment. An explanation for the disordering of the xyz-file may stem from how the structures were build in Avogadro. The order of which atoms or fragments were added seems to play a decisive role on the outcome. While using *py3Dmol* might appear redundant, it becomes clear that it was strictly necessary. Explicitly using the xyz-format and visualising them enabled verification of correct atom selection for various sets of bonds and dihedals to calculate.

Perhaps, a more sophisticated alternative could involve creating a function that determines the relevant atoms for bonds and dihedrals based on their proximity and stores this information. Subsequently, the functions in this work could calculate these metrics based on the stored atom information. This approach would require creating dictionaries based on the connectives to facilitate bonds and dihedral calculations. Additionally, enhancing the transferability of the functions by calculating the metrics with no package dependencies, such as the *numpy* module would make implementation in other programming languages easier.

It must be mentioned that efforts were still made to make a function that calculates all bonds and dihedrals present in the structure, similar to modules in *RDKit*. However, this turned out to be more demanding than rewarding at that point. Especially, since bonds and dihedrals very easily can be calculated by clicking on atoms in Avogadro. However, it must be noted that it is easy to miss the relevant atoms when just relying on clicking.

4.4.7 Optimising Task Execution

Both `ProcessPoolExecutor` and `ThreadPoolExecutor` are classes within the concurrent.futures module. They can be imported to create a pool executor, allowing tasks to be run concurrently. The choice between them depends on the nature of the task. In general, using *thread pool* is preferable when the tasks are *I/O-bound*, i.e. tasks pertaining to reading and writing data. If however a task is *CPU*-bound, involving demanding calculations (such as geometry optimisations), *process pool* might be a more suitable choice[47].

Both methods were tried when adding solvent effects on the equilibrium geometries(on already optimised geometries at cc-pVDZ basis set). When `max_workers = 4`, the `ProcessPoolExecutor` spent 3 hours and 57 minutes and `ThreadPoolExecutor` spent 5 hours and 46 minutes on the same

tasks. This suggests that the `ProcessPoolExecutor` was more efficient in this case, likely because these are CPU-bound tasks and, the executor utilises the CPU more effectively.

4.5 Computational Accuracy

An optimal way for assessing the computational the results, involves comparing them to experimental results. However, this might not always be feasible for instances where experimental findings are scarce, incomparable or inaccessible directly. In such cases, an alternative approach is to verify the computational accuracy by comparing them to more precise calculations, provided they are available [85].

4.5.1 Advantages and Limitations to DFT

The primary limitation in DFT stems from the approximation of the exchange functional. DFT is based on the two fundamental Hohenberg-Kohn theorems, namely the HK existence theorem and the HK variational theorem. However, the latter theorem only holds if the exact exchange-correlation functional is utilised[45, 68]. DFT can, in other words, yield energy lower than the true energy. There are naturally other factors contributing to a certain degree of uncertainty. As a result, choosing an optimal functional is important. When that is said, there is no all-around perfect functional, as their performance tend to depend on the specific problem at hand. The same goes for the basis set. Nevertheless, DFT tend to be more resilient to basis set size than correlated wave function methods. However, despite using the best functional available, poor results may still arise from using a small basis set[19].

Selecting a sufficiently large basis set is essential for to obtain accurate calculations. However, achieving completed basis sets, although ideal, is not feasible. Despite this, limitation settling with an incomplete basis set still provides usefully insights. Another important aspect is to ensure that the basis set chooses converge. Essentially, it is crucial to have some insurance that that the computed values converge towards greater accuracy as the size of the basis set increases. Without convergence, the results may be inaccurate and unreliable, further affecting assumptions and predictions of molecular properties. Therefore achieving basis set convergence is essential for obtaining reliable and predictable results.

In the theory section it was mentioned that for basis set superposition error (BSSE) are especially prone for systems described with a small basis set, as for the case with the HEGly-reaction which did not consider augmented despite one of the products being an anion. Although not used in this work, a method to assess BSSE is to apply the so called *counterpoise-correction*. The method is an approximate way and consist of correcting the spurious energy contribution. The BSSE is inevitable to eliminate as it is present, in various magnitude, in all calculations. However, it's influence depends on the system being described. For instance, it is greater for weakly bound systems such as complexes held together by dispersion forces, and can become as great the interaction itself.

Therefore higher level correlation treatment and basis set is required. For hydrogen bonds there is not as stringent requirement, because they are electrostatic at the core. In general, the interactions energies varies between 100-500 mE_h for covalent bonds, 1-10 mE_h for hydrogen bonds and 50-500 μE_h for covalent bonds[35].

Another important aspect to consider is *spin contamination*, which in this context, refers to that different KS-orbitals are used for α- and β-spins. Because the unrestricted Kohn-Sham (UKS) approach was used, spin contamination is to a certain degree present. In Hartree-Fock (HF) this mainly stems form the exchange-correlation term. This will also be influenced by the functional utilised, as the once that incorporate HF-exchange to a greater degree (such as double hybrid functional), consequently are more susceptible for spin contamination[51]. While the UKS is less affected by spin contamination, as apposed to its HF counter part, it may still introduce errors[10], especially when it is compared to more accurate methods provided by CCSD and CCSD(T)[51]. In general, unrestricted and restricted procedures describe equilibrium geometries similarly for most singlet systems. However, when bonds are stretched, such as for transition state structures, the amount of spin contamination becomes more pronounced. In some cases, unrestricted approaches might overestimate biradical character, leading to a TS becoming a minimum on the PES[40].

4.5.2 Addressing the Choice of Functional and Basis set

The choice of the B3LYP was among several reasons due to its wide application, which further allows for comparison with previous work so the accuracy of the results could be assessed. Another important argument is that B3LYP is known for its performance in thermochemistry [46, 48] and harmonic frequency calculation[48], which has been essential for the results obtained. Despite the even more popular combination using B3LYP with Pople basis set, owing to it affordability, this combination seems to also yield poor results pertaining to thermochemistry[44]. Particularly the B3LYP/6-31G* combination, which is susceptible to basis set superposition error (which will be discussed in a bit) and lacks London dispersion effects[19]. On the other hand, while B3LYP is poor with metals[53, 90]. This could be an argument for choosing a different functional since amine degradation tends to be catalysed by metals such as iron and chromium. However, since none of the calculations in this work considered metals, this was not taken into account.

Although B3LYP does not describe dispersion sufficiently, which moreover is not the main contributor to intramolecular forces in the system as this was hydrogen bonds, they could have had a greater influence on the solvent model, which was a continuum model. Perhaps a functional such as CAM-B3LYP, could have been more appropriate, which aims to correct long range forces such as dispersion[84]. Nevertheless, the B3LYP functional was used throughout this work for consistency.

As already mentioned, the Pople basis sets are considered as relatively computationally affordable, but a major drawback is the unstable basis set convergence, which can affect the accuracy and the precision of the results. This is, among several aspects, Dunning correlation consistent basis,

offer improved basis set convergence[26, 76]. This is essential to obtain reliable results with both accuracy and precision.

The combination with B3LYP and augmented correlation consistent basis set has yielded results comparable to molecular orbital methods[48], and does overall seem to perform better than the Pople basis sets[48, 51]. However, opinions seems to be dividend whether which basis set is the most optimal. A study conducted by Wiberg 2004[82] showed that in addition to 6-311++G**, being more efficient, it yields more satisfactory geometries. Nevertheless, it should be noted that these comparisons were carried out on MP2 and CCSD, and the correlation consistent basis sets are constructed to yield more accurate results for post-Hartree Fock methods.

A relevant assumption based on the conclusion by Kelterer et al., 1994[42] that the basis set might have an influence, perhaps using a different basis set would enable the determination of the remaining conformers. On the other side a study using the same leveel theory did successfully manege to identify all conformers. Furthermore, previous studies that successfully determined these conformers used the Pople basis set. However, as discussed, because it unsuccessfully converged towards the basis set limit, it is out of the question. Bursch et al., 2022[19] have carried out a thorough study to give a comprehensive guide on the *Best Practice DFT Protocols*, based on 25 years of experience in the field. Interestingly, they rule out both the the Pople basis sets 6-31G* and 6-311G** as well as the Dunning-type cc-pVXZ (X = D, T, Q, ...), arguing that the basis set by Ahlrichs are superior to them both pertaining to efficiency and availability acreoss a wider range of the periodic table. Additionally, they highlight *PySCF* as one of the noteworthy quantum chemistry packages.

As the field of DFT continuously evolves, it becomes more and more challenging to make a choice. The selection of functional and basis set to use often depends on the problem at hand, and what is available. Various aspect ought to be carefully considered and it is important to be aware that different functional may perform differently, when they are applied to different systems. Addressing potential drawbacks and assessment regarding the choice from previous works can be valuable when choosing. Overall, selecting the suitable functional and basis set requires careful consideration.

5 | Conclusions and Outlook

The primary objective of this master thesis has been study the conformer stability of the MEA molecule, considered a benchmark solvent in PCC. MEA is furthermore a central component in eucaryotic cell membranes and can function as a complexing agent either as a monodentate or a bidentate. However, a significant drawback with using amine-based solvents, however, is that they all degrade to a certain extent, resulting in increased operational costs due to material wear and solvent replacement. Due to its versatile application and challenges related to degradation, it is of considerable interest to gain insight in its stability. In order to address this, studying the interconversion barriers and map spectrum of conformers and how they connect has been of interest.

Another object has been to make a codes that increase efficiency and versatility, while utilising the software PySCF.

5.1 Concluding remarks

In this work, 25[1] conformers out of the theoretical 27 were determined. The most stable conformer found, aligns with other studies conducted at similar level of theory and higher. Furthermore, results from DFT calculations on the conformers indicate the presence of two conformers that has not been determined in more recent literature. However, because the stationary point could not be affirmed, it is possible that these calculations are in fact erroneous. This might indicate that its presence is less relevant after all.

The transition states of all minima were also determined. In general, the interconversion barriers range from 0.66-5.15 kcal mol^{-1}. The barrier heights are overall very low, and they tend to be even lower for interconversion between two conformers exhibiting HB. With such small barriers and consequently flat PES, which led to convergence issues, the results must be carefully assessed.

Furthermore, rate constants were required to run kMC simulations. These were calculated based on TST, which relies on statistical mechanics to capture the contributions from rotational, vibrational and translational degrees of freedom. The results from the DFT calculations has been used in this

[1] if the uncertain conformers 14g'Gg and 27gG'g' counts

context.

During this work, a great effort was dedicated to make an efficient and user-friendly code within the Python ecosystem, particularly for newcomers to PySC. The aim was to create versatile scripts capable of executing the relevant tasks to eventually be able to calculate the rate constants. Clarity and automation for efficiency has been the main thoughts behind every code line. This has overall contributed to maintain a stable workflow throughout this work.

During the kMC simulations, various time intervals were explored to study the conformational interconversions of MEA. Naturally, the states exhibiting the lowest energy were more populated. The results of the kMC simulation are overall in agreement with a similar work that suggests that MEA tends to exist predominantly, where the O−C−C−N dihedral angle takes the gauche conformation. This can further be explained by the fact that all conformers exhibiting HB, which also are the lowest lying conformers, also take gauche conformation.

Finally, the results from kMC were further used to study a specific secondary reaction pathway, namely the formation of HEGly, with the desire to gain insight in whether the reacting conformer influences the reaction kinetics. In that context three different conformers of MEA were tested. While the results might indicate that the conformer selection possibly have some influence, it is important to note the inherent limitations of the chosen level of theory impose substantial uncertainties. Moreover, the convergence issues and several imaginary frequencies complicates drawing any conclusions.

5.2 Outlook

Further work on this topic may include studying the reaction mechanism for other secondary degradation pathways, using a carefully selected MEA conformer, as computational studies supporting these are few compared to primary degradation paths. Any findings increasing the insight of secondary amine degradation would be valuable. Naturally, the system size and subsequently computational costs, are deceive factors as there is a trade-off between computational accuracy and computational costs for most situations. Combining results from both primary and secondary degradation would be interesting to study in a kMC simulation. On the other hand, secondary pathways consider large systems constituted by very complex multi-steps reaction, which are not very compatible with kMC simulations. In addition, the choice of basis set and functional might also need to be reevaluated since some of the secondary degradation pathways involve mechanisms of radical character, rapid proton transfer and heavy metal complexes, which further catalyse the entire degradation, just to mention some.

Bibliography

[1] In *The MAK-Collection for Occupational Health and Safety*, John Wiley & Sons, Ltd, **2012**, pp. 16–35, DOI 10.1002/3527600418.mb14143e0012.

[2] N. D. Afify, M. B. Sweatman, 'Solvent-mediated modification of thermodynamics and kinetics of monoethanolamine regeneration reaction in amine-stripping carbon capture: Computational chemistry study', *The Journal of Chemical Physics* **2024**, *160*, DOI 10.1063/5.0169382.

[3] S. H. Ali, 'Kinetics of the reaction of carbon dioxide with blends of amines in aqueous media using the stopped-flow technique', *International Journal of Chemical Kinetics* **2005**, *37*, 391–405, DOI 10.1002/kin.20059.

[4] T. C. Allison, D. G. Truhlar in *Modern Methods for Multidimensional Dynamics Computations in Chemistry*, WORLD SCIENTIFIC, **1998**, pp. 618–712, DOI 10.1142/9789812812162_0016.

[5] M. Andersen, C. Panosetti, K. Reuter, 'A Practical Guide to Surface Kinetic Monte Carlo Simulations', *Frontiers in Chemistry* **2019**, *7*, DOI 10.3389/fchem.2019.00202.

[6] E. V. Anslyn, D. A. Dougherty, *Modern Physical Organic Chemistry*, 2nd ed., University Science Books, **2006**, pp. 365–372.

[7] P. Atkins, R. Friedman, *Molecular Quantum Mechanics*, 5th ed., Oxford University Press, **2010**, pp. 239–243, 317–326.

[8] P. Atkins, J. de Paula, J. Keeler, *Atkins' Physical Chemistry*, 11th ed., Oxford University Press, **2018**, pp. 251–252, 343, 544.

[9] P. W. Atkins, *Elements of physical chemistry*, 7th edition, (Eds.: J. de Paula, D. Smith), Oxford University Press, Oxford, **2017**, pp. 269–288.

[10] J. Baker, A. Scheiner, J. Andzelm, 'Spin contamination in density functional theory', *Chemical Physics Letters* **1993**, *216*, 380–388, DOI 10.1016/0009-2614(93)90113-f.

[11] R. Bates, G. Pinching, 'Acidic dissociation constant and related thermodynamic quantities for monoethanolammonium ion in water from 0°C to °C', *Journal of Research of the National Bureau of Standards* **1951**, *46*, 349, DOI 10.6028/jres.046.039.

[12] N. H. F. Beebe, *The Mathematical-Function Computation Handbook*, 2nd ed., Springer International Publishing AG, **2017**, p. 336, DOI 10.1007/978-3-319-64110-2.

[13] R. P. Bell, 'The tunnel effect correction for parabolic potential barriers', *Transactions of the Faraday Society* **1959**, *55*, 1, DOI 10.1039/tf9595500001.

[14] A. Bello, R. O. Idem, 'Comprehensive Study of the Kinetics of the Oxidative Degradation of CO2 Loaded and Concentrated Aqueous Monoethanolamine (MEA) with and without Sodium Metavanadate during CO2 Absorption from Flue Gases', *Industrial &; Engineering Chemistry Research* **2005**, *45*, 2569–2579, DOI 10.1021/ie050562x.

[15] A. Blondel, M. Karplus, 'New formulation for derivatives of torsion angles and improper torsion angles in molecular mechanics: Elimination of singularities', *Journal of Computational Chemistry* **1996**, *17*, 1132–1141, DOI 10.1002/(SICI)1096-987X(19960715)17:9<1132::AID-JCC5>3.0.CO;2-T.

[16] N. Borduas, J. P. D. Abbatt, J. G. Murphy, 'Gas Phase Oxidation of Monoethanolamine (MEA) with OH Radical and Ozone: Kinetics, Products, and Particles', *Environmental Science & Technology* **2013**, *47*, 6377–6383, DOI 10.1021/es401282j.

[17] M. Born, R. Oppenheimer, 'Zur Quantentheorie der Molekeln', *Annalen der Physik* **1927**, *389*, 457–484, DOI 10.1002/andp.19273892002.

[18] G. Buemi, 'Conformational analysis and rotation barriers of 2-aminoethanethiol and 2-aminoethanol: An ab initio study', *International Journal of Quantum Chemistry* **1996**, *59*, 227–237, https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-461X%281996%2959%3A3%3C227%3A%3AAID-QUA6%3E3.0.CO%3B2-%23.

[19] M. Bursch, J.-M. Mewes, A. Hansen, S. Grimme, 'Best-Practice DFT Protocols for Basic Molecular Computational Chemistry**', *Angewandte Chemie International Edition* **2022**, *61*, DOI 10.1002/anie.202205735.

[20] V. Buvik, 'Stability of amines for CO_2 capture', Ph.D. Thesis, Department of Chemical Engineering, Norwegian University of Science and Technology, Trondheim, **2021**, https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2777358/Vanja%20Buvik_PhD.pdf?sequence=1&isAllowed=y.

[21] E. Calzada, O. Onguka, S. M. Claypool in *International Review of Cell and Molecular Biology*, Elsevier, **2016**, pp. 29–88, DOI 10.1016/bs.ircmb.2015.10.001.

[22] S. Canneaux, F. Bohr, E. Henon, 'KiSThelP: A program to predict thermodynamic properties and rate constants from quantum chemistry results†', *Journal of Computational Chemistry* **2013**, *35*, 82–93, DOI 10.1002/jcc.23470.

[23] B. K. Carpenter, 'Dynamic Behavior of Organic Reactive Intermediates', *Angewandte Chemie International Edition* **1998**, *37*, 3340–3350, DOI 10.1002/(sici)1521-3773(19981231)37:24<3340::aid-anie3340>3.0.co;2-1.

[24] Y.-P. Chang, T.-M. Su, T.-W. Li, I. Chao, 'Intramolecular Hydrogen Bonding, Gauche Interactions, and Thermodynamic Functions of 1,2-Ethanediamine, 1,2-Ethanediol, and 2-Aminoethanol: A Global Conformational Analysis', *The Journal of Physical Chemistry A* **1997**, *101*, 6107–6117, DOI 10.1021/jp971022j.

[25] S. Chi, G. T. Rochelle, 'Oxidative Degradation of Monoethanolamine', *Industrial &; Engineering Chemistry Research* **2002**, *41*, 4178–4186, DOI 10.1021/ie010697c.

[26] C. J. Cramer, *Essentials of Computational Chemistry*, 2nd ed., John Wiley & Sons, Ltd, **2004**, pp. 6–10, 94, 249–294.

[27] K. A. Dill, S. Bromberg, *Molecular Driving Forces*, 2nd ed., Garland Science, **2011**, pp. 173, 176, 198–207, 235–243, 364–368.

[28] M. Ernst, J.-P. Melder, F. I. Berger, C. Koch, Ethanolamines and Propanolamines, **2022**, DOI 10.1002/14356007.a10_001.pub2.

[29] H. Eyring, 'The Activated Complex in Chemical Reactions', *The Journal of Chemical Physics* **1935**, *3*, 107–115, DOI 10.1063/1.1749604.

[30] J. D. Figueroa, T. Fout, S. Plasynski, H. McIlvried, R. D. Srivastava, 'Advances in CO2 capture technology—The U.S. Department of Energy's Carbon Sequestration Program', *International Journal of Greenhouse Gas Control* **2008**, *2*, 9–20, DOI 10.1016/s1750-5836(07)00094-1.

[31] D. T. Gillespie, 'Exact stochastic simulation of coupled chemical reactions', *The Journal of Physical Chemistry* **1977**, *81*, 2340–2361, DOI 10.1021/j100540a008.

[32] M. Gupta, S. J. Vevelstad, H. F. Svendsen, 'Mechanisms and Reaction Pathways in MEA Degradation; A Computational Study', *Energy Procedia* **2014**, *63*, 1115–1121, DOI 10.1016/j.egypro.2014.11.120.

[33] M. D. Hanwell, D. E. Curtis, D. C. Lonie, T. Vandermeersch, E. Zurek, G. R. Hutchison, 'Avogadro: an advanced semantic chemical editor, visualization, and analysis platform', *Journal of Cheminformatics* **2012**, *4*, DOI 10.1186/1758-2946-4-17.

[34] J. Hass, C. Heil, M. D. Weir, *Thomas' Calculus in SI Units*, 14th ed., Pearson Education Limited, **2020**, pp. 435–438, 681–686, 689–694, 816–820.

[35] T. Helgaker, P. Jørgensen, J. Olsen, *Molecular Electronic-Structure Theory*, Wiley, **2000**, pp. 315–335.

[36] G. Herzberg, L. Herzberg, 'Rotation-Vibration Spectra of Diatomic and Simple Polyatomic Molecules with Long Absorbing PathsXI. The Spectrum of Carbon Dioxide (Co2) below 1.25$\mu*$', *J. Opt. Soc. Am.* **1953**, *43*, 1037–1044, DOI 10.1364/JOSA.43.001037, https://opg.optica.org/abstract.cfm?URI=josa-43-11-1037.

[37] O. L. Hestad, A.-D. Vuong, I. Madshaven, P.-O. Åstrand, 'Thermal decomposition of cyclohexane by Kinetic Monte Carlo simulations and its relevance to streamer formation', **2016**, DOI 10.1109/ceidp.2016.7785518.

[38] P. Hohenberg, W. Kohn, 'Inhomogeneous Electron Gas', *Physical Review* **1964**, *136*, B864–B871, DOI 10.1103/physrev.136.b864.

[39] A. Jansen, *An Introduction to Kinetic Monte Carlo Simulations of Surface Reactions*, Springer Berlin Heidelberg, **2012**, pp. 1–4, 37–55, DOI 10.1007/978-3-642-29488-4.

[40] F. Jensen, *Introduction to Computational Chemistry*, 3rd ed., John Wiley & Sons Inc, **2017**, pp. 141, 233–269, 455–458, 499, 614–616.

[41] J. H. Jensen, 'Predicting accurate absolute binding energies in aqueous solution: thermodynamic considerations for electronic structure methods', *Physical Chemistry Chemical Physics* **2015**, *17*, 12441–12451, DOI 10.1039/c5cp00628g.

[42] A.-M. Kelterer, M. Ramek, R. F. Frey, M. Cao, L. Schäfer, 'Basis set influence in ab initio calculations: The case of 2-aminoethanol and N-formylproline amide', *Journal of Molecular Structure* **1994**, *310*, 45–53, DOI 10.1016/s0022-2860(10)80055-9.

[43] W. Kohn, L. J. Sham, 'Self-Consistent Equations Including Exchange and Correlation Effects', *Phys. Rev.* **1965**, *140*, A1133–A1138, DOI 10.1103/PhysRev.140.A1133.

[44] H. Kruse, L. Goerigk, S. Grimme, 'Why the Standard B3LYP/6-31G* Model Chemistry Should Not Be Used in DFT Calculations of Molecular Thermochemistry: Understanding and Correcting the Problem', *The Journal of Organic Chemistry* **2012**, *77*, 10824–10834, DOI 10.1021/jo302156p.

[45] E. G. Lewards, *Computational Chenistry - Introduction to the Theory and Applications of Molecular Quantun Mechanics*, 3rd ed., Springer Cham, **2016**, pp. 483–557, DOI 10.1007/978-3-319-30916-3.

[46] L. Lu, H. Hu, H. Hou, B. Wang, 'An improved B3LYP method in the calculation of organic thermochemistry and reactivity', *Computational and Theoretical Chemistry* **2013**, *1015*, 64–71, DOI 10.1016/j.comptc.2013.04.009.

[47] A. Martelli, A. M. Ravenscroft, S. Holden, P. McGuire, *Python in a Nutshell: A Desktop Quick Reference*, 4th ed., O'Reilly Media, **2023**, pp. 323–327, 443, 445, 467–470.

[48] J. M. Martin, J. El-Yazal, J.-P. François, 'Basis set convergence and performance of density functional theory including exact exchange contributions for geometries and harmonic frequencies', *Molecular Physics* **1995**, *86*, 1437–1450, DOI 10.1080/00268979500102841.

[49] D. A. McQuarrie, *Statistical mechanics*, University Science Books, **2000**, pp. 94–110, 129–136.

[50] D. A. McQuarrie, J. D. Simon, *Molecular Thermodynamics*, University Science Books, **1999**, pp. 10, 20, 105, 148–159.

[51] A. M. Mebel, V. V. Kislov, 'The C_2H_3 + O2 Reaction Revisited: Is Multireference Treatment of the Wave Function Really Critical?', *The Journal of Physical Chemistry A* **2005**, *109*, 6993–6997, DOI 10.1021/jp052772t.

[52] Y. V. Novakovskaya, M. N. Rodnikova, 'Ethanolamine: conformational diversity', *Structural Chemistry* **2014**, *26*, 177–187, DOI 10.1007/s11224-014-0530-3.

[53] J. Paier, M. Marsman, G. Kresse, 'Why does the B3LYP hybrid functional fail for metals?', *The Journal of Chemical Physics* **2007**, *127*, DOI 10.1063/1.2747249.

[54] D. Patel, S. N. Witt, 'Ethanolamine and Phosphatidylethanolamine: Partners in Health and Disease', *Oxidative Medicine and Cellular Longevity* **2017**, *2017*, 1–18, DOI 10.1155/2017/4829180.

[55] J. P. Perdew, K. Schmidt in AIP Conference Proceedings, AIP, **2001**, DOI 10.1063/1.1390175.

[56] S. Pirhadi, J. Sunseri, D. R. Koes, 'Open source molecular modeling', *Journal of Molecular Graphics and Modelling* **2016**, *69*, 127–143, DOI 10.1016/j.jmgm.2016.07.008.

[57] P. Prinz, T. Crawford, *C in a Nutshell*, 2nd ed., O'Reilly Media, **2015**, pp. 352–360.

[58] L. Radom, W. A. Lathan, W. J. Hehre, J. A. Pople, 'Molecular orbital theory of the electronic structure of organic compounds. XVII. Internal rotation in 1,2-disubstituted ethanes', *Journal of the American Chemical Society* **1973**, *95*, 693–698, DOI 10.1021/ja00784a008.

[59] L. Radom, W. A. Lathan, W. J. Hehre, J. A. Pople, 'Molecular orbital theory of the electronic structure of organic compounds. XVII. Internal rotation in 1,2-disubstituted ethanes', *Journal of the American Chemical Society* **1973**, *95*, 693–698, DOI 10.1021/ja00784a008.

[60] M. Räsänen, A. Aspiala, L. Homanen, J. Murto, 'IR-induced photorotamerization of 2-aminoethanol in low-temperature matrices. AB initio optimized geometries of conformers', *Journal of Molecular Structure* **1983**, *96*, 81–100, DOI https://doi.org/10.1016/0022-2860(82)90060-6.

[61] G. Rochelle, S. Bishnoi, S. Chi, H. Dang, J. Santos, 'Research needs for CO_2 capture from flue gas by aqueous absorption/stripping', *Research Report for P.O.: No. DE-AF26-99FT01029 of U.S. Department of Energy* **2001**.

[62] F. V. Ryzhkov, Y. E. Ryzhkova, M. N. Elinson, 'Python in Chemistry: Physicochemical Tools', *Processes* **2023**, *11*, 2897, DOI 10.3390/pr11102897.

[63] G. Sartori, D. W. Savage, 'Sterically hindered amines for carbon dioxide removal from gases', *Industrial & Engineering Chemistry Fundamentals* **1983**, *22*, 239–249, DOI 10.1021/i100010a016.

[64] D. S. Sholl, J. A. Steckel, *Density Functional Theory - A Practical Introduction*, 2nd ed., John Wiley & Sons, Ltd, **2009**, pp. 7–30.

[65] C. F. Silva, M. L. T. Duarte, R. Fausto, 'A concerted SCF-MO ab initio and vibrational spectroscopic study of the conformational isomerism in 2-aminoethanol', *Journal of Molecular Structure* **1999**, *482–483*, 591–599, DOI 10.1016/s0022-2860(98)00794-7.

[66] E. F. da Silva, T. Kuznetsova, B. Kvamme, K. M. Merz, 'Molecular Dynamics Study of Ethanolamine as a Pure Liquid and in Aqueous Solution', *The Journal of Physical Chemistry B* **2007**, *111*, 3695–3703, DOI 10.1021/jp068227p.

[67] E. F. da Silva, H. Lepaumier, A. Grimstvedt, S. J. Vevelstad, A. Einbu, K. Vernstad, H. F. Svendsen, K. Zahlsen, 'Understanding 2-Ethanolamine Degradation in Postcombustion CO_2 Capture', *Industrial & Engineering Chemistry Research* **2012**, *51*, 13329–13338, DOI 10.1021/ie300718a.

[68] S. F. Sousa, P. A. Fernandes, M. J. Ramos, 'General Performance of Density Functionals', *The Journal of Physical Chemistry A* **2007**, *111*, 10439–10452, DOI 10.1021/jp0734474.

[69] S. Spicher, S. Grimme, 'Single-Point Hessian Calculations for Improved Vibrational Frequencies and Rigid-Rotor-Harmonic-Oscillator Thermodynamics', *Journal of Chemical Theory and Computation* **2021**, *17*, 1701–1714, DOI 10.1021/acs.jctc.0c01306.

[70] K. Z. Sumon, C. H. Bains, D. J. Markewich, A. Henni, A. L. L. East, 'Semicontinuum Solvation Modeling Improves Predictions of Carbamate Stability in the CO2 + Aqueous Amine Reaction', *The Journal of Physical Chemistry B* **2015**, *119*, 12256–12264, DOI 10.1021/acs.jpcb.5b06076.

[71] J.-Q. Sun, K. Ruedenberg, 'Locating transition states by quadratic image gradient descent on potential energy surfaces', *The Journal of Chemical Physics* **1994**, *101*, 2157–2167, DOI 10.1063/1.467721.

[72] Q. Sun, S. Berkelbach, N. S. Blunt, G. H. Booth, S. Guo, Z. Li, J. Liu, J. D. McClain, E. R. Sayfutyarova, S. Sharma, S. Wouters, G. K.-L. Chan, 'PySCF: the Python-based simulations of chemistry framework', *WIREs Computational Molecular Science* **2018**, *8*, e1340, DOI 10.1002/wcms.1340.

[73] R. Sure, J. Antony, S. Grimme, 'Blind Prediction of Binding Affinities for Charged Supramolecular Host–Guest Systems: Achievements and Shortcomings of DFT-D3', *The Journal of Physical Chemistry B* **2014**, *118*, 3431–3440, DOI 10.1021/jp411616b.

[74] R. Sure, S. Grimme, 'Corrected small basis set Hartree-Fock method for large systems', *Journal of Computational Chemistry* **2013**, *34*, 1672–1685, DOI 10.1002/jcc.23317.

[75] L. H. Thomas, 'The calculation of atomic fields', *Mathematical Proceedings of the Cambridge Philosophical Society* **1927**, *23*, 542–548, DOI 10.1017/s0305004100011683.

[76] A. J. C. Varandas, 'CBS extrapolation of Hartree–Fock energy: Pople and Dunning basis sets hand-to-hand on the endeavour', *Physical Chemistry Chemical Physics* **2019**, *21*, 8022–8034, DOI 10.1039/c8cp07847e.

[77] S. J. Vevelstad, 'CO_2 Absorbent Degradation', Ph.D. Thesis, Department of Chemical Engineering, Norwegian University of Science and Technology, Trondheim, **2013**.

[78] I. Vorobyov, M. C. Yappert, D. B. DuPré, 'Hydrogen Bonding in Monomers and Dimers of 2-Aminoethanol', *The Journal of Physical Chemistry A* **2002**, *106*, 668–679, DOI 10.1021/jp013211e.

[79] A. F. Voter in *NATO Science Series*, Springer Netherlands, **2007**, pp. 1–23, DOI 10.1007/978-1-4020-5295-8_1.

[80] K. Wang, X. Shan, X. Chen, 'Electron propagator theory study of 2-aminoethanol conformers', *Journal of Molecular Structure: THEOCHEM* **2009**, *909*, 91–95, DOI 10.1016/j.theochem.2009.05.030.

[81] L.-P. Wang, C. Song, 'Geometry optimization made simple with translation and rotation coordinates', *The Journal of Chemical Physics* **2016**, *144*, DOI 10.1063/1.4952956.

[82] K. B. Wiberg, 'Basis set effects on calculated geometries: 6-311++G** vs. aug-cc-pVDZ', *Journal of Computational Chemistry* **2004**, *25*, 1342–1346, DOI 10.1002/jcc.20058.

[83] H.-B. Xie, Y. Zhou, Y. Zhang, J. K. Johnson, 'Reaction Mechanism of Monoethanolamine with CO2 in Aqueous Solution from Molecular Modeling', *The Journal of Physical Chemistry A* **2010**, *114*, 11844–11852, DOI 10.1021/jp107516k.

[84] T. Yanai, D. P. Tew, N. C. Handy, 'A new hybrid exchange–correlation functional using the Coulomb-attenuating method (CAM-B3LYP)', *Chemical Physics Letters* **2004**, *393*, 51–57, DOI 10.1016/j.cplett.2004.06.011.

[85] X. Yang, R. J. Rees, W. Conway, G. Puxty, Q. Yang, D. A. Winkler, 'Computational Modeling and Simulation of CO_2 Capture by Aqueous Amines', *Chemical Reviews* **2017**, *117*, 9524–9593, DOI 10.1021/acs.chemrev.6b00662.

[86] T. Yoshikawa, *Foundations of Robotics*, 14th ed., The MIT Press, **1990**, p. 20.

[87] I. Y. Zhang, X. Xu, *A New-Generation Density Functional: Towards Chemical Accuracy for Chemistry of Main Group Elements*, Springer Berlin Heidelberg, **2014**, pp. 1–20, DOI 10.1007/978-3-642-40421-4.

[88] T. Zhang, P. W. Stackhouse, B. Macpherson, J. C. Mikovitz, 'A solar azimuth formula that renders circumstantial treatment unnecessary without compromising mathematical rigor: Mathematical setup, application and extension of a formula based on the subsolar point and atan2 function', *Renewable Energy* **2021**, *172*, 1333–1340, DOI 10.1016/j.renene.2021.03.047.

[89] M. Zhao, A. I. Minett, A. T. Harris, 'A review of techno-economic models for the retrofitting of conventional pulverised-coal power plants for post-combustion capture (PCC) of CO2', *Energy Environ. Sci.* **2013**, *6*, 25–40, DOI 10.1039/c2ee22890d.

[90] Y. Zhao, D. G. Truhlar, 'Density Functionals with Broad Applicability in Chemistry', *Accounts of Chemical Research* **2008**, *41*, 157–167, DOI 10.1021/ar700111a.

A | Proof and Derivation

A.1 Proof of the Hohenberg-Kohn Existence theorem

If the ground state electron density determines the external potential, then all properties of the molecule could in principle be determined by the ground state electron density as well. To prove this, the opposite is presented as a supposition[7].

The Hamiltonian for a many-electron molecule

$$\hat{H} = -\frac{\hbar^2}{2m}\sum_{i=1}^{n}\nabla_i^2 + \sum_{i=1}^{n}\nu(\vec{r}_i) + \frac{1}{2}j_0 \sum_{\substack{\mathrm{i}\,=\,1,\\ \mathrm{j}\,=\,\mathrm{i}\,+\,1}}^{n} \frac{1}{|r_{ij}|} \tag{A.1}$$

Need to use the Slater-Condon rule for one-electron operators

$$\langle\Psi|\hat{\Omega}_1|\Psi\rangle = \sum_{i=1}^{n}\langle\phi_i(1)|\hat{\Omega}(\vec{r}_1)|\phi_i(1)\rangle \tag{A.2}$$

The overall *external potential* is a sum of one-electron terms, and can thus be expressed as

$$\begin{aligned}\langle\Psi|\sum_{i=1}^{n}\nu(\vec{r}_i)|\Psi\rangle &= \sum_{i=1}^{n}\langle\phi_i(1)|\nu(\vec{r}_1)|\phi_i(1)\rangle \\ &= \int\sum_{i}\phi_i^*(\vec{r}_1)\phi_i(\vec{r}_1)\nu(\vec{r}_1)d\vec{r}_1 \\ &= \int\rho(\vec{r})\nu(\vec{r})d\vec{r}\end{aligned} \tag{A.3}$$

Here the electron density is expressed in terms of spin orbitals ϕ_i. The integration in the last line replaces $\vec{r}_1$ by $\vec{r}$.

Next, suppose the two Hamiltonians $\hat{H}$ and $\hat{H}'$ correspond to the same ground state electron density, but their external potential $\nu(\vec{r})$ differ. Consequently, there are two different normalised wave functions Ψ and Ψ'. Using the result from above in eq. (A.3) and that Ψ' is a trail function for $\hat{H}$ the ground state energy can from the variational theorem (2) be written as

$$\begin{aligned}E_0 < \langle\Psi'|\hat{H}|\Psi'\rangle &= \langle\Psi'|\hat{H}'|\Psi'\rangle + \langle\Psi|\hat{H}-\hat{H}'|\Psi\rangle \\ &< E_0' + \int\rho(\vec{r})\left\{\nu(\vec{r})-\nu'(\vec{r})\right\}d\vec{r}\end{aligned} \tag{A.4}$$

Alternatively can Ψ be used as a trail function for $\hat{H}'$

$$\begin{aligned} E_0' &< \langle\Psi|\hat{H}'|\Psi\rangle = \langle\Psi|\hat{H}|\Psi\rangle + \langle\Psi|\hat{H}' - \hat{H}|\Psi\rangle \\ &< E_0 + \int \rho(\vec{r}) \left\{\nu(\vec{r}) - \nu'(\vec{r})\right\} d\vec{r} \end{aligned} \tag{A.5}$$

The sum of the equations (A.4) and (A.5) is

$$E_0 + E_0' < E_0' + E_0 \tag{A.6}$$

which enables a conclusion to be made that the initial presumption leads to a contradiction. Hence, the ground state electron density does in fact corresponds to a unique external potential. Consequently, all properties of the molecule can in principle be determined by the ground state electron density.

A.2 Proof of the Hohenberg-Kohn Variational Theorem

The ground state electron density determines the external potential of a system as already established in A.1. From that it follows, that a trail density function $\rho'(\vec{r})$ that integrates to n electrons and is positive everywhere. It founds a basis for the corresponding external potential $\nu'(\vec{r})$ and consequently the Hamiltonian $\hat{H}'$ and the normalised wave function Ψ'. In the following, the wave function acts as a trail function for the real system with the Hamiltonian $\hat{H}$. Together with the result from the variational theorem, i.e. $\langle\Psi'|\hat{H}|\Psi'\rangle \geq E_0$, it follows

$$\langle\Psi'|\hat{H}|\Psi'\rangle = \hat{T}[\rho'] + \hat{V}_{ee}[\rho'] + \int \rho'(\vec{r})\nu(\vec{r})d\vec{r} = E[\rho'] \tag{A.7}$$

It follows that $E[\rho'] \geq E_0$ because of the variational theorem $\langle\Psi'|\hat{H}|\Psi'\rangle \geq E_0$

A.3 Derivation of the dihedral angle

When calculating the dihedral angle θ, a set of four consecutive atoms i, j, k, l are considered. The dihedral angle corresponds to angle between the two planes constituted by the atoms i, j, k and j, k, l within the range $(-\pi, \pi)$ or equivalently $(-180°, 180°)$. The xyz-coordinates of these four consecutive atoms are relevant for the calculation of the following three vectors:

$$\begin{aligned} \vec{v}_{ij} &= j - i \\ \vec{v}_{jk} &= k - j \\ \vec{v}_{kl} &= l - k \end{aligned} \tag{A.8}$$

Taking the cross product of two non-zero vectors corresponds to a plane formed by the three corresponding atoms. In this plane, the normal vector is perpendicular to the two non-zero vectors

$$\begin{aligned} \vec{n}_{ijk} &= \vec{v}_{ij} \times \vec{v}_{jk} \\ \vec{n}_{jkl} &= \vec{v}_{jk} \times \vec{v}_{kl} \end{aligned} \tag{A.9}$$

The smallest angle between the normal vectors $\vec{n}_{ijk}$ and $\vec{n}_{jkl}$ corresponding to the dihedral angle can be expressed in two ways; in terms of the dot product or the cross product of the two normal vectors[15]

$$\cos\theta = \frac{\vec{n}_{ijk}\cdot\vec{n}_{jkl}}{|\vec{n}_{ijk}||\vec{n}_{jkl}|} = \frac{(\vec{v}_{ij}\times\vec{v}_{jk})\cdot(\vec{v}_{jk}\times\vec{v}_{kl})}{|\vec{v}_{ij}\times\vec{v}_{jk}||\vec{v}_{jk}\times\vec{v}_{kl}|} \tag{A.10}$$

$$\sin\theta = \frac{\vec{v}_{jk}}{|\vec{v}_{jk}|}\cdot\frac{\vec{n}_{ijk}\times\vec{n}_{jkl}}{|\vec{n}_{ijk}||\vec{n}_{jkl}|} = \frac{\vec{v}_{jk}}{|\vec{v}_{jk}|}\cdot\frac{\overbrace{(\vec{v}_{ij}\times\vec{v}_{jk})\times(\vec{v}_{jk}\times\vec{v}_{kl})}^{\text{vector quadruple product}}}{|\vec{v}_{ij}\times\vec{v}_{jk}||\vec{v}_{jk}\times\vec{v}_{kl}|} \tag{A.11}$$

where $\frac{\vec{v}_{jk}}{|\vec{v}_{jk}|}$ is a unit vector corresponding to the cross product of the the normal vectors $\vec{n}_{ijk}$ and $\vec{n}_{jkl}$, i.e. the normalised $\vec{v}_{jk}$. Further on, the *vector quadruple product* can be simplified to a *scalar triple product* making it easier to implement

$$\begin{aligned}
\vec{n}_{ijk}\times\vec{n}_{jkl} &= (\vec{v}_{ij}\times\vec{v}_{jk})\times(\vec{v}_{jk}\times\vec{v}_{kl}) \\
&= -(\vec{v}_{jk}\times\vec{v}_{ij})\times(\vec{v}_{jk}\times\vec{v}_{kl}) \\
&= -[\vec{v}_{jk}\cdot(\vec{v}_{ij}\times\vec{v}_{kl})]\cdot\vec{v}_{jk} \\
&= [\vec{v}_{jk}\cdot(\vec{v}_{kl}\times\vec{v}_{ij})]\cdot\vec{v}_{jk} \\
&= [\vec{v}_{ij}\cdot(\vec{v}_{jk}\times\vec{v}_{kl})]\cdot\vec{v}_{jk}
\end{aligned} \tag{A.12}$$

The second line leverages the *anti-commutative* property of the cross product (i.e. $\vec{a}\times\vec{b} = -\vec{b}\times\vec{a}$). This is necessary in order to rewrite the quadruple product into a scalar triple product ($[\vec{a}\cdot(\vec{b}\times\vec{c})]\vec{a} = (\vec{a}\times\vec{b})\times(\vec{a}\times\vec{c})$), as demonstrated in line 3. Next the anti-commutative relation of cross product is employed as evident in line 4 as for line 1. Line 5 makes use of that circular shift of the three vectors, wherein changing the order yields the same outcome ($\vec{a}\cdot(\vec{b}\times\vec{c}) = \vec{c}\cdot(\vec{a}\times\vec{b}) = \vec{b}\cdot(\vec{c}\times\vec{a})$). Consequently, this gives an even simpler sin expression than the one presented in eq. (A.11)

$$\begin{aligned}
\sin\theta &= \frac{\vec{v}_{jk}}{|\vec{v}_{jk}|}\cdot\frac{[\vec{v}_{ij}\cdot(\vec{v}_{jk}\times\vec{v}_{kl})]\vec{v}_{jk}}{|\vec{v}_{ij}\times\vec{v}_{jk}||\vec{v}_{jk}\times\vec{v}_{kl}|} \\
&= \frac{\vec{v}_{jk}}{|\vec{v}_{jk}|}\cdot\vec{v}_{jk}\frac{\vec{v}_{ij}\cdot(\vec{v}_{jk}\times\vec{v}_{kl})}{|\vec{v}_{ij}\times\vec{v}_{jk}||\vec{v}_{jk}\times\vec{v}_{kl}|} \\
&= \frac{|\vec{v}_{jk}|\vec{v}_{ij}\cdot(\vec{v}_{jk}\times\vec{v}_{kl})]}{|\vec{v}_{ij}\times\vec{v}_{jk}||\vec{v}_{jk}\times\vec{v}_{kl}|}
\end{aligned} \tag{A.13}$$

Nonetheless, when employing either the arccos or arcsin functions, challenges arise with respect to determining the sign and magnitude of the dihedral angle, where both aspects are decisive. These challenges stem from the intrinsic properties of the trigonometric functions, specifically their *domain* (all valid input values) and *range* (all valid output values)[34]

Table A.1: The domain and range for some trigonometric functions[12, 34, 57]

	sin	cos	arcsin	arccos	arctan	arctan2
domain	$(-\infty,\infty)$	$(-\infty,\infty)$	$[-1,1]$	$[-1,1]$	$(-\infty,\infty)$	$(-\infty,\infty)$
range	$[-1,1]$	$[-1,1]$	$[-\frac{\pi}{2},\frac{\pi}{2}]$	$[0,\pi]$	$(-\frac{\pi}{2},\frac{\pi}{2})$	$(-\pi,\pi)$

The arccos function can yield an angle, albeit with an inherent ambiguity in sign. This arises from the fact that arccos operates within the positive range between 0 and π, as illustrated in Table A.1. Conversely, the arcsin function is valid in the negative range but is confined to $[-\frac{\pi}{2}, \frac{\pi}{2}]$. To address this, both functions can be consider and the dihedral angle θ, can thus be calculated using the arctan function. However, dividing $\sin\theta$ by $\cos\theta$ results in sign cancellation, which is a major issue. The solution to this sign preservation is to employ the $\arctan 2$ function. It takes two arguments, $\arctan 2(y, x)$, and conserves the sign information, which is decisive for the resulting quadrant[12, 57].

The first set of equations in (A.14) deals with typical cases when $arctan2$ is well-defined based on the signs of x and y. The function returns an angle in the specified range[57]:

$$arctan2(y,x) = \begin{cases} [0, \frac{\pi}{2}] & \text{if } x > 0 \text{ and } y > 0 \\ [\frac{\pi}{2}, \pi] & \text{if } x < 0 \text{ and } y > 0 \\ [-\pi, -\frac{\pi}{2}] & \text{if } x < 0 \text{ and } y < 0 \\ \text{undefined} & \text{if } x = 0 \text{ and/or } y = 0 \text{ or if } x = \infty \text{ and/or } y = \infty \end{cases} \tag{A.14}$$

Although $arctan2$ is undefined for several special cases (A.14), they are taken care of according to the cases below in (A.15)[12]

$$arctan2(y,x) = \begin{cases} \pm\pi & \text{if } arctan2(\pm 0, -0) \\ \pm 0 & \text{if } arctan2(\pm 0, +0) \\ \pm\pi & \text{if } arctan2(\pm 0, +0) \text{ for } x < 0 \\ \pm 0 & \text{if } arctan2(\pm 0, +0) \text{ for } x > 0 \\ -\frac{\pi}{2} & \text{if } arctan2(y, \pm 0) \text{ for } y < 0 \\ \frac{\pi}{2} & \text{if } arctan2(y, \pm 0) \text{ for } y > 0 \\ \pm\pi & \text{if } arctan2(\pm y, -\infty) \text{ for finite } y > 0 \\ \pm\pi & \text{if } arctan2(\pm y, -\infty) \text{ for finite } y > 0 \\ \pm\frac{\pi}{2} & \text{if } arctan2(\pm\infty, x) \text{ for finite } x \\ \pm\frac{3\pi}{4} & \text{if } arctan2(\pm\infty, -\infty) \\ \pm\frac{\pi}{4} & \text{if } arctan2(\pm\infty, +\infty) \end{cases} \tag{A.15}$$

Moreover, the $arctan2$ function satisfies [86]

$$\theta = arctan2(\sin\theta, \cos\theta) \tag{A.16}$$

Consequently, the dihedral angle ought to be calculated using the $arctan2$ function by inserting eq. (A.13) and (A.10) into eq. (A.16)

$$\begin{aligned} \theta &= arctan2\left(\sin\theta, \cos\theta\right) \\ &= arctan2(\ |\vec{v}_{jk}|\vec{v}_{ij} \cdot (\vec{v}_{jk} \times \vec{v}_{kl}), (\vec{v}_{ij} \times \vec{v}_{jk}) \cdot (\vec{v}_{jk} \times \vec{v}_{kl})) \\ &= arctan2(|\vec{v}_{jk}|\vec{v}_{ij} \cdot \vec{n}_{jkl}, \vec{n}_{ijk} \cdot \vec{n}_{jkl}) \end{aligned} \tag{A.17}$$

providing an elegant implementation.

A.3.1 Calculating the C-C-N-lp Dihedral Angle

When considering a dihedral angle involving a lone pair, the method just given must be modified. Naturally, there is no explicit coordinates for a lone pair of electrons, but its direction can be estimated by considering the position of the adjacent hydrogens on the nitrogen. When looking through the N−C-bond and rotating the amine group so that the lone pair is on top, the hydrogens on the right and left side can be designated as l and m respectively. From this, two new vectors can be defined as

$$\begin{aligned} \vec{v}_{kl} &= l - k \\ \vec{v}_{km} &= m - k \end{aligned} \tag{A.18}$$

By adding these two vectors together, a new vector is obtained

$$\vec{v}_{kl} + \vec{v}_{km} = \vec{v}_{mid} \tag{A.19}$$

which will extended between the two addends vectors. Furthermore, changing the sign of a vector the direction gets inverted

$$\vec{v}_{lp} = -\vec{v}_{mid} \tag{A.20}$$

and gives a vector pointing somewhat in the direction of the lone pair. Next step is to define the normal vector perpendicular to the plane defined by C−N−lp

$$\vec{n}_{klm} = \vec{v}_{jk} \times \vec{v}_{lp} \tag{A.21}$$

where the normal vector $\vec{n}_{klm}$ is supposed to corresponds to the $\vec{n}_{jkl}$ from the original calculation. The dihedral angle with respect to the special case C−C−N−lp can be calculated as accordingly

$$\theta = arctan2(|\vec{v}_{jk}|\vec{v}_{ij} \cdot \vec{n}_{klm}, \vec{n}_{ijk} \cdot \vec{n}_{klm}) \tag{A.22}$$

B | Code

B.1 Process pool - Minimisation

```python
from pyscf import gto
from pyscf.geomopt.geometric_solver import optimize
from pyscf.hessian import thermo
import numpy as np
import os
import sys
# from pyscf.solvent import ddcosmo # Neccessary for PySCF 2.4.0
from concurrent.futures import ThreadPoolExecutor, ProcessPoolExecutor

# Function to find and read files
def find_and_read_files(directory, file_extension):
    name_list = []
    atoms_init_list = []
    file_path_list = []

    for root, dirs, files in os.walk(directory):
        for file in files:
            if file.endswith(file_extension):
                file_path = os.path.join(root, file)
                file_path_list.append(file_path)
                with open(file_path, 'r') as f:
                    atoms_init = f.read()[2:]

                atoms_init_list.append(atoms_init)
                filename = os.path.basename(file)
                modified_name = filename.replace(file_extension, '')
                name_list.append(modified_name)

    return name_list, atoms_init_list, file_path_list

# Function to write xyz-file
def write_xyz(mol_opt, out_xyz):
    with open(out_xyz, 'w+') as f:
        f.write(f'{mol_opt.natm}\n\n')
```

```
        for i in range(mol_opt.natm):
            b2a = mol_opt.atom_coord(i, unit = 'ANG')
            symbol = mol_opt.atom_symbol(i)
            f.write(f'{symbol} \t {b2a[0]} \t {b2a[1]} \t {b2a[2]}\n')
        f.seek(3)   # Move the pointer to the beginning
        opt_coords = f.read()
    return opt_coords

# Function to write the result-file
def write_result(result_out, atoms_init, opt_energy, energies, opt_coords,
    ↪freq_info, thermo_info, vibration_frequencies, rotation_constants,
    ↪symmetry_number):

    with open(result_out, 'w') as f:
        f.write(f'Input Geometry [Angstrom]:{atoms_init}\n\n')

        f.write(f'Optimised Geometry [Angstrom]:\n{opt_coords}\n\n')

        f.write(f'Optimised Energy [hartree]:\n{opt_energy}\n\n')

        f.write(f'The various energy contributions [hartree]:\n')
        for name, value in energies.items():
            f.write(f'{name}:\t{value}\n')

        f.write('\nVibrational Frequencies [cm^-1]:\n' +
                ', '.join([f'{vf.imag}j' if isinstance(vf, complex) and vf.imag !=
                    ↪0 else f'{vf.real}'
                           for vf in vibration_frequencies]))

        f.write(f'\n\nRotation Constants [GHz]:\n{", ".join(map(str,
            ↪rotation_constants))}\n')

        f.write(f'\nSymmetry number:\n{symmetry_number}\n')

        f.write(f'\n\nThermo info:\n')
        for name, value in thermo_info.items():
            f.write(f'{name}:\t\t{value}\n')

        f.write(f'\nFrequency info:\n\n')
        for name, value in freq_info.items():
            f.write(f'{name}:\t\t{value}\n')

# Function to perform geometry optimisation
def geometry_opt(basis, charge, diel, functional, pressure, temperature, verbose,
    ↪atoms_init, result_out, out_xyz, error_log):
    # Open the error log file to redirect the output
    with open(error_log, 'w') as error_file:
        # Redirect stdout and stderr to the error log file
        sys.stdout = sys.stderr = error_file
```

```
79
80          mol = gto.M(atom = atoms_init, basis = basis, charge = charge, verbose =
                ↪verbose)
81          mf = mol.UKS(xc = functional).ddCOSMO()
82          mf.with_solvent.eps = diel
83          mf.run()
84          mol_opt = optimize(mf)
85
86          mf_opt = mol_opt.UKS(xc=functional).ddCOSMO()
87          mf_opt.with_solvent.eps = diel
88          mf_opt.run()
89          opt_energy = mf_opt.e_tot
90          energies = mf_opt.scf_summary
91
92          hess = mf_opt.Hessian().kernel()
93          freq_info = thermo.harmonic_analysis(mol_opt, hess)
94          thermo_info = thermo.thermo(mf_opt, freq_info['freq_au'], temperature,
                ↪pressure)
95          # thermo_info = thermo_info[0] # Neccessary for PySCF 2.4.0
96          vibration_frequencies = np.array(freq_info['freq_wavenumber'])
97          rotation_constants = thermo_info['rot_const'][0]
98          symmetry_number = thermo_info['sym_number'][0]
99
100     # Restore stdout and stderr to their original state
101     sys.stdout = sys.__stdout__
102     sys.stderr = sys.__stderr__
103
104     opt_coords = write_xyz(mol_opt, out_xyz)
105     write_result(result_out, atoms_init, opt_energy, energies, opt_coords,
            ↪freq_info, thermo_info, vibration_frequencies, rotation_constants,
            ↪symmetry_number)
106
107     return opt_energy, energies, opt_coords, freq_info, thermo_info,
            ↪vibration_frequencies, rotation_constants, symmetry_number
108
109 if __name__ == '__main__':
110     basis = 'augccpvdz'
111     charge = 0
112     diel = 78.3553
113     functional = 'b3lyp'
114     pressure = 101325
115     solvent = 'water'
116     temperature = (273.15 + 25)
117     state = 'minimum'
118     verbose = 4
119     workers = 4
120
121     directory_to_search = f'/Users/kristine/pyscf/mea/minima/gasphase/augccpvdz'
122     file_extension_to_find = f'_minimum_b3lyp_augccpvdz.xyz'
```

```
    result_path = f'/Users/kristine/pyscf/mea/water/minima'

    name_list, atoms_init_list, file_path_list = find_and_read_files(
        ↪directory_to_search, file_extension_to_find)

    # Use ProcessPoolExecutor to run tasks concurrently
    with ProcessPoolExecutor(max_workers = workers) as executor:
        task_list = []
        for name, atoms_init, file_path in zip(name_list, atoms_init_list,
            ↪file_path_list):
            mol_dir = os.path.join(result_path, basis, name)
            if not os.path.exists(mol_dir):
                os.makedirs(mol_dir)

                out_xyz = f'{mol_dir}/{name}_{solvent}_{state}_{functional}_{basis
                    ↪}.xyz'
                result_out = f'{mol_dir}/{name}_{solvent}_{state}_{functional}_{
                    ↪basis}.out'
                error_log = f'{mol_dir}/{name}_{solvent}_error.log'

                # Start a new task for each molecule
                task = executor.submit(geometry_opt, basis, charge, diel,
                    ↪functional, pressure, temperature, verbose, atoms_init,
                    ↪result_out, out_xyz, error_log)
                task_list.append(task)

            else:
                print(f'The {mol_dir} directory already exists')

        for task in tasks:
            task.result()
```

Listing B.1: Code to perform geometry optimisation in a given directory with ProcessPoolExecutor

B.2 Process pool - Transition state

```
from pyscf import gto
from pyscf.qsdopt.qsd_optimizer import QSD
from pyscf.hessian import thermo
import numpy as np
import os
import sys
# from pyscf.solvent import ddcosmo # Neccessary for PySCF 2.4.0
from concurrent.futures import ProcessPoolExecutor

# Function to find and read files
def find_and_read_files(directory, file_extension):
    names = []
```

```
    atoms_init_list = []
    file_paths = []

    pattern_to_remove = file_extension.replace('.xyz', '')

    for root, dirs, files in os.walk(directory):
        for file in files:
            if file.endswith(file_extension):
                file_path = os.path.join(root, file)
                with open(file_path, 'r') as f:
                    atoms_init = ''.join(f.readlines()[2:])  # Read file content
                        ↪starting from line 3

                # Modify the name and store it in names
                filename = os.path.basename(file)
                modified_name = filename.replace(pattern_to_remove, '')[:-4]  #
                    ↪Remove pattern and .xyz extension
                names.append(modified_name)
                atoms_init_list.append(atoms_init)
                file_paths.append(file_path)

    return names, atoms_init_list, file_paths

# Function to write xyz-file
def write_xyz(mol, out_xyz):
    with open(out_xyz, 'w+') as f:
        f.write(f'{mol.natm}\n\n')
        for i in range(mol.natm):
            b2a = mol.atom_coord(i, unit = 'ANG')
            symbol = mol.atom_symbol(i)
            f.write(f'{symbol} \t {b2a[0]} \t {b2a[1]} \t {b2a[2]}\n')
        f.seek(3)   # Move the pointer to the beginning
        ts_coords = f.read()
    return ts_coords

def write_result(result_out, conv, atoms_init, ts_energy, energies, ts_coords,
    ↪thermo_info, freq_info, vibration_frequencies, rotation_constants,
    ↪symmetry_number):
    with open(result_out, 'w') as f:
        f.write(f'The calculation has converged:\n{conv}\n\n')

        f.write(f'Input geometry [Angstrom]:\n{atoms_init}\n\n')

        f.write(f'Transition State Coordinates [Angstrom]:\n{ts_coords}\n\n')

        f.write(f'\nTransition State Energy [hartree]:\n{ts_energy}\n\n')

        f.write(f'Energy contributions [hartree]:\n')
        for name, value in energies.items():
```

```
58            f.write(f'{name}:\t{value}\n')
59
60        f.write('\nVibrational Frequencies [cm^-1]:\n' +
61                ', '.join([f'{vf.imag}j' if isinstance(vf, complex) and vf.imag !=
                      ↪0 else f'{vf.real}'
62                           for vf in vibration_frequencies]))
63
64        f.write(f'\n\nRotation Constants [GHz]:\n{", ".join(map(str,
              ↪rotation_constants))}\n')
65
66        f.write(f'\nSymmetry number:\n{symmetry_number}\n')
67
68        f.write(f'\n\nThermo info:\n')
69        for name, value in thermo_info.items():
70            f.write(f'{name}:\t\t{value}\n')
71
72        f.write(f'\nFrequency info:\n\n')
73        for name, value in freq_info.items():
74            f.write(f'{name}:\t\t{value}\n')
75
76 # Function to perform geometry optimization and return optimized energy and
       ↪coordinates
77 def ts_search(basis, charge, diel, functional, pressure, temperature, verbose,
       ↪atoms_init, result_out, out_xyz, error_log):
78     # Open the error log file to redirect the output
79     with open(error_log, 'w') as error_file:
80         # Redirect both stdout and stderr to the error log file
81         sys.stdout = sys.stderr = error_file
82
83         mol = gto.M(atom = atoms_init, basis = basis, charge = charge, verbose =
               ↪verbose)
84         mf = mol.UKS(xc = functional).ddCOSMO()
85         mf.with_solvent.eps = diel
86         mf.run()
87         optimizer = QSD(mf, stationary_point = 'TS')
88         optimizer.kernel(numhess_method = 'central')
89         mf_ts = mol.UKS(xc = functional).ddCOSMO()
90         mf_ts.with_solvent.eps = diel
91         mf_ts.run()
92
93         conv = optimizer.converged
94         ts_energy = mf_ts.e_tot
95         energies = mf_ts.scf_summary
96         hess = mf_ts.Hessian().kernel()
97         freq_info = thermo.harmonic_analysis(mf_ts.mol, hess)
98         thermo_info = thermo.thermo(mf_ts, freq_info['freq_au'], temperature,
               ↪pressure)
99         # thermo_info = thermo_info[0] # Neccessary for PySCF 2.4.0
100        vibration_frequencies = np.array(freq_info['freq_wavenumber'])
```

```
        rotation_constants = thermo_info['rot_const'][0]
        symmetry_number = thermo_info['sym_number'][0]

        ts_coords = write_xyz(mol, out_xyz)
        write_result(result_out, conv, atoms_init, ts_energy, energies, ts_coords,
            ↪freq_info, thermo_info, vibration_frequencies, rotation_constants,
            ↪symmetry_number)

        return ts_energy, energies, ts_coords, freq_info, thermo_info,
            ↪vibration_frequencies, rotation_constants, symmetry_number

if __name__ == '__main__':
    basis = 'augccpvdz'
    charge = 0
    diel = 78.3553
    functional = 'b3lyp'
    pressure = 101325
    solvent = 'water'
    state = 'ts'
    temperature = (273.15 + 25)
    verbose = 4
    workers = 2

    directory_to_search = f'/Users/kristine/pyscf/mea/maxima/gasphase/augccpvdz'
    file_extension_to_find = f'_ts_b3lyp_augccpvdz.xyz'
    result_path = f'/Users/kristine/pyscf/mea/maxima/water'

    names, atoms_init_list, file_paths = find_and_read_files(directory_to_search,
        ↪file_extension_to_find)

   # Use ProcessPoolExecutor to run tasks concurrently
    with ProcessPoolExecutor(max_workers = workers) as executor:
        task_list = []
        for name, atoms_init, file_path in zip(names, atoms_init_list, file_paths):
            mol_dir = os.path.join(result_path, basis, name)
            if not os.path.exists(mol_dir):
                os.makedirs(mol_dir)

                out_xyz = f'{mol_dir}/{name}_{solvent}_{state}_{functional}_{basis
                    ↪}.xyz'
                result_out = f'{mol_dir}/{name}_{solvent}_{state}_{functional}_{
                    ↪basis}.out'
                error_log = f'{mol_dir}/{name}_{solvent}_error.log'

                # Start a new task for each molecule
                task = executor.submit(ts_search, basis, charge, diel, functional,
                    ↪pressure, temperature, verbose, atoms_init, result_out,
                    ↪out_xyz, error_log)
                task_list.append(task)
```

```
142
143             else:
144                 print(f'The {mol_dir} directory already exists')
145
146         # Wait for all tasks to complete
147         for task in tasks:
148             task.result()
149
150     print('All tasks completed.')
```

Listing B.2: Code to perform concurrent transition state searches with ProcessPoolExecutor

B.3 Calculating the Dihedral Angles

```
1  import numpy as np
2
3  def calculate_dihedral(atom1, atom2, atom3, atom4, atom5 = None):
4      i, j, k, l = (np.array(atom1[1:], dtype = float),
5                    np.array(atom2[1:], dtype = float),
6                    np.array(atom3[1:], dtype = float),
7                    np.array(atom4[1:], dtype = float))
8      v_ij = j - i
9      v_jk = k - j
10     v_kl = l - k
11     n_ijk = np.cross(v_ij, v_jk)
12     n_jkl = np.cross(v_jk, v_kl)
13     angle = np.arctan2(np.dot(np.linalg.norm(v_jk) * v_ij, n_jkl), np.dot(n_ijk,
           ↪n_jkl))
14
15     if atom5 is not None:
16         m = (np.array(atom5[1:], dtype = float))
17         v_kl = l - k # v_NHa
18         v_km = m - k # v_NHb
19         v_mid = v_kl + v_km
20         v_lp = -v_mid
21         n_klm = np.cross(v_jk, v_lp)
22         lp_angle = np.arctan2(np.dot(np.linalg.norm(v_jk) * v_ij, n_klm), np.dot(
               ↪n_ijk, n_klm))
23         return np.degrees(lp_angle)
24     return np.degrees(angle)
25
26 mol_xyz = '''11
27
28 O     -2.0707883056580845    -2.0772383162067536    1.9641095613190434
29 H     -1.2840171177008821    -2.0633719862569664    1.392170998536323
30 H     -3.0906921702079555    -0.8596263437115265    -0.23965149420804374
31 C     -2.9775430360837425    -1.9512873356915454    -0.29065195007958095
32 N     -1.6204350806090628    -2.2415361077155866    -0.7852831216668819
```

```
C     -3.124486992553661    -2.5372190156779526    1.110589983559218
H     -1.3893167524636094   -1.6491535185050932    -1.578212928907465
H     -4.0755120065002        -2.225545040004764     1.561865209276729
H     -3.7808887141288854   -2.345632145696079     -0.9364173468472151
H     -1.5575232562408994   -3.2031698499543726    -1.1148001967755625
H     -3.1136614068480255   -3.6408232737091035    1.0601567058153487
'''
xyzs = [line.split() for line in mol_xyz.strip().split('\n')[2:] if line.strip()]
dihedral_set_list = [(5, 3, 4, 6, 9), # Dihedral C-C-N-Ha-Hb: [5, 3, 4, 6, 9]
                     (0, 5, 3, 4),    # Dihedral O-C-C-N: [0, 5, 3, 4]
                     (3, 5, 0, 1),    # Dihedral C-C-O-H: [3, 5, 0, 1]
                     ]
dihedral_list = []
for dihedral_set in dihedral_set_list:
    dihedral = calculate_dihedral(*[xyzs[i] for i in dihedral_set])
    dihedral_list.append(dihedral)
for i, dihedral in enumerate(dihedral_list, start = 1):
    symbol_dihedral = [xyzs[j][0] for j in dihedral_set_list[i - 1]]
    print(f'Dihedral angle {"-".join(symbol_dihedral)}: {dihedral:.1f} degrees')
```

Listing B.3: This script calculates the dihedral angles (in degrees), where a set of predefined dihedrals are provided

B.4 Calculating Bond Lengths

```
import numpy as np

def calculate_bond_length(xyzs):
    bond_list = []
    for i in range(len(xyzs)):
        for j in range(i + 1, len(xyzs)):
            atom_i = xyzs[i]
            atom_j = xyzs[j]
            if atom_i[0] == 'H' and atom_j[0] == 'H': continue
            coord_i, coord_j = (np.array(atom_i[1:], dtype=float),
                                np.array(atom_j[1:], dtype=float))
            bond = np.linalg.norm(coord_i - coord_j)
            if bond < 2:
                bond_list.append((i, j, bond))
    return bond_list

mol_xyz = '''11

O     -2.0707883056580845   -2.0772383162067536   1.9641095613190434
H     -1.2840171177008821   -2.0633719862569664   1.392170998536323
H     -3.0906921702079555   -0.8596263437115265   -0.23965149420804374
C     -2.9775430360837425   -1.9512873356915454   -0.29065195007958095
N     -1.6204350806090628   -2.2415361077155866   -0.7852831216668819
```

```
C     -3.124486992553661    -2.5372190156779526    1.110589983559218
H     -1.3893167524636094   -1.6491535185050932    -1.578212928907465
H     -4.0755120065002        -2.225545040004764     1.561865209276729
H     -3.7808887141288854   -2.345632145696079     -0.9364173468472151
H     -1.5575232562408994   -3.2031698499543726    -1.1148001967755625
H     -3.1136614068480255   -3.6408232737091035    1.0601567058153487
'''

xyzs = [line.split() for line in mol_xyz.strip().split('\n')[2:] if line.strip()]

bonds = calculate_bond_length(xyzs)

print('Bond lengths:')
for count, (idx_i, idx_j, length) in enumerate(bonds, start=1):
    atom_i = xyzs[idx_i][0]
    atom_j = xyzs[idx_j][0]
    print(f'{count}: {atom_i}({idx_i})-{atom_j}({idx_j}):\t{length:.1f} angstroms')
```

Listing B.4: This script calculates all bond lengths (< 2 angstrom) in a xyz-multistring obtained from a .xyz-file

B.5 Visualise Molecules with py3Dmol

```
import py3Dmol

def view_mol(mol_xyz):
    view = py3Dmol.view(width = 400, height = 400)
    view.addModel(mol_xyz, 'xyz')
    view.setStyle({'stick': {}, 'sphere': {'scale': 0.3}})
    view.zoomTo()
    return view.show()
```

Listing B.5: This code displays the molecule using the py3Dmol package, where mol_xyz is the file content of a xyz-file

B.6 Calculating Rate Constants

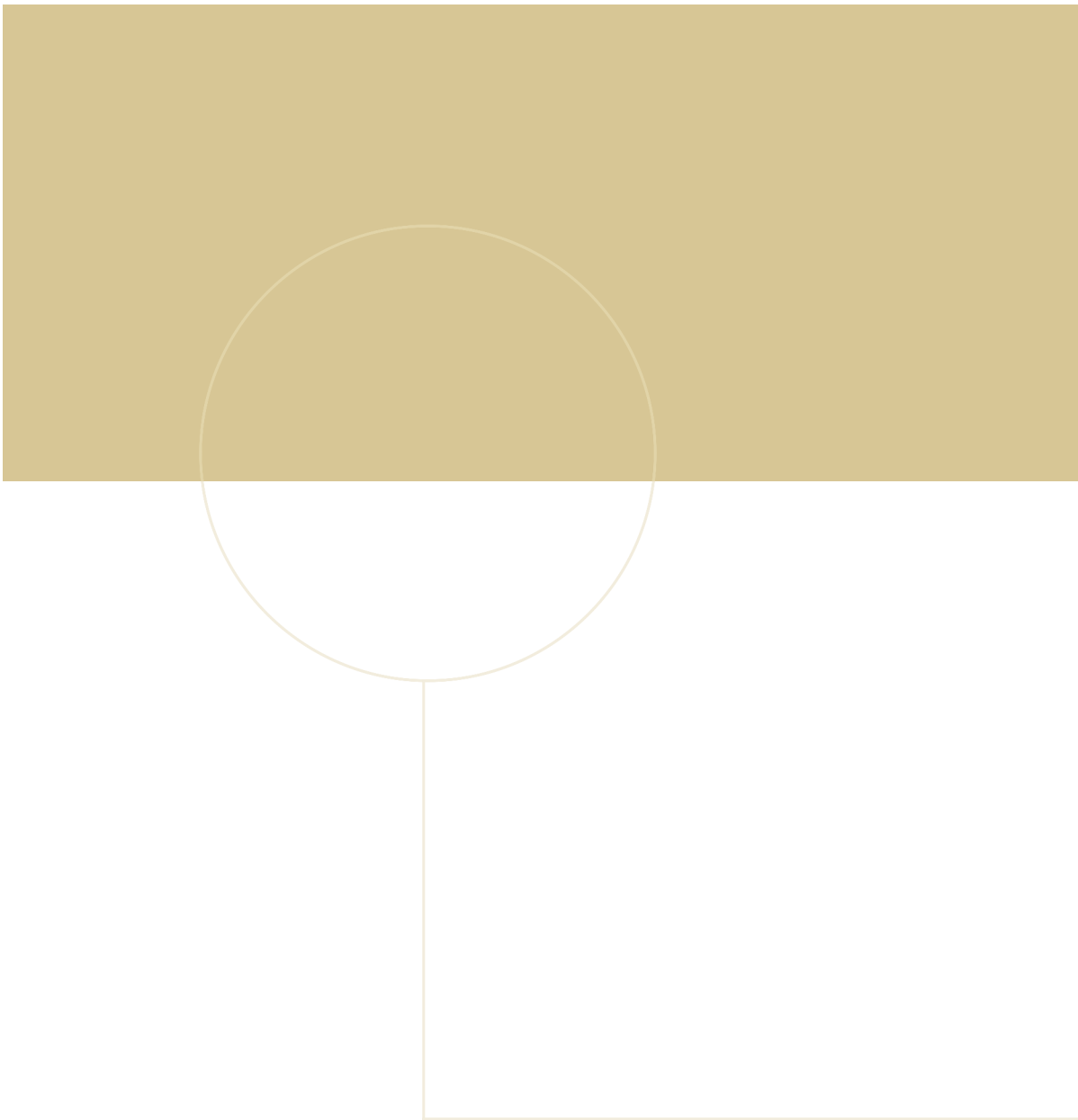
```
import numpy as np
from pyscf import gto

def read_xyz(input_xyz):
    with open(input_xyz, 'r') as f:
        opt_coords = f.read()[2:]   # Except the first two lines
    return opt_coords

# --- From thermo.py --- START
```

```
def _get_rotor_type(rot_const):
    if np.all(rot_const > 1e8):
        rotor_type = 'ATOM'
    elif rot_const[0] > 1e8 and (rot_const[1] - rot_const[2] < 1e-3):
        rotor_type = 'LINEAR'
    else:
        rotor_type = 'REGULAR'
    return rotor_type

def calculate_q_rot(rot_const, sym_number):
    rotor_type = _get_rotor_type(rot_const)
    if rotor_type == 'LINEAR':
        B = rot_const[1] * 1e9
        q_rot = kB * temperature / (sym_number * h * B)
    else:
        ABC = rot_const * 1e9
        q_rot = (kB*temperature/h)**1.5 * np.pi**0.5 / (sym_number * np.prod(ABC)
            ↪**0.5)
    return q_rot

def calculate_q_trans(mol):
    mass = mol.atom_mass_list(isotope_avg = True)
    mass_tot = mass.sum() * atomic_mass
    q_trans = (2 * np.pi * mass_tot * kB * temperature / h**2)**1.5 * kB *
        ↪temperature / pressure
    return q_trans

def calculate_q_vib(freq):
    freq = freq * c * 1e2
    vib_temp_list = freq * h / kB

    q_vib_list = 1/(1 - np.exp(- vib_temp_list / temperature))
    q_vib = np.prod(q_vib_list)
    return q_vib
# --- From thermo.py --- END

def calculate_k(mol_min, mol_ts, rot_const_min, rot_const_ts, freq_min, freq_ts,
    ↪energy_min, energy_ts):
    q_rot_min = calculate_q_rot(rot_const_min, sym_number_min)
    q_trans_min = calculate_q_trans(mol_min)
    q_vib_min = calculate_q_vib(freq_min)

    q_rot_ts = calculate_q_rot(rot_const_ts, sym_number_ts)
    q_trans_ts = calculate_q_trans(mol_ts)
    q_vib_ts = calculate_q_vib(freq_ts)

    K_r = q_rot_ts/q_rot_min
    K_t = q_trans_ts/q_trans_min
    K_v = q_vib_ts/q_vib_min
```

```
56
57      k = (kB * temperature/h) * K_t * K_r * K_v * np.exp(-deltaE/(kB * temperature))
58      return k
59
60
61  if  __name__ == "__main__":
62      avogadro = 6.022140857e23
63      atomic_mass = 1e-3/avogadro
64      bohr = 0.52917721092
65      bohr_si = bohr * 1e-10
66      c = 299792458
67      e_mass = 9.10938356e-31
68      h = 6.626070040e-34
69      hbar = h/(2 * np.pi)
70      hartree2J = hbar**2 / (e_mass * bohr_si**2)
71      au2hz = (hartree2J / (atomic_mass * bohr_si**2))**0.5 / (2 * np.pi)
72      kB = 1.38064852e-23
73      pressure = 101325
74      temperature = (273.15 + 25)
75
76      sym_number_min = 1
77      sym_number_ts = 1
78      sym_number_min2 = 1
79
80      energy_min = -210.43341086557925
81      energy_ts = -210.42520525626108
82      energy_min2 = -210.43018776401314
83
84      deltaE_f = (energy_ts - energy_min) * hartree2J
85      deltaE_r = (energy_ts - energy_min2) * hartree2J
86
87      min = ts = '1g-Gg-'
88      min2 = '24g-Tg-'
89
90      input_xyz_min = f'{min}_water_minimum_b3lyp_augccpvdz.xyz'
91      input_xyz_ts = f'{ts}_water_ts_b3lyp_augccpvdz.xyz'
92      input_xyz_min2 = f'{min2}_water_minimum_b3lyp_augccpvdz.xyz'
93
94      xyz_min = read_xyz(input_xyz_min)
95      xyz_ts = read_xyz(input_xyz_ts)
96      xyz_min2 = read_xyz(input_xyz_min2)
97
98      rot_const_min = np.array([14.22, 5.61, 4.56])
99      rot_const_ts = np.array([21.59, 3.92, 3.89])
100     rot_const_min2 = np.array([28.18, 3.82, 3.58])
101
102     freq_min = np.array([164.82, 257.67, 320.09, 512.00, 540.64, 804.91, 875.91,
            ↪913.39, 998.06, 1057.27, 1104.15, 1174.52, 1241.62, 1310.09, 1350.61,
            ↪1388.92, 1426.23, 1476.93, 1487.48, 1639.82, 2962.39, 2984.94, 3077.91,
```

```
        ↪3087.33, 3495.07, 3582.88, 3724.43])
103     freq_ts = np.array([259.78, 324.21, 412.41, 444.57, 815.09, 852.34, 965.72,
            ↪999.68, 1056.18, 1095.77, 1177.27, 1214.33, 1309.76, 1361.69, 1388.53,
            ↪1418.13, 1474.29, 1491.19, 1643.94, 2966.02, 3012.11, 3069.87, 3106.52,
            ↪3473.78, 3552.53, 3814.25])
104     freq_min2 = np.array([128.71, 229.89, 269.37, 301.84, 473.09, 788.25, 830.39,
            ↪960.30, 1056.50, 1072.05, 1096.94, 1123.87, 1267.55, 1302.89, 1338.65,
            ↪1387.34, 1417.95, 1482.31, 1495.05, 1642.63, 2982.60, 3024.22, 3045.08,
            ↪3084.38, 3490.93, 3574.47, 3813.87])
105
106     mol_min = gto.M(atom = xyz_min)
107     mol_ts = gto.M(atom = xyz_ts)
108     mol_min2 = gto.M(atom = xyz_min2)
109
110     kf = calculate_k(mol_min, mol_ts, rot_const_min, rot_const_ts, freq_min,
            ↪freq_ts, deltaE_f)
111     kr = calculate_k(mol_min2, mol_ts, rot_const_min2, rot_const_ts, freq_min2,
            ↪freq_ts, deltaE_r)
112
113     print(f'The forward rate constant:\t{kf:e} s-1')
114     print(f'The backward rate constant:\t{kr:e} s-1')
```

Listing B.6: This is the code for calculating the rate constants derived from transition state theory



NTNU

Norwegian University of
Science and Technology