UNIVERSITY OF SOUTHAMPTON

Methods for Using Large-Scale First Principles Quantum Mechanical Calculations to Compute Free Energies of Binding

by

Christopher M. Sampson

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the

Faculty of Natural and Environmental Sciences

School of Chemistry

September 2015

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF NATURAL AND ENVIRONMENTAL SCIENCES

SCHOOL OF CHEMISTRY

Doctor of Philosophy

by Christopher M. Sampson

The work presented within this thesis uses quantum mechanical (QM) calculations to improve free energies of binding computed with classical (MM) force fields. Initially a direct approach was taken, where snapshots were taken at equally spaced distances throughout the classical simulation and each structure underwent a quantum single point energy calculation. This direct approach was possible by using the Zwanzig equation. However, one disadvantage of using the Zwanzig equation is it's extreme sensitivity to fluctuations in the energy difference. This led to the quantum corrected free energy being dominated by a few snapshots and convergence could not be achieved. This led to the application of an acceptance criterion, where instead of just using each evenly spaced structure from the classical simulation, each structure would have to be accepted into a target potential. In the work presented here, our target potential was a QM potential. For a simple test system of N_2 in vacuum we achieved a high acceptance and converged free energies, however, for more complex systems little to no acceptance was found. The poor acceptance can be attributed to the difference between the MM and QM potential energy surfaces. Similarities were found, however, on the minima of these potential energy surfaces between the MM and QM, which led to the application of a bias to ensure that sampling was only taken from the minima. However, similar to the Zwanzig equation, this method proved to be too sensitive to difference in energy, thus convergence could not be achieved.

In order to "smooth" the transition between the MM and QM a "stepping stone" approach was used. The first step was to accept structures from a classical simulation to a QM/MM ensemble, then we used a direct approach again using the Zwanzig equation to move from the QM/MM potential to the QM. Using this approach, we find very small convergence errors (< 1 kJ/mol). This method was validated by calculating hydration free energies for a variety of ligands.

Finally, the free energy of binding was calculated for trypsin with several benzamidine derivatives using a QM-PBSA approach, which involved running QM calculations on the entire protein-ligand complex. The final results, however, showed no overall improvement of the calculated free energies between the MM and QM. It was found that the inclusion of QM methods lowered the free energy in each case.

Contents

List of Figures

List of Tables

I, Christopher Martin Sampson declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research.

[title of thesis] Methods for Using Large-Scale First Principles Quantum Mechanical Calculations to Compute Free Energies of Binding

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Either none of this work has been published before submission, or parts of this work have been published as: C. Sampson, T. Fox, C. Tautermann, C. Woods and C.-K. Skylaris. A "stepping-stone" approach for obtaining quantum free energies of hydration. *Journal of Physical Chemistry B*, 119:7030-7040;

Signed:

Date:

Acknowledgements

There are many people that have helped me in getting to this point, far too many to fit all of them in this short section. Firstly I would like to thank Prof. Skylaris for his supervision over the last four years as well as my industrial supervisors Dr. Thomas Fox and Dr. Christofer Tautermann, whose advice was invaluable. I would also like to thank my academic advisor Prof. Jon Essex for advise given throughout the project.

I also need to further thank Boehringer Ingelheim, and the BBSRC for funding part of my PhD studentship and to my wife, Lucy, for funding the rest. Most of this work would not have been possible without the support of the HPC team at Southampton, specifically Ivan Wolton, who deserves a special mention.

When it came to checking through my thesis I need to thank Prof. Skylaris, Dr. Christofer Tautermann and Dr. Karl Wilkinson for corrections, Dr. Chris Pittock for a general read through and Mr. Valerio Vitale for corrections on the QM section of the background. A massive thanks goes to Mr. Max Phipps, who acted as my hands in Southampton when I moved to Devon.

Thanks to my examiners for taking the time to read my thesis and conduct my viva. And thanks to future readers, I hope my work can be of some use to you.

I'd like to thank my wife, Lucy, my parents and my sister, Becky, for supporting me throughout this whole studentship and listening, with a sympathetic ear, to me moan, at length, when things weren't going so well! My colleagues and friends, Mr. Jonny Cave, Dr. Karl Wilkinson, Mr. Max Phipps, Mr. Jolyon Aarons, Mr. Valerio Vitale, Dr. Zohra Ouaray and Mr. Frank Longford, it was the schemes and side projects that kept me going!

I'm sure I have missed lots of people, My final and largest thanks must go to my wife. Without her, I could not have completed my studentship or thesis.

To Lucy...

There are more things in heaven and earth, Horatio, Than are dreamt of in your philosophy.

Hamlet (1.5.167-8)

Chapter 1

Background

1.1 Molecular Mechanics

Balancing computational accuracy with time is one of the main challenges within theoretical chemistry. Molecular Mechanics (MM) uses classical mechanics to describe the behaviour of molecules, but by doing this the accuracy is compromised. The benefit, however, is that the energy of large systems, for example proteins, can be calculated quickly, typically in less than a second. It is because of this speed that it is typically paired with Molecular Dynamics (section 1.3) or Monte Carlo (section 1.4) to generate many conformations of a system quickly.

MM uses the approximation that atoms can be represented by hard spheres and bonds as springs, although there are some exceptions to this, such as Lennard-Jones spheres. As such, no electrons exist within this method and any value that depends explicitly on the electronic structure cannot be calculated. All parameters required for MM are contained within force fields. A successful force field is one with a high degree of transferability, such that it can describe a number of systems. One of the most widely used potential energy functions used for protein force fields is the following [1],

$$\begin{aligned} V(r) \;=\; & \sum_{bonds} k_b(b-b_0)^2 + \sum_{angles} k_\theta(\theta-\theta_0)^2 + \sum_{torsions} k_\phi[cos(n\phi+\delta)+1] \\ & + \sum_{nbp}\left[\frac{q_iq_j}{4\pi\epsilon_0 r_{ij}} + \frac{A_{ij}}{r_{ij}^{12}} - \frac{C_{ij}}{r_{ij}^{6}}\right]. \end{aligned} \tag{1.1}$$

The first term on the right hand side of equation 1.1 uses a harmonic potential to describe all bonding interactions. k_b is the force constant and controls how steep the walls of the potential are, b is the current length of the bond and b_0 is the equilibrium bond length, the center of the potential. Using a harmonic potential to describe bonding interactions is quick and simple, however, within this model bonds cannot break. A similar term exists for the angles, k_θ is the force constant, θ is the current value for the angle and θ_0 is the equilibrium angle.

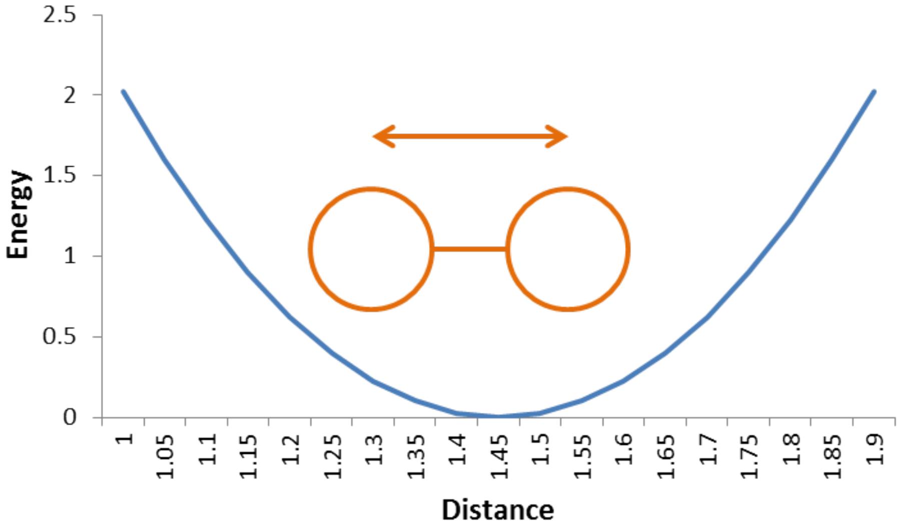


FIGURE 1.1: Harmonic potential used to describe bonds

The third term on the right hand side of equation 1.1 describes the dihedral angles. k_ϕ is the force constant, n controls how many peaks present within the potential (the multiplicity) and δ shifts the potential to the left or right (the phase).

The last term in equation 1.1 describes the nonbonding terms and are summed over all non-bonded pairs. The term is commonly split into two parts, the electrostatics and the

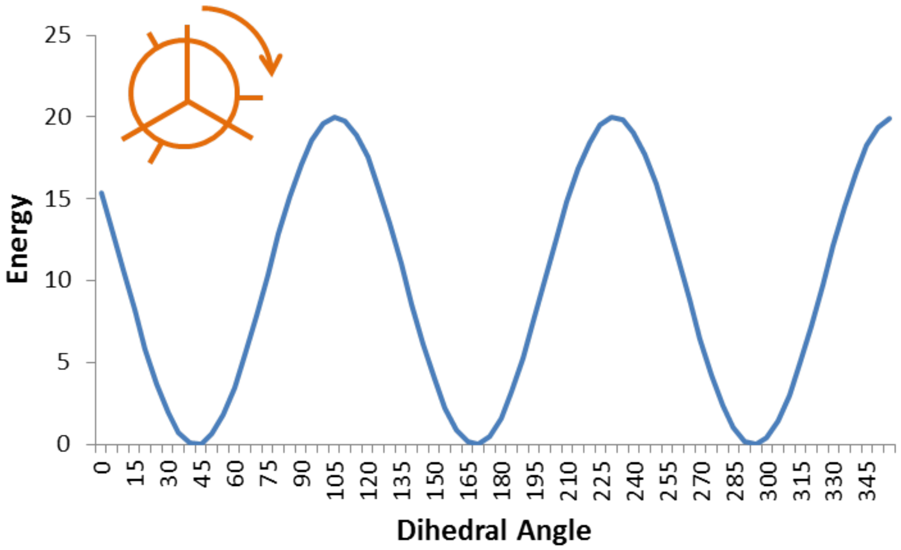


FIGURE 1.2: Potential used to describe dihedral angles

dispersion. The electrostatic interactions are modelled using Coulomb's law,

$$U_{elec} = \sum_{i=1}^{N} \sum_{j=1}^{N-1} \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}}. \tag{1.2}$$

Where q_i is the charge on atom i, r_{ij} is the distance between atom i and j, N is the total number of atoms present and ϵ_0 is the dielectric constant.

If electrostatics were the only interaction considered, atoms with opposing partial charges would be attracted to unrealistically small distances. By including Van der Waals forces, the attraction caused by the partial charges at small distances is less than the repulsive force considered for the Van der Waals interactions. A Lennard-Jones potential is commonly used to describe the interaction between two neutral atoms,

$$U_{disp} = \sum_{i=1}^{N} \sum_{j=1}^{N-1} 4\epsilon \left[\left(\frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}} \right)^{6} \right]. \tag{1.3}$$

Where ϵ is the well depth, σ_{ij} is the Lennard-Jones sphere radius and is related to the position of the minimum r^*_{ij} by the following [2],

$$r_{ij}^{*} = 2^{1/6}\sigma_{ij}. \tag{1.4}$$

$(\sigma/r)^{12}$ described the repulsive forces and $(\sigma/r)^{6}$ describes the attractive forces (dispersion).

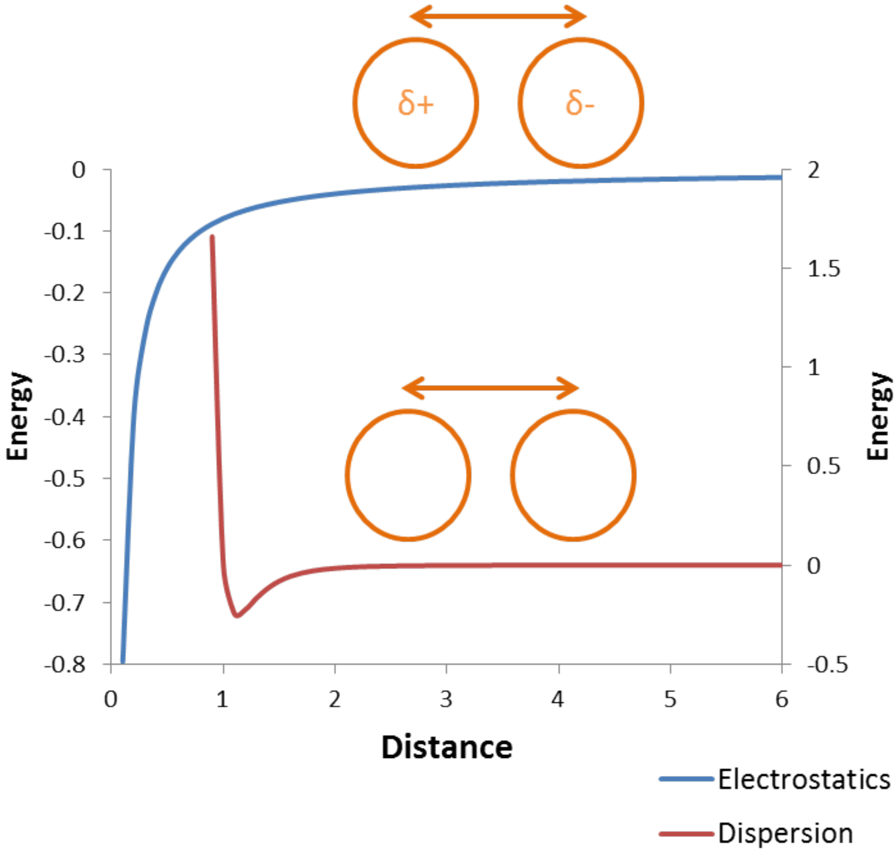


Figure 1.3: Potentials used to describe nonbonded interactions, where the right side energy is the interaction energy between 2 neutral atoms and the left side is the electrostatic energy

The example of Figure 1.3 shows that the dispersion forces converge to 0 rapidly compared to the electrostatic forces, which do not converge to 0 within the range of the figure. The nonbonded forces are the most computationally expensive calculation within the classical potential energy because of the computational scaling with the system size which is formally order N^2, where N is the number of atoms. To lower this cost, long-range methods are used to calculate these interactions after a cut off distance. It is common to only calculate the interactions explicitly within a user-defined cut off. However, this cut off is unphysical, so not possible within experiment and changing it can have serious implications for the total energy and properties of the system [3].

Within AMBER, Van der Waals interactions are calculated using the following [3],

$$U_{full} = U_{r<r_c} + U_{r>r_c}. \tag{1.5}$$

Where r is the distance between two particles, r_c is the cutoff distance and therefore $U_{r<r_c}$ are the short-range van der Waal interactions and are calculated with the standard Lennard-Jones formula. In AMBER, the long-range interactions, are calculated using the following potential [3],

$$\begin{aligned}
U_{r>r_c} &= \frac{N\rho}{2}\frac{1}{N(N-1)}\sum_{i<j}^{n}\int_{r_c}^{\infty} 4\epsilon\left[\left(\frac{\sigma_{ij}}{r}\right)^{12} - \left(\frac{\sigma_{ij}}{r}\right)^{6}\right] g(r)4\pi r^2 dr \\
&= 2\pi N\rho\int_{r_c}^{\infty}\left(\langle 4\epsilon_{ij}\sigma_{ij}^{12}\rangle r^{-12} - \langle 4\epsilon_{ij}\sigma_{ij}^{6}\rangle r^{-6}\right) r^2 dr \\
&= 8\pi N\rho\left[\frac{1}{9}\langle\epsilon_{ij}\sigma_{ij}^{12}\rangle r_c^{-9} - \frac{1}{3}\langle\epsilon_{ij}\sigma_{ij}^{6}\rangle r_c^{-3}\right]. \qquad (1.6)
\end{aligned}$$

Where N is the total number of particles within the system, ρ is the average density of the system, ϵ is the well depth, σ_{ij} is the Lennard-Jones characteristic radius and $g(r)$ is the radial distribution function which is assumed to be equal to one outside the cutoff distance. Equation 1.6 will be dominated by the r^{-6} term, the attractive term. However, outside of the cutoff, this will be small so a direct cutoff is commonly applied in other codes.

Long-range van der Waals energies scale to r^{-6}, so are small in comparison to long-range electrostatic interactions. There are many methods currently available to handle long-range electrostatic interactions. These include the Ewald summation, particle-mesh ewald, reaction field and isotropic periodic sum. A direct cutoff approach to electrostatics will not converge.

One approach to long range electrostatic interactions is the direct sum approach. By using a sufficiently large cutoff so that the change in energy would be negligible by increasing the cutoff, each interaction can be calculated explicitly. In order to achieve this convergence, the cutoff must be very large, a method that is computationally very expensive. An alternative way is to make use of the the Ewald Summation [4].

1.1.1 Ewald Summation

By splitting the electrostatic interactions of a periodic system into two parts, a short-range and a long-range part, the computational time required can be significantly reduced. This is possible due to the following relationship,

$$\frac{1}{r} = \frac{f(r)}{r} - \frac{1 - f(r)}{r}. \tag{1.7}$$

When an appropriate choice for $f(r)$ is used, it will handle the rapid variation at short distances and the slow decay at large distances[5].

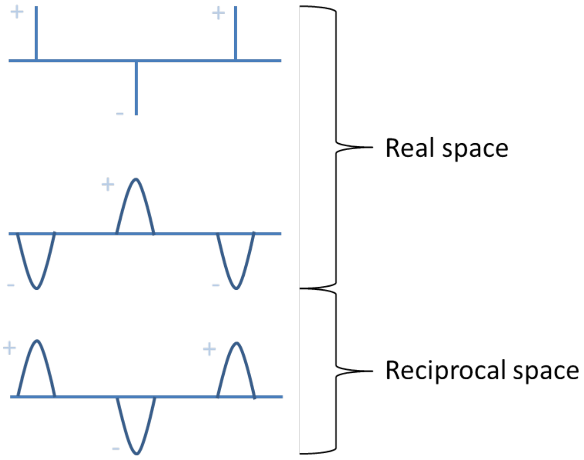


Figure 1.4: Gaussian distributions of equal and opposite charge added to the real space term, then corrected in the reciprocal space term by adding Gaussians of the correct charge.

The Ewald summation ensures that no long-range interactions are considered for the short-range part of the sum by masking the charges with Gaussian distributions of equal magnitude, but opposite charge. This is illustrated in figure 1.4. The short-range calculation (which includes the interaction between charges and the charge distributions used to neutralise them) is then performed by [6],

$$V_{r<r_c} = \frac{1}{8\pi\epsilon_0} \sum_{|\mathbf{n}|=0}^{\prime} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{q_i q_j \mathrm{erfc}(\alpha|\mathbf{r}_{ij} + \mathbf{n}|)}{|\mathbf{r}_{ij} + \mathbf{n}|}. \tag{1.8}$$

Where α controls the width of the Gaussians used to mask the charges and is directly related to the cutoff distance, commonly by $\alpha = 3.5/r_c$[7], and $\mathbf{n}$ is a lattice translation and the prime on the summation indicates that the self interaction, when $i = j$ is not included for the lattice box $|\mathbf{n}| = 0$. Careful selection of $alpha$ can lead to a calculation time of the order $N\log N$ instead of N^2. $\mathrm{erfc}(x)$ is the complementary error function,

$$\mathrm{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2} dt. \tag{1.9}$$

The long-range calculation must then cancel out the Gaussian distributions used in the short-range calculations, this calculation will not converge in real space[6], and as such is performed in reciprocal space using the following sum [5],

$$V_{r>r_c} = \frac{1}{8\pi\epsilon_0} \sum_{k\neq 0} \sum_{i=0}^{N} \sum_{j=0}^{N} \frac{1}{\pi L^3} \frac{q_i q_j 4\pi^2}{k^2} e^{-k^2/4\alpha^2} \cos(\mathbf{k}\cdot\mathbf{r}_{ij}). \tag{1.10}$$

Where $\mathbf{k}$ are vectors in reciprocal space and are defined as $\mathbf{k} = 2\pi\mathbf{n}/L$. This long-range sum corresponds to $f(r)$ and $erfc(x)$ corresponds to $1 - f(r)$ shown in equation 1.7. In order to cancel out the self interactions that are calculated as a by-product of the long-range sum, an additional term is subtracted from the equation.

$$V_{self} = -\frac{\alpha}{\sqrt{\pi}} \sum_{k=1}^{N} \frac{q_k^2}{4\pi\epsilon_0} \tag{1.11}$$

If the simulation box is surrounded by an infinite medium with a dielectric constant then no further corrections are needed, however, if the surrounding medium is a vacuum, then an additional correction is needed.

$$V_{correction} = \frac{2\pi}{3L^3}\left|\sum_{i=1}^{N}\frac{q_i}{4\pi\epsilon_0}\mathbf{r}_i\right|^2. \tag{1.12}$$

The total Ewald summation is then,

$$\begin{aligned} V_{electrostatic} &= \frac{1}{8\pi\epsilon_0}\sum_{|\mathbf{n}|=0}^{\prime}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{q_i q_j erfc(\alpha|\mathbf{r}_{ij}+\mathbf{n}|)}{|\mathbf{r}_{ij}+\mathbf{n}|} \\ &+\frac{1}{8\pi\epsilon_0}\sum_{k\neq 0}\sum_{i=0}^{N}\sum_{j=0}^{N}\frac{1}{\pi L^3}\frac{q_i q_j 4\pi^2}{k^2}e^{-k^2/4\alpha^2}\cos(\mathbf{k}\cdot\mathbf{r}_{ij}) \\ &-\frac{\alpha}{\sqrt{\pi}}\sum_{k=1}^{N}\frac{q_k^2}{4\pi\epsilon_0}+\frac{2\pi}{3L^3}\left|\sum_{i=1}^{N}\frac{q_i}{4\pi\epsilon_0}\mathbf{r}_i\right|^2. \end{aligned} \tag{1.13}$$

1.1.2 Particle Mesh Ewald

The use of the Ewald summation to calculate long-range interactions scales as N^2 with the number of charges N [8]. For this reason, as the system size becomes larger it becomes unsuitable to use. Approximate methods were therefore developed, one such method is particle mesh Ewald (PME) [9]. PME deals with the long range interactions by using fast Fourier transforms (FFTs). These require spatial points to be interpolated onto a grid or mesh of density values. The potential can then be solved by using the Poisson equation. This scales much more favourable at $N \cdot \log(N)$.

1.1.3 Solvation models

Simulating the effect solvent has on a system can be vital to obtaining accurate systematic information. It is therefore important to consider this when calculating energies and running simulations. Several methods exist to take account of the interactions between water and solute. The most common methods are either implicit or explicit, while there are also intermediate approaches such as the 3D-RISM model. The explicit and implicit methods are discussed briefly here.

1.1.3.1 Explicit solvation

There are three common descriptions used for water models: the 3-site, 4-site and 5-site. These are illustrated in figure 1.5.

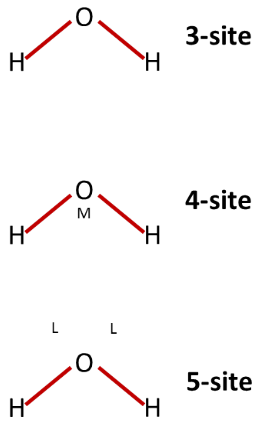


FIGURE 1.5: The common models used for water. 3-site, 4-site and 5-site differ by the different number of interaction sites.

As expected, as the number of interaction sites increases the number of calculations required are increased. The 3-site model is the most simple and has only three points of interaction, where the charges and Lennard-Jones spheres are centred on the atoms. The 4-site moves the negative charge present on the oxygen closer to the hydrogen atoms (position M), leaving the Lennard-Jones sphere centred on the oxygen. The 5-site moves the negative charge to represent the lone pairs.

Several examples are shown in table 1.1.3.1 along with the parameters used within equation 1.1. In order to lower the computational cost when explicit water is used, it is common to freeze the bonds and angle, such that the degrees of freedom are reduced to only translational and rotational.

1.1.3.2 Implicit Solvation

Implicit solvation attempts to calculate the effects water has on a solute while providing an average description of the effect of the solvent molecules.

Table 1.1: Water parameters used for common water models.

	SPC [10]	TIP3P [11]	TIP4P [11]	TIP5P [12]
r_{OH} (Å)	1.0000	0.9572	0.9572	0.9572
θ_{HOH}	109.47	104.52	104.52	104.52
A × 10^{-3} (kcal Å^{12}/mol)	629.4	582.0	600.0	544.5
C (kcalÅ^{6}/mol)	625.5	595.0	610.0	590.3
q_O (e)	-0.820	-0.834	-1.040	-0.241
q_H (e)	0.410	0.417	0.520	0.241
r_{OM} (Å)	-	-	0.15	-
r_{OL} (Å)	-	-	-	0.7
θ_{LOL}	-	-	-	109.47

Classical electrostatics uses Coulomb's law to describe electrostatics,

$$E(\mathbf{r}) = kq_1 \frac{\mathbf{r}-\mathbf{s}}{|\mathbf{r}-\mathbf{s}|^3}. \tag{1.14}$$

$E(\mathbf{r})$ is an electric field at position $\mathbf{r}$ due to a point charge q_1 at position $\mathbf{s}$ and k is a constant known as Coulomb's constant. If there is a continuous charge distribution a charge density can be used [13].

$$E(\mathbf{r}) = k \int \rho(\mathbf{s}) \frac{\mathbf{r}-\mathbf{s}}{|\mathbf{r}-\mathbf{s}|^3} d^3\mathbf{s}. \tag{1.15}$$

Where $\rho(\mathbf{s})$ is known as the free charge density (the bound charge density relates to polarisation and is not covered here). Calculating the divergence[1] of both sides of equation 1.15 with respect to $\mathbf{r}$ yields,

$$\begin{aligned} \nabla \cdot E(\mathbf{r}) &= \nabla \cdot k \int \rho(\mathbf{s}) \frac{\mathbf{r}-\mathbf{s}}{|\mathbf{r}-\mathbf{s}|^3} d^3\mathbf{s} & (1.16) \\ &= k \int \rho(\mathbf{s}) \nabla \cdot \frac{\mathbf{r}-\mathbf{s}}{|\mathbf{r}-\mathbf{s}|^3} d^3\mathbf{s}. & (1.17) \end{aligned}$$

The following function has special properties,

$$\nabla \cdot \frac{\mathbf{r}-\mathbf{s}}{|\mathbf{r}-\mathbf{s}|^3}. \tag{1.18}$$

[1] Divergence is a measure of how much the velocity increases in an outward motion, it is the volume density of the outward flux of a vector field

It is zero everywhere except the origin and when integrated the result is 4π if the origin is within the integrated space. This is very similar to a Dirac delta function $\delta(\mathbf{r}-\mathbf{s})$ [14]. The divergence of the electric field is then,

$$\nabla \cdot E(\mathbf{r}) = k4\pi \int \rho(\mathbf{s})\delta(\mathbf{r}-\mathbf{s})d^3\mathbf{s}. \tag{1.19}$$

Then using the sifting property of a dirac delta function and using $k = 1/4\pi\epsilon_0$,

$$\nabla \cdot E(\mathbf{r}) = \frac{1}{\epsilon_0}\rho(\mathbf{r}). \tag{1.20}$$

Where ϵ_0 is the permittivity of free space[2]. This relates the charge density to the electric field and is known as Gauss's law.

The electric field $E(\mathbf{r})$ can be written as the gradient of a potental $\phi(\mathbf{r})$,

$$\begin{aligned} E(\mathbf{r}) &= -\nabla\phi(\mathbf{r}) && (1.21)\\ \nabla^2\phi(\mathbf{r}) &= -\frac{\rho(\mathbf{r})}{\epsilon_0}. && (1.22) \end{aligned}$$

Equation 1.22 is the Poisson equation. If the charge density follows a Boltzmann distribution, the Poisson-Boltzmann equation must be used[15]. In order to move from the Poisson equation to the Poisson-Boltzmann equation, we assume that the local concentration $c_i(\mathbf{r})$ of species i with bulk concentration c_i^0 at position $\mathbf{r}$ can be calculated by the following Boltzmann distribution,

$$c_i(\mathbf{r}) = c_i^0 \exp\left(-\frac{z_i e\phi(\mathbf{r})}{k_B T}\right). \tag{1.23}$$

Where $z_i e$ is the charge of species i, which, for example is +1 for a Na^+ ion, +2 for Ca^{2+}, etc. The free charge density can be given by,

[2] This is a measure of how much resistance is present when an electric field is formed

$$\rho(\mathbf{r}) = F \sum_i c_i^0 z_i. \tag{1.24}$$

Where F is the Faraday constant. Putting this back into the Poisson equation, we obtain the Poisson-Boltzmann equation,

$$\nabla^2 \phi(\mathbf{r}) = -\frac{F}{\epsilon_0} \sum_i c_i^0 z_i \exp\left(-\frac{z_i e \phi(\mathbf{r})}{k_B T}\right). \tag{1.25}$$

In order to solve equation 1.25, numerical approaches are applied. One such method is finite difference. A grid is superimposed onto the system and each grid point is assigned a value for the charge density, electrostatic potential, ionic strength and dielectric constant [16]. The derivatives required for equation 1.25 are then calculated using finite difference. Each grid point has an effect on its surrounding points, thus this is an iterative process.

The non-polar interactions are then approximated, these include the cavitation energy and the Van der Waals energy. The cavity energy is the energy required to make a cavity in a water box, and the Van der Waals energy is the interaction between the protein and the surrounding water. These tend to be parametrised using experimental values [16].

1.1.4 Force fields

This section so far has dealt with functional forms, solvation and how long range forces are dealt with. The equations require parameters in order to calculate energies. These parameters are based on values obtained from *ab initio* methods and experimental data and collected together in force fields. Some example force fields that are available in AMBER[17] are listed below.

ff94[18]

Charges are based on multiple-conformation HF/6-31G* calculations, the forcefield was developed for use with solvated proteins. The exaggerated dipole moment present within HF/6-31G* was thought to simulate the polarisation present within an aqueous system.

ff96[19]

Empirical modifications of the backbone parameters within the ff94 force field were applied, in order to better match experiment. This change led to improvement in the relative energies between MM and QM.

ff99[20]

Small changes to the protein parameters of the ff96 force field. Parameters were introduced for nucleosides.

ff99SB[21]

New parameters for the backbone dihedral angles. This improved the over estimation of α-helices of the previous force fields, but the charges are still based on HF gas phase calculations.

ff14SB[22]

Slight empirical improvements were made to the side chain and backbone parameters present within the ff99SB. Certain residues were sampling structures that were not present in experiment within the ff99SB, so the side chain corrections aimed to provide a better match between theory and experiment.

GAFF[23]

Generalised amber force field, designed to be used with small organic molecules. For simplicity, the atom types are more general in order to cover most organic chemistry requirements.

1.2 Quantum Mechanics

Quantum Mechanics describes the behaviour of subatomic particles. Within chemistry, the most applicable of these are the electrons and nuclei, which make up the molecules. The classical description of atoms acting as hard spheres is a crude approximation and quantum theory needs to be used for a realistic description. In the quantum world, particles have the qualities typical of both matter and waves and are described by a wavefunction. This wave-particle duality was suggested by Louis de Broglie [24] when he proposed that the momentum of a particle is inversely proportional to the wavelength,

$$p = \frac{h}{\lambda}. \tag{1.26}$$

Where p is the momentum, λ is the wavelength and h is Plank's constant [25].

This is Heisenberg's uncertainty principle[26], the more accurately the position is known the less accurately the momentum can be known.

$$\Delta x \Delta p \geq \frac{h}{4\pi}. \tag{1.27}$$

Δx is the uncertainty on the position and Δp the uncertainty in the momentum. $\frac{h}{4\pi}$ is the limit that the position and momentum can be known.

1.2.1 The wavefunction Ψ

A position of an electron can never be known exactly, so probabilities are used instead. If a region of space has a high probability of finding an electron, then the electron density is high[27].

The wavefunction contains information on the quantum state of a system. It is linked to the electron density by squaring it, i.e. the square of the wavefunction gives the electron density. In order to obtain any physically relevant information from a wavefunction, the use of operators is required. By using the relevant operator on a wavefunction, any observable can be obtained. To obtain real-valued eigenvalues, the operators need to be Hermitian, such that[28],

$$\langle A|\hat{O}|B\rangle = \langle B|\hat{O}|A\rangle^{*}. \tag{1.28}$$

Where the superscript $*$ represents the complex conjugate. Additionally the probability density can be obtained from the wavefunction ψ by multiplication with its complex conjugate.

The time independent Schrödinger equation[29] is used to obtain the energy and the wavefunction using the Hamiltonian operator $\hat{\mathrm{H}}$, it is impossible to solve exactly except for the most simple cases.

$$\hat{H}\psi = E\psi. \tag{1.29}$$

Where E is the energy eigenvalue of the system, ψ is the eigenfunction and the Hamiltonian ($\hat{\mathrm{H}}$) can be split into the operator for the kinetic energy $\hat{\mathrm{T}}$ and the potential energy $\hat{\mathrm{V}}$.

$$\begin{align} \hat{H} &= \hat{V} + \hat{T} \tag{1.30} \\ &= \hat{V}_{NN}\{R\} + \hat{V}_{Ne}\{R;r\} + \hat{V}_{ee}\{r\} + \hat{T}_{N}\{R\} + \hat{T}_{e}\{r\} \tag{1.31} \end{align}$$

Where, for the case of a molecule, $\hat{V}_{NN}\{R\}$ is the nuclear-nuclear repulsion, $\hat{V}_{Ne}\{R;r\}$ is the nuclear-electron attraction, $\hat{V}_{ee}\{r\}$ is the electron-electron repulsion, $\hat{T}_{N}\{R\}$ is the kinetic energy of the nuclei and $\hat{T}_{e}\{r\}$ is the kinetic energy of the electrons. These can be further expanded into,

$$\begin{aligned}\hat{H} &= \frac{1}{2}\sum_{k=1}^{N}\sum_{l\neq k}^{N}\frac{Z_k Z_l e^2}{4\pi\epsilon_0 R_{kl}} - \sum_{i=1}^{n}\sum_{k=1}^{N}\frac{Z_k e^2}{4\pi\epsilon_0 R_{ik}} \\ &+\frac{1}{2}\sum_{i=1}^{n}\sum_{j\neq i}^{n}\frac{e^2}{4\pi\epsilon_0 r_{ij}} - \frac{\hbar^2}{2}\sum\frac{\nabla^2_{R_k}}{m_n} - \frac{\hbar^2}{2}\sum\frac{\nabla^2_{r_i}}{m_e}. \end{aligned} \tag{1.32}$$

Where Z_k is the atomic charge on atom k, R_{kl} is the distance between atom k and l, N is the number of atoms, n is the number of electrons, subscript i and j are electrons and e is the elementary charge. ∇^2 is the Laplacian operator and in a three dimensional cartesian frame takes the form,

$$\nabla^2 = \frac{\partial^2}{\partial x_i^2} + \frac{\partial^2}{\partial y_i^2} + \frac{\partial^2}{\partial z_i^2}. \tag{1.33}$$

1.2.2 The Born-Oppenheimer Approximation

Equation 1.31 can be simplified by using the Born-Oppenheimer approximation[30]. The mass of protons and neutrons are around 1800 times that of the mass of electrons[2] and it can therefore be approximated that electrons move much faster than the protons and neutrons. If taking this approximation into account the electrons can be thought to rearrange themselves instantaneously to any nuclear movement. Equation 1.31 can then be simplified to,

$$\hat{H}_{elec} = \hat{V}_{Ne}\{R;r\} + \hat{V}_{ee}\{r\} + \hat{T}_e\{r\}. \tag{1.34}$$

Where the nuclear kinetic energy can now be ignored, the nuclear-nuclear repulsion is constant and the nuclear-electron attraction can be thought of as an external potential acting on the electrons. $\hat{H}_{elec}$ is the electronic Hamiltonian, which makes up the electronic Schrödinger equation,

$$\hat{H}_{elec}\psi_{elec} = E_{elec}\psi_{elec}. \tag{1.35}$$

1.2.3 Potential Energy Surfaces

Without the Born-Oppenheimer approximation, the notion of a potential energy surface could not exist. Potential energy surfaces are utilised a great deal in the field of computational chemistry, as they provide a relationship between the energy and geometry of a system. By changing the nuclear coordinates and calculating the electronic energy, these surfaces can show where any minima (equilibrium geometries) or transition states are. Figure 1.6 shows an example potential energy surface for the C-O bond and the C-O-H angle with a fixed dihedral angle in methanol.

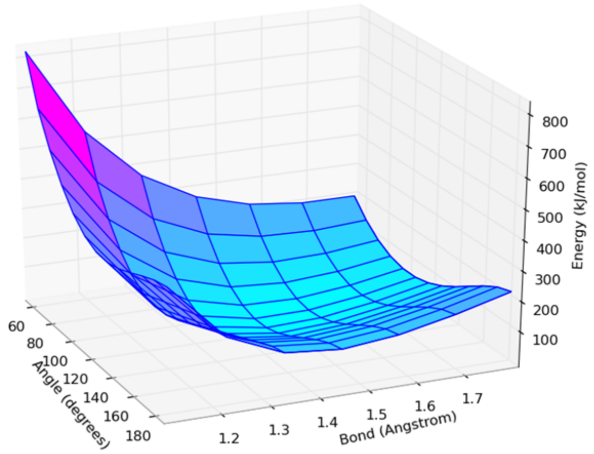


FIGURE 1.6: Potential Energy surface for 2 degrees of freedom in methanol

In order to construct a potential energy surface, after equation 1.35 has been solved an additional term must be included to account for the nuclear-nuclear repulsion,

$$E_{PES} = E_{elec} + \sum_{k=1}^{N} \sum_{l \neq 1}^{N} \frac{Z_k Z_l}{|\mathbf{R}_k - \mathbf{R}_l|}. \tag{1.36}$$

1.2.4 The Variational Principle

The variational principle is used to provide an approximation to the ground state energy. Wavefunctions can be combined into a single trial function by a linear combination of wave functions with weighting coefficients.

$$\Phi = \sum_{\alpha} c_\alpha \Phi_\alpha \tag{1.37}$$

Where orthonormal wavefunctions are orthogonal such that $\int \psi_i \psi_j d\mathbf{r} = 0$ if $i \neq j$ and normalised, $\int \psi_i^2 d\mathbf{r} = 1$.

This trial function must be subject to the same boundary conditions as the wave functions and $\sum_\alpha |c_\alpha|^2 = 1$. The variational principle states that the energy obtained from a trial function will be greater than or equal to the true ground state energy[31].

$$\langle \Phi | H | \Phi \rangle \geq E_0 \tag{1.38}$$

An approximation to the ground state energy is found by minimising the energy with respect to the weighting coefficients. Proof of the theorem can be seen below[32].

$$\begin{aligned}
\langle \Phi | \Phi \rangle &= 1 && (1.39)\\
\sum_\alpha |\Phi_\alpha\rangle\langle\Phi_\alpha| &= 1 && (1.40)\\
\langle \Phi_\alpha | \Phi_\beta \rangle &= \delta_{\alpha\beta} && (1.41)\\
\langle \Phi | \Phi \rangle &= \sum_{\alpha\beta} \langle \Phi | \Phi_\alpha \rangle \langle \Phi_\alpha | \Phi_\beta \rangle \langle \Phi_\beta | \Phi \rangle \\
&= \sum_{\alpha\beta} \langle \Phi | \Phi_\alpha \rangle \delta_{\alpha\beta} \langle \Phi_\beta | \Phi \rangle \\
&= \sum_{\alpha} \langle \Phi | \Phi_\alpha \rangle \langle \Phi_\alpha | \Phi \rangle \\
&= \sum_{\alpha} |\langle \Phi_\alpha | \Phi \rangle|^2 && (1.42)\\
\langle \Phi | H | \Phi \rangle &= \sum_{\alpha\beta} \langle \Phi | \Phi_\alpha \rangle \langle \Phi_\alpha | H | \Phi_\beta \rangle \langle \Phi_\beta | \Phi \rangle \\
&= \sum_{\alpha} E_\alpha |\langle \Phi_\alpha | \Phi \rangle|^2. && (1.43)
\end{aligned}$$

Since $E_\alpha \geq E_0$,

$$\begin{aligned}\langle\Phi|H|\Phi\rangle &\geq \sum_{\alpha} E_0|\langle\Phi_\alpha|\Phi\rangle|^2 \\ &\geq E_0\sum_{\alpha}|\langle\Phi_\alpha|\Phi\rangle|^2 \\ &\geq E_0. \end{aligned} \tag{1.44}$$

Where equations 1.39, 1.40 and 1.41 are true due to the orthonormality of wave functions and in equation 1.43 equation 1.42 has been applied. Therefore, we have shown than an approximate wavefunction Φ will always have energy higher or equal to the ground state energy E_0.

1.2.5 Hartree-Fock Theory

The Hartree-Fock approximation (HF) provides a method to solve the electronic Schrödinger equation (equation 1.35). It deals with the many electron problem as many one-electron problems with an average electron-electron repulsion acting on each electron. It is because of this that it is sometimes referred to as "mean-field theory". Each electron occupies a spin orbital $\chi(\mathbf{x})$, which is a molecular orbital $\psi(\mathbf{r})$ with spin coordinates (either $\alpha(\omega)$ or $\beta(\omega)$)[33]. The Pauli exclusion principle states that the wavefunction must change sign if two electrons are swapped within the system[34], such that $\Psi(\mathbf{x}_1, \mathbf{x}_2) = -\Psi(\mathbf{x}_2, \mathbf{x}_1)$. This anti-symmetric behaviour is upheld within HF by the use of Slater determinant wavefunctions. For example, a two electron system would have the following determinant,

$$\Psi(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\frac{1}{2!}}\begin{bmatrix}\chi_1(\mathbf{x}_1) & \chi_2(\mathbf{x}_1) \\ \chi_1(\mathbf{x}_2) & \chi_2(\mathbf{x}_2)\end{bmatrix} \tag{1.45}$$

$$\Psi(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\frac{1}{2!}}(\chi_1(\mathbf{x}_1)\chi_2(\mathbf{x}_2) - \chi_2(\mathbf{x}_1)\chi_1(\mathbf{x}_2)). \tag{1.46}$$

If the electrons change place, the determinant then becomes,

$$\begin{aligned} \Psi(\mathbf{x}_2, \mathbf{x}_1) &= \sqrt{\frac{1}{2!}}(\chi_1(\mathbf{x}_2)\chi_2(\mathbf{x}_1) - \chi_2(\mathbf{x}_2)\chi_1(\mathbf{x}_1)) && (1.47) \\ \therefore \Psi(\mathbf{x}_1, \mathbf{x}_2) &= -\Psi(\mathbf{r}_2, \mathbf{r}_1). && (1.48) \end{aligned}$$

The single electron problem with average electron-electron repulsion is expressed within the Fock operator,

$$\begin{aligned} \hat{f}_i &= -\frac{1}{2}\hat{\nabla}_i^2 - \sum_{k=1}^{N_{atoms}} \frac{Z_k}{\mathbf{R}_{ik}} + \hat{v}_i && (1.49) \\ &= \hat{h}_i + \hat{v}_i. && (1.50) \end{aligned}$$

Where $\hat{v}_i$ is the average potential felt by electron i due to all other electrons. This energy is composed of Coulomb and exchange integrals, and is shown in equation 1.51 and 1.52 respectively.

$$\begin{aligned} \hat{J}_{ij} &= \int\int \chi_i^*(\mathbf{x}_1)\chi_j^*(\mathbf{x}_2)\frac{1}{r_{ij}}\chi_i(\mathbf{x}_1)\chi_j(\mathbf{x}_2)d\mathbf{x}_1 d\mathbf{x}_2 && (1.51) \\ \hat{K}_{ij} &= \int\int \chi_i^*(\mathbf{x}_1)\chi_j^*(\mathbf{x}_2)\frac{1}{r_{ij}}\chi_j(\mathbf{x}_1)\chi_i(\mathbf{x}_2)d(\mathbf{x}_1)d(\mathbf{x}_2) && (1.52) \end{aligned}$$

The exchange operator is present because of the asymmetric nature of the wavefunction, which is another consequence of the Pauli exclusion principle, stating that no two electrons can have the same quantum number.

The Fock operator can then be written as,

$$\hat{f}_i = \hat{h}_i + \sum_{j}^{n/2}(2\hat{J}_j(\mathbf{r}_i) - \hat{K}_j(\mathbf{r}_i)). \qquad (1.53)$$

The energy of the system is obtained self-consistently as the Fock operator depends on the molecular orbitals that we are trying to calculate. This process is shown in figure 1.7.

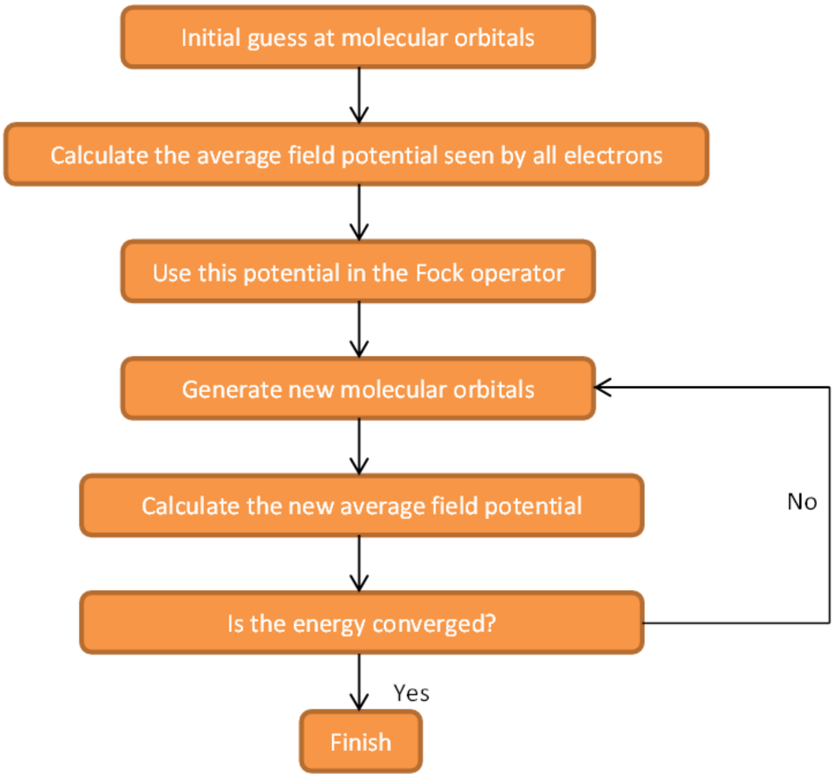

Initial guess at molecular orbitals
Calculate the average field potential seen by all electrons
Use this potential in the Fock operator
Generate new molecular orbitals
Calculate the new average field potential
No
Is the energy converged?
Yes
Finish

FIGURE 1.7: The SCF process used within the HF approximation

As described in section 1.2.4 (The Variational Principle), the energy is minimised to the lowest possible value E_{HF}. The difference between E_{HF} and the ground state energy, E_0 is known as the correlation energy.

1.2.6 Density Functional Theory

In contrast to wavefunction approaches, like HF, density functional theory (DFT) uses the electronic density to calculate the energy. By using the density instead of the wavefunction, the cost of any calculations is limited: for an n electron system, the wavefuntion is a $3n$-dimensional quantity, whereas the density remains 3-dimensional. This lower cost makes DFT very computationally appealing and explains its rise in popularity. As previously mentioned, to get properties from the wavefunction operators are used, to get properties from the density, however, functionals must be used[3]. Hohenberg and Kohn[35] state that all ground state electronic properties can be obtained from the density of the system in this manner.

1.2.6.1 Hohenberg and Kohn's first theorem

The first theorem states that the external potential v_{ext} is determined by the ground state density [35, 36]. This statement can be proved by *reductio ad absurdum.*

Assume that there are two external potentials that provide the same density $n(\mathbf{r})$, but have different ground states Ψ_1 and Ψ_2. So for each ground state, we have Hamiltonian $\hat{H}_1$ and $\hat{H}_2$ and ground state energy E_1 and E_2 with external potentials $v_{ext,1}$ and $v_{ext,2}$. Then including the variational principle, the following is true,

$$\begin{aligned}
E_2 &= \langle\Psi_2|\hat{H}_2|\Psi_2\rangle < \langle\Psi_1|\hat{H}_2|\Psi_1\rangle && (1.54)\\
&= \langle\Psi_1|\hat{H}_1|\Psi_1\rangle + \langle\Psi_1|\hat{H}_2 - \hat{H}_1|\Psi_1\rangle && (1.55)\\
E_2 &< E_1 + \int [v_{ext,2}(\mathbf{r}) - v_{ext,1}(\mathbf{r})]n(\mathbf{r})d\mathbf{r}. && (1.56)
\end{aligned}$$

Similarly for E_1,

[3] A functional operates on a function to produce a number

$$\begin{aligned} E_1 &= \langle\Psi_1|\hat{H}_1|\Psi_1\rangle < \langle\Psi_2|\hat{H}_1|\Psi_2\rangle && (1.57) \\ &= \langle\Psi_2|\hat{H}_2|\Psi_2\rangle + \langle\Psi_2|\hat{H}_1 - \hat{H}_2|\Psi_2\rangle && (1.58) \\ E_1 &< E_2 + \int [v_{ext,1}(\mathbf{r}) - v_{ext,2}(\mathbf{r})]n(\mathbf{r})d\mathbf{r}. && (1.59) \end{aligned}$$

Combining equation 1.56 and 1.59 gives,

$$E_2 + E_1 < E_1 + E_2. \quad (1.60)$$

Which cannot be true and therefore proves that only one external potential exists for the density.

1.2.6.2 Hohenberg and Kohn's second theorem

The kinetic and electron-electron interaction operators are functionals of $n\mathbf{r}$. Therefore, a universal functional ($F[n(\mathbf{r})]$) can be defined that is not dependent on the number of particles and is valid for any external potential,

$$\begin{aligned} F[n(\mathbf{r})] &= \langle\Psi|\hat{T} + \hat{V}_{ee}|\Psi\rangle && (1.61) \\ E[n(\mathbf{r})] &= \int v_{ext}(\mathbf{r})n(\mathbf{r})d\mathbf{r} + F[n(\mathbf{r})]. && (1.62) \end{aligned}$$

Examination of equation 1.62 shows that for the correct density, the ground state energy can be found, which is the focus of the second theorem. To prove this, consider the following energy,

$$E[\Psi'] = \langle\Psi'|\hat{V}_{ext}|\Psi'\rangle + \langle\Psi'|\hat{T} + \hat{V}_{ee}|\Psi'\rangle, \quad (1.63)$$

where Ψ' is an approximation to the ground state wavefunction, thus has a minimum at the ground state energy. Therefore,

$$\begin{aligned} E[\Psi'] &= \int v_{ext}(\mathbf{r})n'(\mathbf{r})d\mathbf{r} + F[n'(\mathbf{r})] \quad (1.64) \\ E[n] &= \int v_{ext}(\mathbf{r})n(\mathbf{r})d\mathbf{r} + F[n(\mathbf{r})] \quad (1.65) \\ E[n] &< E[\Psi']. \quad (1.66) \end{aligned}$$

If $F[n(\mathbf{r})]$ were known exactly, then the calculation of the ground state energy would be trivial, however, determination of the universal functional is highly non-trivial.

1.2.6.3 Kohn-Sham Theory

The current popularity of DFT is largely down to the Kohn-Sham[37] reformulation. The issue of finding a universal function is dealt with by representing the ground state density with the density of a system of non-interacting electrons $n_s(\mathbf{r})$. The energy is then found using the following,

$$E[n_s] = T_s[n_s] + U[n_s] + V_{ext}[n_s] + E_{xc}[n_s]. \quad (1.67)$$

Where $T_s[n_s]$ is the kinetic energy for non-interacting electrons, $U[n_s]$ is the Coulomb energy given by the Fock equation, $V_{ext}[n_s]$ is the external potential, $E_{xc}[n_s]$ is known as the exchange correlation functional and can be split into the kinetic correlation energy, $T_c[n_s]$, and the exchange and correlation energy $V_{ee}[n_s]$.

$E_{xc}[n_s]$ is used to correct for the fact that a non-interacting system has been used,

$$\begin{aligned} E_{xc}[n_s] &= T_c[n_s] + V_{ee}[n_s] \quad (1.68) \\ T_c[n_s] &= T[n_0] - T_s[n_s] \quad (1.69) \\ V_{ee}[n_s] &= V_{ee}[n_0] - U[n_s]. \quad (1.70) \end{aligned}$$

The exchange energy is a completely non-classical term caused by the exchange of two same spin electrons. Equation 1.67 can be written in terms of Kohn-Sham orbitals,

$$\begin{aligned} E[n_s] &= \sum_{i=1}^{n}\int \chi_i^*(\mathbf{r})\left[-\frac{1}{2}\nabla^2\right]\chi_i(\mathbf{r})d\mathbf{r} \\ &- \sum_{i=1}^{n}\sum_{k=1}^{N}\int \chi_i^*(\mathbf{r})\left[\frac{Z_k}{|\mathbf{r}_i-\mathbf{R}_k|}\right]\chi_i(\mathbf{r})d\mathbf{r} \\ &+ \sum_{i=1}^{n}\int \chi_i^*(\mathbf{r})\left[\frac{1}{2}\int\frac{n_s(\mathbf{r})}{|\mathbf{r}_i-\mathbf{r}'|}d(\mathbf{r}')\right]\chi_i(\mathbf{r})d\mathbf{r} + E_{xc}[n_s]. \end{aligned} \tag{1.71}$$

No exact expression exists for the exchange-correlation functional, if there was then Kohn-Sham theory would be exact.

1.2.6.4 Exchange-Correlation Functional

The importance of the exchange-correlation term within Kohn-Sham DFT is highlighted above. It is common practice to separate this functional into the exchange term and the correlation term,

$$E_{xc}[n_s] = E_x[n_s] + E_c[n_s]. \tag{1.72}$$

These components can then be treated as individual entities. Some approximations used to calculate the exchange-correlation energy are described in the next two sections.

Local Density Approximation

The simplest method to calculate the exchange-correlation energy is the local density approximation (LDA). The exchange energy is calculated within LDA by using the local density of a region and using in this region the exchange energy that the uniform electron gas with the same density has. The exchange energy is then calculated for this fictitious system using the following,

$$E_x^{LDA}[n_s] = -\frac{3}{4}\left(\frac{3}{\pi}\right)^{\frac{1}{3}} \int n_s(\mathbf{r})^{\frac{4}{3}} d\mathbf{r}. \tag{1.73}$$

No equivalent term exists for the correlation energy[2]. However, the correlation energy has been calculated using quantum Monte Carlo techniques and, by using a suitable analytical interpolation formula, the energy can be fitted[38]. Such methods are used within the VWN (Vosko, Wilk and Nusair)[39] functional.

LDA works well for systems with uniform density. However, molecules show a cusp in density around nuclei, and because of this, LDA is not a suitable method for use with molecules.

Generalised Gradient Approximation

The generalised gradient approximation (GGA) uses the value from LDA and adds a correctional term to take into account the gradient of the density. The exchange-correlation energy then takes the following form,

$$E_x^{GGA}[n_s(\mathbf{r})] = E_x^{LDA}[n_s(\mathbf{r})] + \Delta E_x \left[\frac{|\nabla n_s(\mathbf{r})|}{n_s(\mathbf{r})^{\frac{4}{3}}}\right]. \tag{1.74}$$

The first exchange functional was developed by Becke (B88)[40],

$$E_x^{GGA}[n_s(\mathbf{r})] = E_x^{LDA}[n_s(\mathbf{r})] + E_x^{B88}[n_s(\mathbf{r})]. \tag{1.75}$$

Many functionals were developed based on the B88, where no empirical data is used and everything is calculated from first principles. One example of this, which is used throughout this thesis is the PBE functional[41].

The correlation energy is also improved within GGA. Some functionals correct the LDA correlation energy, in a similar manner to equation 1.75, like the PW91 functional[42]. The LYP (Lee, Yang and Parr) functional[43] calculates the correlation energy by empirical fitting based on simulations performed on the helium atom.

In practice GGA functionals fit within two distinct categories,

First Those that are fitted based on experimental data.

Second Those that are constructed based on physical identities and limits.

Although not used within this thesis, other types of functionals are available. These include hybrid functionals and meta GGAs. Hybrid functionals include the exchange energy as calculated by HF, an example of this is the B3LYP functional [44] and meta GGAs include a higher order derivative of the density.

1.2.6.5 Basis Sets

The wavefunction has an unknown functional form, so in practice it can be approximated by many known functions, a basis set. This is not an approximation: in the event of an infinite basis set, any other functions can be made. When considering computational effort, however, an infinite basis set is infeasible. So, as is so common within computational chemistry, a compromise must be made between the computational time and accuracy for any calculation. As such, the development of more accurate basis sets has three goals [2],

- minimise the number of basis functions,
- be chosen such that molecular integrals can be completed in an efficient manner,
- be large in the same regions that the probability density is large.

Slater-Type orbitals

In 1930 Slater introduced Slater-type orbitals [45]. These functions exponentially decay at long range and show a cusp at the nucleus which is exact for the hydrogen atom. However, no radial nodes exist within an STO, so these are made by linear combination of many STOs [38]. The functional form can be seen below,

$$\chi_{\zeta,n,l,m}(r,\theta,\varphi) = NY_{l,m}(\theta,\varphi)r^{n-1}e^{-\zeta r}. \tag{1.76}$$

Where N is a normalisation constant, $Y_{l,m}$ are spherical harmonic functions, n is the principle quantum number, r is the distance of the electron from the nucleus and ζ is a constant relating to the effective charge of the nucleus. Figure 1.8 shows an example Slater-Type orbital.

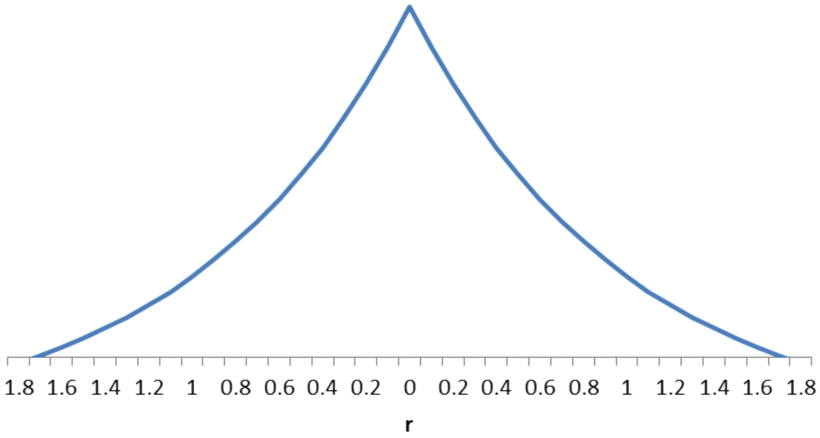


FIGURE 1.8: Slater-Type orbital

The disadvantage with STOs is that there is no analytical solution to the two electron four-centre integrals, meaning these have to be computed numerically. This is computationally costly, thus the use of STOs is limited to small systems.

Gaussian-Type orbitals

In 1950 Boys introduced a new type of orbital that has an analytical solution to the multi-centred integrals [46]. This new orbital changed the e^{-r} term within the STO functional form to e^{-r^2}, which provides a Gaussian distribution. The functional form then becomes simply,

$$\chi_{\zeta,n,l,m}(r,\theta,\varphi) = NY_{l,m}(\theta,\varphi)r^{n-1}e^{-\zeta r^2}. \tag{1.77}$$

However, an individual (or primitive) Gaussian provides a poor representation of the orbital, so multiple Gaussians are combined by linear combination (contracted Gaussians). Using this process, more complex shapes can be formed and orbitals are better represented. The orbitals are then represented by,

$$\chi_{\zeta,n,l,m}(r,\theta,\varphi) = \sum_{\alpha=1}^{M} C_\alpha N Y_{l,m}(\theta,\varphi) r^{n-1} e^{-\zeta r^2}. \tag{1.78}$$

The number of primitive Gaussians used within contracted Gaussians can be controlled to balance the computational accuracy with efficiency. Commonly, a split valence is used, this occurs when a different number of primitive Gaussians have been used to describe the core and valence electrons. For example the 3-21G basis set uses 3 primitives within a contracted Gaussian to describe core electrons and two contracted Gaussians, one with 2 primitive Gaussians and one with 1 primitive, to describe the valence electrons.

Basis Set Superposition Error

Using GTO or STO to calculate the interaction energy of a system ($\Delta E = E_{COMPLEX} - E_{HOST} - E_{LIGAND}$) or any atom centred basis set, will result in basis set superposition error (BSSE). The error arises from an overlap of the basis set in the complex, i.e. the basis function centred on the ligand overlaps with the host. This effect is not duplicated when calculating either the host or ligand in the absence of the other, which will overestimate interaction energies. One way of solving this issue is the introduction of ghost atoms within the host and ligand calculations, so the basis functions are still available to overlap. This is called the counterpoise correction [47].

Plane Wave Basis Sets

In contrast to the previously mentioned basis sets, the plane wave basis is uniform in space and not on atoms. Because of this it does not not suffer from BSSE, however, the box size will have an effect on the speed of the calculation. Plane waves are used when periodic boundaries must be present within the simulation and are the solution to the periodic Schrödinger equation. The form of a plane wave for a cubic box with side length l is,

$$\begin{aligned} \psi_{\mathbf{k}}(\mathbf{r}) &= \frac{1}{l^{\frac{3}{2}}} e^{i(k_x x + k_y y + k_z z)} && (1.79) \\ &= \frac{1}{V^{\frac{1}{2}}} e^{i\mathbf{k}\cdot\mathbf{r}}. && (1.80) \end{aligned}$$

Where $k_x = \frac{2\pi}{l} n_x$ and $n_x, n_y, n_z \in \mathbb{Z}$.[4] Plane waves are distributed evenly over a grid within the box and because of this cannot accurately represent the electrons tightly bound around the nucleus, so are used primarily to describe the valence density. To overcome this "effective core potentials" (ECP) or pseudopotentials are commonly used to describe the behaviour of the core electrons.

Pseudopotentials

In 1934 Hans G. A. Hellmann introduced the pseudopotential [48]. The use of a pseudopotential eliminates the need for core electrons to be described by a basis set, and instead uses an effective potential. Using this potential makes the approximation that the core electrons can be considered, along with the nucleus, as rigid and non-polarisable. Without the use of pseudopotentials, basis sets such as the plane wave basis sets would be far too computationally expensive to be useful. This is because of the rapid oscillations of the wavefunction at small distances from the nucleus.

When deriving suitable pseudopotentials, it is required that the energy and magnitude of the potential be the same after a cut off r_c. For soft pseudopotentials, this cut off is large, which leads to faster convergence, but makes the pseudopotential less transferable.

1.2.6.6 ONETEP

ONETEP (Order-N Electronic Total Energy Program) can perform DFT calculations on thousands of atoms by making use of linear scaling DFT methods. As previously mentioned, the computational effort of DFT methods scale as $\mathcal{O}(N^3)$ where N is the number of electrons present within a system. This restriction means that traditional DFT can only be performed on relatively small systems. Linear scaling DFT, however, makes use of the exponential decay of the density matrix for systems with a bandgap[50]. In practice this is done by applying a cutoff within the single particle density matrix $\rho(\mathbf{r}, \mathbf{r}')$, where the single particle density matrix is formed by,

$$\rho(\mathbf{r}, \mathbf{r}') = \sum_n |\psi_n(\mathbf{r})\rangle f_n \langle \psi_n(\mathbf{r}')| \tag{1.81}$$

[4] Where $\mathbb{Z}$ is a set of integers and $\in$ means "is an element of"

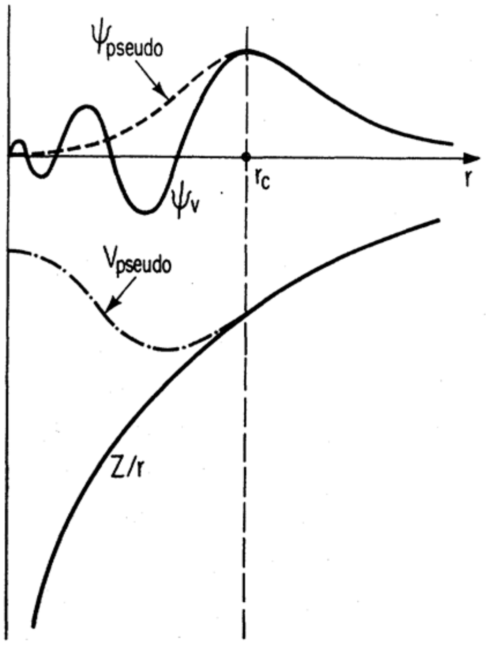


FIGURE 1.9: Pseudopotential (dashed line) with an all electron potential (solid line), taken from reference [49]

Where f_n is the occupation number. The diagonal of the density matrix provides the density, and within traditional DFT methods this is evaluated by direct diagonalisation of the Hamiltonian. For large systems this is not feasible, so a cutoff is applied to the density matrix based on distance $|\mathbf{r}-\mathbf{r}'|$. A decrease in this cutoff will increase the speed of the calculation, but lower the accuracy.

Within ONETEP, the density matrix is written as,

$$\rho(\mathbf{r},\mathbf{r}') = \sum_{\alpha\beta} \phi_\alpha(\mathbf{r}) K^{\alpha\beta} \phi^*_\beta(\mathbf{r}'). \tag{1.82}$$

Where $\phi_\alpha(\mathbf{r})$ are a set of spatially localised non-orthogonal generalised Wannier functions[51] (NGWFs), an example of which is shown in figure 1.10. $K^{\alpha\beta}$ is the density Kernel[52], which is the representation of the density matrix in a set of duals of the NGWFs, and a generalisation of the occupation numbers f_n to nonorthogonal functions.

Linear scaling is achieved within ONETEP by applying a cutoff, as described above, to the density kernel and by strict localisation of the NGWFs to atomic centres. ONETEP

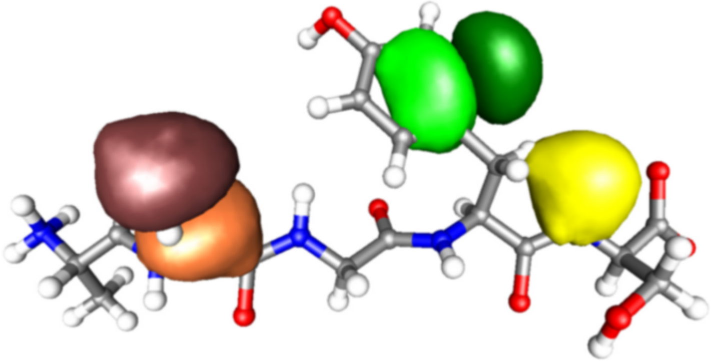


Figure 1.10: Three NGWFs on an oligopeptide, taken from [53]

operates by employing two nested loops, an outer loop which controls the optimisation of the NGWFs and an inner loop which controls the optimisation of the density kernel. The loops continue in a self consistent manner until convergence has been achieved. Once the NGWFs have been optimised, they contain the same information as the Kohn-Sham orbitals of the system and because they are optimised *in situ* they alter to their chemical environment. Similarities are present between the combination of the NGWFs and the combination of delocalised orbitals within traditional cubic scaling DFT up to the boundary of the NGWF. Therefore by altering the radii of the NGWF, the same accuracy as cubic scaling DFT can be achieved. This comparable accuracy is shown in reference [54].

The optimisation procedure of the NGWFs is performed by expansion in a psinc (periodic cardinal sine) basis[55]. Psinc functions are related to plane waves by Fourier transform and are highly localised and orthogonal by nature. Each psinc function is centered on a grid point and therefore the basis set can be controlled by changing the spacing between each point. An example of a psinc function is shown in figure 1.11.

In a similar manner to a plane wave basis, because a psinc basis is not atomic centred, ONETEP does not suffer from BSSE.

The attractive dispersion forces present within systems are not explicitly calculated by DFT. As such an empirical correction is applied within ONETEP[57] in a similar fashion to the DFT+D approach described by Grimme et al.[58]. These corrections

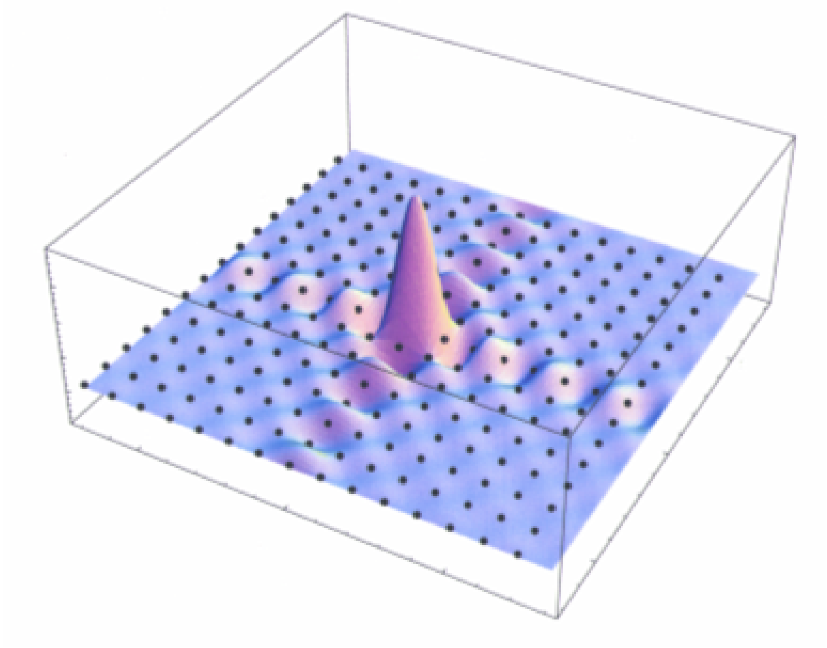


Figure 1.11: A psinc function taken from reference [56]

include the Elstner[59] and Grimme D2[58] approaches and are parameterised for specific functionals.

1.3 Molecular Dynamics

The previous sections explained how the energy of a stationary molecule can be calculated. However, the most interesting properties of molecules comes from an average of many structures. As such, methods exist to accurately generate an ensemble of structures. One of these methods, molecular dynamics (MD) [60], achieves this by calculating the motion of a molecule when forces are applied to it [34]. If the ergodic hypothesis is satisfied, i.e. the average properties obtained from the system over time are the same as the average properties over the whole statistical ensemble, then accurate macroscopic thermodynamic properties can be calculated [61].

Newton's second law (equation 1.83) describes the relationship between the force $\mathbf{F}_i$ and acceleration $\mathbf{a}_i$ of a particle with mass m_i. By integrating this equation and with knowledge of the forces applied, the positions of the atoms can be found.

$$\begin{aligned} \mathbf{F}_i &= m_i \mathbf{a}_i & (1.83) \\ &= m_i \frac{d\mathbf{v}_i}{dt} & (1.84) \\ &= m_i \frac{d^2\mathbf{r}_i}{dt^2}. & (1.85) \end{aligned}$$

Equations 1.84 and 1.85 use the relationship between the acceleration and velocity v_i and the position $\mathbf{r}_i$ respectively. The force can also be calculated from the gradient of the potential energy,

$$\mathbf{F}_i = -\nabla_i U. \tag{1.86}$$

Combination of equation 1.85 and 1.86 yields a relationship between the potential energy and position of the atoms at time t,

$$-\frac{dU}{d\mathbf{r}_i} = m_i \frac{d^2\mathbf{r}_i}{dt^2}. \tag{1.87}$$

The simplest application is to assume that the acceleration is a constant. Using this assumption the velocities and positions can be found simply by integrating the following equations with respect to time,

$$\begin{aligned} \mathbf{a}_i &= \frac{d\mathbf{v}_i}{dt} && (1.88) \\ \mathbf{v}_i &= \frac{d\mathbf{r}_i}{dt}. && (1.89) \end{aligned}$$

Which gives respectively,

$$\begin{aligned} \mathbf{v}_i &= \mathbf{a}_i t + \mathbf{v_0}_i && (1.90) \\ \mathbf{r}_i &= \mathbf{v}_i t + \mathbf{r_0}_i. && (1.91) \end{aligned}$$

The combination of equation 1.90 and 1.91 gives,

$$\mathbf{r}_i = \mathbf{a}_i t^2 + \mathbf{v_0}_i t + \mathbf{r_0}_i. \qquad (1.92)$$

The initial velocities are often chosen from a Gaussian distribution, with the following functional form,

$$E_{KIN} = \prod_{i=1}^{n} \frac{\sqrt{m_i}}{\sqrt{2\pi k_B T}} \exp\left[-\beta \frac{\mathbf{p}_i^2}{2m_i}\right]. \qquad (1.93)$$

This is identical to a selection using a Maxwell-Boltzmann distribution or a Boltzmann distribution of kinetic energies. In order to clarify this, we must start at the Boltzmann distribution of the kinetic energy,

$$\rho(E_{KIN}) = \exp[-\beta E_{KIN}]. \qquad (1.94)$$

Where $\rho(E_{KIN})$ is the probability of kinetic energy E_{KIN}. From here we can derive both the Maxwell-Boltzmann distribution. By splitting the kinetic energy into the x, y and z components we obtain,

$$E_{KIN} = \frac{1}{2m}p_x^2 + \frac{1}{2m}p_y^2 + \frac{1}{2m}p_z^2. \tag{1.95}$$

This is then put back into equation 1.93. The Maxwell-Boltzmann distribution then provides us with the probability a molecule will have velocity between v_x and $v_x + dv_x$, v_y and $v_y + dv_y$, v_z and $v_z + dv_z$, summing these together will give the probability that a molecule will have velocity between v and $v + dv$. This forms a spherical shell of thickness dv, the volume of this shell is $4\pi\mathbf{v}^2$. If this is placed back into the Gaussian distribution the Maxwell-Boltzmann distribution is achieved.

$$E_{KIN} = \prod_{i=1}^{n} 4\pi \left(\frac{m}{2\pi k_B T}\right)^{3/2} \mathbf{v}^2 \exp\left[-\frac{\beta m \mathbf{v}^2}{2}\right]. \tag{1.96}$$

1.3.1 Integration Algorithms

Direct application of equation 1.92 is numerically unstable for all but the most simplistic systems. As such integration algorithms exist with the following aims:

- Energy and momentum must be conserved
- Computational efficiency
- Long time steps.

Three integrators that are commonly used are the Verlet, velocity-Verlet and leapfrog algorithms. All of which use Taylor expansions to solve the equations of motion.

$$\begin{aligned}\mathbf{r}(t+\delta t) &= \mathbf{r}(t)+\mathbf{v}(t)\delta t+\frac{1}{2}\mathbf{a}(t)\delta t^2+\cdots && (1.97)\\ \mathbf{v}(t+\delta t) &= \mathbf{v}(t)+\mathbf{a}(t)\delta t+\frac{1}{2}\mathbf{b}(t)\delta t^2+\cdots && (1.98)\\ \mathbf{a}(t+\delta t) &= \mathbf{a}(t)+\mathbf{b}(t)\delta t+\cdots . && (1.99)\end{aligned}$$

Where the velocity $\mathbf{v}(t)$ is the first derivative, the acceleration $\mathbf{a}(t)$ is the second derivative and $\mathbf{b}(t)$ is the third derivative and so on. The Verlet algorithm [62] uses two Taylor series expansions, one at $t + \delta t$ and one at $t - \delta t$. Where the Taylor series for $t - \delta t$ is shown in equation 1.100 and the combination of this with equation 1.97 $(t + \delta t)$ is shown in equation 1.101.

$$\begin{aligned}\mathbf{r}(t-\delta t) &= \mathbf{r}(t)-\mathbf{v}(t)\delta t+\frac{1}{2}\mathbf{a}(t)\delta t^2+\cdots && (1.100)\\ \mathbf{r}(t+\delta t) &= 2\mathbf{r}(t)-\mathbf{r}(t-\delta t)+\mathbf{a}\delta t^2. && (1.101)\end{aligned}$$

One should note, however, that velocities are not explicitly calculated using this method. Therefore, the velocities are calculated by the following,

$$\mathbf{v}(t)=\frac{\mathbf{r}(t+\delta t)-\mathbf{r}(t-\delta t)}{2\delta t}. \tag{1.102}$$

In addition to not calculating the velocity explicitly, to calculate the next position $\mathbf{r}(t+\delta t)$ the current and previous positions must be known. This algorithm was later adapted into the velocity-Verlet algorithm [63], where the velocities are explicitly calculated.

$$\begin{aligned}\mathbf{r}(t+\delta t) &= \mathbf{r}(t)+\mathbf{v}(t)\delta t+\frac{1}{2}\mathbf{a}(t)\delta t^2 && (1.103)\\ \mathbf{v}(t+\delta t) &= \mathbf{v}(t)+\frac{1}{2}[\mathbf{a}(t)+\mathbf{a}(t+\delta t)]\delta t. && (1.104)\end{aligned}$$

Another variation based on the Verlet algorithm is the leap-frog algorithm. Within the leap-frog algorithm the velocities are calculated at half time-steps.

$$\begin{aligned}\mathbf{r}(t+\delta t) &= \mathbf{r}(t)+\mathbf{v}\left(t+\frac{1}{2}\delta t\right)\delta t \qquad (1.105)\\ \mathbf{v}\left(t+\frac{1}{2}\delta t\right) &= \mathbf{v}\left(t-\frac{1}{2}\delta t\right)+\mathbf{a}(t)\delta t. \qquad (1.106)\end{aligned}$$

1.3.2 Ensembles

Depending on the purpose of the simulation, MD can be performed in a number of statistical ensembles. Within this thesis only three ensembles are used, the microcanonical (NVE) the canonical (NVT) and the isothermal-isobaric (NPT). The NVE ensemble is an example of a thermodynamic closed system, no particles or energy are exchanged outside of the system. The N within NVE stands for constant number of particles, the V for constant volume and the E for constant energy. The NVT and NPT ensemble are examples of thermodynamic closed systems, energy can be transferred into and out of the system, but the particles can not. The T in NVT and NPT stands for constant temperature and the P for constant pressure.

Both NVT and NPT can be thought of as closed system within an isolated system, where outside the closed system is a heat bath allowing the flow of heat energy into the system and controlling the temperature. This is illustrated in figure 1.12.

Within the NPT and NVT ensembles it is necessary to control the temperature, this is done by use of a thermostat. Two commonly used examples are the Berendsen [8] and Langevin [64] thermostats. The Berendsen thermostat regulates the temperature by simply scaling the velocities. The rate in temperature change is given by,

$$\frac{dT}{dt}=\frac{1}{\tau}(T_0-T). \qquad (1.107)$$

Where τ is a time constant and T_0 is the target temperature. The velocities are then scaled, $\mathbf{v}(t)'=\lambda\mathbf{v}(t)$, to match the target temperature [65]. The Berendsen thermostat

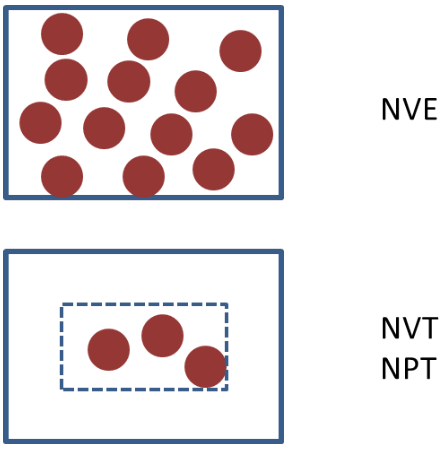


Figure 1.12: Illustration describing the statistical ensembles, NVE is an isolated system, both NVT and NPT are closed systems within an isolated system

is known to have issues with producing correct canonical ensembles for small systems, however, for large system sizes the approximation produces accurate properties [66].

The Langevin thermostat controls the temperature by solving the Langevin equations of motion, which differ to the Newton equations by an introduction of a friction constant ζ. The equations of motion are then,

$$m_i \mathbf{a}_i = \mathbf{F}_i + \zeta \mathbf{v}_i + f'. \tag{1.108}$$

Where f' is a random force determined from a Gaussian distribution.

Similarly, within an NPT ensemble the pressure must be regulated too. This is done by using a barostat. Pressure can be calculated using the Claussius Virial Theorem [5],

$$P = \frac{1}{V}\left[N k_B T - \frac{1}{3}\sum_{i=1}^{N}\sum_{j=i+1}^{N} r_{ij} f_{ij} \right]. \tag{1.109}$$

Where P is the pressure, V is the volume, r_{ij} is the distance between particle i and j and f_{ij} is the force acting between those particles. Within the Berendsen barostat the volume is scaled to adjust the pressure using the scaling parameter μ, in a similar way as within the thermostat [67].

$$\mu = \left[1 - \frac{\beta \partial t}{\tau} (P_0 - P) \right]^{\frac{1}{3}}. \tag{1.110}$$

Where β is the isothermal compressibility and P_0 is the target pressure.

1.3.3 Time Efficiency

Running MD simulations on large systems can be costly, because of this many methods exist to lower these costs. One such method is the application of periodic boundaries, by running a smaller system with periodic boundaries, bulk properties can be calculated. This is illustrated in figure 1.13.

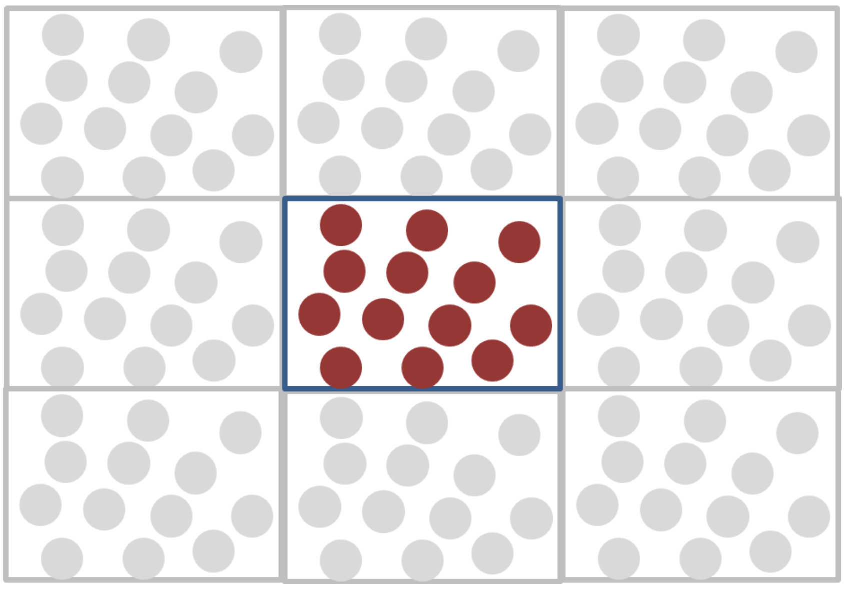


FIGURE 1.13: Illustration describing periodic boundaries, the center box is calculated explictly and the forces from the other boxes are applied to it

In addition, the time step can be increased by freezing out the fastest vibrations, such as bonds involving hydrogen. Two constraint algorithms that do this were used in this thesis, these were SHAKE [68] and LINCS [69].

1.4 Monte Carlo

Another way to generate an ensemble of structures is to use Monte Carlo. Whereas molecular dynamics allows the movement of a system as it evolves through time, Monte Carlo applies random moves and then accepts or rejects them. As such, it does not require the kinetic energy and so can use the configurational partition function (see section 1.5 for a definition). Monte Carlo is a stochastic approach to generating structures, in order to ensure that it remains random (unlike the deterministic approach of MD) a Markov chain is constructed. Structures are accepted into a Markov chain by comparison with the structure that immediately preceded it only.

A Markov chain must satisfy the balance condition [70],

$$\sum_m \rho_m \pi_{mn} = \rho_n. \tag{1.111}$$

When

$$\sum_n \pi_{mn} = 1. \tag{1.112}$$

Where ρ is the state probability and π is the transition probability. One way to ensure that this is satisfied is to apply the overstrict detailed balance condition,

$$\rho_m \pi_{mn} = \rho_n \pi_{nm}. \tag{1.113}$$

There are two aspects of the transition matrix, the trial probability t_{mn} and the acceptance probability π^{acc}_{mn}. The trial probability controls how large the movements are between structures [71], for example this could be the displacement of one atom.

Rearranging equation 1.113 gives,

$$\frac{\pi_{mn}}{\pi_{nm}} = \frac{\rho_n}{\rho_m}. \tag{1.114}$$

Then using the definition of the transition probability,

$$\frac{\pi_{mn}^{acc}}{\pi_{nm}^{acc}} = \frac{t_{nm}\rho_n}{t_{mn}\rho_m}. \tag{1.115}$$

Where ρ is given by the Boltzmann probability,

$$\rho = \frac{\exp -\beta U(\mathbf{r})}{Z(N,V,T)}. \tag{1.116}$$

Cancellation of the partition functions leaves

$$\begin{aligned} \frac{\pi_{mn}^{acc}}{\pi_{nm}^{acc}} &= \frac{\exp[-\beta U_2]}{\exp[-\beta U_1]} && (1.117) \\ &= \exp[-\beta U_2 + \beta U_1] && (1.118) \\ &= \exp[-\beta(U_2 - U_1)] && (1.119) \end{aligned}$$

When $U_2 < U_1$ then the move will always occur, so a *min* function can be introduced,

$$\pi_{mn}^{acc} = min[1, \exp(-\beta(U_2 - U_1))]. \tag{1.120}$$

This equation is known as the Metropolis-Hastings criterion [72]. It can be shown that it satisfies detailed balance [73],

$$\begin{aligned} \frac{\pi_{mn}^{acc}}{\pi_{nm}^{acc}} &= \frac{min[1, \exp(-\beta(U_2 - U_1))]}{min[1, \exp(-\beta(U_1 - U_2))]} && (1.121) \\ &= \begin{cases} \frac{\exp[-\beta(U_2-U_1)]}{1} \text{if } U_2 > U_1 \\ \frac{1}{\exp[-\beta(U_1-U_2)]} \text{if } U_1 > U_2 \end{cases} . && (1.122) \end{aligned}$$

1.5 Free Energy

Free energy is the easiest way to make comparisons directly from theory to experiment. It is is calculated using the rules of statistical mechanics, the object of which is to provide mathematical relations to equilibrium properties of macroscopic systems [74], i.e. it provides the link between what is observed using theoretical methods and what is observed in the macroscopic world. Statistical mechanics introduces a function that if known, all thermodynamic values (entropy, free energy, temperature etc..) can be exactly calculated [75]. This function is known as the partition function and the classical canonical partition function has the following functional form,

$$Q(N,V,T) = \frac{1}{h^{3N}N!}\int \exp(-\beta H(\mathbf{p},\mathbf{r}))d\mathbf{p}d\mathbf{r}. \tag{1.123}$$

To know the value of the partition function requires the knowledge of all energy levels and their occupancy within a system. The only way to know this is to sample infinitely, which is obviously not feasible. Because of this relative free energies are commonly calculated, this will be further explained later.

The probability of selecting a structure with energy H is,

$$\rho(\mathbf{r},\mathbf{p}) = \frac{\exp(-\beta H(\mathbf{r},\mathbf{p}))}{Q(N,V,T)}. \tag{1.124}$$

For simplicity the kinetic energy part of the Hamiltonian can be integrated out and an analytical formula can be found.

$$\begin{aligned}
Q(N,V,T) &= \frac{1}{h^{3N}N!}\int \exp(-\beta U(\mathbf{r}))d\mathbf{r}\int \exp(-\beta K(\mathbf{p}))d\mathbf{p} \quad (1.125)\\
\int \exp(-\beta K(\mathbf{p}))d\mathbf{p} &= (2\pi k_B T m)^{\frac{3N}{2}} \quad (1.126)\\
Q(N,V,T) &= \frac{(2\pi k_B T m)^{\frac{3N}{2}}}{h^{3N}N!}\int \exp(-\beta U(\mathbf{r}))d\mathbf{r} \quad (1.127)\\
&= \frac{1}{N!}\left(\frac{2\pi k_B T m}{h^2}\right)^{\frac{3N}{2}}\int \exp(-\beta U(\mathbf{r}))d\mathbf{r} \quad (1.128)\\
Z(N,V,T) &= \int \exp(-\beta U(\mathbf{r}))d\mathbf{r} \quad (1.129)\\
Q(N,V,T) &= \frac{1}{N![\Lambda(T)]^{3N}}Z(N,V,T). \quad (1.130)
\end{aligned}$$

Where $\Lambda(T)^{3N}$ is the thermal de Broglie wavelength (the average wavelength of gas particles in an ideal gas at temperature T) and the step in equation 1.126 is performed using the following,

$$\begin{aligned}
\int \exp(-\beta K(\mathbf{p}))d\mathbf{p} &= \int \exp\left(-\beta\frac{\mathbf{p}^2}{2m}\right) \quad (1.131)\\
y &= \sqrt{\beta\frac{\mathbf{p}^2}{2m}} \quad (1.132)\\
&= \left(\frac{\beta}{2m}\right)^{\frac{1}{2}}\mathbf{p} \quad (1.133)\\
\frac{dy}{d\mathbf{p}} &= \left(\frac{\beta}{2m}\right)^{\frac{1}{2}} \quad (1.134)\\
d\mathbf{p} &= \frac{dy}{\left(\frac{\beta}{2m}\right)^{\frac{1}{2}}} \quad (1.135)\\
\sqrt{2mk_BT}\int \exp(-y^2)dy &= \sqrt{2\pi m k_B T}. \quad (1.136)
\end{aligned}$$

$Z(N,V,T)$ is the configurational partition function, by using this we can calculate the probability of selecting a conformation based on the potential energy.

$$\rho(\mathbf{r}) = \frac{\exp(-\beta U(\mathbf{r}))}{Z(N,V,T)}. \quad (1.137)$$

The knowledge of partition functions and how to use them to obtain probabilities can then be combined to calculate free energies. As mentioned previously, absolute free energies (free energies relative to an empty cavity) require infinite sampling, because of this relative free energies are commonly calculated. There are several methods that allow the calculation of relative free energies, two of the most common among these are thermodynamic integration and free energy perturbation. Both methods have advantages and disadvantages and will be described in more detail.

1.5.1 Thermodynamic Integration

Thermodynamic Integration (TI) introduces a switching function to smoothly move from one state to another. By using a switching function the degree of overlap between the start and end states is irrelevant as a number of intermediate states are introduced. These states do not have to have any physical significance and are merely a tool used to smooth the path. The switching function λ is introduced into the partition function,

$$Q(N,V,T,\lambda) = \frac{1}{h^{3N}N!}\int \exp(-\beta U(\mathbf{r},\lambda))d\mathbf{r}\int \exp(-\beta K(\mathbf{p}))d\mathbf{p}. \tag{1.138}$$

Where $U(\mathbf{r},\lambda)$ has the following functional form,

$$U(\mathbf{r},\lambda) = (1-\lambda)U_A(\mathbf{r}) + \lambda U_B(\mathbf{r}). \tag{1.139}$$

So when $\lambda = 0$ we are sampling from state A an when $\lambda = 1$ state B is sampled. The use of this partition function leads to the following expression for the Helmholtz free energy [7],

$$A(N,V,T,\lambda) = -k_B T \ln Q(N,V,T,\lambda). \tag{1.140}$$

Then, using partial differentiation,

$$
\begin{aligned}
\frac{\partial A(N,V,T,\lambda)}{\partial\lambda} &= -\frac{k_BT}{Q(N,V,T,\lambda)}\frac{\partial Q(N,V,T,\lambda)}{\partial\lambda} \quad (1.141)\\
&= -\frac{k_BT}{Z(N,V,T,\lambda)}\frac{\partial Z(N,V,T,\lambda)}{\partial\lambda}. \quad (1.142)
\end{aligned}
$$

The step in equation 1.141 is performed by application of the chain rule, and the step in equation 1.142 comes from analytical integration of the kinetic energy cancelling. Although this may not be the case if the mutation occurs between different size atoms, if the full free energy cycle is applied the kinetic energy is compensated [15].

The partial differential of $Z(N,V,T,\lambda)$ is then,

$$
\begin{aligned}
-\frac{k_BT}{Z(N,V,T,\lambda)}\frac{\partial Z(N,V,T,\lambda)}{\partial\lambda} &= -\frac{k_BT}{Z(N,V,T,\lambda)}\frac{\partial}{\partial\lambda}\int \exp(-\beta U(\mathbf{r},\lambda))d\mathbf{r} \quad (1.143)\\
&= -\frac{k_BT}{Z(N,V,T,\lambda)}\int\left(-\beta\frac{\partial U}{\partial\lambda}\right)\\
&\quad \exp(-\beta U(\mathbf{r},\lambda))d\mathbf{r} \quad (1.144)\\
&= \left\langle\frac{\partial U}{\partial\lambda}\right\rangle. \quad (1.145)
\end{aligned}
$$

The free energy difference between state A and B can be found using the following,

$$
\begin{aligned}
\Delta A_{AB} &= \int_0^1 \frac{\partial A}{\partial\lambda}d\lambda \quad (1.146)\\
&= \int_0^1 \left\langle\frac{\partial U}{\partial\lambda}\right\rangle_\lambda d\lambda. \quad (1.147)
\end{aligned}
$$

This process allows an alchemical mutation between two states. In practice there are two ways to obtain this data, single and dual topology. Single topology involves shrinking and growing bonds and atoms as the value of λ changes. Within dual topology both systems are present and λ controls the degree of how “present” a system is. If a dual topology method is used the end states ($\lambda = 0$ and $\lambda = 1$) can be numerically unstable, this is due to the particles still having mass and velocities, but is not coupled to any other degree of freedom, so can cause convergence issues [15].

In either case, if a standard potential is used for the mutation, the energies would be massively repulsive, because of this, soft-core potentials must be used. A widely used soft-core potential for the Lennard-Jones energy is [76],

$$U_{LJ} = 4\epsilon_{ij}(1-\lambda)\left([\alpha\lambda^2 + (r_{ij}/\sigma_{ij})^6]^{-2} - [\alpha\lambda^2 + (r_{ij}/\sigma_{ij})^6]^{-1}\right). \tag{1.148}$$

ϵ_{ij} and σ_{ij} are standard Lennard-Jones parameters, r_{ij} is the interatomic distance between particle i and j and α controls how "soft" the potential is. With the soft-core potential for the Lennard-Jones energy, it is possible to perform thermodynamic integration by first switching off the electroststatic energy. Another option is to use an additional soft-core potential for the electrostatic energy. Steinbrecher et al. [76] describe this and show a comparison between one step (using soft-core potentials for both electrostatic and Lennard-Jones energy) and two step (switching electrostatics off first, mutating, then switching electrostatics back on) within AMBER [17].

The disadvantage of using TI to obtain free energies is that it must be feasible to sample both states. When mutating between two classical states, this is not an issue and the error can be calculated by hysteresis (performing the forward and reverse calculation and comparing the difference of the magnitude). By introducing intermediate states a potential of mean force (PMF) is produced and can be viewed to ensure a smooth path between states. If any large jumps present, it could be a sign that more λ values are needed.

1.5.2 Free Energy Perturbation

By using free energy perturbation it is possible to move from one state to another without the necessity of sampling any configurations from the more expensive state. In principle this is exact, but in practice the free energies only converge if the two states have a high degree of configurational space overlap [77].

In order to understand how it is possible to calculate the free energy difference without sampling from the second state, consider the following [7],

$$U_B(\mathbf{r}) = U_A(\mathbf{r}) + U_1(\mathbf{r}). \tag{1.149}$$

Where $U_A(\mathbf{r})$ is our current potential, $U_B(\mathbf{r})$ is the desired potential and $U_1(\mathbf{r})$ is the perturbation required to move between these potentials. The configurational partition function for $U_B(\mathbf{r})$ is,

$$Z_B(N,V,T) = \int \exp[-\beta U_B(\mathbf{r})] d\mathbf{r}. \tag{1.150}$$

Combining equation 1.149 and 1.150 gives,

$$\begin{aligned} Z_B(N,V,T) &= \int \exp[-\beta U_A(\mathbf{r})] \exp[-\beta U_1(\mathbf{r})] d\mathbf{r} && (1.151)\\ &= \frac{Z_A(N,V,T)}{Z_A(N,V,T)} \int \exp[-\beta U_A(\mathbf{r})] \exp[-\beta U_1(\mathbf{r})] d\mathbf{r} && (1.152)\\ &= Z_A(N,V,T) \langle \exp[-\beta U_1(\mathbf{r})] \rangle_A. && (1.153) \end{aligned}$$

The free energy for potential $U_B(\mathbf{r})$ is calculated using the following,

$$\begin{aligned} A_B(N,V,T) &= -k_B T \ln Z_B(N,V,T) && (1.154)\\ &= -k_B T \ln \left(Z_A(N,V,T) \langle \exp[-\beta U_1(\mathbf{r})] \rangle_A \right) && (1.155)\\ &= -k_B T \left(\ln Z_A(N,V,T) + \ln \langle \exp[-\beta U_1(\mathbf{r})] \rangle_A \right) && (1.156)\\ &= -k_B T \ln Z_A(N,V,T) - k_B T \ln \langle \exp[-\beta U_1(\mathbf{r})] \rangle_A. && (1.157) \end{aligned}$$

The free energy required for state B has been split into the free energy for state A and the free energy required to perturb between the systems. The perturbation free energy requires the calculation of an ensemble average and not the full configurational partition function, therefore it can be used to calculate relative free energies.

$$\Delta_{BA}A_1(N,V,T) = -k_BT\ln\langle\exp[-\beta U_1]\rangle_A \tag{1.158}$$

This equation is known as the Zwanzig equation [78].

1.5.3 MM-PBSA

MM-PBSA (molecular mechanics - Poisson-Boltzman surface area) is an example of a more computationally efficient, but less rigorous approach to calculate the free energy[79]. It is an example of a continuum method, where solvent effects are obtained by approximating the solvent as a polarisable continuum. This then uses the Poisson-Boltzman equation (equation 1.25) to approximate the solvent effects. These free energies are then combined with the interaction energy and entropy to provide the free energy of binding.

The calculation of the interaction energies can be performed by running three separate simulations, one for the complex, one for the host and one for the ligand. This approach, although correct, suffers from convergence issues, therefore commonly this is condensed into a single simulation.

The interaction energy is then calculated by equation 1.160, where the three simulation approach calculates the interaction energy by equation 1.159. So each average in equation 1.159 comes from a different simulation.

$$\Delta U_{int} = \langle U_{complex}\rangle - \langle U_{host}\rangle - \langle U_{ligand}\rangle \tag{1.159}$$

$$\Delta U_{int} = \langle U_{complex} - U_{host} - U_{ligand}\rangle \tag{1.160}$$

The final free energy is then given by,

$$\Delta G = \Delta U_{int} + \Delta G_{pbsa} - T\Delta S. \tag{1.161}$$

Within the non-polar solvation term used within equation 1.161 implicitly contains an estimation of the entropic cost of inserting the solute into a box of solvent. This, however, does not consider any entropic cost associated with the actual binding process[80]. To calculate this many methods can be used, the only one used within this thesis however, is normal mode analysis.

1.5.3.1 Normal Mode Analysis

Within Normal mode analysis the protein-ligand system is restricted to a harmonic approximation of a minimum. As such, each system first undergoes a rigorous minimisation procedure to find the lowest energy local minimum. The vibrational entropy is then calculated from the frequency of the harmonic potential (equation 1.162). Lower frequencies represent wider potentials, with more conformations available, so the entropy is higher[81].

$$S_{vib} = \sum_{i=1}^{3N} \left(\frac{h\nu_i}{k_B T}\right) \frac{1}{\exp\left[\frac{h\nu_i}{k_B T}\right] - 1} - \ln\left(1 - \exp\left[-\frac{h\nu_i}{k_B T}\right]\right). \tag{1.162}$$

This is then used as an approximation to the conformational entropy within the free energy of binding.

Chapter 2

Introduction

Since the development of classical mechanics by Isaac Newton, models and theories have been used to explain natural phenomena. It wasn't until the early 1900's that physicists discovered that classical mechanics cannot accurately describe the behaviour of small particles and the theory of quantum mechanics was developed. These theories have been adopted by chemists who developed computational chemistry approaches in order to understand properties and processes of molecules.

Pharmaceutical companies would like to use computational methods to predict interactions between proteins and ligands before potentially expensive experimental techniques are performed. Because of this, it is crucial to be able to perform accurate calculations, as such any method is validated extensively against "test" systems (systems with known experimental results). In the case of protein-ligand binding, free energies are used to compare theoretical with experimental results. The reason for this that the experimental equilibrium constant K can be easily converted to the free energy ΔG, which can be calculated using statistical mechanics [82]. K is known as the association constant and is defined as,

$$K = \frac{[AB]}{[A][B]}. \tag{2.1}$$

Where [] represents the concentration. The free energy can be calculated using the following equation,

$$\Delta G_{bind} = -k_B T \ln K. \tag{2.2}$$

It is clear to see that the calculation of accurate free energies of binding is crucial to the pharmaceutical industry. The conventional approach to calculate these free energies is to use classical mechanics to describe the system in combination with molecular dynamics (MD) and/or Monte Carlo (MC) to generate structures [83, 84]. Enough structures are needed to obtain converged ensemble averages, for MD this means that the ergodic hypothesis must be satisfied. The ergodic hypothesis states that the time average of a system is equal to the ensemble average[38]. Because free energies require a large amount of sampling to converge[84], classical mechanics is the obvious choice over quantum methods due to the lower computational cost associated with it. This approach relies on the transferability of force fields [16], and cannot possibly model every interaction relevant to calculate accurate binding free energies. For example, classical mechanics does not explicitly include charge transfer, polarisation or the exchange energy. In standard force fields, such as the AMBER force field [20], the polarisation is included implicitly within the parameterisation [85]. However, these force fields cannot show the dependence of the electronic structure of a molecule to its environment [86]. This has led to the development of force fields that can explicitly calculate polarisation, known as polarisable force fields. Polarisation is taken into account within these force fields by an additional term within the functional form, U_{ele}^{ind} [82]. The interactive induction scheme for example, will calculate the effect the surrounding electrostatics has on a polarisable site, forming a new dipole, then the effect the this dipole has on the surroundings, and repeats until convergence [15]. This process can become computationally expensive and still provides no solution to other non-classical interactions.

Ideally, the ensemble of structures generated would be as a result of *ab initio* molecular dynamics, but this is far too computationally expensive to be feasible. This leads to the use of hybrid methods and guiding potentials. QM/MM is an example of a hybrid method and was first suggested by Warshel et al.[87], which is the combination of using QM for a small region of interest and MM for the surroundings. The energy then becomes,

$$E_{QM/MM} = E_{QM} + E_{MM} + E_{QM-MM}. \tag{2.3}$$

Where E_{QM} is the quantum potential energy of the region of interest, E_{MM} is the potential energy of the surroundings and E_{QM-MM} is the interaction of the two regions. Some sources suggest that this is all that is needed to accurately describe interactions within protein-ligand systems, as the electron correlation is primarily short-range [82]. But this still relies heavily on the forcefield to describe interactions accurately. Difficulties arise when the QM region is bonded covalently to the MM region and a great deal of expertise is required to perform calculations using these hybrid methods [2]. The implementation of QM/MM can be described by two categories: mechanical embedding and electrostatic embedding [88]. The former uses the classical force field to describe electrostatics and dispersion. The latter calculates the electrostatics and polarisation at a QM level. The latter of the two being the more accurate [89].

The use of a guiding potential is another common way to obtain accurate free energies, these involve using a cheap potential to sample structures, then a perturbation to a more expensive potential. This approach uses the extended thermodynamic cycle introduced by Štrajbl et al. [90], which can be seen in figure 2.1.

$$\begin{tikzcd}
A^{QM}_{complex} & A^{MM}_{complex} \arrow[l, "\Delta G_8"'] \arrow[r, "\Delta G_1"] & B^{MM}_{complex} \arrow[r, "\Delta G_2"] & B^{QM}_{complex} \\
A^{QM}_{solv} \arrow[u, "\Delta G_7"] & A^{MM}_{solv} \arrow[l, "\Delta G_6"'] \arrow[r, "\Delta G_5"] \arrow[u, "\Delta G_9"] & B^{MM}_{solv} \arrow[r, "\Delta G_4"] \arrow[u, "\Delta G_{10}"] & B^{QM}_{solv} \arrow[u, "\Delta G_3"']
\end{tikzcd}$$

Figure 2.1: The extended thermodynamic cycle

In this cycle, ΔG_1 and ΔG_5 are classical alchemical mutations of the ligand when bound in a protein and when in solvent respectively. ΔG_2, ΔG_4, ΔG_6 and ΔG_8 are the perturbations from a classical system to a quantum system. ΔG_7 and ΔG_3 are the desired free energies, but are much too computationally expensive to calculate. However, with all the other free energies, the relative free energy ($\Delta G_7 - \Delta G_3$) can be calculated by $\Delta G_1 + \Delta G_2 - \Delta G_4 - \Delta G_5 + \Delta G_6 - \Delta G_8$.

Application of the extended thermodynamic cycle has seen many uses, from reaction mechanisms [90, 91] to correcting binding free energies [92, 93, 94] For the purposes of this thesis, the application will be directed toward correcting binding free energies. Beierlein et al. [92] correct the binding free energy using the Coulomb energies in a single step perturbation from the classical potential to a QM/MM potential. Fox et al. [93] performed a similar task on the solvation free energies using the interaction energies this time perturbing from the classical to a fully quantum potential. In both studies the Zwanzig equation [78] was used (equation 2.4), however, convergence was achieved by using interaction energies. Formally, this equation requires the use of total energies.

$$\Delta G = -k_B T \ln \left\langle \exp\left[-\frac{E_{MM} - E_{QM}}{k_B T} \right] \right\rangle \tag{2.4}$$

Where E_{MM} is the classical potential energy and E_{QM} is the quantum potential energy, when interaction energies are used, these become ΔE_{MM} and ΔE_{QM}. Interaction energies can simply be described by the following,

$$\Delta E = E_{complex} - E_{ligand} - E_{host}. \tag{2.5}$$

In the case of potentials that include only two-body interactions, this cancels out any intramolecular terms and intermolecular interactions within the host. The complex is the entire system e.g. a ligand bound to a protein, and within this example the host would be the protein in the same geometry but without the ligand present. This makes the approximation that this conformation of the protein would be sampled in the absence of the ligand. Although an approximation, it is an approximation that can provide converged free energies [92, 93]. One potential issue of using total energies is the overlap of configurational space. However, Woods et al. [94] developed a method that allows the perturbation from a classical system to a QM/MM system with the use of total energies. Doing this involved evaluating whether a classically obtained snapshot was a good representation at the QM/MM level and accepting or rejecting it.

The search for a method that can provide accurate free energies of binding that is relatively inexpensive is one of the main challenges within computational chemistry. As such, within this thesis, we would like to explore the possibilities of finding a method that can accurately calculate quantum corrected free energies for a range of systems when using total energy approaches.

Chapter 3

Calculating Quantum Corrected Free Energies Using a Single Step Perturbation Approach

3.1 Introduction

The introduction explained the importance of calculating accurate binding free energies for all systems using a relatively inexpensive method. Also that, to find a rigorous method that obeys the rules of statistical mechanics, total energies should be used. This chapter provides the description and results of how this task was initially tackled. Following a similar method described by Fox et al. [93] where classical mechanics will be used to generate an ensemble of structures cheaply, then the Zwanzig equation [78] will be used to perturb from the generated classical ensemble to a fully quantum one, making use of the extended thermodynamic cycle. Fox et al. [93] briefly describe a few issues associated with this, the first being that due to the large size difference between the MM and QM total energies, the calculation of the exponential required for the Zwanzig equation (shown in equation 3.1) is numerically unstable, however, numerical techniques can be used to calculate this [95]. The second is that the number of classical snapshots required for convergence is too high. This poses more of an issue, and it is because of this that we will first apply the method to small test systems with considerably less configurational space available to sample than a larger system, e.g. a protein, to test convergence. As such, the hydration free energies of two simple systems with slightly different chemical characteristics were calculated. These selected systems were ethane and ethanol in aqueous solvent.

3.2 Methods

3.2.1 Classical Simulation Setup

All classical simulations were performed in AMBER 10 [17], the initial structures for ethane and ethanol were generated using the MOE programme [96] and parameterised using antechamber [97]. TIP3P [11] was used as a solvent model and the generalised AMBER force field (GAFF) [23] was used to describe ethane and ethanol.

A fifteen step equilibration procedure was applied to two test systems, ethane in 133 TIP3P water molecules and ethanol in 138 TIP3P water molecules. A small box size was selected to keep quantum simulations fast, as a caveat of this, only a 5 Å nonbonded cutoff could be applied, this may cause convergence issues, but is required to keep the computational time required down. The equilibration procedure was as follows. Initially the structure was minimised using 1000 steps of steepest descent and 1000 steps of conjugate gradient with a restraint of 1000 kcal/mol-Å^2 on all non-hydrogen atoms. This step was then repeated with the restraint removed on the oxygen atoms within the water. After this the system was heated from 100 to 300K over 100ps, SHAKE [68] algorithm was used to allow for a 2 fs time step. This was performed in a canonical ensemble with a Langevin thermostat with a 3 ps^{-1} collision constant with the same restraint as the previous step applied. Then the volume was equilibrated by running within a isothermal-isobaric ensemble for 500ps. The simulation was then cooled to 100K over 100ps. Step one was repeated with the same restraint as the previous step. This step was repeated a further eight times lowering the restraint each time to 500, 100, 50, 20, 10, 5, 2 kcal/mol-Å^2 and removing it completely. The system was then heated back up to 300K with no restraints present.

After the equilibration an MD simulation of 40 ns was performed in the canonical ensemble for ethane in 133 water molecules and snapshots extracted at regular intervals. For ethanol in water, a simulation length of 200 ns was run, the increase in length was to ensure the results were not being affected by correlated structures. So for ethane, ultimately each snapshot was 20ps apart, whereas for ethanol each snapshot was 100ps apart. Although 20ps should be long enough to ensure that the structure is not correlated to the previous structure, by extending this to 100ps we make correlation much less likely.

An additional test that will be described in more detail in the results section involved classical charge perturbations. The starting point for these simulations was the equilibrated ethane in water system, and a simulation of 200 ns was run before extracting snapshots at regular intervals.

3.2.2 Quantum Simulation Setup

All quantum calculations were performed using ONETEP [54] either using NGWFs or pseudo-atomic orbitals (PAOs). If NGWFs were used, the setup involved using NGWFs with radii of 8 bohr with 4 NGWFs describing heavy atoms and 1 for hydrogen. When PAOs were used, a single zeta + polarisation basis set was used. NGWFs with 8 bohr radii have been found to be of similar accuracy to a cc-PVTZ basis set [98].

3.2.3 Application of a Single Step Perturbation

In order to perturb between two ensembles in a single step, the Zwanzig equation can be used. This can be seen in equation 3.1.

$$\Delta G = -k_B T \ln \left\langle \exp \left[-\frac{\Delta E_{QM-MM}}{k_B T} \right] \right\rangle_{MM} \tag{3.1}$$

The large size difference between E_{MM} and E_{QM} makes the direct application of equation 3.1 numerically unstable. As such, a method explained by Berg [95] can be used. The method follows,

$$
\begin{aligned}
C &= A + B && (3.2)\\
\ln C &= \ln(A+B) && (3.3)\\
&= \ln\left[\mathrm{max(A,B)}\left(1+\frac{\mathrm{min(A,B)}}{\mathrm{max(A,B)}}\right)\right] && (3.4)\\
&= \mathrm{max(\ln A, \ln B)} + \ln(1+\exp[\mathrm{min(\ln A, \ln B)} - \mathrm{max(\ln A, \ln B)}]). && (3.5)
\end{aligned}
$$

Where A and B are equal to the following equation, and ΔE_{QM-MM} changes based on the energy difference of the snapshots included within the equation,

$$\exp\left(-\frac{\Delta E_{QM-MM}}{k_B T}\right). \tag{3.6}$$

The formula can then be used recursively until all snapshots have been included.

3.3 Results

3.3.1 Applying a Single Step Perturbation Approach to Non-Polar and Polar Test Systems

The results when applying the perturbation to the structures generated from the 40ns MD simulation for ethane in water are shown below in Table 3.1. These results have been split into two, state A represents structures obtained from the first 20ns of the MD simulation and state B from the second 20ns. By doing this, convergence is achieved if both state A and state B have the same free energies, i.e. ΔΔ G = 0. In this case the quantum calculations were run using NGWFS.

When run within ONETEP, some of the structures generated from the classical MD would not converge and as such were discarded from the results. Examination of these structures yielded no explanations. The convergence issues came from the outer loop of ONETEP, the NGWF optimisation. However, even excluding these results, the free energy does not converge and shows a difference of 5.44 kcal/mol between the two halves

Table 3.1: MM to QM perturbation applied to ethane using NGWFs, allowing up to 10% of snapshots to be omitted due to SCF convergence errors. All energies shown are in kcal/mol

Number of snapshots	1^{st} 20.0 ns	2^{nd} 20.0 ns	Difference
	Δ G	Δ G	$\Delta\Delta$ G
400	-1447906.57	-1447912.01	5.44

of the MD simulation. As such, it is clear that 400 snapshots from state A and B are not enough to provide convergence of the method. So additional snapshots were used, but for speed, these snapshots were calculated using PAOs instead of NGWFs. Simulation times for the single point energy calculations when using NGWFs were approximately 7 hours on 6 cores, whereas by using PAOs, this computational time lowered to 4 hours on 6 cores. Additionally, by using PAOs the previous convergence issues were not present as only the density kernel is optimised. However, when increasing the number of snapshots from each state to 1000 and applying the single step perturbation (SSP) method, $\Delta\Delta$G does not converge to 0. These results can be seen in table 3.2.

Table 3.2: MM to QM perturbation applied to ethane using PAOs. All energies shown are in kcal/mol

Number of snapshots	1^{st} 20.0 ns	2^{nd} 20.0 ns	Difference
	Δ G	Δ G	$\Delta\Delta$ G
1000	-1442495.51	-1442499.82	4.31

On first inspection, it appears that the difference is lowered and could point to convergence being attainable if a great deal more snapshots were included. However, further investigation into the lack of convergence within the $\Delta\Delta$ G has led to calculating the free energy as a function of the number of snapshots included. In this way if the free energy is being dominated by one or a few snapshots it will become apparent. The resulting graph can be seen in figure 3.1.

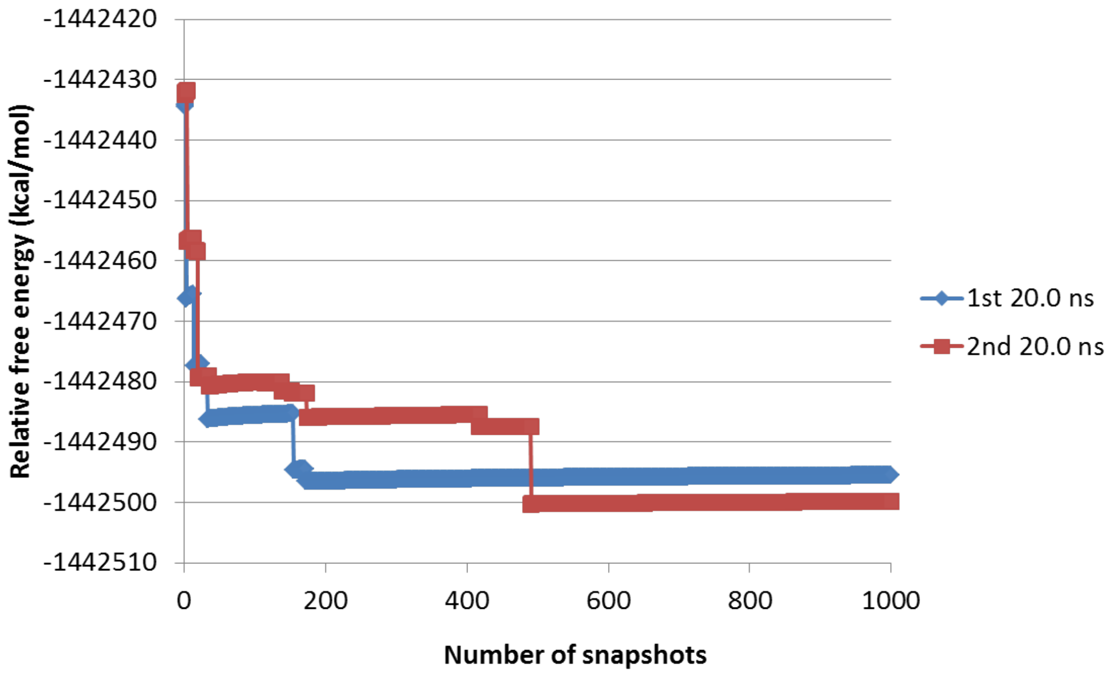


FIGURE 3.1: Free energy as a function of the number of snapshots for ethane in 133 water, all values shown are in kcal/mol

The "sawtooth" shape of the graph in figure 3.1 is characteristic of poor sampling [99]. In order to check if the lack of convergence in the ΔΔ G was related to using a basis set with poor accuracy, when applying the single step perturbation (SSP) method to ethanol in 138 water, NGWFs were used. Additionally the original MD simulation used to generate configurations was run for a longer time of 200ns to provide certainty of obtaining uncorrelated structures. However, regardless of these additional measures taken to achieve convergence of ΔΔ G, the difference is much larger (table 3.3) and 10% of structures would not converge, an issue that, again, was not explained by further examination of these structures, but can be attributed to the outer loop.

TABLE 3.3: MM to QM perturbation applied to ethanol using NGWFs, allowing upto 10% of snapshots to be omitted due to SCF convergence errors. All energies shown are in kcal/mol

Number of snapshots	1^{st} 100.0 ns Δ G	2^{nd} 100.0 ns Δ G	Difference ΔΔ G
1000	-1511726.64	-1511765.45	38.81

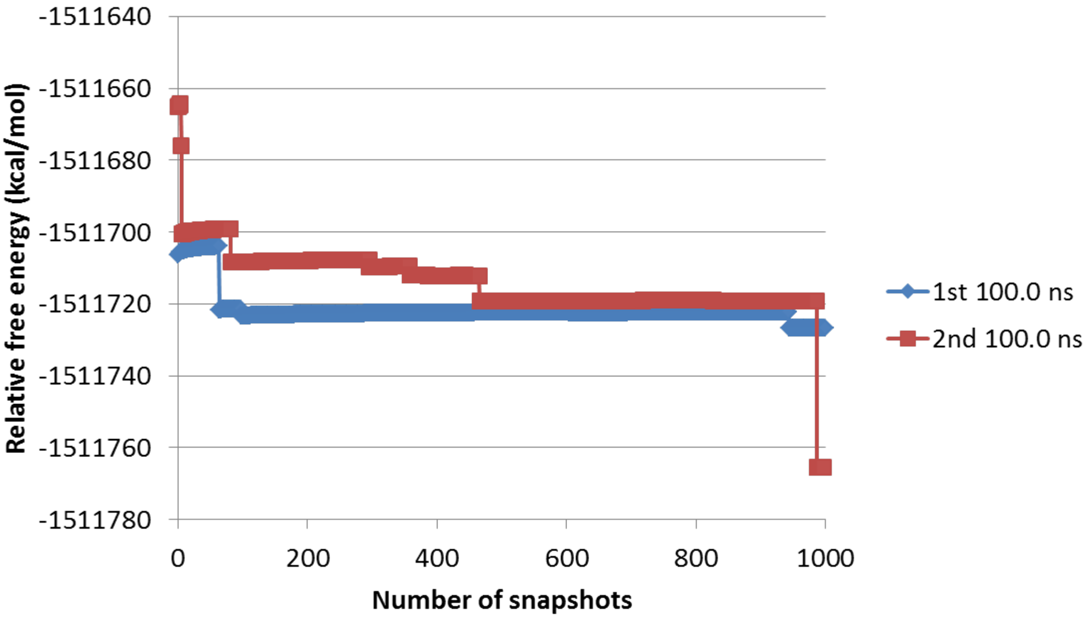


Figure 3.2: Free energy as a function of the number of snapshots for ethanol in 138 water, all values shown are in kcal/mol

Again plotting ΔΔ G as a function of the number of snapshots, was then produced for ethanol in 138 water and shown in figure 3.2. This figure shows the catastrophic effect that one of these outlier structures can have on the free energy. Where the largest effect to the free energy occurred within this 2nd 100 ns simulation of ethanol in 138 water and occurred at almost the end of the MD simulation. This implies that these snapshots that have a large effect on the free energy can occur anywhere. The distributions of energies produced by the structures within this simulation were then compared with an aim to find a reason behind the problematic structures.

Three energy distributions were produced from this simulation, these energies were the MM, QM and QM-MM distributions. To analyse these distributions, they were plotted within box and whisker plots. The “box” in the box and whisker plots shows the interquartile range (IQR) of the data, and the line in the middle of the box is the median. This whisker length is 1.5 times the length of the IQR, any snapshots outside of this length can be thought of as an energetical outlier.

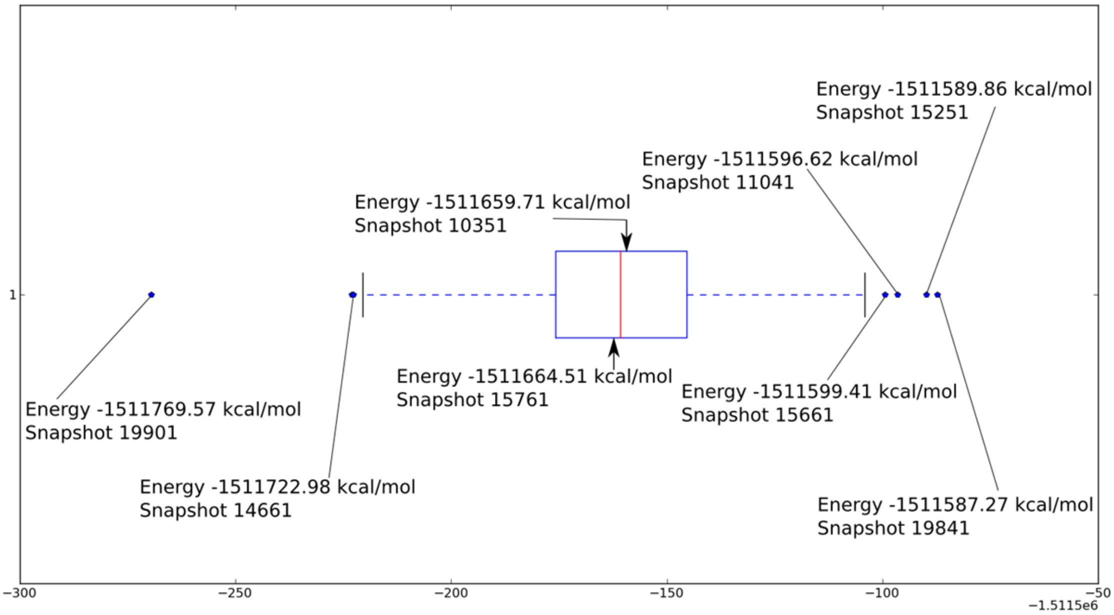


Figure 3.3: Energy difference distribution for the ethanol in 138 water, 2nd 200 ns, plotted in a box and whisker plot

Six outlier snapshots can be identified from figure 3.3 and, for analysis purposes, two snapshots close to the median value were selected to show their location within the other distributions. Something that should be noted is the large range of energy difference present within the box and whisker plot of 182.30 kcal/mol. Producing similar plots for the MM and QM energy distributions provides some interesting results.

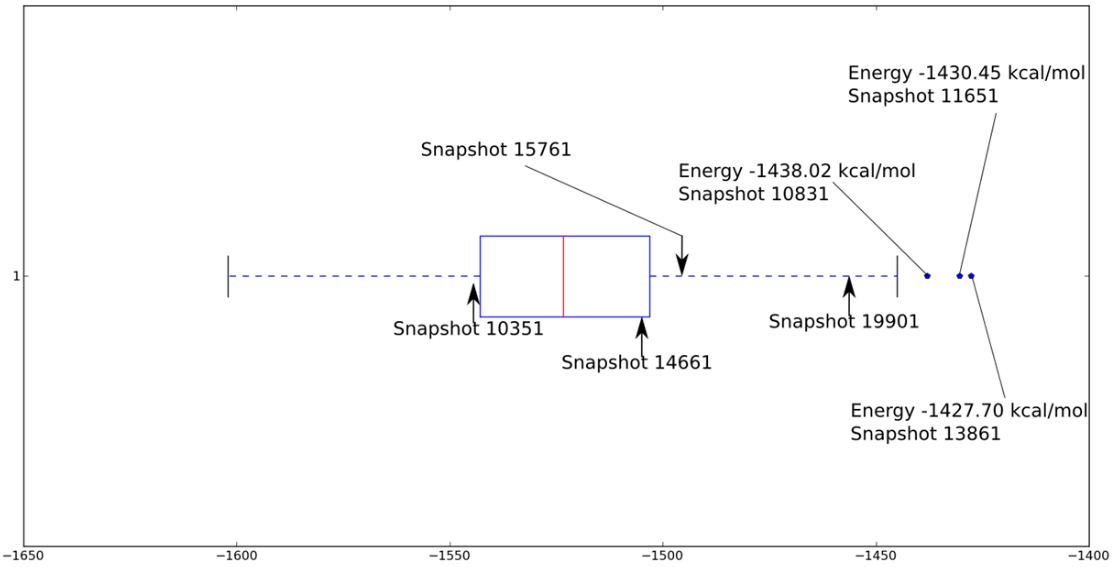


Figure 3.4: MM energy distribution for the ethanol in 138 water, 2nd 200 ns, plotted in a box and whisker plot

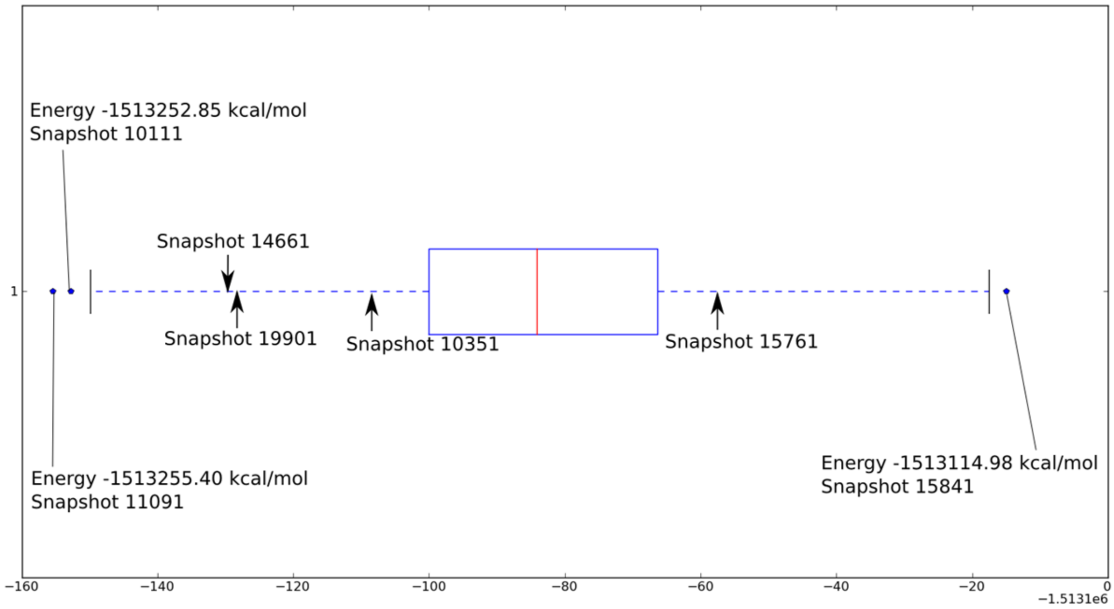


Figure 3.5: QM energy distribution for the ethanol in 138 water, 2nd 200 ns, plotted in a box and whisker plot

Comparing which snapshots were shown to be outlier snapshots in each distribution produced the following table (table 3.4).

Table 3.4: Outlier snapshots

Energy Difference	QM	MM
19901	11901	10831
14661	10111	11651
15661	15841	13861
11041		
15251		
19841		

Table 3.4 shows that for all three distribution, there are no common outliers. Which proves that the large jumps present in figure 3.3 attributed to the energy difference outliers are not present due to a structure being anomalous in either the MM or QM distributions. Additionally, the examination of the positions of the snapshots identified

as close to the median in the energy differences within the MM and QM distributions show that, in both cases, the snapshots are outside of the IQR. However, because both of the snapshots appear on the same side of the MM and QM distributions i.e. either both snapshots appear on the more negative side of the IQR or both snapshots appear on the less negative side of the IQR, the snapshots appear close to the median in the energy difference distribution. With regard to the outlier snapshots within the energy difference distribution, the more negative side of the distribution will have a larger effect on the free energy. This is due to the calculation of the exponential of the negative energy present within the Zwanzig equation (equation 3.1). Finding these snapshots within the MM and QM distributions show them to be on different sides of the distributions. This difference shows that the MM and QM configurational space does not overlap exactly, and this inexact overlap of configurational space causes these energy difference outliers.

3.3.2 Controlling the Level of Configurational Overlap to Measure the Convergence of the Free Energy

In order to investigate the perturbation between different potentials, while controlling the degree of configurational space overlap, a model system was set up. Calculating quantum energies are orders of magnitude more computationally expensive than classical energies, as such, the test case was between classical systems. The system of ethane in 133 water was used with the setup described within the methods section. The MD simulation was then divided into two parts and 10000 snapshots were extracted at regular intervals and the charges present on ethane were then perturbed. The standard charges on ethane used were obtained from antechamber [97], these were -0.0941 e on carbon and 0.0317 e on hydrogen. The perturbations then involved changing these charges and calculating the new energies, these charges were changed by doubling them, tripling them etc.. up to eight times the charge. The convergence of $\Delta\Delta$ G was then measured while lowering the configurational space overlap.

Table 3.5 shows that by lowering the amount of overlap present between the two states involved in the perturbation, the free energy does not converge. The trend of $\Delta\Delta$ G is as expected, when perturbations are small the free energy converges. The configurational

Table 3.5: Calculating the free energy required to perturb the charges on ethane in 133 water from standard charges to increased charges. Sampling performed using standard charges. All energies shown are in kcal/mol

Increased charge	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
	Δ G	Δ G	ΔΔ G
Double	2.800	2.798	0.002
Triple	7.439	7.434	0.005
Quadruple	13.898	13.886	0.012
Pentuple	22.148	22.111	0.037
Sextuple	32.231	32.108	0.123
Septuple	43.957	43.635	0.322
Octuple	57.365	56.751	1.206

space overlap is demonstrated in figure 3.6 and 3.7. Where, in figure 3.6, the carbon carbon bond length is measured when an MD simulation is performed using the altered charges. Figure 3.7 shows a similar analysis, but using the hydrogen-carbon-carbon angle of these MD simulations. These figures show that as the charge is increased, both the bond length and angle is increased slightly.

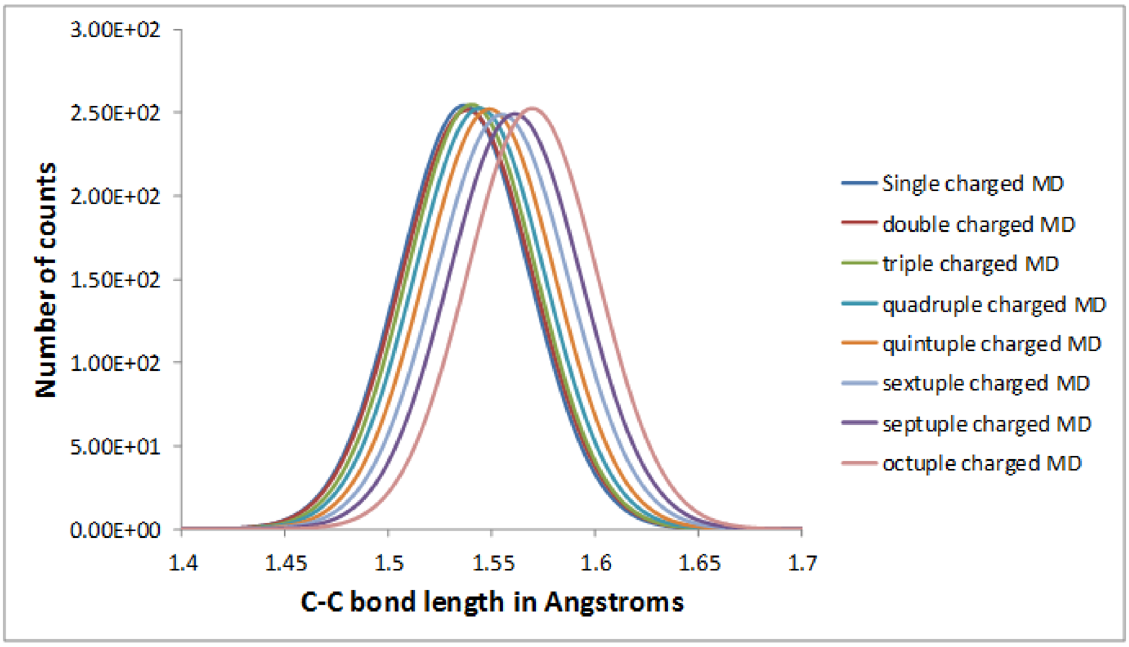


Figure 3.6: Carbon-carbon bond length within 8 different MD simulations with different partial charges on ethane

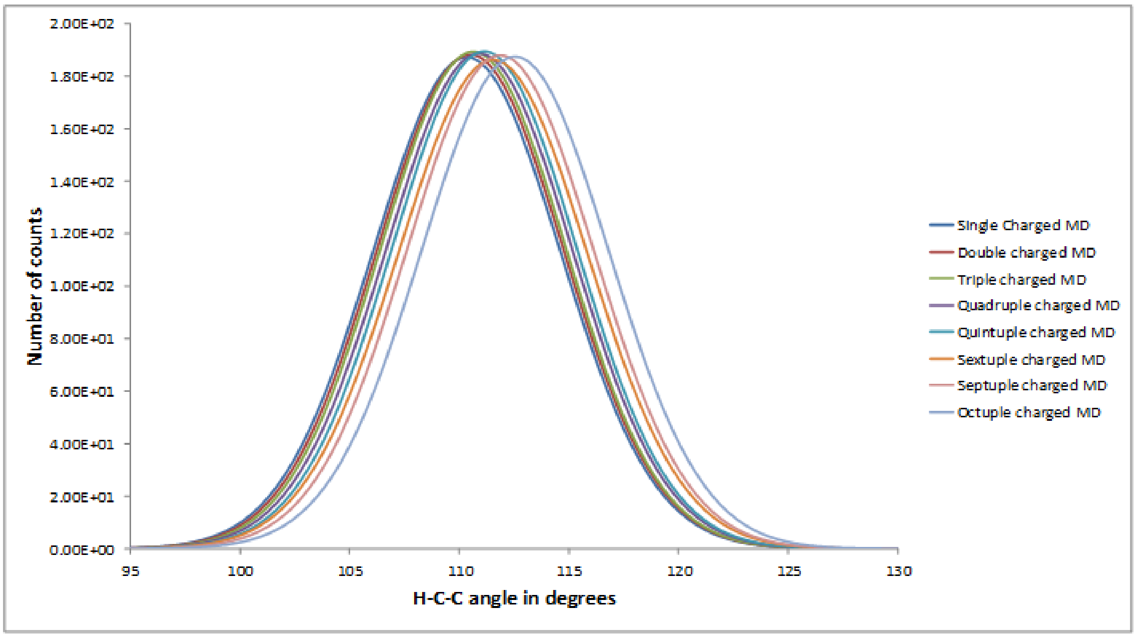


Figure 3.7: Carbon-carbon-hydrogen angle within 8 different MD simulations with different partial charges on ethane

Noticing how poor the configurational space overlap is between the standard and octuple charged systems, the reverse calculation was performed. This involved running an MD simulation at the desired charge and perturbing to a standard charged system. Free energy is a state function, so the reverse process should be the negative of the forward process.

Table 3.6: Calculating the free energy required to perturb the charges on ethane in 133 water, from standard charges to an increased charge. Sampling performed using the increased charges. All energies shown are in kcal/mol

Increased charge	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
	Δ G	Δ G	ΔΔ G
Double	2.773	2.771	0.002
Triple	7.290	7.293	-0.003
Quadruple	13.386	13.428	-0.042
Pentuple	20.864	20.795	0.069
Sextuple	29.422	29.228	0.194
Septuple	36.360	37.766	-1.406
Octuple	43.939	45.746	-1.807

The results show that when the configurational space does not overlap exactly the forward and reverse process does not converge. The lack of overlap here is small, but the

TABLE 3.7: The difference between perturbations, each value is the average of the 1^{st} and 2^{nd} 100ns. All energies shown are in kcal/mol

Increased charge	standard to increased charge Δ G	increased charge to standard Δ G	Difference ΔΔ G
Double	2.799	2.772	0.027
Triple	7.437	7.292	0.145
Quadruple	13.892	13.407	0.485
Pentuple	22.130	20.830	1.300
Sextuple	32.170	29.325	2.845
Septuple	43.796	37.063	6.733
Octuple	57.058	44.843	12.215

overlap between classical and quantum configurational space has been shown to be poor. This offers some explanation toward the poor convergence of the MM to QM. It is clear that without sampling a large amount of configurational space, the free energy from MM to QM will not converge. Also the cost of running quantum calculations on 1000 snapshots for ethane in 133 water is a manageable computational cost, but for a larger, more biologically relevant system, 1000 snapshots will be extremely computationally expensive. So several post-processing methods were applied to try and converge these free energies and the results are shown in the next section.

3.3.3 Methods to Converge the Free Energy

The largest effect an energy difference outlier had on the convergence of free energies was observed for ethanol in 138 water in the 2^{nd} 100 ns. Because of this, all attempts to converge the free energy, using the data already obtained, will be for this system. Section 3.3.1 shows that it is impossible to predict where energy difference outliers will occur by examining only the classical or quantum energies. Because of this, the first attempt to better converge the free energy was to subtract the average energy from both energy distibutions,

$$\Delta E = (E_i^{MM} - \langle E^{MM} \rangle) - (E_i^{QM} - \langle E^{QM} \rangle). \tag{3.7}$$

Where the subscript i represents the i^{th} structure, $\langle E_i^{MM} \rangle$ is the average classical energy.

The results, shown in table 3.8, display the same poor convergence of table 3.3. Applying the Zwanzig equation [78] as a function of the number of snapshots, provides large "jumps" in free energy again (figure 3.8), showing that subtracting the average from each energy distribution cannot converge the free energy.

Table 3.8: MM to QM perturbation applied to ethanol using NGWFs, each energy distribution has the average energy subtracted from it. Allowing upto 10% of snapshots to be omitted due to SCF convergence errors. All energies shown are in kcal/mol

Number of snapshots	1^{st} 20.0 ns	2^{nd} 20.0 ns	Difference
	Δ G	Δ G	ΔΔ G
1000	-66.04	-104.51	38.47

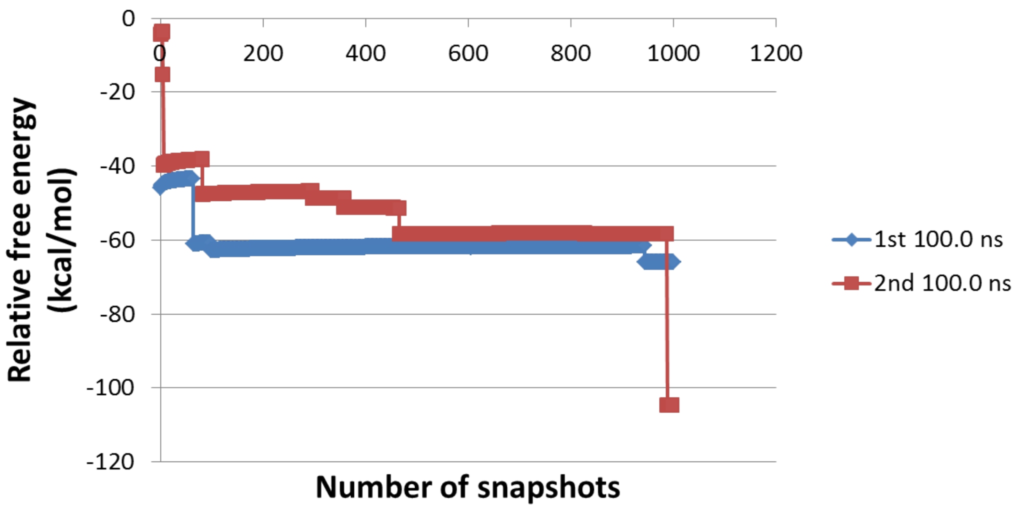


Figure 3.8: Free energy as a function of the number of snapshots included for ethanol in 138 water, subtracting the average energy from energy distributions

Following this, the distribution of the energy differences was examined and found to be Gaussian. This is reinforced when considering the central limit theorem, which states that the sum of a large number of independent distributed variables will be a Gaussian. The combination of the energy differences for the 1^{st} and 2^{nd} 100ns is shown in figure 3.9. Although the distribution fits a Gaussian well, the energy differences show a large range: approximately 138 kcal/mol. When considering that it is the exponential of these energy differences that are calculated, 138 kcal/mol is extremely large. Nevertheless, by introducing a Gaussian to act as a smoothing function, the low-energy tail may be better described and provide convergence of the free energy.

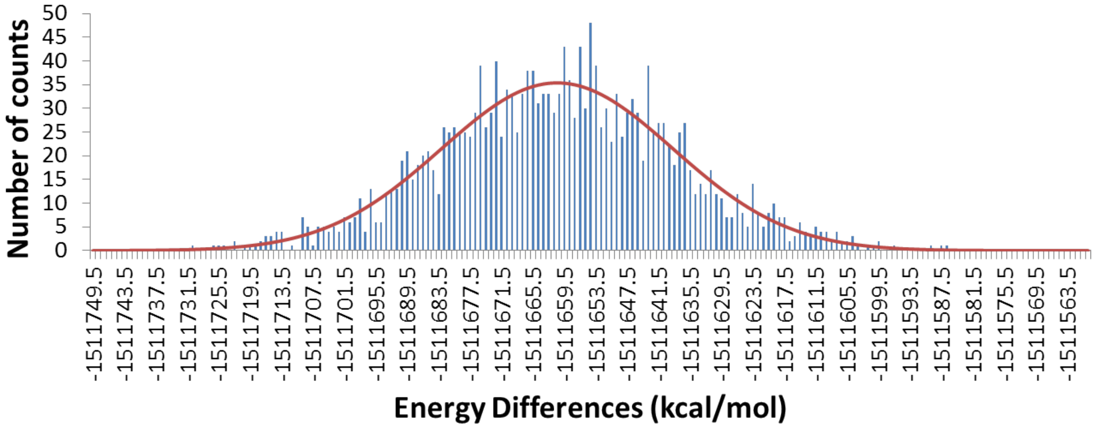


FIGURE 3.9: Energy differences for the full 200 ns of data for ethanol in 138 water

Using a Gaussian distribution to describe the energy differences for both the 1^{st} and 2^{nd} 100ns separately, then applying equation 3.8, the free energy can be found. By using these Gaussian distributions, we approximate how the distribution would look if we could sample infinitely. To do this, the Gaussian is used as the probability density for ΔE, then the following can be used[99],

$$\begin{aligned}\Delta G &= -k_B T \ln \int P(\Delta E) \exp\left[-\frac{\Delta E}{k_B T}\right] dE \\ &= -k_B T \left\langle -\frac{\Delta E}{k_B T} \right\rangle . \end{aligned} \tag{3.8}$$

Where $P(\Delta E)$ is the probability density of ΔE, k_B is the Boltzmann constant and T is the temperature.

In practice, however, the distribution was divided into small segments and the following equation was used along with the trapezoid rule for integration,

$$\Delta G = -k_B T \ln \sum_{i=1}^{n_{Bars}} P(\Delta E_i) \exp\left[-\frac{\Delta E_i}{k_B T}\right] . \tag{3.9}$$

In order to calculate the required exponential of the energy difference for equation 3.9, Berg's [95] formula for the calculation of large exponentials (equation 3.5) must be used, as before in section 3.3.1. In this case, however, by using the rules for dealing with logarithms, $\ln A$ will be the following,

$$\ln A = \ln(P(\Delta E_i)) - \frac{\Delta E_i}{k_B T}. \tag{3.10}$$

When using bin widths of 1 kcal/mol, where "bin" in this case refers to the small segments the Gaussian distribution is divided into, the results can be seen in table 3.9.

TABLE 3.9: Using Gaussian distributions for the 1^{st} and 2^{nd} 100 ns, then integrating over the probability distribution

1000 snapshots Bin width (kcal/mol)	1^{st} 100.0 ns Δ G	2^{nd} 100.0 ns Δ G	Difference ΔΔ G
1	-1512064.46	-1512070.75	6.29

Although the free energy is still not converged, the difference is a great deal smaller, down to 6.29 kcal/mol. This is a promising result, and the application of smoothing functions will be further investigated. For comparison, this method was applied directly to the histogram that was used to generate the Gaussians. The histograms were initially normalised so that the area underneath them was equal to 1, equation 3.5 was then applied using this new count. The results are shown in table 3.10.

TABLE 3.10: Using the histograms for the 1^{st} and 2^{nd} 100 ns

1000 snapshots Bin width (kcal/mol)	1^{st} 100.0 ns Δ G	2^{nd} 100.0 ns Δ G	Difference ΔΔ G
10	-1511729.23	-1511759.23	30.00
1	-1511723.55	-1511762.55	39.00
0.1	-1511722.84	-1511761.63	38.79

As expected without a smoothing function, the free energy does not converge. The functional form of the Gaussian used as the smoothing function was the following,

$$P(\Delta E) = N \exp\left[-\alpha(E - E_0)^2\right]. \tag{3.11}$$

By integration of the following equation (equation 3.13) the analytical solution to the free energy can be obtained.

$$\begin{aligned}\Delta G &= -k_B T \ln \left\langle \exp\left[-\frac{\Delta E}{k_B T}\right]\right\rangle && (3.12)\\ &= -k_B T \ln \frac{\int P(\Delta E)\exp\left[\frac{-\Delta E}{k_B T}\right] d\Delta E}{\int P(\Delta E) d\Delta E} && (3.13)\\ &= -k_B T \ln \frac{I_1}{I_2} && (3.14)\end{aligned}$$

I_2 is the simpler of the two integrals, as such it will be solved first.

$$I_2 = \int_{-\infty}^{\infty} \exp\left[-\alpha(\Delta E - \Delta E_0)^2\right] dE. \qquad (3.15)$$

Setting $y = \sqrt{\alpha}(\Delta E - \Delta E_0)$,

$$\begin{aligned} y &= \sqrt{\alpha}\Delta E - \sqrt{\alpha}\Delta E_0 && (3.16)\\ \frac{dy}{d\Delta E} &= \sqrt{\alpha} && (3.17)\\ d\Delta E &= \frac{1}{\sqrt{\alpha}} dy. && (3.18)\end{aligned}$$

Substituting equation 3.18 into equation 3.15 and using the rule of integrating Gaussians, $\int_{-\infty}^{\infty} \exp[-x^2]dx = \sqrt{\pi}$, leads to,

$$I_2 = \frac{1}{\sqrt{\alpha}} \int_{-\infty}^{\infty} \exp[-y^2] dy = \sqrt{\frac{\pi}{\alpha}}. \qquad (3.19)$$

Now, examining the numerator of equation 3.13,

$$\begin{aligned} I_1 &= \int_{-\infty}^{\infty} \exp\left[-\beta\Delta E\right] \exp\left[-\alpha(\Delta E - \Delta E_0)^2\right] d\Delta E && (3.20) \\ &= \int_{-\infty}^{\infty} \exp\left[-\beta\Delta E - \alpha(\Delta E - \Delta E_0)^2\right] d\Delta E && (3.21) \end{aligned}$$

Then, working on the exponent,

$$\begin{aligned} & -\beta\Delta E - \alpha(\Delta E - \Delta E_0)^2 && (3.22) \\ = & -\left[\beta\Delta E + \alpha\Delta E^2 + \alpha\Delta E_0^2 - 2\alpha\Delta E\Delta E_0\right] && (3.23) \\ = & -\left[\Delta E\left(\beta - 2\alpha\Delta E_0\right) + \alpha\Delta E^2 + \alpha\Delta E_0^2\right] && (3.24) \\ = & -\alpha\left[\Delta E\left(\frac{\beta}{\alpha} - 2\Delta E_0\right) + \Delta E^2 + \Delta E_0^2\right] && (3.25) \\ = & -\alpha[\Delta E^2 + 2\Delta E\left(\frac{\beta}{2\alpha} - \Delta E_0\right) + \left(\frac{\beta}{2\alpha} - \Delta E_0\right)^2 + \Delta E_0^2 \\ & -\left(\frac{\beta}{2\alpha} - \Delta E_0\right)^2] && (3.26) \\ = & -\alpha\left[\left(\Delta E + \left(\frac{\beta}{2\alpha} - \Delta E_0\right)\right)^2 + \Delta E_0 - \left(\frac{\beta}{2\alpha} - \Delta E_0\right)^2\right] && (3.27) \\ = & -\alpha\left[\left(\Delta E + \left(\frac{\beta}{2\alpha} - \Delta E_0\right)\right)^2 - \frac{\beta^2}{4\alpha^2} + \frac{\beta\Delta E_0}{\alpha}\right] && (3.28) \\ = & -\alpha\left(\Delta E + \left(\frac{\beta}{2\alpha} - \Delta E_0\right)\right)^2 - \frac{\beta^2}{4\alpha} + \beta\Delta E_0. && (3.29) \end{aligned}$$

Therefore,

$$\begin{aligned} I_1 &= \int_{-\infty}^{\infty} \exp\left[-\alpha(\Delta E - \Delta E_0)^2 - \beta\Delta E\right] d\Delta E && (3.30) \\ &= \int_{-\infty}^{\infty} \exp\left[-\alpha\left(\Delta E + \left(\frac{\beta}{2\alpha} - \Delta E_0\right)\right)^2 - \frac{\beta^2}{4\alpha} + \beta\Delta E_0\right] d\Delta E && (3.31) \\ &= \exp\left[-\frac{\beta^2}{4\alpha} + \beta\Delta E_0\right] \int_{\infty}^{\infty} \exp\left[-\alpha\left(\Delta E + \left(\frac{\beta}{2\alpha} - \Delta E_0\right)\right)^2\right] d\Delta E. && (3.32) \end{aligned}$$

Then,

$$
\begin{aligned}
x &= \sqrt{\alpha}\left[\Delta E + \left(\frac{\beta}{2\alpha} - \Delta E_0\right)\right] && (3.33)\\
\frac{dx}{d\Delta E} &= \sqrt{\alpha} && (3.34)\\
d\Delta E &= \frac{1}{\sqrt{\alpha}} dx. && (3.35)
\end{aligned}
$$

I_1 is then,

$$
\begin{aligned}
I_1 &= \frac{1}{\sqrt{\alpha}} \exp\left[-\frac{\beta^2}{4\alpha} + \beta\Delta E_0\right] \int_{-\infty}^{\infty} \exp\left[-x^2\right] dx && (3.36)\\
&= \sqrt{\frac{\pi}{\alpha}} \exp\left[-\frac{\beta^2}{4\alpha} + \beta\Delta E_0\right]. && (3.37)
\end{aligned}
$$

Combining I_1 and I_2 gives,

$$
\begin{aligned}
\frac{I_1}{I_2} &= \langle \exp\left[-\beta\Delta E\right]\rangle && (3.38)\\
&= \exp\left[-\frac{\beta^2}{4\alpha} + \beta\Delta E_0\right]. && (3.39)
\end{aligned}
$$

Inserting this back into the Zwanzig equation,

$$
\begin{aligned}
\Delta G &= -\beta^{-1} \ln \langle \exp\left[-\beta\Delta E\right]\rangle && (3.40)\\
&= -\beta^{-1} \ln \frac{I_1}{I_2} && (3.41)\\
&= -\beta^{-1} \ln \left(\exp\left[-\frac{\beta^2}{4\alpha} + \beta\Delta E_0\right]\right) && (3.42)\\
&= -\beta^{-1} \left(\frac{\beta^2}{4\alpha} - \beta\Delta E_0\right) && (3.43)\\
&= \Delta E_0 - \frac{\beta}{4\alpha} && (3.44)
\end{aligned}
$$

The free energy can then be calculated directly from the exponents used from the Gaussian used to fit the distribution.

The exponents fitted when using the full histogram are shown in table 3.11 and the free energies calculated when using these exponents within equation 3.44 are shown in table 3.12. Along with these free energies, are the resulting free energies when the sample size for the histogram is reduced and the Gaussian re-fitted. When lowering the sample size, the snapshots selected were spread evenly throughout the simulation.

Table 3.11: Exponents used for the Gaussian distributions when using the full histograms for the 1^{st} and 2^{nd} 100 ns

coefficient	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
α	0.0010276	0.0010126	0.000015
E_0	-1511661.01	-1511660.96	-0.06

Table 3.12: Free energies obtained when using a Gaussian probability density along with equation 3.44 for the 1^{st} and 2^{nd} 100 ns. The number of snapshots used provides information on how many snapshots were used to make fit the gaussian distribution.

Number of snapshots used	1^{st} 100.0 ns Δ G (kcal/mol)	2^{nd} 100.0 ns Δ G (kcal/mol)	Difference $\Delta\Delta$ G (kcal/mol)
1000	-1512069.09	-1512075.07	5.98
500	-1512087.09	-1512076.93	-10.16
100	-1512232.73	-1512010.76	-221.97

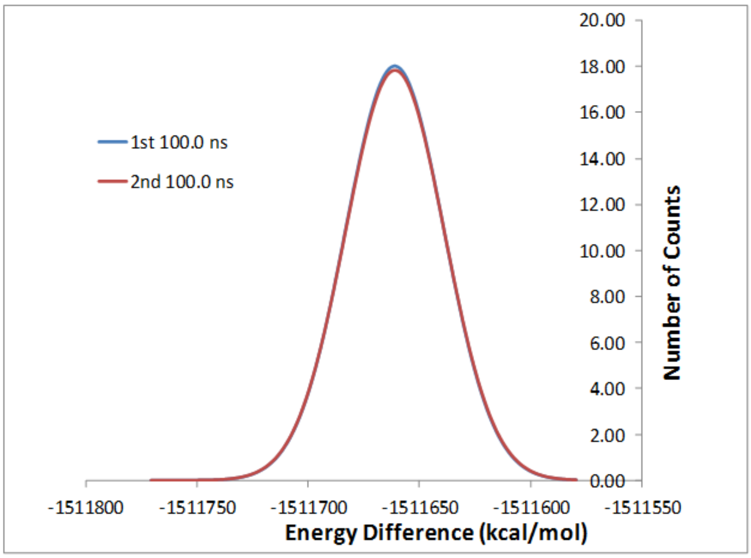


Figure 3.10: The Gaussian distributions for the 1^{st} and 2^{nd} 100 ns

The difference in free energies is surprising when considering the small difference between the exponents of the Gaussian distribution. Figure 3.10 shows the two distributions overlapping to give an indication of how similar these distributions are. These results led to a simple numerical test to find how sensitive these exponents are. First, ΔΔ G can be calculated by,

$$\begin{aligned}
\Delta\Delta G &= \left(E_0^{(1)} - \frac{1}{4\alpha_1 k_B T}\right) - \left(E_0^{(2)} - \frac{1}{4\alpha_2 k_B T}\right) && (3.45)\\
&= \Delta E_0 - \frac{1}{4\alpha_1 k_B T} + \frac{1}{4\alpha_2 k_B T} && (3.46)\\
&= \Delta E_0 - \frac{1}{\alpha_1} + \frac{1}{\alpha_2}. && (3.47)
\end{aligned}$$

By setting ΔE_0 to 0, the α exponents can be tested. The aforementioned numerical test then involved setting α_1 to values ranging from 0.001 to 0.999 in incremental steps of 0.001 and α_2 was calculated to be $\alpha_1 + 1\%$ and $\alpha_1 - 1\%$. The two $\Delta\Delta G$ values obtained were then compared and are shown in figure 3.11.

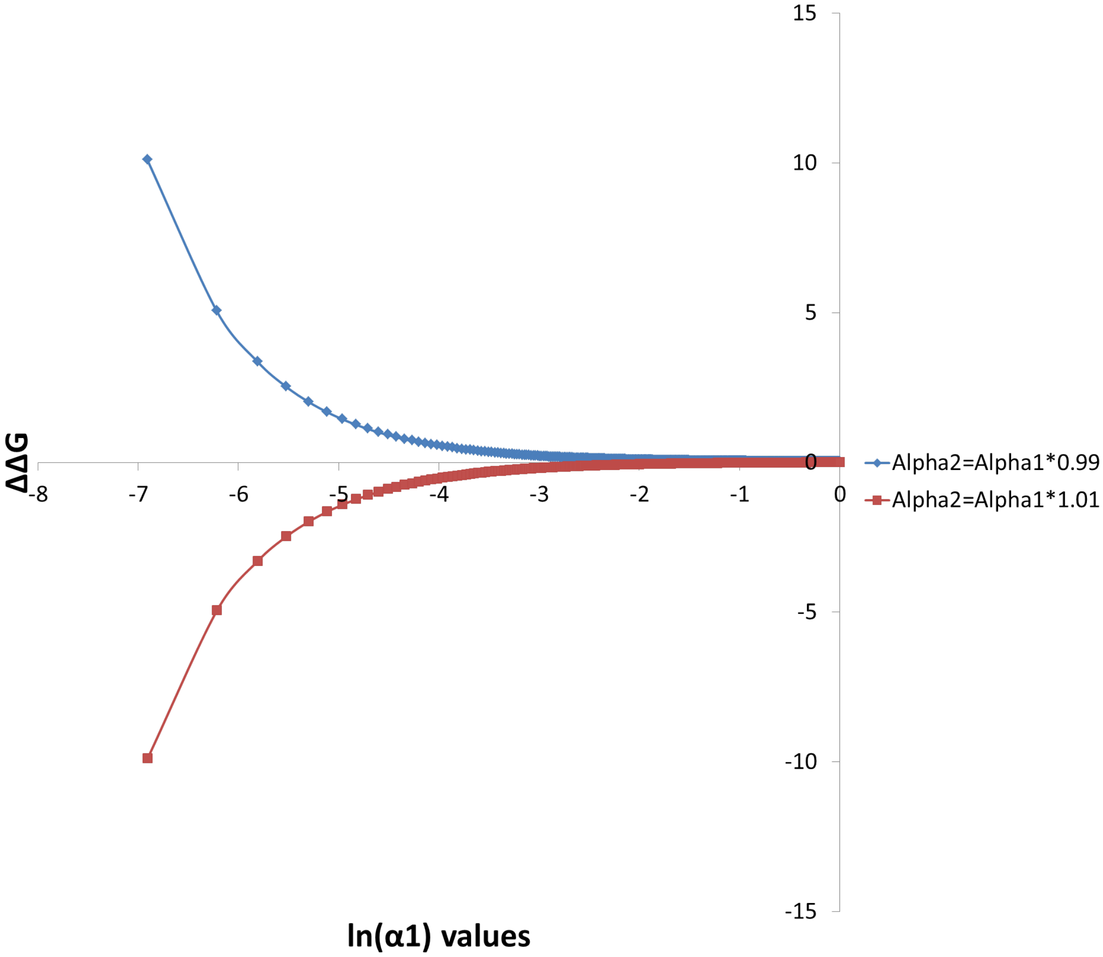


Figure 3.11: The sensitivity of the α exponents used in equation 3.44

It is clear that when α is very small there is little room for any error. However, the relative free energy for the system in table 3.12 shows a reduction from applying the Zwanzig equation directly. The reason for this can be found when considering that the exponential average is not calculated, so the low-energy tail of the distribution is not given unfair weighting. To avoid this exponential averaging issue, the cumulant expansion of the free energy was considered[100]. This is simply the expansion of the Zwanzig equation (equation 3.8) in a Taylor series [7], the derivation follows.

Firstly we must specify the functional form that will be used,

$$
\begin{aligned}
\Delta G &= -\frac{1}{\beta}\ln\langle\exp\left[-\beta\Delta E\right]\rangle & (3.48)\\
&= -\frac{1}{\beta}\ln\left\langle\sum_{l=0}^{\infty}\frac{(-\beta\Delta E)^l}{l!}\right\rangle & (3.49)\\
&= -\frac{1}{\beta}\ln\left(1+\sum_{l=1}^{\infty}\frac{(-\beta)^l}{l!}\left\langle\Delta E^l\right\rangle_0\right). & (3.50)
\end{aligned}
$$

The step from equation 3.49 to 3.50 is performed as $\ln(1+x)$ can be expanded in a Maclaurin series (expansion of a function using a Taylor series at 0).

$$
\begin{aligned}
f(x) &= \sum_{n=0}^{\infty} f^{(n)}(a)\frac{(x-a)^n}{n!} & (3.51)\\
f(a) &= \ln(1+a)\\
f'(a) &= (1+a)^{-1}\\
f''(a) &= -(1+a)^{-2}\\
f'''(a) &= 2(1+a)^{-3}\\
f''''(a) &= -6(1+a)^{-4}\\
&\dots\\
a &= 0\\
f(x) &= \ln(1+0)+(1+0)^{-1}\frac{x}{1!}+-(1+0)^{-2}\frac{x^2}{2!}+2(1+0)^{-3}\frac{x^3}{3!}\\
&-6(1+0)^{-4}\frac{x^4}{4!} & (3.52)\\
&= 0+x-\frac{x^2}{2}+\frac{2x^3}{6}-\frac{6x^4}{24} & (3.53)\\
&= 0+x-\frac{x^2}{2}+\frac{x^3}{3}+\frac{x^4}{4} & (3.54)\\
&= \sum_{k=1}^{\infty}(-1)^{n-1}\frac{x^n}{n} & (3.55)
\end{aligned}
$$

Combining equation 3.50 and 3.55,

$$\Delta G = -\frac{1}{\beta}\sum_{k=1}^{\infty}(-1)^{k-1}\frac{1}{k}\left(\sum_{l=1}^{\infty}\frac{(-\beta)^l}{l!}\left\langle \Delta E_1^l \right\rangle\right) \tag{3.56}$$

Equating 3.56 to the cumulant generating function, equation 3.57, and cancelling a factor of $1/\beta$,

$$\Delta G = \sum_{k=1}^{\infty}\frac{(-\beta)^{k-1}}{k!}\omega_k. \tag{3.57}$$

Extracting powers of β^1

$$\begin{aligned} -\beta\omega_1 &= -\beta\langle\Delta E\rangle_0 \\ \omega_1 &= \langle\Delta E\rangle_0. \end{aligned} \tag{3.58}$$

Next, extracting powers of β^2,

$$\begin{aligned} \frac{(-\beta)^2}{2}\omega_2 &= \left(\frac{(-\beta^2)}{2}\langle\Delta E^2\rangle_0\right) - \frac{1}{2}\left(\frac{(-\beta)}{1}\langle\Delta E\rangle_0\right)^2 && (3.59)\\ &= \frac{(-\beta)^2}{2}\langle\Delta E^2\rangle_0 - \frac{(-\beta)^2}{2}\langle\Delta E\rangle_0^2 && (3.60)\\ \omega_2 &= \langle\Delta E^2\rangle_0 - \langle\Delta E\rangle_0^2 && (3.61)\\ \langle\Delta E^2\rangle_0 - \langle\Delta E\rangle_0^2 &= \langle(\Delta E - \langle\Delta E\rangle_0)^2\rangle_0 && (3.62)\\ &= \langle\Delta E^2 + \langle\Delta E\rangle^2 - 2\Delta E\langle\Delta E\rangle_0\rangle_0 && (3.63)\\ &= \langle\Delta E^2\rangle_0 + \langle\Delta E\rangle_0^2 - 2\langle\Delta E\rangle_0^2 && (3.64)\\ &= \langle\Delta E^2\rangle_0 - \langle\Delta E\rangle_0^2. && (3.65) \end{aligned}$$

Extracting β^3,

$$\begin{aligned}
\frac{-\beta^3}{6}\omega_3 &= -\beta\langle\Delta E\rangle_0 + \frac{(-\beta)^2}{2}\langle\Delta E^2\rangle_0 + \frac{(-\beta)^3}{6}\langle\Delta E^3\rangle_0 \\
&- \frac{1}{2}\left(-\beta\langle\Delta E\rangle_0 + \frac{(-\beta)^2}{2}\langle\Delta E^2\rangle_0 + \frac{(-\beta)^3}{6}\langle\Delta E^3\rangle_0\right)^2 \\
&+ \frac{1}{3}\left(-\beta\langle\Delta E\rangle_0 + \frac{(-\beta)^2}{2}\langle\Delta E^2\rangle_0 + \frac{(-\beta)^3}{6}\langle\Delta E^3\rangle_0\right)^3 \qquad (3.66) \\
&= \frac{(-\beta)^3}{6}\langle\Delta E^3\rangle_0 - \frac{(-\beta)^3}{4}\langle\Delta E^2\rangle_0\langle\Delta E\rangle_0 + \frac{(-\beta)^3}{3}\langle\Delta E\rangle_0^3 \qquad (3.67) \\
\omega_3 &= \langle\Delta E^3\rangle_0 - 3\langle\Delta E^2\rangle_0\langle\Delta E\rangle_0 + 2\langle\Delta E\rangle_0^3. \qquad (3.68)
\end{aligned}$$

Additional powers of β can be extracted in the same manner, however the higher the power, the more complicated the derivation. When calculating the free energy using the cumulant expansion and applying a Gaussian distribution as a smoothing function, the terms naturally truncate after the first 2 cumulants [100]. This should not come as a surprise, examination of the first 2 cumulants, $\langle\Delta E\rangle_0$ and $\langle\Delta E^2\rangle_0 - \langle\Delta E\rangle_0^2$ are equal to the average energy and the variance respectively. With these two pieces of information a Gaussian distribution can be plotted. The third cumulant is equal to the slant of the data, as a Gaussian has no slant, this is equal to 0. Additionally, what should be noticed is that, when applied to a Gaussian, this approach is identical to the analytical approach previously derived,

$$\begin{aligned}
\Delta E_0 &= \langle\Delta E\rangle_0 \qquad (3.69) \\
\sigma^2 &= \frac{1}{2\alpha} \qquad (3.70) \\
\sigma^2 &= \langle\Delta E^2\rangle_0 - \langle\Delta E\rangle_0^2 \qquad (3.71) \\
\Delta G &= \langle\Delta E\rangle_0 - \frac{\beta}{2}(\langle\Delta E^2\rangle_0 - \langle\Delta E\rangle_0^2) \qquad (3.72) \\
&= \Delta E_0 - \frac{\beta}{2}\left(\frac{1}{2\alpha}\right). \qquad (3.73)
\end{aligned}$$

Applying this method directly to the histogram without the use of a smoothing function produced the following results shown in table 3.13.

TABLE 3.13: Applying the cumulant expansion to free energy directly to the histogram, all values in kcal/mol

Cumulant truncation	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
First	-1511660.60	-1511660.94	0.34
Second	-1511245.27	-1511233.63	-11.64
Third	-1511725.47	-1512194.03	468.56

Truncation after the first cumulant shows good convergence between the two simulation runs, however, this is just the average energy difference and cannot be considered a full solution the free energy. Truncating at the second cumulant proves that the free energy convergence is strongly negatively effected by the variance, i.e. the distribution is spread over a large range of data.

In an attempt to lower the effect of the variance on the free energy, a method similar to that of bootstrapping for extracting random samples was used. The process follows,

1. Extract 500 snapshots
2. Fit a Gaussian distribution
3. Repeat 100 times
4. Average the exponents
5. Apply the analytical equation for the free energy (equation 3.44).

It was hoped that by only including 500 snapshots from the data set, that the rare events that dominate the free energy would be selected very infrequently and the effect would be lessened. The results for the average exponents obtained from this approach are shown in table 3.14 and the free energy in table 3.15. The results show a smaller difference in the average for the data, but a worse variance. The variance has the largest effect on the free energy, as such convergence is actually decreased.

As has been shown here and previously described [99], it is the low energy tail of the energy difference distribution that dominates the free energy. In order to avoid this, a cutoff method was implemented. In this approach the mean of the histogram is used as

Table 3.14: Average exponents after a "random samples" approach

Coefficient	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
α	0.0010383	0.0010194	0.0000189
ΔE_0	-1511661.49	-1511661.48	-0.01

Table 3.15: Free energies calculated using "random sampling" approach

	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
Snapshots	ΔG (kcal/mol)	ΔG (kcal/mol)	$\Delta\Delta G$ (kcal/mol)
1000	-1512065.38	-1512072.84	7.46

a center point, then the number of snapshots included within the free energy calculation is controlled by this cutoff. For instance, a cutoff of 10 allows any snapshot higher or lower than the average energy by 10 kcal/mol. A Gaussian is then fitted to this new-shorter histogram and the free energy is calculated using the analytical approach. The average energy has been shown to be consistent between separate simulation runs, so by using increasing amounts of data either side of this average, Gaussian distributions of the same shape may be fitted and the free energy may converge.

Table 3.16: Free energies calculated by applying a cutoff directly to the histogram and fitting a Gaussian, then using the analytical formula. A cutoff of 10 will include data 10 kcal/mol above and below the mean

Cutoff	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
Snapshots	ΔG (kcal/mol)	ΔG (kcal/mol)	$\Delta\Delta G$ (kcal/mol)
200	-1512069.90	-1512075.07	5.98
100	-1512068.19	-1512069.70	1.51
50	-1512292.93	-1512048.03	-244.90
10	-1519177.07	-1546594.99	27417.92

The inclusion of 200 kcal/mol above and below the mean, a total of 400 kcal/mol, was to ensure the method was working. Comparison with table 3.12 shows exactly the same values in the absence of a cut off. The aforementioned cut off is large enough to include all the energy differences within the histogram, so the method can be considered to be working. When changing this cut off the results do not show any consistency, a cut off of 100 kcal/mol provides much better convergence, whereas a smaller cut off, only including the very center of the histogram show much worse convergence. These results are too inconsistent to conclude that this method could provide convergence when using a set cut off.

The issue with only including a smaller data set is that the fitted Gaussian does not have enough information to be an accurate fit. Indeed, this is the same issue that causes a lack of convergence when using the entire histogram, in the event of infinite sampling, the tail of the distribution would be sampled entirely and fitting a Gaussian each time would yield the same exponents, thus converge the free energy. Including more snapshots allows fitting a Gaussian with less error, so this cut off method was modified and applied to the Gaussian distributions fitted on the entire histogram. Gaussian widths were used as the controlling variable for the cut off. The halfwidth of a Gaussian is defined as the width at half the height, so continuations, such as the quarterwidth is the width at quarter the height etc. The free energies were found by numerical integration of the distribution by using equation 3.13.

Table 3.17: Free energies calculated by applying a cutoff to the Gaussian, defined by the halfwidth, and integrating over the remaining distribution

Cutoff	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
Width	ΔG (kcal/mol)	ΔG (kcal/mol)	$\Delta\Delta G$ (kcal/mol)
Half	-1511681.48	-1511681.48	0.00
Quarter	-1511691.86	-1511691.86	0.00
Eighth	-1511699.35	-1511699.36	0.01
Hundredth	-1511719.62	-1511719.65	0.03
Ten thousandth	-1511744.66	-1511744.73	0.07
Infinite	-1512064.46	-1512070.75	6.29

Results shown in table 3.17 highlight which part of the distribution is causing the lack of convergence within the free energies. It is surprising to see that even including the Gaussian from one ten thousandth height, convergence is achieved. However, there is no scientific reason to ignore the data below the ten thousandth height, as such, it cannot be ignored. A study by Hummer et al.[101] applied to calculating electrostatic solvation free energies found that the electrostatic energies of the solute interaction with the solvent can be better described by using multiple Gaussian distributions and selecting the appropriate substate diagnostic parameters. In this same manner, a multistate Gaussian approach was applied to the calculation of the free energies perturbing from

MM to QM. Multiple Gaussian distributions result in multiple low-energy tails and by choosing the substate diagnostic parameters the energy difference outliers could be "smoothed" over.

The probability density for the energy difference is then no longer a single Gaussian but multiple Gaussian distributions, combined by the following,

$$P(\Delta E) = \sum_n c_n P_n(\Delta E). \tag{3.74}$$

The process for generating these new distributions was then simply to generate new histograms from the energy differences, using a substate diagnostic parameter. For example, the QM energy was divided into five groups. These groups were decided upon by by the range of QM energy, e.g. if the range was 10 kcal/mol, each group would be 2 kcal/mol wide. Histograms of the energy difference were then calculated for each of these groups, i.e. an energy difference would count toward the histogram only if the QM energy fell into the range described by the current QM energy group.

It was decided for the energy difference that there could only be two sensible choices for the substate diagnostic parameters: the MM and QM energy. As such, both were used and the results compared. The resulting distributions are shown in figure 3.12, 3.13, 3.14 and 3.15.

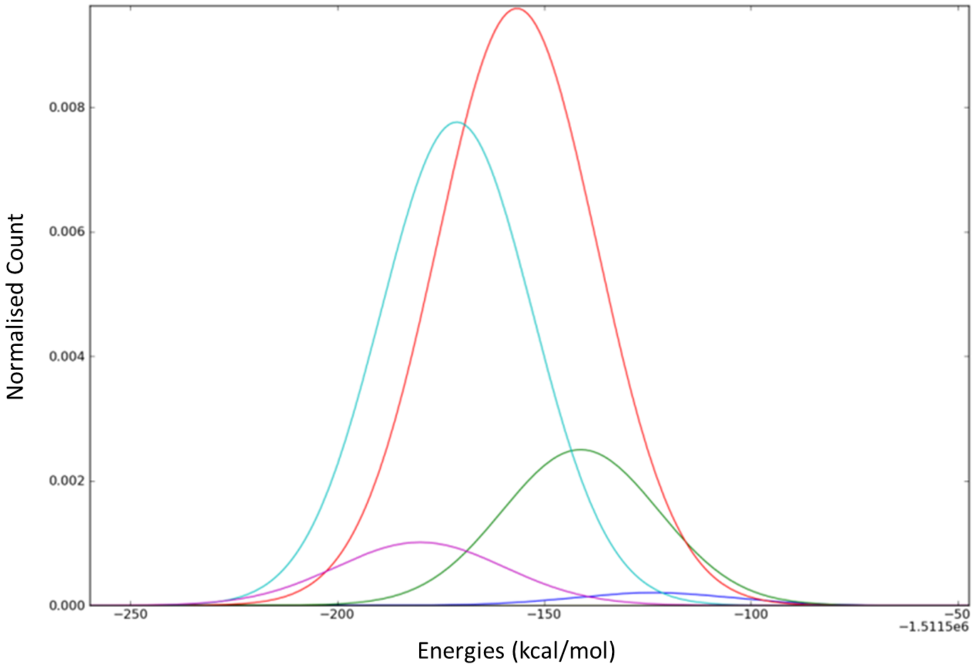


Figure 3.12: Gaussian distributions used to calculate the free energy when using the 1^{st} 100ns MM energies as the substate control parameter

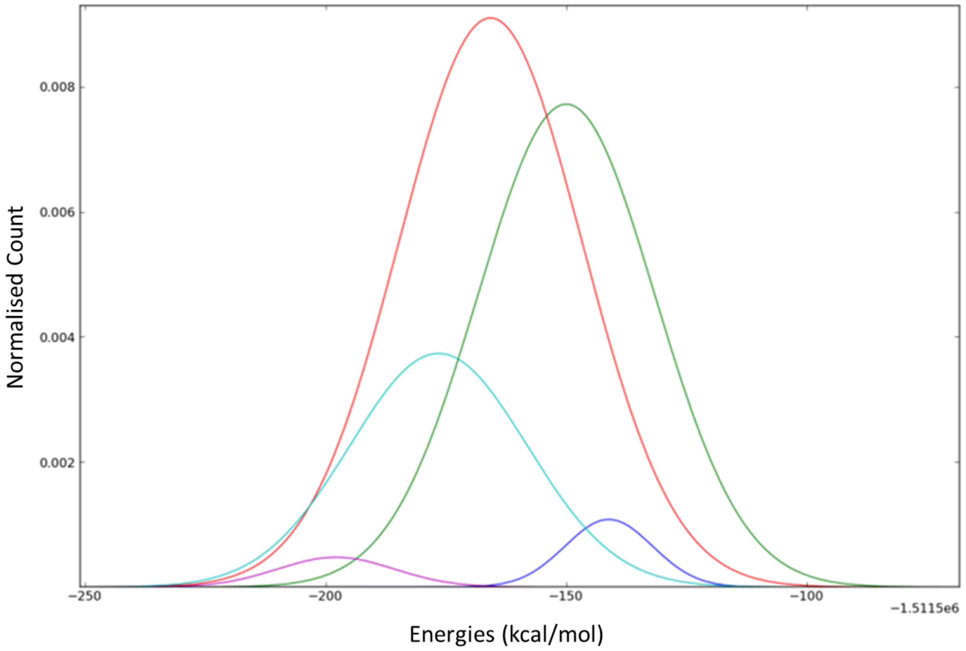


Figure 3.13: Gaussian distributions used to calculate the free energy when using the 2^{nd} 100ns MM energies as the substate control parameter

The line boundaries for the MM data are shown below:

	Minimum boundary	
Line colour	1^{st} 100 ns	2^{nd} 100 ns
Blue	-1631.580	-1602.100
Green	-1593.174	-1567.220
Red	-1554.768	-1532.340
Cyan	-1516.362	-1497.460
Pink	-1477.956	-1462.580
MAX VALUE	-1439.550	-1427.700

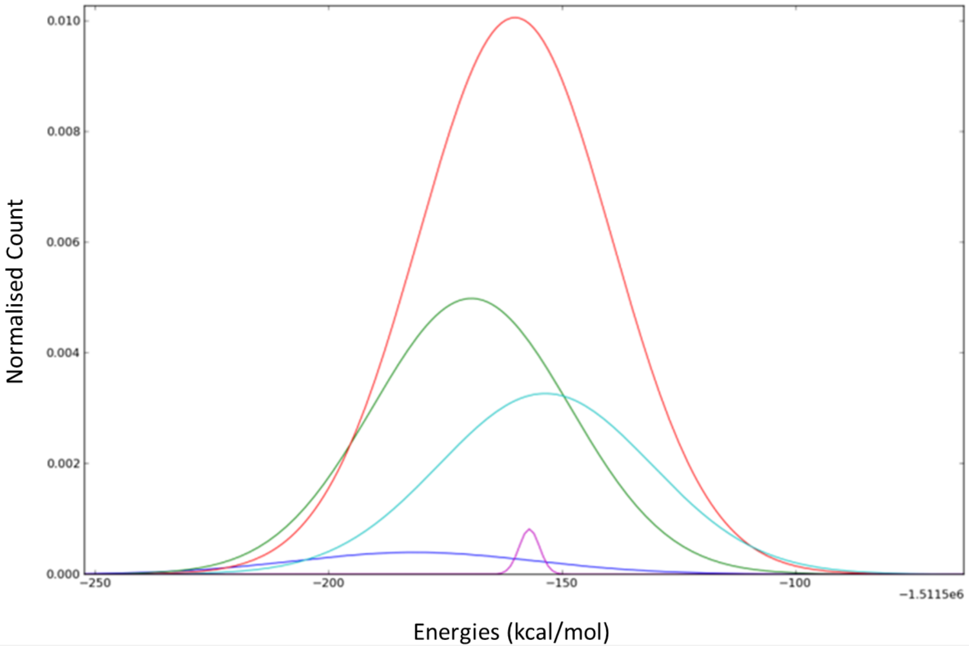


Figure 3.14: Gaussian distributions used to calculate the free energy when using the 1^{st} 100ns QM energies as the substate control parameter

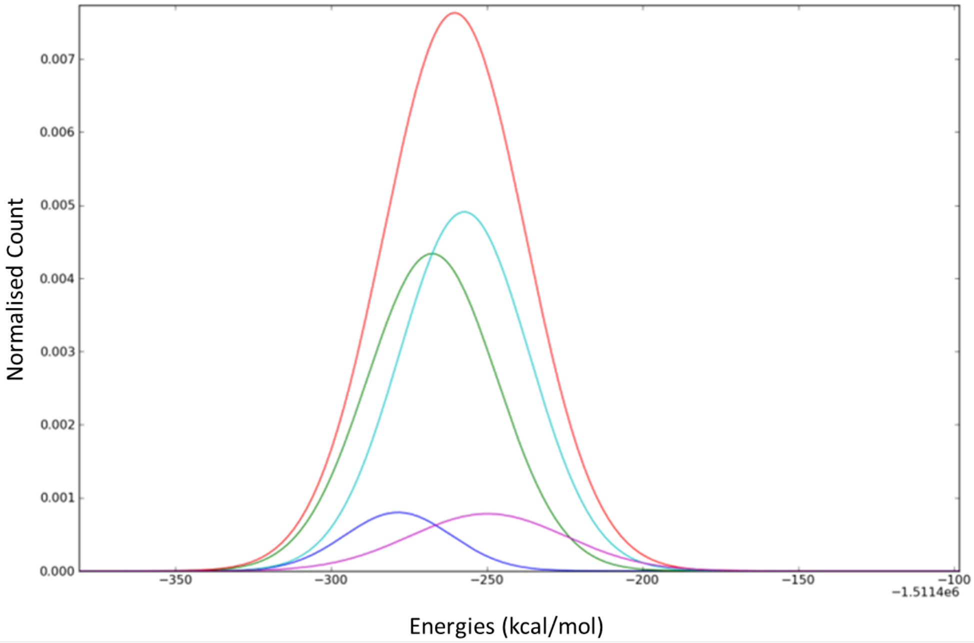


Figure 3.15: Gaussian distributions used to calculate the free energy when using the 2^{nd} 100ns QM energies as the substate control parameter

The line boundaries for the QM data are shown below:

	Minimum boundary	
Line colour	1^{st} 100 ns	2^{nd} 100 ns
Blue	-1513266.870	-1513255.400
Green	-1513231.992	-1513227.316
Red	-1513197.114	-1513199.232
Cyan	-1513162.236	-1513171.148
Pink	-1513127.358	-1513143.064
MAX VALUE	-1513092.480	-1513114.980

The free energy can then be found by applying the following equation,

$$\Delta G = -k_B T \ln \int \sum_n c_n P_n(\Delta E) \exp\left[-\frac{\Delta E}{k_B T}\right] d\Delta E. \tag{3.75}$$

The resultant free energy can be seen in table 3.18.

Table 3.18: Free energies calculated by using multiple Gaussian distributions to describe the probability density, where the energies within the table are the free energies, calculated using equation 3.75.

Substate parameter	1^{st} 100.0 ns	2^{nd} 100.0 ns	Difference
MM energy	-1512019.13	-1511968.77	-50.36
QM energy	-1512221.67	-1512171.68	-49.99

Examination of the multiple Gaussian distributions used as the probability density shows in every case the center Gaussian (red) is the largest and the Gaussians used for the extremities (pink and blue) are the smallest, both of which are indications that there must be some degree of overlap for the configurational space between the MM and QM. If there was no overlap at all, the substate Gaussians would have different proportional sizes. This further enforces the examination of the energy distributions within the box and whisker plots previously, showing that the issue of outliers comes from rare events that are on different sides of the distributions. Several of the substate Gaussians have low energy tails overlapping, this could have better described the low energy energy differences and perhaps have provided better convergence. However, the produced free energies show poor convergence between simulation runs (table 3.18), but very good convergence between the $\Delta\Delta G$ produced by different substate diagnostic parameters.

3.4 Conclusions

The work presented within this chapter has made some progress toward calculating quantum free energies when using classical mechanics to sample structure. Issues in convergence can be largely attributed to the inexact overlap of the configurational space. The effect of changing this overlap was demonstrated when using charge perturbations, it is clear that even a small perturbation can cause convergence problems. However, by using the multistate Gaussians, it has been shown that there is no bias towards either the high energy or low energy structures in the QM distributions, which proves some degree of overlap. The box and whisker plots also defend this conclusion, with snapshots close to the median of the energy difference distribution being found on the same side of the MM and QM energy distributions. Outlier snapshots are caused by rare events when the configurational space does not overlap, however, because of the calculation involved within the Zwanzig equation (equation 3.1), these rare events dominate the free energy.

In the event of infinite sampling, this single step perturbation (SSP) method will work because all rare events will be sampled, however, this is obviously infeasible. An alternative to this is to apply a method that will analyse whether the classically obtained structure is a good "fit" for a quantum ensemble. Woods et al. [94] use an acceptance criterion to converge the free energy when perturbing from MM to QM/MM. The next chapter will apply this method, using QM/MM only as a stepping stone to the full QM.

Chapter 4

Application of Monte Carlo Sampling to Calculate Quantum Binding Free Energies from DFT Total Energies

4.1 Introduction

The previous chapter showed that for systems that do not share a high degree of configurational space overlap, a direct perturbation will not converge. The classical and quantum ensemble face this issue, where certain snapshots that are not energetical outliers in either the quantum or classical ensemble, become energetical outliers in the energy difference ensemble and have a drastic effect on the free energy. In order to overcome this issue, Woods et al. [94] applied an acceptance criterion that allowed them to use MM to generate structures that were statistically correct for a QM/MM ensemble. The application of this method employs a Monte Carlo (MC) technique to generate MM structures and a Metropolis Hastings MC criterion to accept or reject these structures to the QM/MM ensemble.

An alternative approach for generating a correct quantum ensemble would be to use the Hybrid Monte Carlo (HMC) technique [102] which is based on MD and therefore allows larger moves between acceptance tests and does not suffer random walk errors associated with standard Monte Carlo techniques [103, 104]. In this chapter we propose a HMC based approach for correcting classical free energies with quantum techniques. To do this, we use HMC to generate from MM an ensemble of QM/MM structures that is a much closer representation of the fully quantum ensemble. By doing this, the errors associated with sampling unfavourable structures are significantly lessened and also states intermediate between MM and QM/MM are generated which allow us to compute the MM to QM/MM change in free energy using thermodynamic integration (TI) which has stable convergence. We can then apply a single step free energy perturbation from our generated QM/MM ensemble to the fully quantum ensemble.

This stepwise approach to obtain the quantum corrected free energies is possible because free energy is a state function. Within this chapter we have chosen to move initially to a QM/MM ensemble as this should share a higher degree of configurational space overlap with the full QM ensemble. If this proves to be the case, then a direct perturbation from the QM/MM accepted structures to the full QM will converge.

In section 4.2 we describe the theory behind this method, then section 4.3 details which programs have been used. Finally, section 4.4 presents the results for systems with increasing complexity.

4.2 Theory

The method presented here uses HMC to generate a QM/MM ensemble of structures, using MD as the underlying engine then applying an acceptance test to check the validity of the structures in the desired (QM/MM) potential.

4.2.1 Hybrid Monte Carlo

In order to apply the Metropolis-Hastings criterion when perturbing from MM to QM/MM, the generation of MM structures must obey the detailed balance condition (equation 4.1). This can be satisfied by using pure Monte Carlo or by Hybrid Monte Carlo [102]. By using Hybrid Monte Carlo, the random walk errors associated with Monte Carlo are avoided and the conformational changes between $\mathbf{R}$ and $\mathbf{R}'$ can be larger [103]. The detailed balance can be seen here,

$$\rho(\mathbf{R})\pi(\mathbf{R} \rightarrow \mathbf{R}') = \rho(\mathbf{R}')\pi(\mathbf{R}' \rightarrow \mathbf{R}). \tag{4.1}$$

Where $\rho(\mathbf{R})$ is the probability of occupying a certain conformation $\mathbf{R}$, from here onwards this will be referred to as state probability. $\pi(\mathbf{R} \rightarrow \mathbf{R}')$ is the transition probability shown in equation 4.2, which consists of the trial probability and the acceptance probability.

$$\pi(\mathbf{R} \rightarrow \mathbf{R}') = \pi_{acc}(\mathbf{R} \rightarrow \mathbf{R}')t(\mathbf{R} \rightarrow \mathbf{R}'), \tag{4.2}$$

$\pi_{acc}(\mathbf{R} \rightarrow \mathbf{R}')$ is the acceptance probability, given by a Metropolis-Hastings criterion, and can be seen in equation 4.3, and $t(\mathbf{R} \rightarrow \mathbf{R}')$ is the trial probability, which is the probability of moving to a new conformation.

$$\pi_{acc}(\mathbf{R} \rightarrow \mathbf{R}') = min\left\{1, \frac{\rho(\mathbf{R}')t(\mathbf{R}' \rightarrow \mathbf{R})}{\rho(\mathbf{R})t(\mathbf{R} \rightarrow \mathbf{R}')}\right\} \tag{4.3}$$

The generation of new conformations in hybrid Monte Carlo is performed by an underlying MD simulation. In order to satisfy detailed balance the MD simulation here is in the microcanonical ensemble. To run a simulation in the microcanonical ensemble, only two inputs are required: the initial structure and velocities. The acceptance criterion shown in equation 4.3 also requires two values and their complimentary terms. One is the state probability of the original conformation (the complimentary in this case is the state probability of the new conformation) and the other is the trial probability. The state probability is given by the following Boltzmann distribution,

$$\rho(\mathbf{R}) = \frac{\exp(-\beta U(\mathbf{R}))}{Z(N,V,T)}. \tag{4.4}$$

Where $Z(N,V,T)$ is the configurational partition function.

The trial probability required for the acceptance criterion is related to the selected momenta. To ensure a Boltzmann distribution of kinetic energy, the momenta are randomly selected from a distribution using the Marsaglia Polar method [105]. The Marsaglia Polar method selects random numbers on a normal distribution by a simple process; First two pseudo random numbers μ_1 and μ_2 are generated between -1 and 1, such that $\mu_3 = \mu_1^2 + \mu_2^2 < 1$. The random numbers are then generated by the following,

$$X = \mu_1 \left[-2\frac{\ln(\mu_3)}{\mu_3}\right]^{1/2} \tag{4.5}$$

$$Y = \mu_2 \left[-2\frac{\ln(\mu_3)}{\mu_3}\right]^{1/2}. \tag{4.6}$$

Finally, the random numbers X and Y are multiplied by the required standard deviation. By selecting momentum, the temperature of the simulation can be controlled, thus acting like a thermostat.

$$t(\mathbf{R} \to \mathbf{R}') \propto \prod_{i=1}^{n} \frac{1}{\sqrt{2m_i \pi k_B T}} \exp\left[-\beta \frac{\mathbf{p}_i^2}{2m_i}\right] \tag{4.7}$$

To derive a useful form of the acceptance criterion we place equation 4.4 and 4.7 into equation 4.3, noticing that equation 4.7 is the functional form of kinetic energy, we obtain,

$$
\begin{aligned}
\pi_{acc}(\mathbf{R} \rightarrow \mathbf{R}') &= min\left\{1, \frac{\exp(-\beta U(\mathbf{R}'))\exp(-\beta K(\mathbf{P}'))}{\exp(-\beta U(\mathbf{R}))\exp(-\beta K(\mathbf{P}))}\right\} \\
&= min\left\{1, \frac{\exp(-\beta H(\mathbf{R}', \mathbf{P}'))}{\exp(-\beta H(\mathbf{R}, \mathbf{P}))}\right\} \\
&= min\left\{1, \exp(-\beta(H(\mathbf{R}', \mathbf{P}') - H(\mathbf{R}, \mathbf{P})))\right\} \\
&= min\left\{1, \exp(-\beta \Delta H(\mathbf{R}, \mathbf{P}))\right\}. \qquad (4.8)
\end{aligned}
$$

When applying this method to generate a classical NVT ensemble of structures the acceptance should be 100%. This is because the underlying MD simulation is performed in the NVE ensemble, as such the total energy $H(\mathbf{R}, \mathbf{P})$ should be constant, and the difference between the initial and final total energies will equal 0. So if the acceptance is below 100% this is due to errors in the MD integrator, which can be minimised by reducing the timestep, i.e. the smaller the timestep the better energy conservation. This method will produce an accurate classical canonical ensemble of structures.

The next section deals with the case when we are moving to a QM/MM NVT ensemble.

4.2.2 Accepting to the QM/MM ensemble

The QM/MM model used within this method was initially the ONIOM approach (mechanical embedding)[106] later electrostatic embedding was used. ONIOM provides a simple QM/MM description of the system by the following,

$$
U^{QM/MM}_{complex} = U^{MM}_{complex} - U^{MM}_{region} + U^{QM}_{region}. \qquad (4.9)
$$

Where $U^{MM}_{complex}$ is the total complex energy of the system when calculated using a classical potential and U^{QM}_{region} is the quantum potential energy of the region of interest (e.g. a binding pocket and ligand).

Every time the acceptance test is applied we build up the QM/MM ensemble by an additional structure. In practice the acceptance in equation 4.8 is applied by taking the initial kinetic energy directly from the generated momentum, the initial potential energy is the QM/MM energy of the initial structure. The final kinetic energy is taken directly from the MD simulation and the final potential energy is the QM/MM energy of the final structure. So the classical potential energy is not used at all within the QM/MM acceptance. If a structure is accepted, then it is used within the QM/MM ensemble. However, if a structure is rejected, the last accepted structure counts again in the QM/MM ensemble.

This will produce a QM/MM ensemble of structures at the selected temperature, in this case 300K.

4.2.3 Stratification

In order to make the transition from MM to QM/MM smoothly, additional λ steps can be introduced. To do this, a simulation is performed in an identical fashion as previously described, with the only exception within the acceptance test (equation 4.8). Where the potential energy $U(\mathbf{r})$ becomes $U(\mathbf{r}) = (1-\lambda)U_{classical} + \lambda U_{QM/MM}$.

The free energy is then obtained using thermodynamic integration (TI),

$$\begin{aligned}
\Delta G &= \int_0^1 d\lambda \left\langle \frac{\partial U(\lambda)}{\partial \lambda} \right\rangle_\lambda \\
\frac{\partial U(\lambda)}{\partial \lambda} &= U_{QM/MM} - U_{classical} \\
\Delta G &= \int_0^1 d\lambda \left\langle U_{QM/MM} - U_{classical} \right\rangle_\lambda
\end{aligned}$$

4.2.4 Transitioning to the full QM

If we make the assumption that the $U_{QM/MM}$ ensemble is an accurate representation of the U_{QM} ensemble, the transition from QM/MM to QM is simple. To perform this perturbation, the energy differences $U_{QM} - U_{QM/MM}$ are calculated then the Zwanzig equation can be used as shown in equation 4.10.

$$\Delta G = -k_B T \ln \left\langle \exp\left[-\beta \Delta U_{QM-QM/MM}\right] \right\rangle_{QM/MM} . \tag{4.10}$$

4.3 Methods

4.3.1 Classical details

All parameters for non-waters, unless otherwise specified, were obtained using antechamber [97]. The equilibration procedure was undertaken within Gromacs v4.6.2 [107] using the leapfrog integrator and is as follows. For ethane and ethanol in 448 and 450 waters respectively and alanine dipeptide in vacuum, structures were initially minimised with the steepest descent algorithm for 5000 steps with a cutoff of 11 Å, long range electrostatics were treated with PME and a cut-off method was used for Van der Waals interactions. Following this, a 500 ps simulation in the canonical ensemble was run using a timestep of 1 fs, the Berendsen thermostat was used to heat the system smoothly to 300 K, finally a 1 ns isothermal-isobaric simulation was performed using the Berendsen barostat. After equilibration the box size decreased such that only a maximum of an 8 Å cut-off could to be used for the non-bonded interactions.

The process for N_2 in vacuum was similar, although the pressure equilibration was skipped and a cutoff method was used for long range electrostatics. N_2 in varying solvents (Ar, CH_4 and TIP3P water) was equilibrated following the above procedure, although a cutoff method was used for long range electrostatics within Ar and the parameters for Ar were taken from White (1999) [108].

The equilibration for cyclodextrin for the minimisation and isothermal-isobarric equilibration followed the above process, the simulation within the canonical ensemble differed slightly, in that the system was heated over 500ps and then allowed to run at 300K for an additional 1.5 ns.

The classical simulations within hybrid Monte Carlo were performed using the velocity-verlet algorithm, with a 0.25 fs timestep, the reason for such a small timestep can be seen in figure 4.1, which shows the effect of changing the timestep for an MD simulation that was performed 3 times. The overall energy is conserved, but the fluctuations cause a lower classical acceptance rate when any larger timestep is used. Each simulation is run for 20 ps and then an acceptance test is applied.

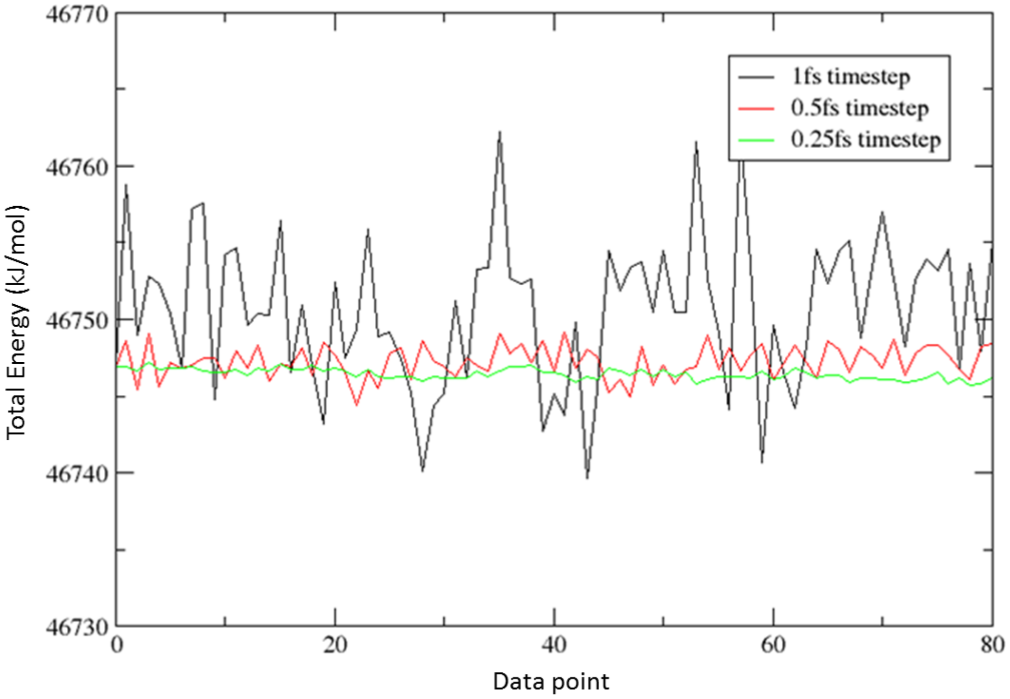


FIGURE 4.1: Running an MD simulation using different timesteps to show fluctuations of the total energy in the microcanonical ensemble

4.3.2 QM details

The quantum energies were calculated using ONETEP [54]. The first 100 applications of the acceptance test (steps) were discounted and used as equilibration. The setup for ONETEP, in the case of N_2, was that 4 NGWFs were used with a 7.0 a_0 localisation radius. For ethane, ethanol, alanine dipeptide and cyclodextrin 4 NGWFs were used on heavy atoms and 1 for light atoms, all were 8 a_0 and in both cases a kinetic energy cutoff of 800eV was used.

4.4 Results

4.4.1 Applying HMC to N_2 while Modifying the Equilibrium Bond Length

A simple test system of N_2 in vacuum was selected to show the accuracy of the generated QM ensemble, when applying the Metropolis-Hastings criterion in comparison to the direct method of extracting snapshots from an MD trajectory at regular intervals and applying a single step free energy perturbation to the QM level (in the same manner shown in the previous chapter). The advantage of such a simple test system is that there is no need to define a classical region, meaning that we accept into a purely quantum ensemble.

For simplicity, the snapshots required for the single step perturbation approach (SSP) were taken from the ensemble when $\lambda = 0$ of the Metropolis-Hastings applied method (MHA). These snapshots accurately represent the classical canonical ensemble.

All parameters were obtained for N_2 from antechamber [97] with the exception of the equilibrium bond length which was set to the experimental length of 1.098 Å [109]. The bond length within the force field was then increased in increments of 0.01 Å and the effect on the proposed quantum ensemble was observed. The results can be seen in table 4.1.

1.098	SSP	MHA	
+	Average Bond Length	Average Bond Length	Acceptance at λ=1
0.00	1.098	1.104	62.8
0.01	1.109	1.106	62.0
0.02	1.121	1.107	54.2
0.03	1.132	1.108	42.6
0.04	1.140	1.109	33.2

TABLE 4.1: The effect on the average bond length (shown in Å) of the quantum ensemble when changing the equilibrium bond length within the force field. The acceptance rate is shown as a percentage and is for λ=1

As expected, taking snapshots directly from an MD trajectory when the force field is parameterised badly, produces a poor representation of a quantum ensemble. This can be seen by examining the average bond lengths of Table 4.1. However, when structures have to go through an acceptance criterion, as in MHA, the average bond length is conserved close to the experimental value. By forcing the classical and quantum configurational space to overlap less, the acceptance at $\lambda = 1$ is drastically effected, in some cases no structures were accepted post-equilibration. This shows the acceptance rate is a good way to gauge how well the force field represents the quantum region: if the acceptance rate is very low then the force field is badly paramterised. In a simple system such as this, identifying which parameter is the main cause of the poor acceptance rate is trivial, however, when moving to larger systems this will be highly non-trivial. So, although the issue is identified the solution remains difficult to obtain.

The free energies of moving from the MM potential to the QM potential were then calculated and are shown in figure 4.2.

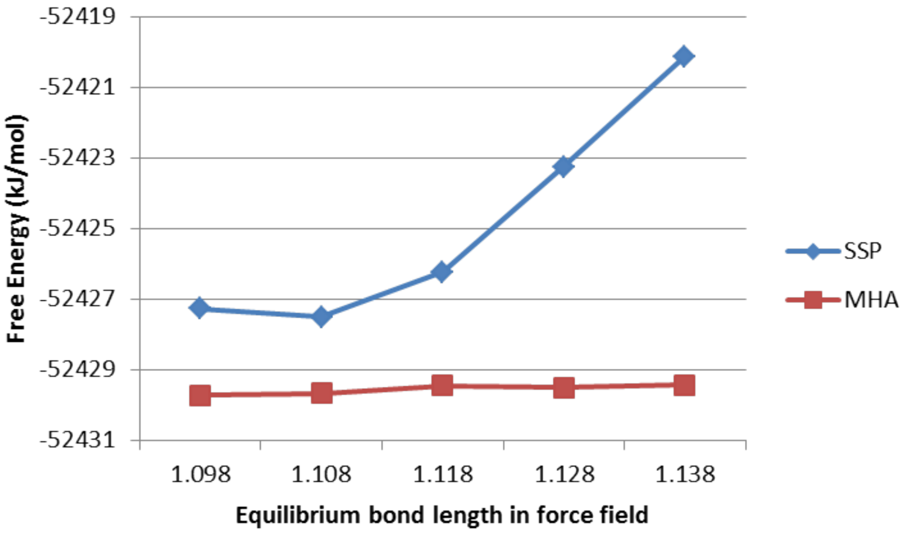


Figure 4.2: Comparison of free energy calculated when using the SSP and MHA methods while changing the classical equilibrium bond length of N_2

It is apparent by examination of figure 4.2 that the free energies diverge from each other as the equilibrium bond length is increased in the classical potential. The explanation for this is found when considering the potential of mean force (PMF) plots produced when using λ windows. These are shown in figure 4.3.

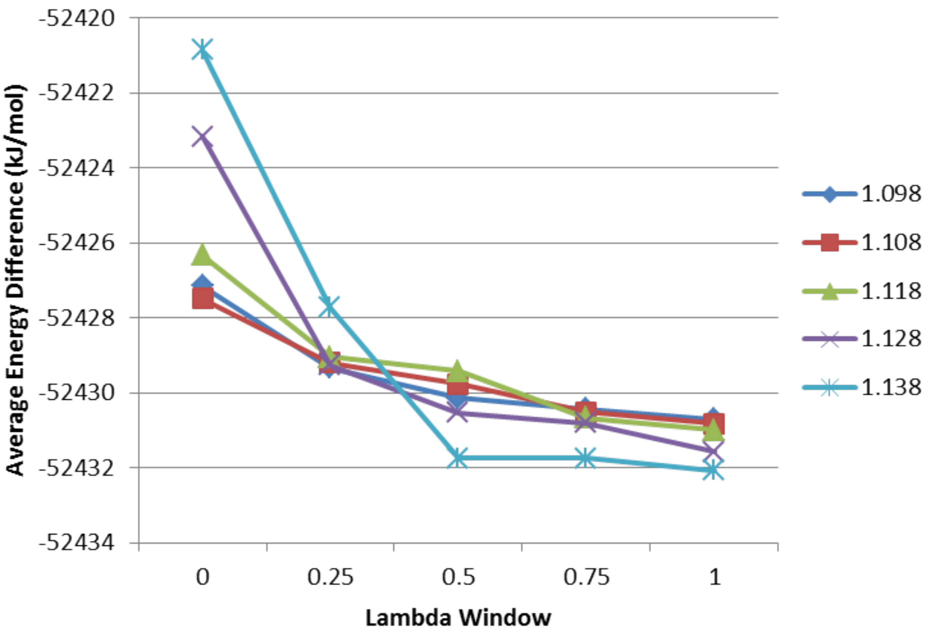


FIGURE 4.3: PMFs when moving from MM ($\lambda = 0$) to QM ($\lambda = 1$) for N_2 when altering the classical equilibrium bond length

The Zwanzig equation (equation 4.10) assumes the PMF will be a straight line between the MM ($\lambda = 0$) and QM ($\lambda = 1$). This assumption proves to be catastrophic to the free energies as the force field parameters move from the ideal values. These results provide us with confidence that the MHA method can improve upon the results obtained when using the SSP method.

Previously we showed convergence issues when perturbing from QM/MM to QM. In order to identify the cause of these convergence issues, N_2 in solvents of increasing complexity was used. These solvents were Ar, CH_4 and TIP3P water.

4.4.2 N_2 in Ar

As mentioned above, the first test is the simplest as there is no electrostatic or polarisation terms required to model the 250 Ar atoms. Because of this the classical potential should provide an accurate description of the ensemble. The first step to calculate the free energies is to generate ensembles at different λ values between the MM and QM/MM descriptions for the system. The resultant PMFs in figure 4.4 shows excellent convergence between the two simulation runs, the difference between the free energies for these runs was 0.09 kJ/mol.

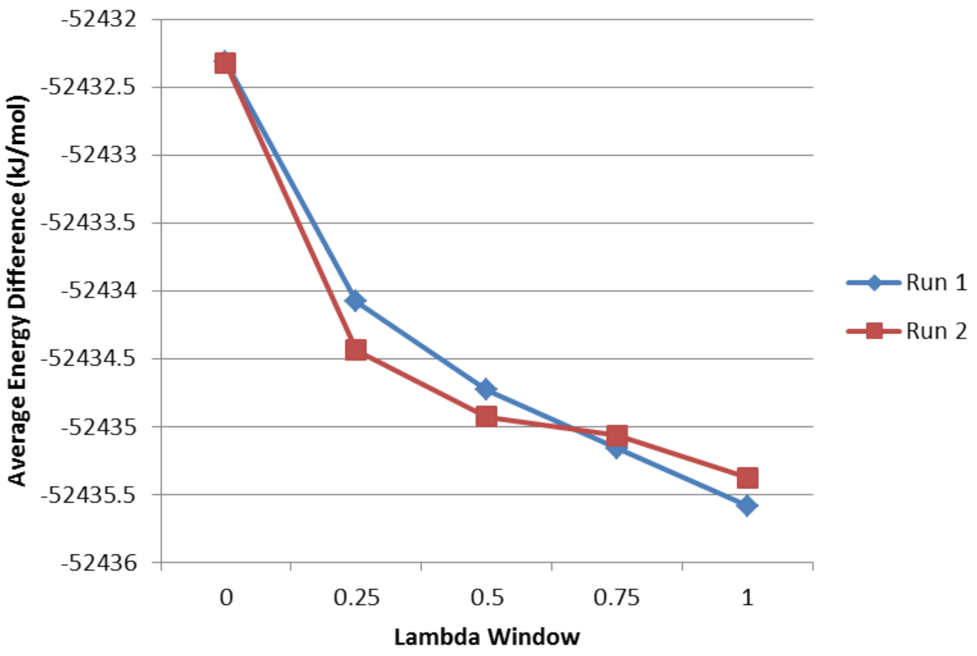


FIGURE 4.4: PMFs when moving from MM ($\lambda = 0$) to QM/MM ($\lambda = 1$) for N_2 in 250 Ar

Using the structures that were accepted at $\lambda = 1$ (the QM/MM ensemble), the fully QM energy was calculated on the entire system. Then a single step perturbation was performed, the produced free energy was then added to the free energy required to move from an MM description to a QM/MM description of the system. The resultant values can be seen in Table 4.2.

This small difference within the free energies is within thermal error. Further analysis of the free energy calculated between the QM/MM and QM was performed by increasing

	Free Energy (kJ/mol)
Run 1	-2297316.80
Run 2	-2297315.90
Difference	1.10

Table 4.2: Free energy when moving from MM to QM for N_2 in 250 Ar

the number of snapshots included within the Zwanzig equation in increments of 1. The produced graph will show if a single or a few snapshots are dominating the free energy.

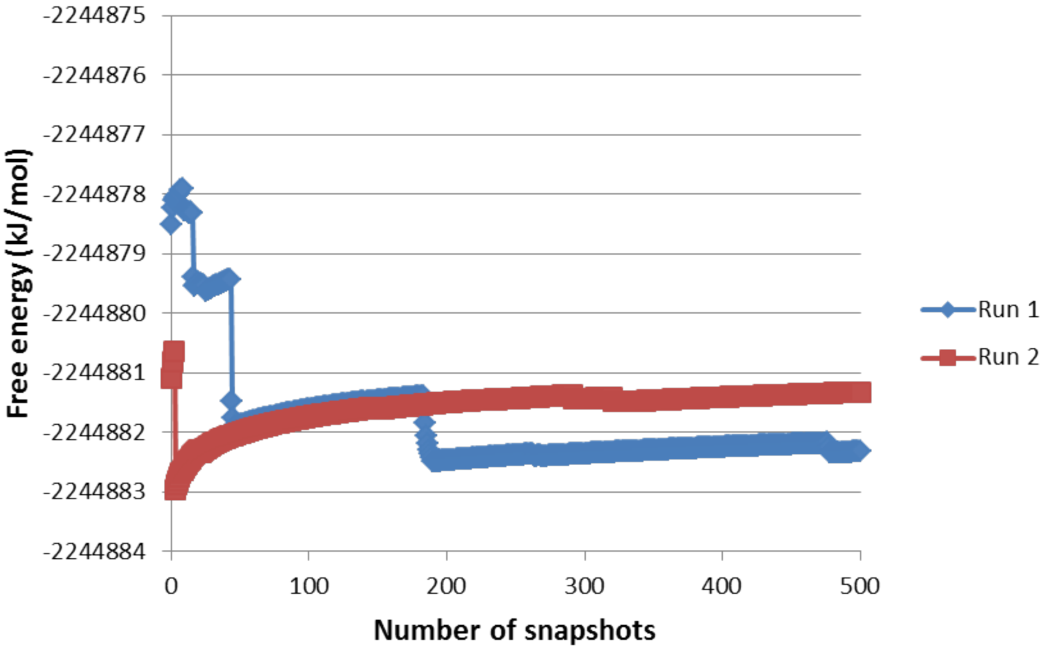


Figure 4.5: Increasing the number of snapshots included in the Zwanzig equation when perturbing from QM/MM to QM

The results shown in figure 4.5 display a “sawtooth” shape which is characteristic of poor sampling [99]. However, the free energy for the two runs is similar because the changes within the free energy are small, which can be largely attributed to the simplicity of this system. By increasing the complexity of the system, it is predicted that this “sawtooth” sampling will have a much larger effect on the free energy.

4.4.3 N_2 in CH_4

By introducing the solvent of 250 CH_4, there are now small partial charges and the complexity of the system is increased. The intermolecular interactions are now no longer solely dispersion driven. As before, first the free energy moving from an MM description to a QM/MM description was found.

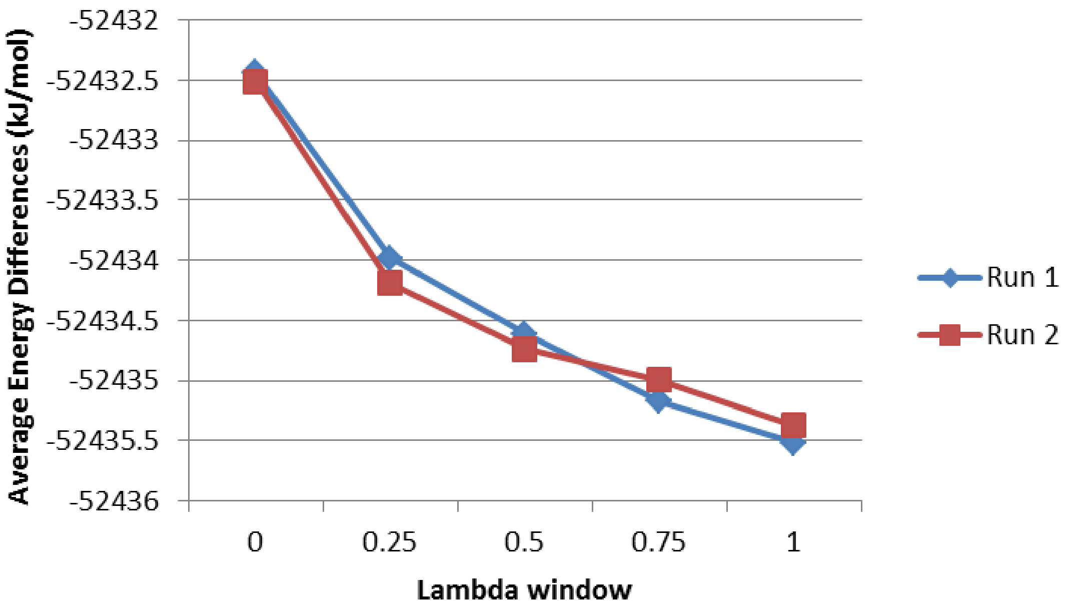


FIGURE 4.6: Free energy when moving from MM to QM for N_2 in 250 CH_4

Figure 4.6 shows the two PMFs are almost identical and this is reflected in the 0.03 kJ/mol free energy difference between them.

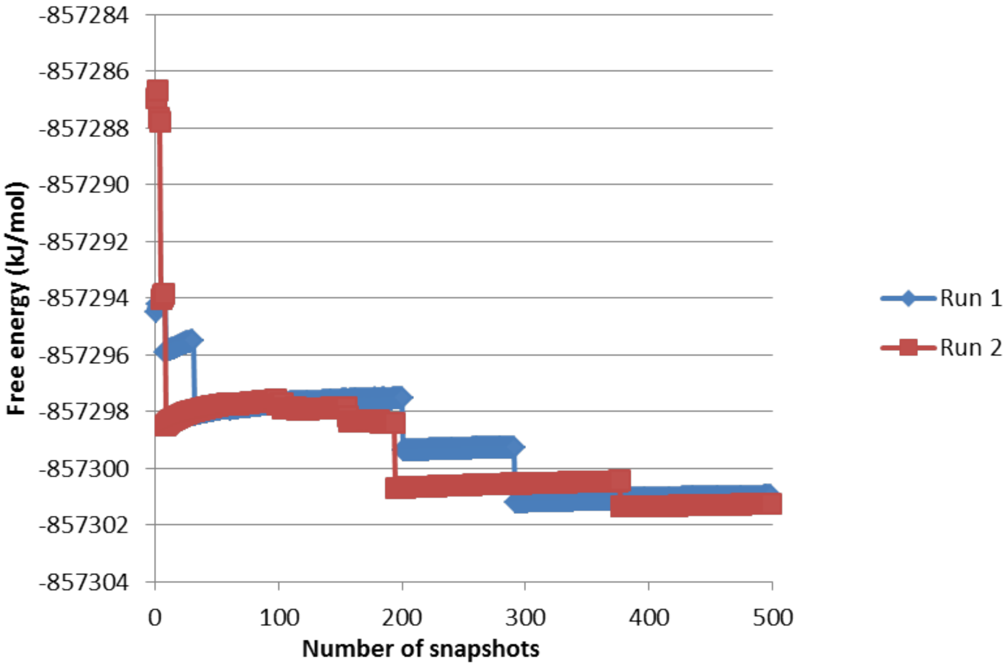


Figure 4.7: Increasing the number of snapshots included in the Zwanzig equation when perturbing from QM/MM to QM for N_2 in CH_4

Performing the same analysis as before: increasing the number of snapshots included within the Zwanzig equation, figure 4.7 was produced. The "sawtooth" shape is again present, but the final snapshots show no large changes in either simulation runs. The final difference in the free energy when moving from MM to QM can be seen in Table 4.3. This excellent agreement shows that the way the classical force field has modelled the intermolecular interactions here agrees well with the quantum description.

	Free Energy (kJ/mol)
Run 1	-909735.43
Run 2	-909735.72
Difference	0.29

Table 4.3: Free energy when moving from MM to QM for N_2 in 250 CH_4

4.4.4 N_2 in TIP3P Water

The final test system is the most complex and most interesting system. By introducing a higher partial charge, electrostatics will be the main driving force for the intermolecular interactions and polarisation effects will be present, although it should be noted that the charge on nitrogen is 0, but the intermolecular interactions of the solvent will be driven primarily by electrostatics. Polarisation is included implicitly within the force field [85] and, as this is mechanical embedding QM/MM, it will not be modelled explicitly by the quantum calculation. To keep the box size similar and using a realistic density, 499 waters were used. The PMFs generated by moving from a classical to a QM/MM description are shown in figure 4.8. The convergence between the simulation runs was 0.12 kJ/mol.

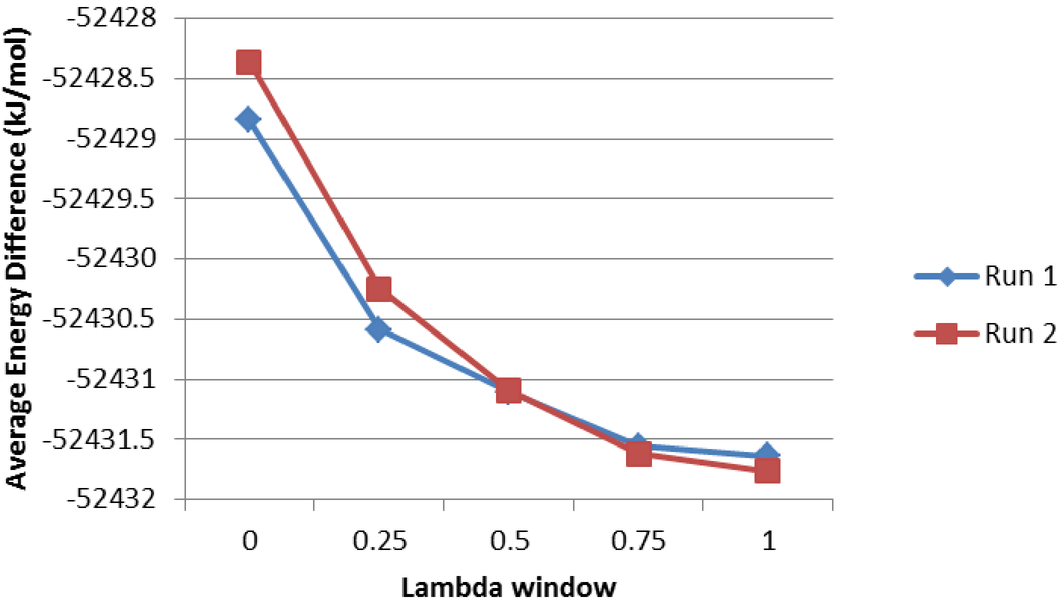


FIGURE 4.8: Free energy when moving from MM to QM/MM for N_2 in 499 water

Next, the free energy between the QM/MM and full QM was calculated. As before this was performed is a stepwise manner to ensure convergence, the results can be seen in figure 4.9. Again the “sawtooth” shape is present, and the two runs have not converged to a single value. Also notice that the “jumps” in the graph are larger than in any other solvent, which implicates that the electrostatics have the largest effect on the convergence. If this is the case then using a QM/MM description that uses electrostatic embedding instead of mechanical embedding should improve the quality of the ensemble at the QM/MM level.

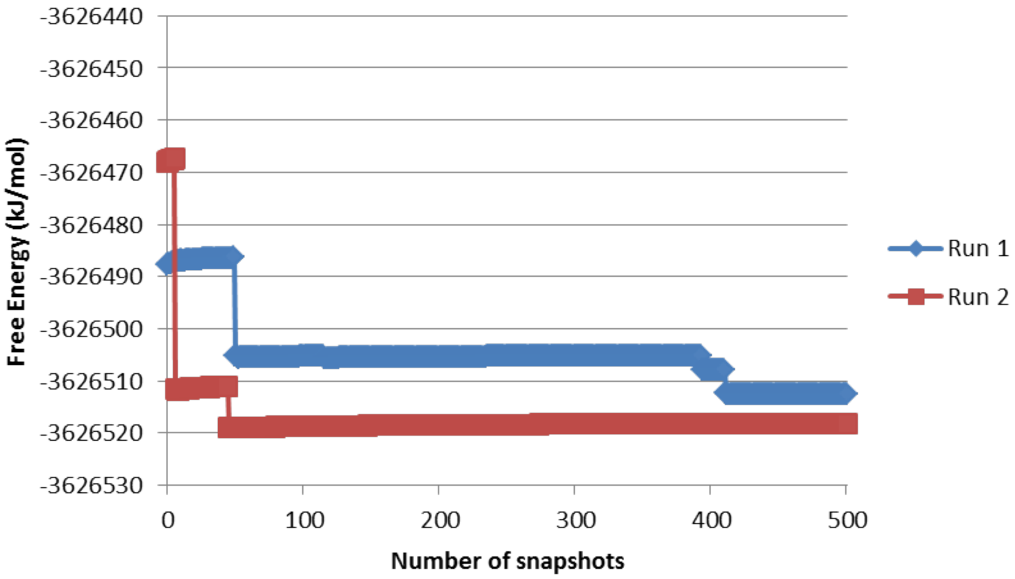


FIGURE 4.9: Increasing the number of snapshots included in the Zwanzig equation when perturbing from QM/MM to QM for N_2 in TIP3P

The total free energy moving from MM to QM can be seen in Table 4.4.

	Free Energy (kJ/mol)
Run 1	-3678943.45
Run 2	-3678948.83
Difference	5.38

TABLE 4.4: Free energy when moving from MM to QM for N_2 in 250 CH_4

Regardless of the solvent used convergence was achieved to the QM/MM level. The perturbation the full QM displayed "sawtooth" shaped graphs, which are stereotypical of poor sampling. This indicates that the QM/MM description is not close enough to the full QM description, which could be improved by using electrostatic embedding instead of the mechanical embedding used here.

The "jumps" within the graphs increase in size as the main intermolecular interactions between solute and solvent shift between dispersion and electrostatics, because of this electrostatic embedding was introduced to improve configurational overlap between the QM/MM and QM.

4.4.5 Introducing electrostatic embedding

Electrostatic embedding differs from mechanical embedding in that the QM part of the system is surrounded by embedded point charges that represent the rest of the system. For systems in vacuum or when mechanical embedding was used, equilibration was achieved rapidly. However, the introduction of point charges led to slower equilibration. In order to increase the speed at which this could be achieved, a classical equilibration was included initially, i.e. after the system had been minimised, heated and the box size equilibrated, a further classical equilibration was performed using HMC to accept to a classical NVT ensemble.

Although N_2 in various solvents has been used heavily as a test system, in order to appreciate the additional accuracy provided by using electrostatic embedding an solute that has a larger interaction with the solvent was chosen, in this case ethanol in 450 TIP3P water. TIP3P was used because of the additional acceptance that occurred when comparing rigid with flexible waters previously.

However, it was found that once out of the 100 steps of classical equilibration and 100 steps of QM/MM equilibration, no snapshots were accepted. By comparing the MM and QM potential energy figure 4.10 is produced. Although the QM/MM Hamiltonian energy is used within the acceptance criterion, the potential energies of both QM/MM and MM can be compared because of the following relationship,

$$
\begin{aligned}
\Delta H &= (E^j_{QM} + E^j_{KIN}) - (E^i_{QM} + E^i_{KIN}) && (4.11)\\
&= E^j_{QM} - E^i_{QM} + E^j_{KIN} - E^i_{KIN} && (4.12)\\
&= E^j_{QM} - E^i_{QM} + E^i_{MM} - E^j_{MM} && (4.13)\\
&= (E^j_{QM} - E^j_{MM}) - (E^i_{QM} - E^i_{MM}). && (4.14)
\end{aligned}
$$

Where the step in equation 4.13 can be performed because the MD is performed in the microcanonical ensemble, i.e. (assuming no integrator errors) the difference in kinetic energy and classical potential energy will be equal and opposite, $\Delta E_{KIN} = -\Delta E_{MM}$, i represents the current snapshot and j represents the new snapshot.

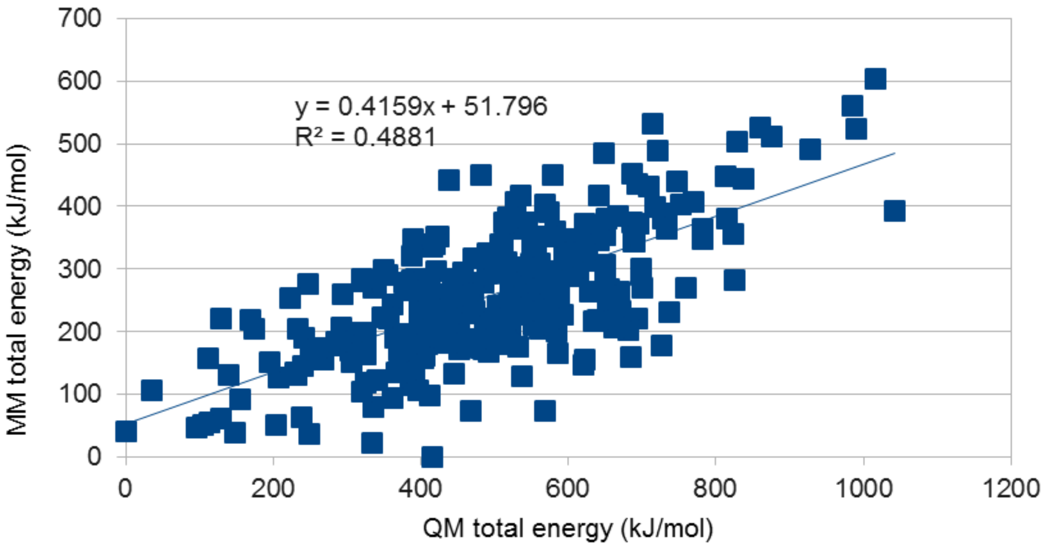


Figure 4.10: Comparison of MM and QM/MM potential energy for ethanol in 450 TIP3P waters

The positive correlation shown in figure 4.10 shows that there is a relationship between the potential energies, but the correlation is not high enough for a good acceptance rate. However, this correlation can be drastically improved by including the classical dispersion within the QM/MM potential energy. The result can be seen in figure 4.11.

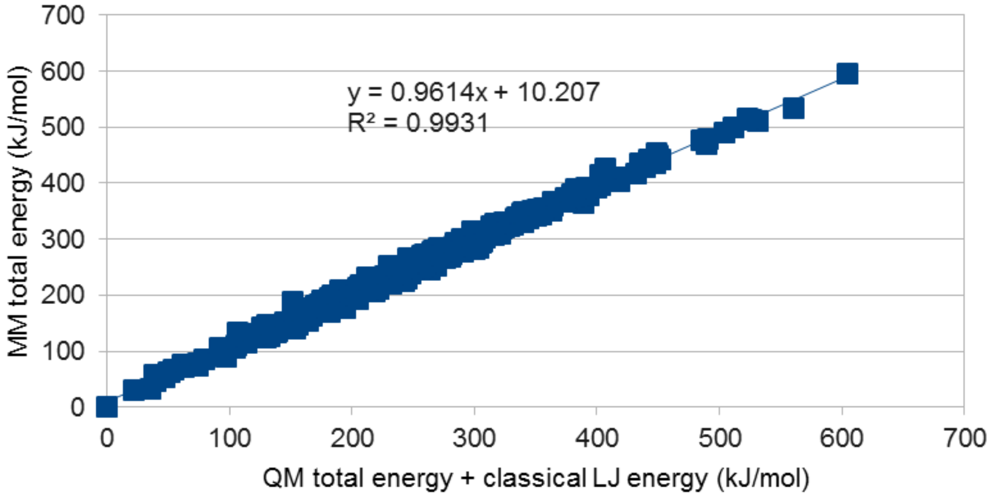


Figure 4.11: Comparison of MM and QM/MM + LJ potential energy for ethanol in 450 TIP3P waters

Even with an R^2 value of 0.99 the MM and QM/MM + LJ overlap is still insufficient to obtain any acceptance outside of the equilibration steps. The reasoning for this becomes clear when examining figure 4.12 and 4.13.

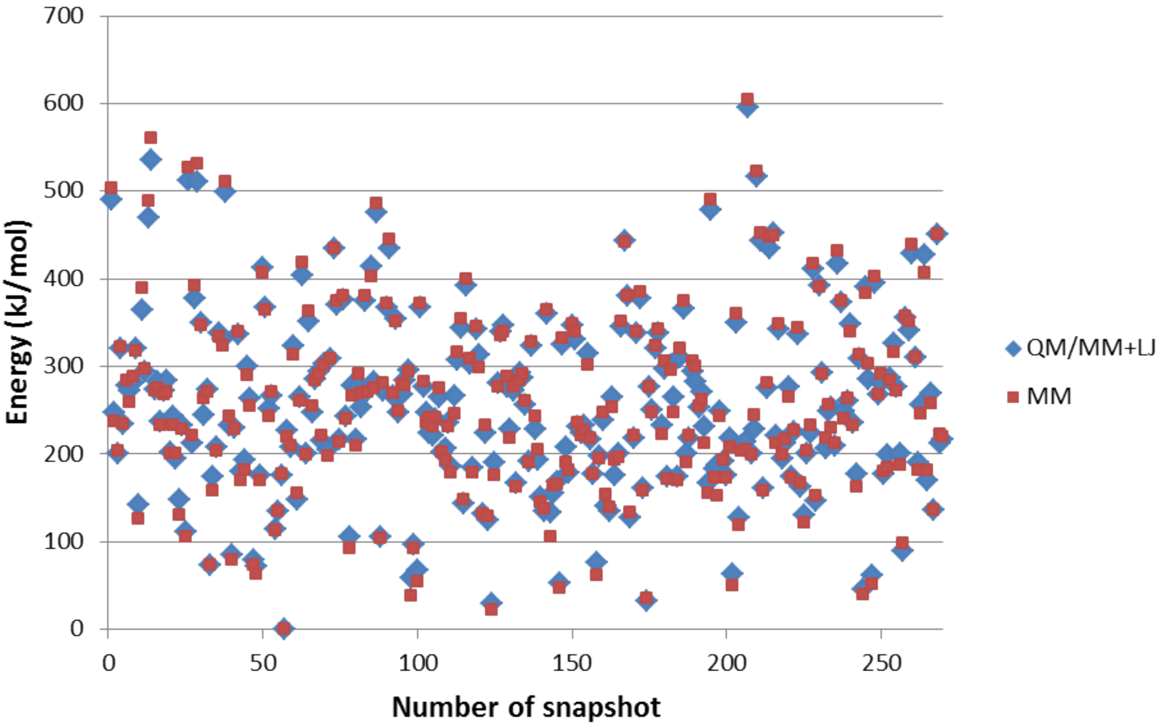


FIGURE 4.12: Level of overlap between the MM and QM/MM+LJ, both have been shifted down so the lowest energy snapshots sit at 0

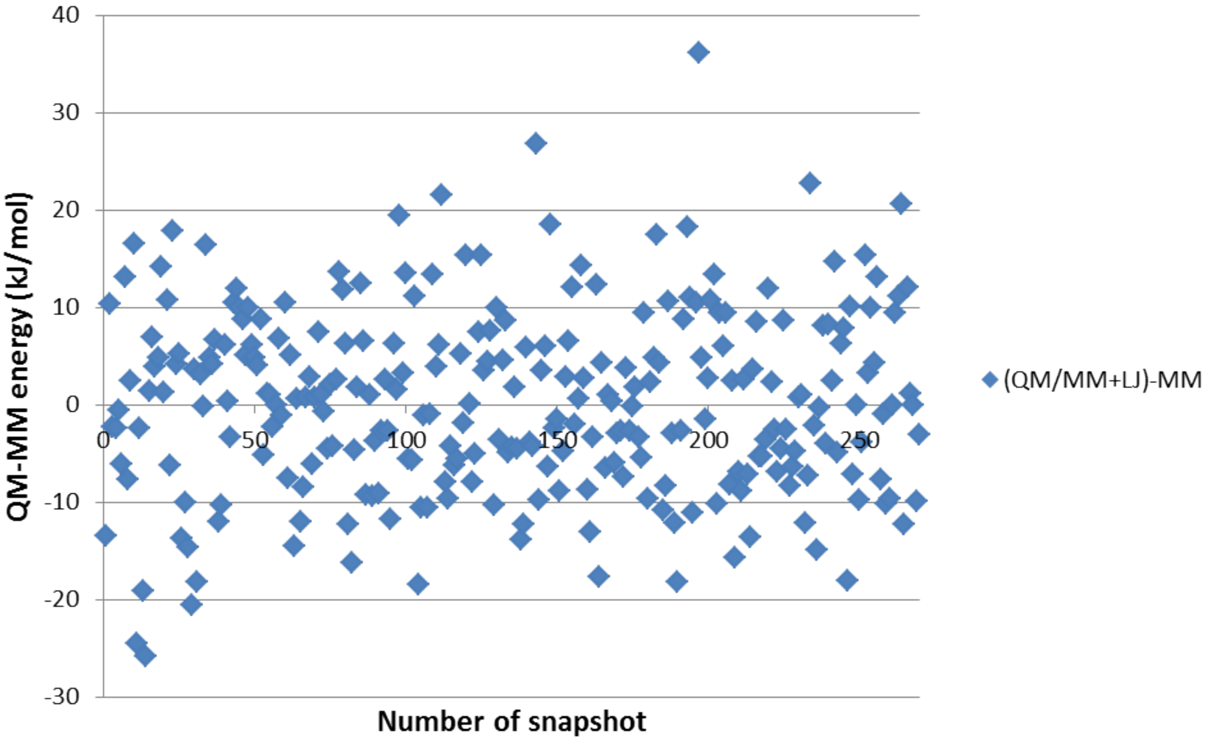


Figure 4.13: The difference between the QM/MM+LJ and MM potential energy, the average energy difference has been subtracted from each energy difference

These figures (figure 4.12 and 4.13) highlight a flaw in the method, if the QM/MM+LJ energy is lower than the MM energy in figure 4.12 then the energy difference is negative and is accepted, regardless of the true probability to select this snapshot from a QM/MM+LJ potential energy surface. By examining the energy differences and subtracting the average energy difference, figure 4.13 is produced, which shows a large number of the snapshots appearing around 0 kJ/mol. This shows that the MM is, on the whole, a good representation of the QM/MM+LJ surface. However, rare events cause a large negative energy difference, which then cause the simulation to get stuck. For example, in figure 4.13, the lowest energy difference occurs very early on, as such after this snapshot no other is accepted. This is caused by the inexact overlap between the MM and QM/MM+LJ.

There are several possible solutions to this issue, the first being to alter the force field to improve the level of overlap, alternatively the MC steps can be made much smaller to make use of the correlation between structures to ensure that energy differences are close together.

4.4.5.1 Small MC Steps

By lowering the movement of the system between subsequent snapshots the acceptance rate should increase. In order to lower the movement, the length of the MD simulation used to generate structures was made shorter. By lowering the time of the MD to 0.25fs, a single timestep, the acceptance for ethanol in 450 electrostatic embedded waters increases to 99%. The disadvantage with this is that each snapshot is correlated, so cannot give an accurate representation of the free energy.

Lowering the time between structures was tested on a larger system of cyclodextrin in complex with chlorobenzene in vacuum (figure 4.14). Both MD lengths of 2 fs and 5 fs were tested, and in both cases the acceptance was below 10%.

Although in theory using much smaller moves works, using a method like this will take a long time to converge the free energy, the bottleneck being the calculation of the QM/MM.

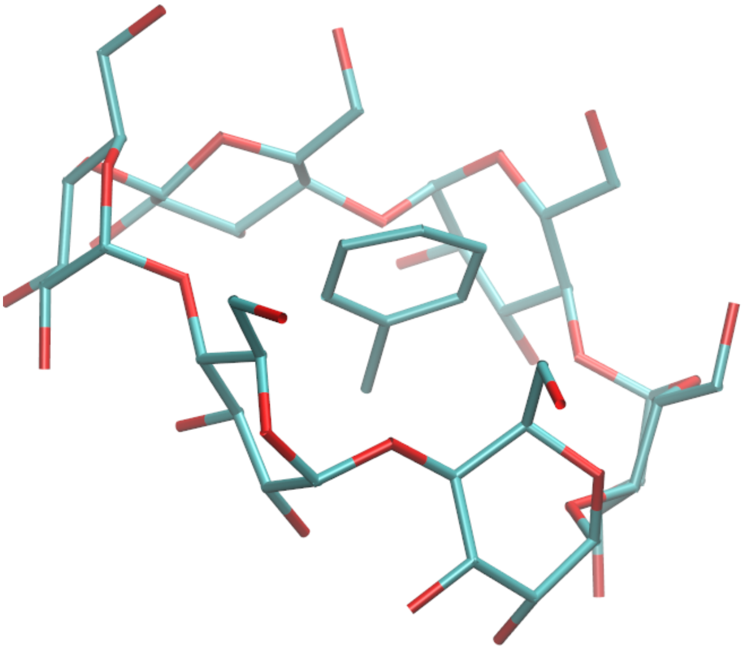


Figure 4.14: PhCl bound to Cyclodextrin

4.4.5.2 Altering the Force Field/Removing Degrees of Freedom

In order to achieve higher acceptance rates, alterations to the force field were considered. If the classical potential energy surface was a better match for the QM/MM potential energy surface, then the acceptance will increase. In order to test this a simple system of ethanol in vacuum was used.

The quantum parameters were obtained by geometry optimisation using ONETEP with the same setup as for the single point energy calculations used throughout this study.

The equilibrium values in the force field were then altered to match the values for the quantum optimised structure. Table 4.5 shows the results of changing these values

Parameter changed	Acceptance %
Bond lengths	< 5*
Angles	< 2*
None*^{1}	27.6
Fixed Bond Lengths LINCS*^{1}	30.2
Fixed Bond Lengths SHAKE*^{1}	25.0
Higher Bond force constant	< 5*
Fixed Angles	< 10*

TABLE 4.5: *simulations cancelled before end of equilibration due to extremely low acceptance, *^{1}simulations using standard classical parameters. Acceptance when altering the classical potential

It is clear from Table 4.5 that altering the force field in such a simple manner does not improve the overlap with the quantum surface. This was all tried for ethanol, which is not a standard residue. The test was then repeated for alanine dipeptide, which uses the AMBER residues, ACE, NME and ALA (figure 4.15). The results are shown in Table 4.6.

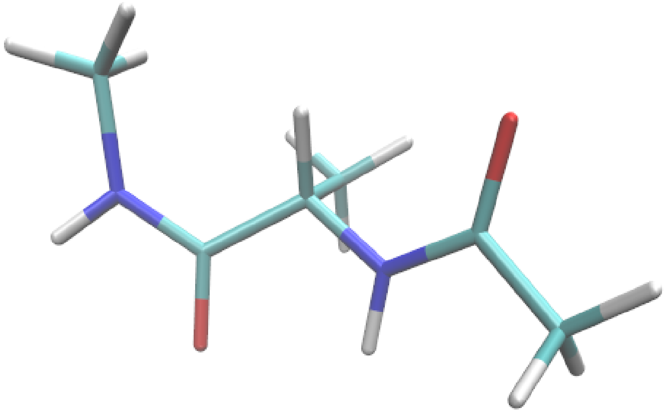


FIGURE 4.15: Alanine dipeptide, residues from left to right are NME, ALA and ACE

Parameter changed	Acceptance %
None	$< 3^{*1}$
Frozen bond LINCS	$< 5^{*1}$

Table 4.6: ^{*1}simulations using standard classical parameters. Acceptance when altering the classical potential for alanine dipeptide

Although alanine dipeptide uses standard residues the acceptance is still very poor. By freezing out the bond vibrations the acceptance slightly increases, however, this could just be an artefact of the stochastic nature of the method.

The next step was to freeze part of the system, initially this was selected to be the QM region, i.e. the solute. The ligand was allowed to move for the equilibration, then frozen and the water allowed to move around it. The system selected was N_2 in 499 water and the acceptance was 13.2 %. If the ligand is allowed to move throughout the simulation, this acceptance drops to below 1 %. Although an improvement, this acceptance is still a little low. Following this the water was frozen and the ligand allowed to move. The results can be seen in Table 4.7.

	Acceptance % at $\lambda = 1$	Free Energy (kJ/mol)
Run 1	39.8	-487623.80
Run 2	20.8	-488176.35

Table 4.7: Free energy and acceptance rate when allowing N_2 to move in frozen water

A difference in the free energy of 552.55 kJ/mol for a simple system such as N_2 in water suggests major convergence problems. If this method were to be applied to a more flexible ligand the results would be expected to show even worse convergence.

4.5 Conclusions

Using classical mechanics as a guiding potential to generate a QM/MM ensemble of structures, in practice, only works if the potential energy surfaces match. The acceptance test used is sensitive to differences in energy, similar to that found when using the single step perturbation approach in the previous chapter. However, unlike when using the single step perturbation approach, the problem is more subtle. The acceptance test uses the QM/MM potential energy and kinetic energy, so in theory should not depend upon the classical potential energy. However, because the underlying MD simulation must be run in the microcanonical ensemble, the change in the kinetic energy, will be equal-but-opposite the change in the classical potential energy. If this is taken into account, the resultant acceptance test is based upon the differences between the QM/MM potential energy and the MM potential energy. Such that, if the difference of a new snapshot is lower than the original snapshot, it will always be accepted.

The cause of the mismatch between the potential energy surfaces can be attributed to a number of things; For small ligands, with relatively few degrees of freedom, it is caused by either the interaction between the ligand and the electrostatic embedded water molecules, or the intermolecular interaction in the solvent. Whereas, for larger ligands, the degrees of freedom start to have an impact on the acceptance.

In order to improve the acceptance, at least for the smaller ligands, a bias could be applied, so that only if the energy difference is around a central point will the snapshot be accepted. This bias will be the focus of the next chapter.

Chapter 5

Improving convergence between MM and QM by application of a bias

5.1 Introduction

By using hybrid Monte Carlo [102] a desired potential can be sampled by using a cheaper, guiding potential. In this work so far, QM/MM is the desired potential and MM is the cheap guiding potential. The acceptance criterion for such a method is,

$$A_{ij} = \min\{1, \exp[-\beta((E^{j}_{QM/MM} + E^{j}_{KIN}) - (E^{i}_{QM/MM} + E^{i}_{KIN}))]\}. \tag{5.1}$$

$E^{i}_{QM/MM}$ is the QM/MM potential energy of snapshot i, the current snapshot. E^{i}_{KIN} is the kinetic energy, for snapshot i this is taken from a Gaussian distribution, for j this is extracted from the underlying MD simulation. MD is used as an engine to generate uncorrelated structures quickly and is run with constant Hamiltonian energy (i.e. the microcanonical ensemble). By using a small MD time step of 0.25 fs the errors in the integrator will be negligible and the difference in the QM/MM Hamiltonian used within equation 5.1 can be rewritten as,

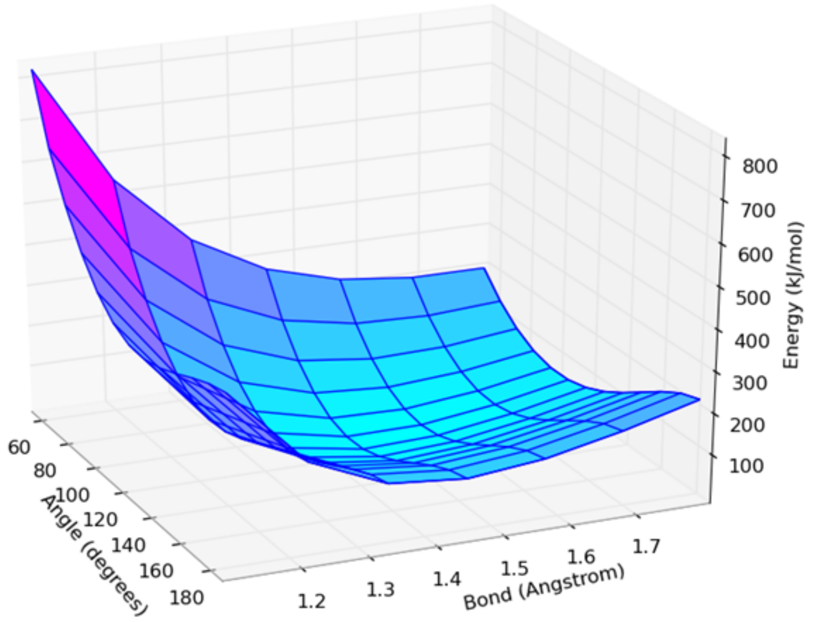


FIGURE 5.1: QM potential energy surface for methanol

$$\begin{aligned}\Delta H &= (E^j_{QM/MM} + E^j_{KIN}) - (E^i_{QM/MM} + E^i_{KIN}) & (5.2)\\ &= E^j_{QM/MM} - E^i_{QM/MM} + E^j_{KIN} - E^i_{KIN} & (5.3)\\ &= E^j_{QM/MM} - E^i_{QM/MM} + E^i_{MM} - E^j_{MM} & (5.4)\\ &= (E^j_{QM/MM} - E^j_{MM}) - (E^i_{QM/MM} - E^i_{MM}). & (5.5)\end{aligned}$$

Where the step in equation 5.4 is possible as the MD simulation is run within the NVE ensemble, as such $\Delta E_{KIN} = -\Delta E_{MM}$. This shows that the acceptance of structures into the QM/MM ensemble is related to the level of overlap between the MM and QM/MM.

In order to examine the extent at which the MM and QM potential energy surfaces differ, a model system was used. This model system was selected to be methanol with all degrees of freedom frozen except the C-O bond and the C-O-H angle. By using this system, two dimensional plots can be easily generated and compared and give and indication of how well and where the potential energy surfaces match.

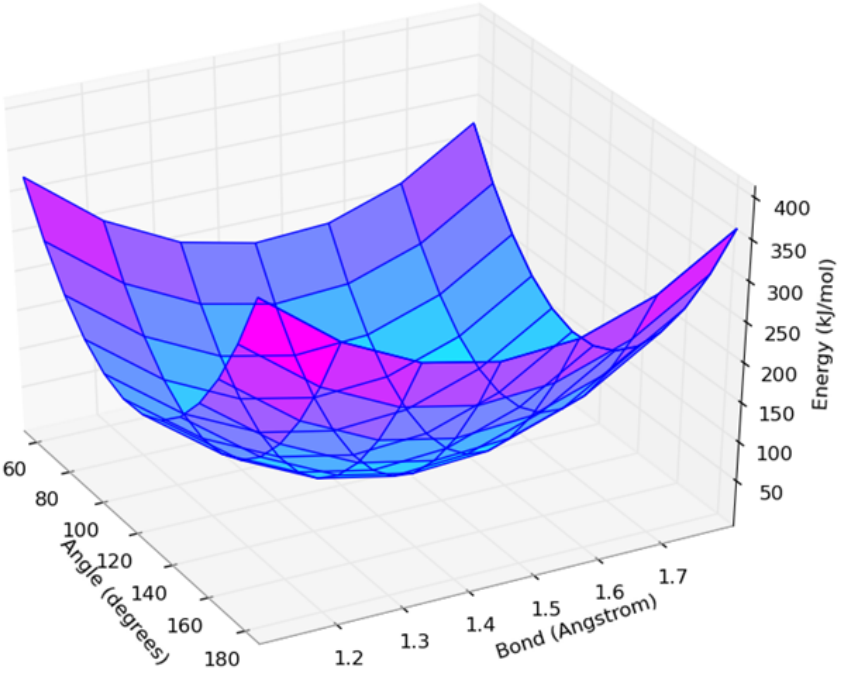


FIGURE 5.2: MM potential energy surface for methanol

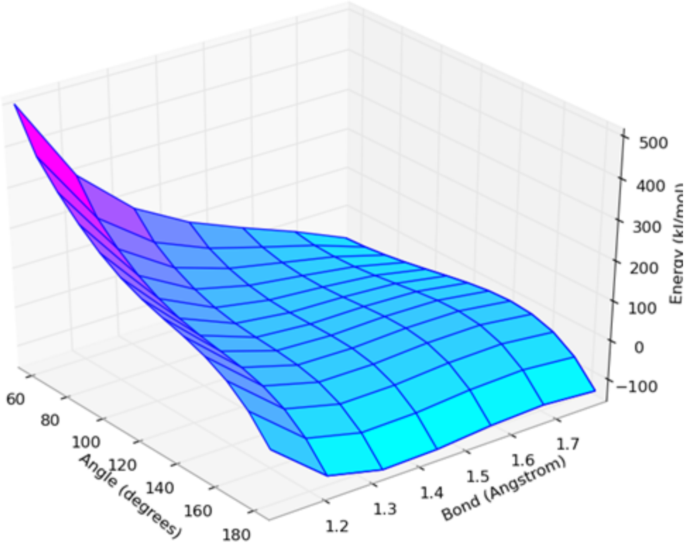


FIGURE 5.3: QM - MM energy difference surface for methanol

From figure 5.1 and 5.2 it is clear to see the poor overlap which results in figure 5.3. The regions of negative energy difference have been caused by the different types of surface (Morse like vs Harmonic). During a hybrid Monte Carlo simulation, an excessive weight is given to these low energy difference structures, regardless of the position on the QM potential energy surface. So for the above example, if a structure is in this low energy difference region, it will be accepted. However, this is a high energy QM region and not a good representation of the quantum ensemble. A similar conclusion was also drawn from Iftimie et al. [104] where they fit the classical potential such that it approximately matches the *ab initio* potential energy surface.

Notice however, that the position of the minimum is the same for both the MM and QM surfaces. Therefore, it should be possible to apply a bias to ensure better sampling of the minimum. We hope that such an approach will prevent the simulation from getting stuck in a low energy difference region. The use of biasing potentials within free energy calculations is not novel [110, 111, 112, 113]. Reference [111] uses the energy difference to bias the free energy, then unbiases by using the relationship described in reference [110],

$$\langle X \rangle = \frac{\langle X \exp(\beta\omega) \rangle_{bias}}{\langle \exp(\beta\omega) \rangle_{bias}}. \tag{5.6}$$

Where X is the property of interest, ω is the applied biasing potential and the subscript $_{bias}$ refers to ensembles generated in the presence of the biasing potential.

By including the bias into the acceptance criterion, a biased ensemble can be generated. The acceptance criterion is then,

$$A_{ij} = \exp[-\beta\{[(U^f_{QM} + \omega^f) - U^f_{MM}] - [(U^i_{QM} + \omega^i) - U^i_{MM}]\}]. \tag{5.7}$$

By setting ω to a simple function, such as $\frac{1}{2}\kappa(\Delta U - \langle \Delta U \rangle)^2$ there are only two unknowns to identify: κ and $\langle \Delta U \rangle$. ΔU in this case is the difference in the QM and MM energies i.e. $\Delta U = U_{QM} - U_{MM}$. Identifying the average energy difference is a process that is updated at every proposed snapshot. The process to find the constant κ is slightly more complicated.

After X steps of the simulation being stuck in a low energy difference, a bias is applied such that if the next snapshot proposed was equal to the average energy difference, it would be accepted. i.e.,

$$\langle \Delta U \rangle = (\Delta U^i) + \omega^i. \tag{5.8}$$

Then using the value of ω^i the value for κ is found,

$$\kappa = \frac{2\omega^i}{(\Delta U^i - \langle \Delta U \rangle)^2} \tag{5.9}$$

The value for ω^f is then calculated using κ. κ will remain unchanged unless the simulation gets stuck in a low energy difference snapshot for X steps again.

After 100 steps of hybrid Monte Carlo, the average energy difference and the constant κ are considered converged and held fixed for subsequent steps.

Once the biased ensemble has been generated, the property of interest to unbias using equation 5.6 is the energy difference required for TI,

$$\begin{aligned}
\left\langle \frac{\partial U}{\partial \lambda} \right\rangle &= \frac{\left\langle \frac{\partial U}{\partial \lambda} \exp[-\beta\omega] \right\rangle_{bias}}{\langle \exp[-\beta\omega] \rangle_{bias}} && (5.10)\\
U &= \lambda U_{QM} + (1-\lambda) U_{MM} && (5.11)\\
\frac{\partial U}{\partial \lambda} &= U_{QM} - U_{MM}. && (5.12)
\end{aligned}$$

For the perturbation between the QM/MM and QM, the Zwanzig equation (equation 4.10) is used. The property to unbias in this case is the following,

$$\left\langle \exp[-\beta(U_{QM} - U_{QM/MM})\right\rangle = \frac{\left\langle \exp[-\beta(U_{QM} - U_{QM/MM})] \exp[\beta\omega]\right\rangle_{bias}}{\left\langle \exp[\beta\omega]\right\rangle_{bias}}. \tag{5.13}$$

5.2 Methods

The details of the simulations are identical to the previous chapter, but are re-capped here for clarity.

5.2.1 Classical details

The parameters for ethanol and N_2 were obtained by using antechamber [97]. The equilibration procedure was undertaken within Gromacs v4.6.2 [107] using the leapfrog integrator and is as follows. All systems, N_2, ethanol and alanine dipeptide in vacuum, were initially minimised with the steepest descent algorithm for 5000 steps with a cutoff of 11 Å, long range electrostatics and Van der Waals were treated with the cut-off method. Following this, a 500 ps simulation in the canonical ensemble was run using a timestep of 1 fs, the Berendsen thermostat was used to heat the system smoothly to 300 K. After equilibration the box size decreased such that only a maximum of an 8 Å cut-off could to be used for the non-bonded interactions.

The classical simulations within hybrid Monte Carlo were performed using the velocity-verlet algorithm, with a 0.25 fs timestep.

5.2.2 Quantum details

The quantum energies were calculated using ONETEP [54]. The first 100 applications of the acceptance test (steps) were discounted and used as equilibration. The setup for ONETEP, in the case of N_2, was that 4 NGWFs were used with a 7.0 a_0 localisation radius. For ethanol and alanine dipeptide 4 NGWFs were used on heavy atoms and 1 for light atoms, all were 8 a_0 and in both cases a kinetic energy cutoff of 800eV was used.

5.3 Results

In order to validate the proposed method, N_2 in vacuum was selected to act as a test system. By using such a simple test case, the unbiased simulation can be performed and can be used as a reference against which to compare the biased simulation results. Because the unbiased simulation can be performed, it is unlikely that the simulation will get stuck in a low energy difference well, therefore a bias will never be applied. In order to circumvent this, a pre-set bias constant (κ) of 0.5 (kJ/mol)$^{-2}$ was applied. The initial comparison between the obtained average energy difference for the unbiased and de-biased simulation can be seen in Table 5.1. Where the de-biased simulation is a biased simulation that has had the bias removed.

	Unbiased	De-biased
Average energy difference (kJ/mol)	-52430.70	-52434.95

Table 5.1: Comparison of average energy difference between the QM and MM at $\lambda = 1$

The initial results shown in the above Table display a 4 kJ/mol difference between the average energy difference obtained from the de-biased and unbiased simulations. This result was surprising considering only a single degree of freedom is present within N_2. To discover the reason for this difference the equation used to unbias the biased simulation was examined,

$$\left\langle \frac{\partial U}{\partial \lambda} \right\rangle = \frac{\left\langle \frac{\partial U}{\partial \lambda} \exp[\beta\omega] \right\rangle_{bias}}{\langle \exp[\beta\omega] \rangle_{bias}}. \tag{5.14}$$

There are two aspects to this equation, the numerator and the denominator. By examining the average for these values as the number of included snapshots is increased, an idea of how converged the results are can be obtained.

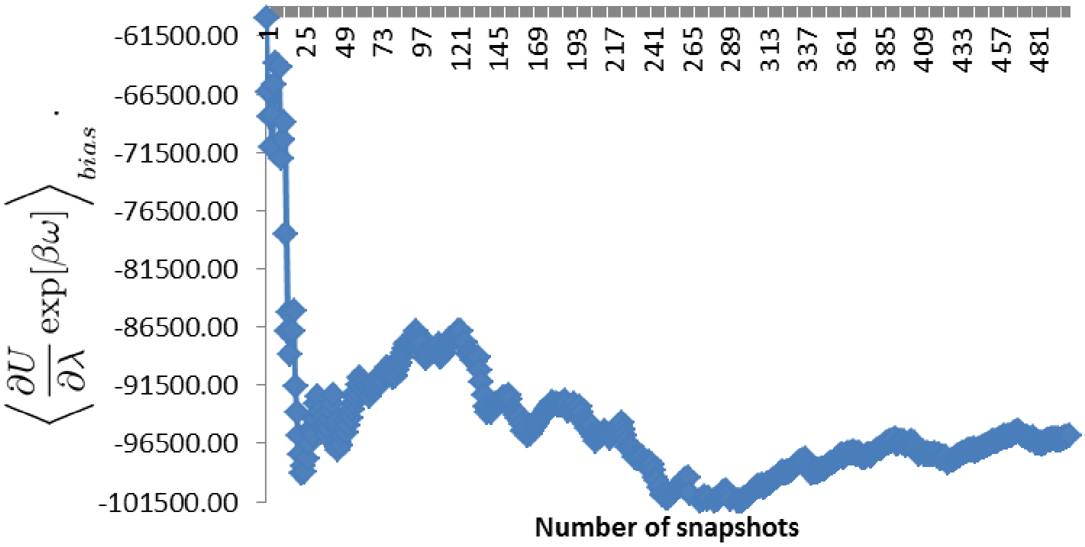


FIGURE 5.4: Calculating the numerator of equation 5.14 as the number of snapshots is increased

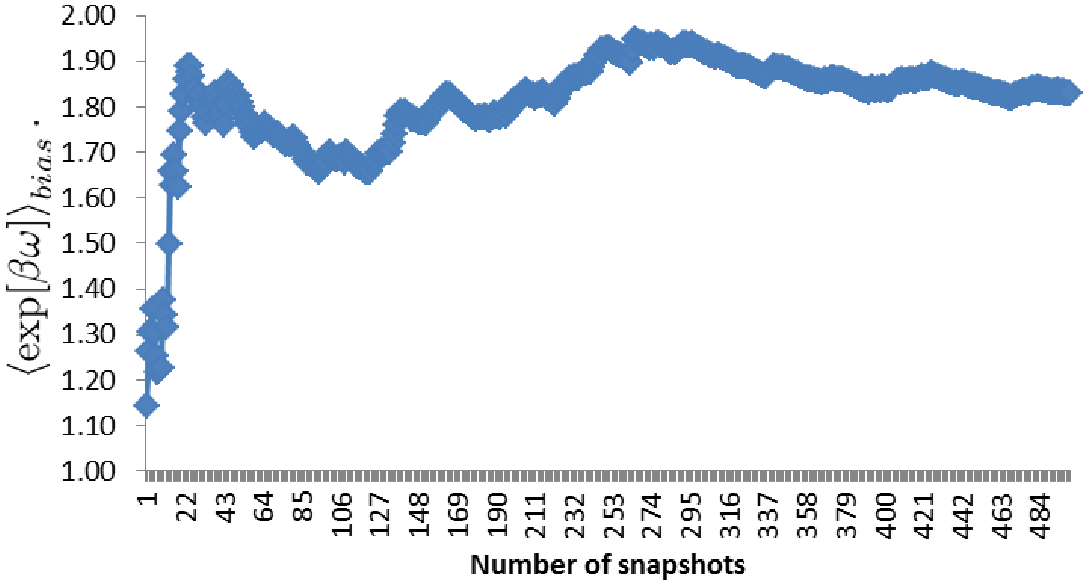


FIGURE 5.5: Calculating the denominator of equation 5.14 as the number of snapshots is increased (the y-axis is unitless)

The denominator very quickly converges to between 1.8 and 1.9, however, the numerator continues to show large fluctuations throughout the simulation. When considering how small the fluctuations of the denominator are, it is clear that the fluctuations within the numerator must be due to large differences between the MM and QM energies.

Further tests were then run, these include running an additional 500 hybrid Monte Carlo steps using the same constant (0.5 $(\text{kJ/mol})^{-2}$) and center of bias as the previous simulation, using a smaller constant (0.1 $(\text{kJ/mol})^{-2}$) and finally, doubling the equilibration steps within hybrid Monte Carlo from 100 to 200.

	0.5 constant (cont)	0.1 constant	200 step equilibration
De-biased average energy difference (kJ/mol)	-52434.82	-52434.96	-52434.92

Table 5.2: Comparison of average energy difference between the QM and MM at $\lambda = 1$ varying the constant and equilibration length

The results for the additional test simulations in Table 5.2 do not match the average energy difference obtained from the unbiased simulation in Table 5.1. Interestingly, all the results that were obtained from a biased simulation compare very well to one another, falling within a range of 0.1 kJ/mol. This excellent convergence led to further investigation of the unbiased free energy term in equation 5.14 as generated by the original simulation. In a similar process to the examination of the numerator and denominator, we viewed the unbiased free energy as a function of the number of snapshots included. The resulting graph (figure 5.6) shows very little ($<$ 0.2 KJ/mol) fluctuation after 300 snapshots.

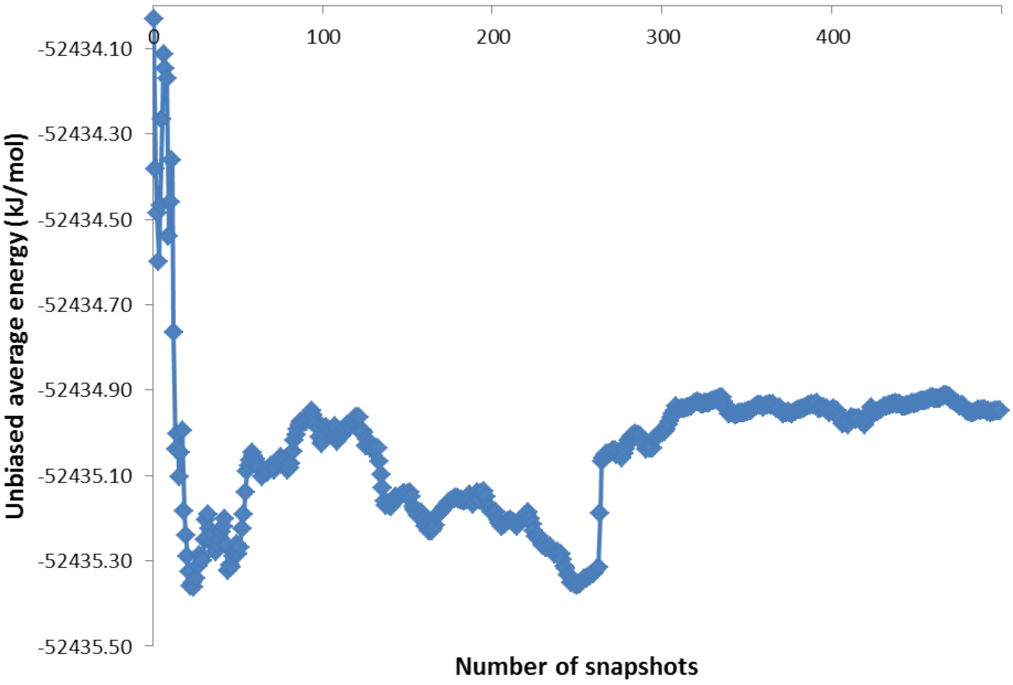


Figure 5.6: Calculating the unbasied energy as the number of snapshots is increased

The converged free energy for the simple system led to the application on more complicated test systems. Initially ethanol in vacuum was used. Again this system can be used within the unbiased simulation so a pre-set set bias of 0.5 was applied and the standard 500 hybrid Monte Carlo steps were used.

	Unbiased simulation	De-biased simulation
Average energy difference (kJ/mol)	-81546.81	-81544.51

Table 5.3: Comparison of average energy difference between the QM and MM at $\lambda = 1$ for ethanol in vacuum

The results in Table 5.3 again show a small difference between the average energy difference obtained from the biased and unbiased simulations. In order to check convergence, the biased simulation was continued for an additional 1000 hybrid Monte Carlo steps.

	Unbiased simulation	De-biased simulation (cont)
Average energy difference (kJ/mol)	-81546.81	-81544.22

Table 5.4: Comparison of average energy difference between the QM and MM at $\lambda = 1$ for ethanol in vacuum

Performing an additional 1000 steps shows no change in the average energy difference. As before, the average energy difference was calculated as a function of the number of snapshots included and the result is shown in figure 5.7.

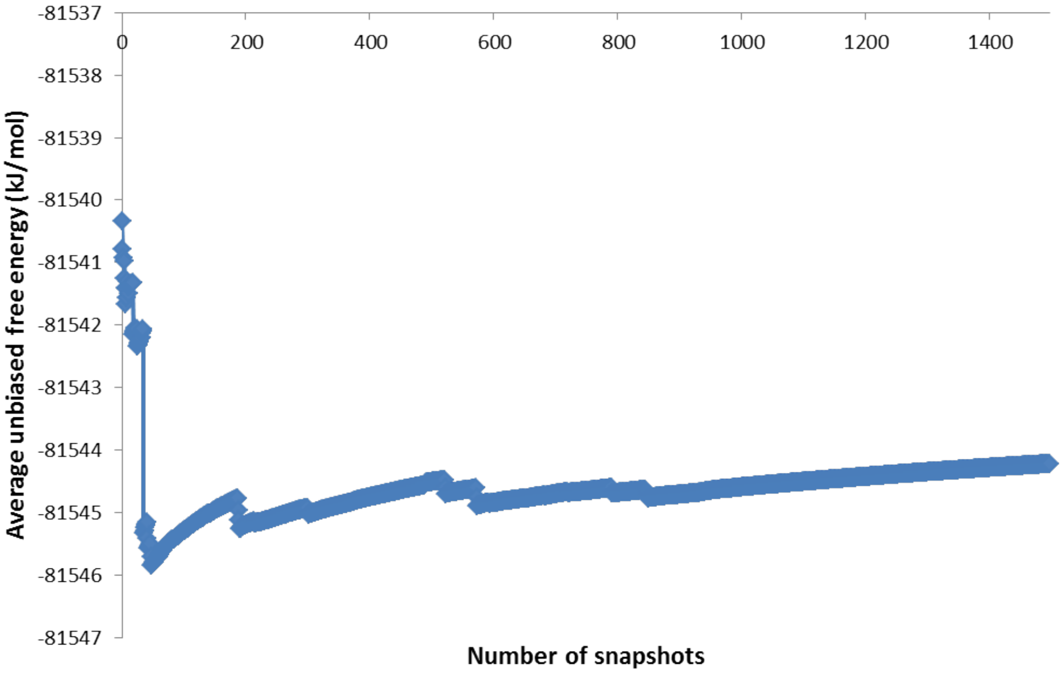


Figure 5.7: Calculating the unbasied energy as the number of snapshots is increased

The slight "jumps" present within the above figure show the first signs of "sawtooth" behaviour, characteristic of poor sampling[99]. However, these "jumps" have little effect on the free energy for this system. To identify if a larger effect is seen when the system used has more degrees of freedom, alanine dipeptide was used. However, as was shown in the previous chapter, the unbiased method cannot be used to reliably sample alanine dipeptide. As such the unbiased simulation was not performed, so three repeats of the biased simulation were run to check convergence. The results can be seen in Table 5.5.

	Run 1	Run 2	Run 3
De-biased average energy difference (kJ/mol)	-245749.95	-245755.24	-245759.52
Center of bias (kJ/mol)	-245743.87	-245740.52	-245741.59
Constant $(kJ/mol)^{-2}$	0.34299	0.13284	0.07103
Acceptance %	30.6	40.8	36.2

Table 5.5: Comparison of simulation details for three repeats of alanine dipeptide in vacuum

It is clear from Table 5.5 that convergence has not been achieved, with a range of values around 10 kJ/mol. However, the center of the bias for each simulation, had a much lower range of values. Following these results, a simulation was performed with a high constant of 10 $(kJ/mol)^{-2}$. The acceptance rate dropped to 7.4% as would be expected from a more intense bias. The final result was -245743.68 kJ/mol, which is within 6 kJ/mol of run 1. This may be attributed to the higher constant within run 1, indeed, it from Table 5.5 it is apparent that the higher constant, the closer to the unbiased average energy will be to the bias center. The convergence for the high constant simulation for alanine dipeptide can be seen in figure 5.8.

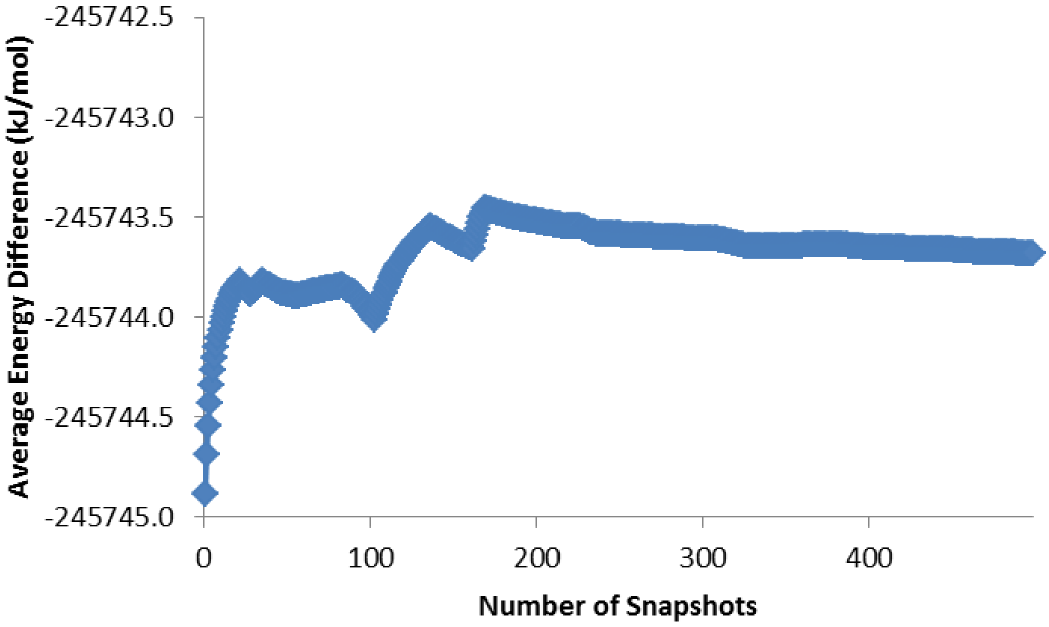


Figure 5.8: Calculating the unbasied energy as the number of snapshots is increased for alanine dipeptide in vacuum with a constant of 10 $(kJ/mol)^{-1}$

To test if the unbiased average energy can be reproduced if the same bias center and constant is used, run 1 was performed again. This time an unbiased average energy of -245751.31 kJ/mol was achieved. This value differs with the original value of -245749.95 kJ/mol by 1.36 kJ/mol. This difference is within thermal error and shows some promise, however, more sampling is required to make a firm conclusion. This is problematic as each simulation run involves quantum calculations that take between 10-15 minutes of cpu time on 4 cores.

With the consideration that more sampling is needed to make any firm conclusions, the computational cost of the quantum calculations of simple systems and that these systems are only simple "test" systems. A numerical model was designed with the intention to provide some insight on what would happen on larger systems without the prohibitive cost of the calculations. Chapter 3 showed that the QM-MM energy values are a Gaussian distribution, as such the numerical model selected "energies" from a Gaussian using the Marsaglia Polar method[105] detailed in Chapter 4. A constant of 0.3 $(\mathrm{kJ/mol})^2$ was used as this is similar to the constant generated from the simulation of alanine dipeptide in vacuum and within the range of the pre-set value for ethanol in vacuum. The bias center was set to -80,000, similar to that of ethanol in vacuum, and the standard deviation was altered to represent systems of increasing complexity, i.e. the larger the system, the worse the overlap between the MM and QM configurational space, the more diverse the energy differences, so the larger the standard deviation of the Gaussian distribution. The computational cost of this numerical test is negligible, so 1,000,000 values were collected for each run. The results are shown below.

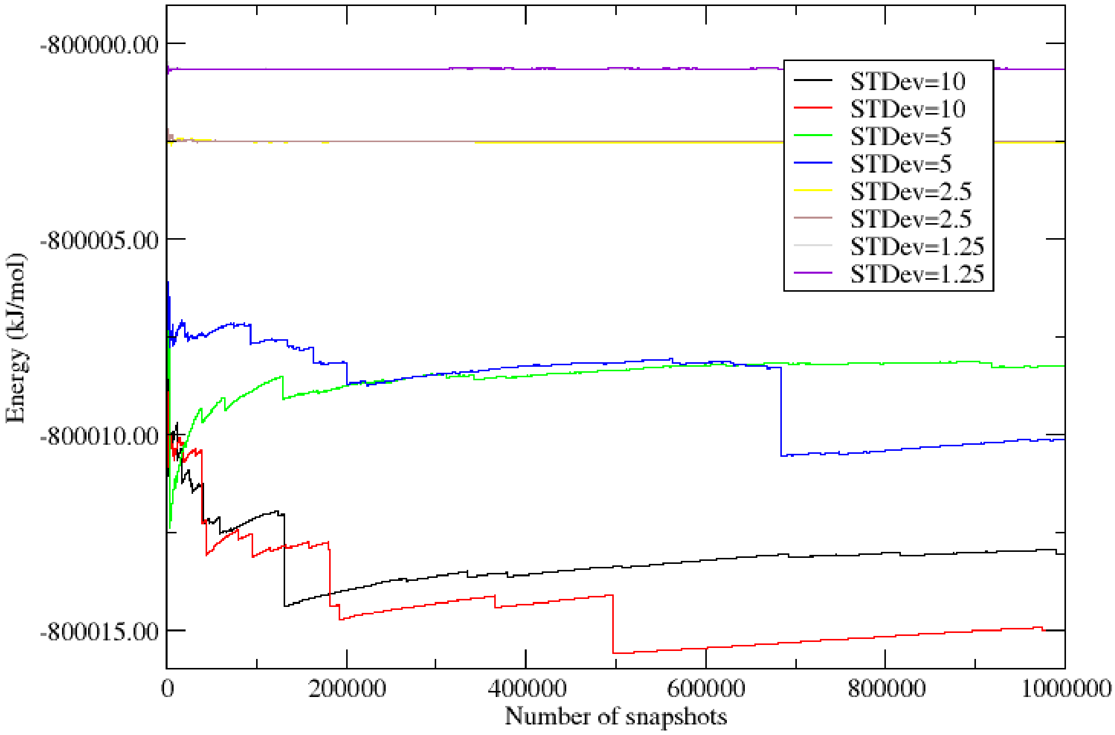


Figure 5.9: Results for the numerical test, bias center at -80000 kJ/mol with a constant of 0.3 $(\mathrm{kJ/mol})^{-2}$

Figure 5.9 shows that convergence is linked to the standard deviation. Indeed, for a standard deviation of 5 kJ/mol and above, convergence is never achieved and a "sawtooth" shape is present. Each standard deviation was used in 2 numerical tests, to test whether convergence could be achieved. The standard deviation affects the width of the Gaussian distribution, so higher standard deviations represent systems with more degrees of freedom. If the Gaussian distribution presented in Chapter 3 (figure 3.9) is considered, the range is 138 kcal/mol (577.39 kJ/mol) and the standard deviation is approximately 144 kJ/mol, which is considerably larger than the standard deviations used within the numerical model. Considering the system that produced this energy difference Gaussian was a relatively simple one of ethanol in 138 water molecules, it does not bode well for complex biologically relevant systems, such as proteins with many more degrees of freedom. This leads to the conclusion that the biased method presented within this chapter will not converge for systems of biological interest.

5.4 Conclusions

This chapter initially showed the differences within the MM and QM potential energy surfaces of a simple molecule (methanol). These differences highlighted the problem with direct applications of HMC, in that the resulting generated ensemble can get stuck in a low-energy difference structure, which is actually a high energy structure in the desired potential. As such a bias was applied to ensure that simulations would not get stuck in unfavourable structures.

This new biased method provided converged results for N_2 and ethanol in vacuum, but the results for ethanol started to show signs of "sawtooth" sampling. In both cases the free energies from the unbiased simulation did not match those from the biased. However, a variety of experiments were attempted using N_2 in vacuum, and although not matching the unbiased simulation, all the biased free energies matched.

We then moved on to a slightly more realistic system of alanine dipeptide in vacuum, which could not be compared to an unbiased simulation due to poor acceptance (see the previous chapter for more information). Three independent simulations were performed, which did not converge, however, when the parameters used within the bias were set to match the parameters generated for the first run and the simulation re-run, the resultant free energy was within thermal error of the value originally produced.

Following this, a numerical model was designed to represent what would happen if this method were applied to systems of increasing complexity with a much higher degree of sampling. The results showed that after a standard deviation of 5 kJ/mol convergence between two simulation runs could not be achieved. This was then compared to real values produced within Chapter 3, where the standard deviation was 144 kJ/mol for ethanol in 138 water.

This method has some merit when sampling simple systems, and can produce converged free energies for more complex systems than the unbiased method, but still faces issues when the systems are more complex. In order to improve the convergence the guiding potential must better match the desired potential.

Chapter 6

A "Stepping Stone" Approach for Obtaining Quantum Free Energies of Hydration

6.1 Introduction

In previous chapters, it has been shown that the use of hybrid Monte Carlo [102] (HMC) can generate an ensemble that is relevant for a more expensive potential. However, if the overlap is insufficient, the acceptance rate is too low to provide a feasible method. Chapter 5 showed this inexact overlap in more detail and that even when a bias is applied, convergence cannot be achieved.

This chapter deals with the inexact overlap of the configurational space between the MM and QM, by accepting to an ensemble closer to the MM potential, but one that includes information on polarisation. The structures from this new potential are then used within the Zwanzig equation (equation 3.1) to calculate the quantum free energy. Both interaction energies and total energies were tested.

The method presented here was applied to five solutes that represent a variety of chemical behaviours, including, crucially, different polarities. The hydration free energies for these ligands were calculated in order to test the method. Although not presented here, we expect that it should be possible to extend this method to include protein-ligand binding.

The work within this chapter has been published in J. Phys. Chem. B [114].

6.2 Methods

Our method first generates an ensemble of structures closer to the fully quantum ensemble by applying the Hybrid Monte Carlo method using classical molecular dynamics to sample configurations, but accepting to a QM/MM ensemble. The QM/MM ensemble energy is calculated as the energy of the MM system but with the Coulombic component of its interaction energy replaced by the equivalent interaction energy in the quantum description. This takes into account the electronic polarisation of the solute as a result of the surrounding solvent. Once an ensemble of QM/MM structures has been generated then a single step perturbation approach can be applied to calculate the free energy of mutation from the QM/MM to the full QM ensemble.

6.2.1 Theoretical Details

6.2.1.1 The Hybrid Monte Carlo Method

The theory behind HMC can be seen in Chapter 4. The acceptance criterion can also be seen below,

$$\begin{aligned}\pi_{acc}(\mathbf{R} \rightarrow \mathbf{R}') &= min\left\{1, \frac{\exp(-\beta U(\mathbf{R}'))\exp(-\beta K(\mathbf{P}'))}{\exp(-\beta U(\mathbf{R}))\exp(-\beta K(\mathbf{P}))}\right\} \\ &= min\left\{1, \frac{\exp(-\beta H(\mathbf{R}', \mathbf{P}'))}{\exp(-\beta H(\mathbf{R}, \mathbf{P}))}\right\}. \qquad (6.1)\end{aligned}$$

The method presented here aims to mutate a solvent-ligand complex from its MM representation to a "quantum corrected" representation where the classical electrostatic interactions have been replaced by interactions from a QM/MM calculation. The acceptance criterion is then,

$$\pi_{acc}(\mathbf{R} \to \mathbf{R}') = min \left\{ 1, \frac{\exp(-\beta U_{MM+QM^{int}_{Coul}}(\mathbf{R}')) \exp(-\beta K(\mathbf{P}'))}{\exp(-\beta U_{MM+QM^{int}_{Coul}}(\mathbf{R})) \exp(-\beta K(\mathbf{P}))} \right\}. \tag{6.2}$$

Where the target potential energy is calculated by,

$$\begin{aligned} U_{MM+QM^{int}_{Coul}} &= U^{com}_{MM} \\ &- [U^{com}_{MM_{Coul}} - U^{host}_{MM_{Coul}} - U^{lig}_{MM_{Coul}}] \\ &+ [U^{com}_{QM/MM} - U^{host}_{QM/MM} - U^{lig}_{QM/MM}] \qquad (6.3) \\ &= U^{com}_{MM} - U^{int}_{MM_{Coul}} + U^{int}_{QM_{Coul}}. \qquad (6.4) \end{aligned}$$

The above equation uses the classical potential energy (U^{com}_{MM}) for the whole complex (ligand in solvent) and subtracts from it the electrostatic (Coulomb, "Coul") contribution of the classical interaction energy ($U^{int}_{MM_{Coul}}$). This interaction energy is replaced by the interaction energy between the QM/MM ligand ($U^{lig}_{QM/MM}$) and the host ($U^{host}_{QM/MM}$) from the QM/MM calculation (in this work the host is the solvent), where the QM/MM description used here is the quantum ligand surrounded by classical point charges. The dispersion (Lennard-Jones) part of the interaction energy is not replaced - it still comes from the MM calculation.

6.2.1.2 Transitioning between MM and QM/MM

In order to make the transition from U_{MM} to $U_{MM+QM^{int}_{Coul}}$ smoothly, intermediate $\lambda_{QM/MM}$ steps can be introduced. The introduction of these steps is trivial, and can be performed by replacing the $U_{MM+QM^{int}_{Coul}}(\mathbf{R})$term of equation 6.2 with the following,

$$U_{MM+QM^{int}_{Coul}}(\mathbf{R}; \lambda_{QM/MM}) = (1 - \lambda_{QM/MM})U_{MM} + (\lambda_{QM/MM})U_{MM+QM^{int}_{Coul}}. \tag{6.5}$$

HMC is run for all λ states between the MM and QM/MM, such that at $\lambda_{QM/MM} = 0$ we are accepting to a classical canonical ensemble and at $\lambda_{QM/MM} = 1$ to the QM/MM corrected canonical ensemble. The difference for each lambda value is in the MC acceptance criterion which influences the progress of the MD simulations accordingly. The free energy between the MM and QM/MM is then obtained using thermodynamic integration (TI),

$$\begin{aligned}
\Delta G &= \int_0^1 d\lambda_{QM/MM} \left\langle \frac{\partial U(\lambda_{QM/MM})}{\partial \lambda_{QM/MM}} \right\rangle_{\lambda_{QM/MM}} \\
\frac{\partial U(\lambda_{QM/MM})}{\partial \lambda_{QM/MM}} &= U_{MM+QM^{int}_{Coul}} - U_{MM} \\
\Delta G &= \int_0^1 d\lambda_{QM/MM} \left\langle U_{MM+QM^{int}_{Coul}} - U_{MM} \right\rangle_{\lambda_{QM/MM}} \qquad (6.6) \\
&= \int_0^1 d\lambda_{QM/MM} \langle U^{int}_{QM_{Coul}} - U^{int}_{MM_{Coul}} \rangle_{\lambda_{QM/MM}} . \qquad (6.7)
\end{aligned}$$

Each lambda window is built up by running classical MD simulations and applying the acceptance test shown in equation 6.2. To clarify, no MD simulation that involves a mix of MM and QM/MM are needed.

6.2.1.3 Transitioning to the full QM

We expect that the QM/MM ensemble is a much closer representation of the QM ensemble, so the transition from the QM/MM to QM should be possible with a method which is simpler than the HMC method. Therefore, to perform this perturbation, the energy differences $U_{QM} - U_{MM+QM^{int}_{Coul}}$ are calculated and then the Zwanzig equation is used to calculate the free energy change from QM/MM to full QM.

We have also computed this free energy by using the interaction energy, as defined in equation 2.5, of the full quantum system, in place of the total energy U_{QM}. The full QM interaction energy in DFT contains also the attractive component of dispersion interactions which are given by various types of empirical correction [57, 58]. We have therefore also investigated the effect of changing this empirical dispersion correction between different dispersion methods.

As we are doing the transition to full QM via a QM/MM state, our full thermodynamic cycle has three steps as shown in Figure 6.1. Figure 6.2 shows the whole correction process. This differs from attempts in previous chapters by the inclusion of the QM/MM "step" before moving to the QM.

$$\begin{tikzcd}
A^{solv}_{QM} & A^{solv}_{MM+QM^{int}_{coul}} \arrow[l, "Zwanzig"'] & A^{solv}_{MM} \arrow[l, "TI(HMC)"'] \arrow[r] \arrow[dr, phantom, "I"] & B^{solv}_{MM} \arrow[r, "TI(HMC)"] \arrow[dr, phantom, "II"] & B^{solv}_{MM+QM^{int}_{coul}} \arrow[r, "Zwanzig"] \arrow[dr, phantom, "III"] & B^{solv}_{QM} \\
A^{vac}_{QM} \arrow[u] & A^{vac}_{MM+QM^{int}_{coul}} \arrow[u] \arrow[l] & A^{vac}_{MM} \arrow[u] \arrow[l] \arrow[r] & B^{vac}_{MM} \arrow[u] \arrow[r] & B^{vac}_{MM+QM^{int}_{coul}} \arrow[u] \arrow[r] & B^{vac}_{QM} \arrow[u]
\end{tikzcd}$$

FIGURE 6.1: The three-step thermodynamic cycle. Cycle I shows the classical mutation from ligand A to B, cycle II describes the transition from MM to QM/MM and finally cycle III shows the transition from QM/MM to full QM

6.2.2 Computational details

This chapter aims to validate this new method, by using it in the calculation of hydration free energies, as preparation for application to more challenging host-ligand systems in the future. We applied our method to the calculation of hydration free energies for ethanol, ethane, ethylene glycol, dimethyl ether and propane, each in a simulation cell with explicit waters. These molecules were chosen due to their different degree of polarity, ranging from non-polar (hydrophobic) such as ethane and propane to highly polar (hydrophilic) such as ethylene glycol.

6.2.2.1 Classical Simulations details

The charges for the solute were obtained from AM1-BCC calculations using Antechamber [97] and the force field parameters were taken from the GAFF Force field [23]. The water model used throughout was the TIP3P model [11]. Electrostatics were treated

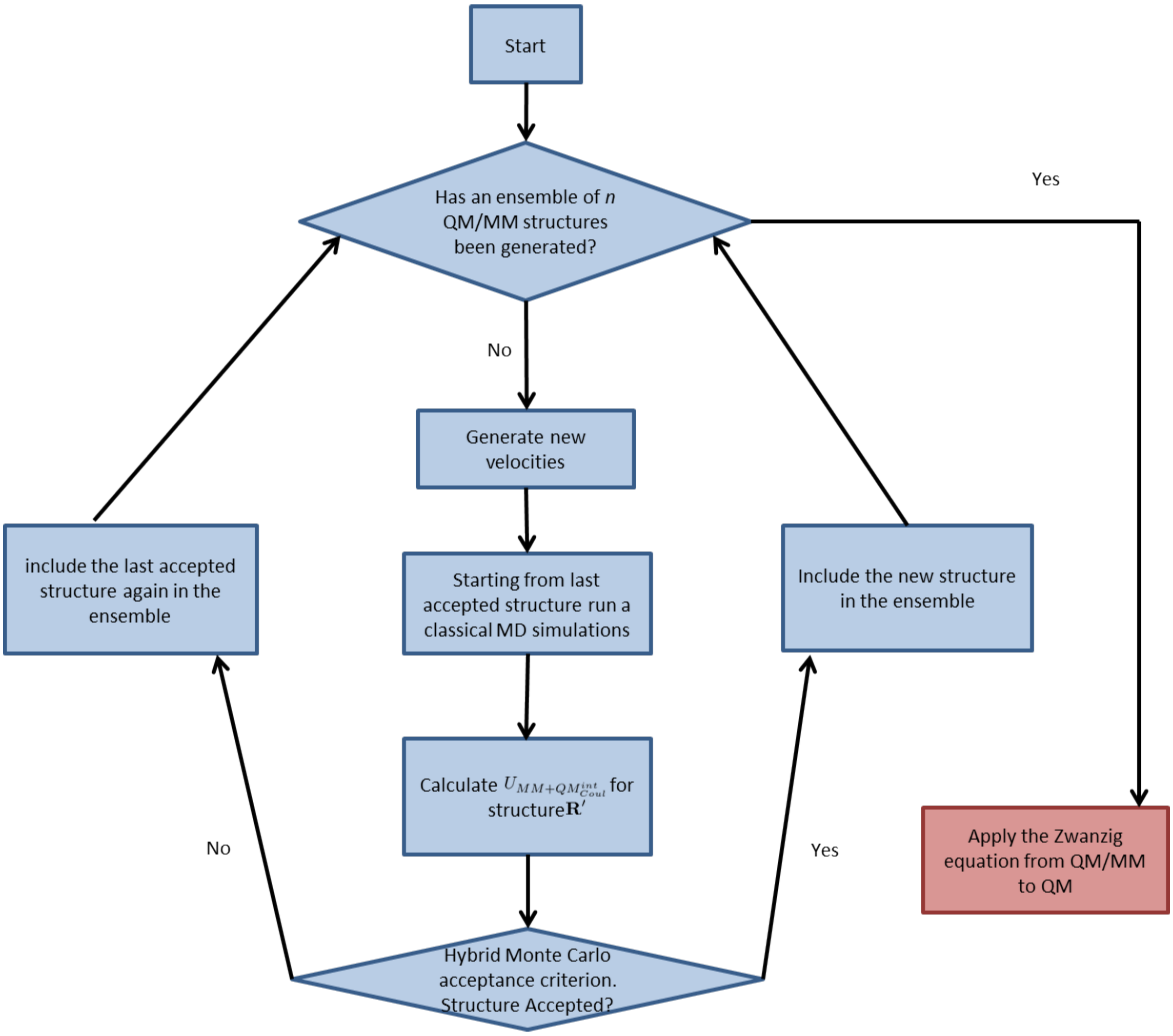

Start
Has an ensemble of n QM/MM structures been generated?
Yes
No
Generate new velocities
include the last accepted structure again in the ensemble
Starting from last accepted structure run a classical MD simulations
Include the new structure in the ensemble
Calculate $U_{MM+QM^{int}_{Coul}}$ for structure$\mathbf{R}'$
Apply the Zwanzig equation from QM/MM to QM
No
Yes
Hybrid Monte Carlo acceptance criterion. Structure Accepted?

FIGURE 6.2: Flow chart for computing steps 2 and 3 of the 3-step thermodynamic cycle, of Figure 6.1.

with PME and a cut-off of 8Å was used for Van der Waals interactions. All classical MD simulations were performed within the double precision version of Gromacs v4.6.5 [107], to allow us to use the Velocity Verlet integrator as access to kinetic energies is required for the HMC method. The double precision was necessary to ensure good energy conservation in the NVE simulations.

Each ligand was solvated with 450 waters and the whole system was equilibrated using the following procedure. Structures were initially minimised with the steepest descent algorithm for 5000 steps. Following this a 500 ps simulation in the canonical ensemble was run using a timestep of 1 fs and the Berendsen thermostat was used to heat the system raising the temperature linearly from 100K to 300K. Finally a 1 ns isothermal-isobaric simulation was performed using the Berendsen barostat. After equilibration the box size for each ligand was around 24 Å^3.

Using the NPT equilibrated structures, further equilibration was then applied using the HMC method where structures were accepted from the NVE ensemble into a classical NVT ensemble for 100 HMC steps. As the MD simulations used within the HMC method were run in the microcanonical ensemble, we expect to have an acceptance rate of 100% as the total Hamiltonian energy should be constant. However, due to fluctuations within the total energy during the NVE simulation, this is commonly not the case. To minimise these fluctuations a timestep of 0.25 fs was used. The length of these MD simulations was determined by tests described in section 6.2.3.

Classical relative free energies between systems were calculated using TI. 17 classical λ_{MM} windows were used, mutating the charges and VdW interactions at the same time using soft-core potentials. The classical λ_{MM} windows had the values 0.0, 0.002, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99, 0.998, 1.0, where the small λ_{MM} differences at the end points are to tackle discontinuities. Each classical λ_{MM} window underwent an equilibration procedure which is the same as that mentioned above, with the exception that the NPT simulation to equilibrate the box size was 500ps. Following this, a production simulation of 2ns was run. Errors associated with the TI free energies were calculated by hysteresis and the free energies are the average of the forward and reverse calculations.

6.2.2.2 Quantum simulations details

Following the classical 100 step equilibration within the HMC method, we switched from going from NVE (MM) to NVT (MM) to going from NVE (MM) to NVT (QM/MM). The first 100 steps of this process were also taken as equilibration where the QM/MM energy was computed according to equation 6.3. After this, 500 steps were run and counted as production, for the transition to QM/MM. This process was repeated, accepting to different values of $\lambda_{QM/MM}$. We used three values, $\lambda_{QM/MM} = 0, 0.5, 1$.

Each quantum simulation was performed using ONETEP [54]. 4 NGWFs were used on heavy atoms and 1 on hydrogen, all with a radius of 8.0 a_0. Calculations used a psinc kinetic energy cutoff of 800eV. Embedding charges[98] were used to represent water within the calculations, so only the ligand was fully represented by QM. Only the electrostatics were corrected by quantum mechanics, so dispersion was included at the classical level, as in equation 6.3.

The fully quantum calculations for cycle III of the thermodynamic cycle of Figure 6.1 were obtained by restoring the embedding charges to explicit atoms, keeping all other parameters the same with the exception of the dispersion component which, within the total energy perturbation, was treated by a damped London potential [57] using the "Elstner" method[59]. Within the interaction energy perturbation approach the "Elstner"[59], "Grimme D2"[58], "Grimme D3" [115] and the dispersion component of the force field were all tested. When the force field dispersion was used within the correction, this was extracted from the classical simulation and used within the QM interaction energy.

6.2.3 Determining the length of the MD simulations used within the HMC method

In order to optimise the length of the MD simulations required to run the HMC method with a reasonable acceptance rate, the MD runs were increased progressively from 0.1 ps to 1 ns. This was applied to ethanol in 450 waters and was repeated for two runs to ensure convergence. Each HMC run consisted of 500 production steps following the equilibration procedure described in subsection 6.2.2.2. The results can be seen in Table 6.1. The "Energy" refers to the average energy differences ($U^{int}_{QM_{Coul}} - U^{int}_{MM_{Coul}}$) at $\lambda_{QM/MM} = 1$. These energy differences are required for TI (equation 6.7). By examining the energy differences at $\lambda_{QM/MM} = 1$ where the acceptance will be lowest, we can confirm the acceptance at other windows will also be sufficient.

Table 6.1: Determining the MD simulation length within the HMC method for ethanol in 450 waters. Energy is the average of $(U^{int}_{QM_{Coul}} - U^{int}_{MM_{Coul}})$ at $\lambda_{QM/MM} = 1$

	Energy (kJ/mol) / Acceptance (%)	
HMC step size (ps)	Run 1	Run 2
0.1	-16.08 / 14.0	-16.90 / 17.5
1	-15.46 / 16.2	-16.56 / 11.5
10	-16.01 / 13.6	-14.99 / 13.3
100	-16.78 / 8.6	-14.27 / 16.6
1000	-13.64 / 21.6	-16.21 / 11.2

The largest difference between two HMC runs shown is 2.51 kJ/mol which is around thermal error (k_BT, 2.5 kJ/mol). Similarly there is only a small amount of error between simulations of different lengths. This suggests that the energy difference is essentially independent of the length of the MD simulation used for this system. In order to ensure uncorrelated snapshots while keeping the length of CPU time required for each MD manageable, 100 ps was chosen as the HMC step size, for all our subsequent HMC simulations. The 1000 ps simulations show no substantial improvement on the 100 ps run, while taking an order of magnitude longer to run.

6.3 Results and Discussion

6.3.1 Classical TI

The classical free energy of mutation (relative hydration free energies) between our ligands was calculated using ethanol as the reference. The values are shown in Table 6.4 in the Classical column. All the experimental values were obtained from reference [116] with the exception of that for ethylene glycol, which was taken from reference [117].

The mutation to propane shows the largest hysteresis which is 1.2 kJ/mol, which can be attributed either to the force field or to the introduction of two additional atoms, whereas all other mutations involved either one or no additional atoms. All calculated free energies were within thermal error (2.5 kJ/mol) from the experimental free energies, with the exception of ethanol to ethane which, while being a well converged mutation, is 7.2 kJ/mol away from the experimental value, pointing to force field limitations.

6.3.2 Calculating the free energy of mutation from MM to QM/MM

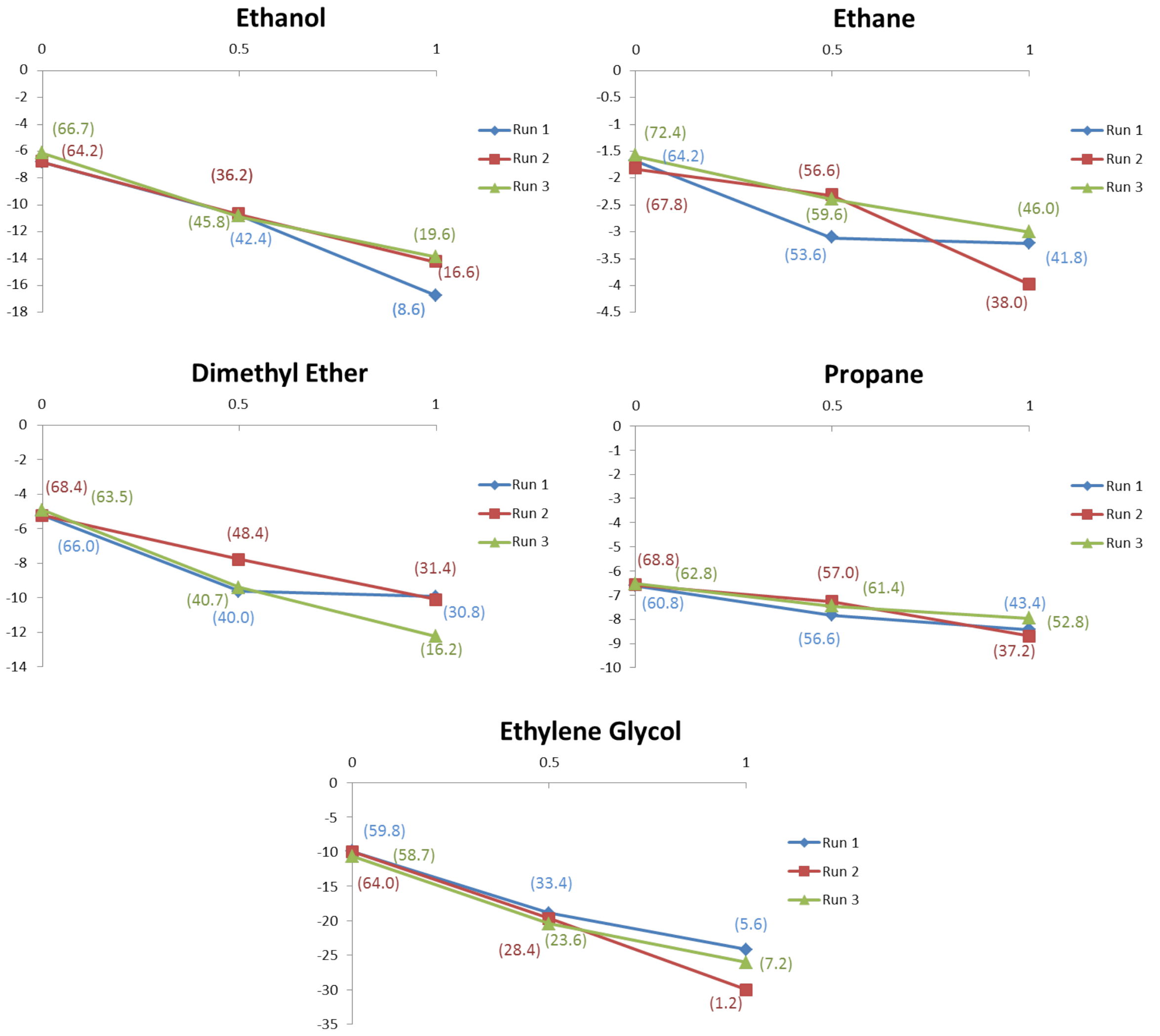


FIGURE 6.3: PMFs for moving between U_{MM} and $U_{MM+QM^{int}_{Coul}}$. Each point shows the average energy difference required for equation 6.7 in kJ/mol. These energy differences were calculated for three $\lambda_{QM/MM}$ windows for each system. The value in brackets shows the acceptance.

The resulting $dV/d\lambda_{QM/MM}$ values as a function of $\lambda_{QM/MM}$ value from the application of the HMC method to all test systems are shown in figure 6.3. Three runs were performed for each ligand and in each case they are consistent and converge to the same value.

The largest differences between same $\lambda_{QM/MM}$ window are shown by ethanol and ethylene glycol. In the case of ethanol the largest difference of 2.5 kJ/mol between the three runs is observed at $\lambda_{QM/MM} = 1$ and the difference for ethylene glycol is much larger at 5.8 kJ/mol again at $\lambda_{QM/MM} = 1$. This can be explained by the presence of the hydroxyl groups forming much stronger bonds with the surrounding water than any of the other ligands used, leading to low acceptance. Indeed this is reflected by the values shown within the graph. Ethane shows the smallest energy difference in going from $\lambda_{QM/MM} = 0$ to $\lambda_{QM/MM} = 1$. This can be expected as it is the most apolar, meaning that the quantum Coulombic contribution to the interaction energy was very close to the classical equivalent. Following this trend, propane is the next apolar, thus has the next smallest energy difference, then dimethyl ether, followed by ethanol, then ethylene glycol.

This trend is also reflected within the acceptance ratios (shown in brackets within figure 6.3). Again, the lowest acceptance is found when applying the HMC method to ethylene glycol, with an acceptance below 10%. This low acceptance is caused by large mismatch between the energy differences, which shows an inconsistency between the QM and MM electrostatic interaction energies. This fluctuation of energy differences could be explained by the strong electronic polarisation of polar ligands. Polarisation is implicitly included within the AMBER force field and explicitly within the QM/MM description. Implicit polarisation may be inaccurate in a chemical environment that varies from that of water (e.g. a binding pocket), but the force field nevertheless shows excellent correlation with experimental hydration free energies (Classical column, Table 6.4).

In order to assess the polarisation effect on these ligands, the dipole moments of structures that were accepted to the QM/MM ensemble for dimethyl ether and ethylene glycol were calculated both classically and within ONETEP. The classical dipole moment was calculated once per structure, given that the force field is not polarisable, so the dipole does not change between vacuum and solvent, but the quantum dipole moment was calculated twice, once in solvent (embedding point charges) and once in vacuum. The results are shown in Table 6.3.2 for 5 structures for each molecule that span the range of the dipole moments obtained by each calculation method.

Table 6.2: Magnitude of the dipole moments in eÅ for five example structures of ethylene glycol and dimethyl ether accepted into the QM/MM ensemble

Ligand	Structure	QM/MM Solvated	QM/MM Vacuum	Classical
Ethylene glycol	1	0.457	0.340	0.313
	2	0.635	0.530	0.579
	3	0.725	0.566	0.515
	4	0.720	0.492	0.641
	5	0.895	0.735	0.460
Dimethyl ether	1	0.237	0.121	0.338
	2	0.304	0.218	0.378
	3	0.300	0.230	0.371
	4	0.318	0.229	0.320
	5	0.331	0.220	0.329

The classical dipole moment for dimethyl ether is fairly constant between the five sampled snapshots, whereas the dipole moments for ethylene glycol fluctuate. The discrepancy between the classical dipole moments and those of the QM/MM system in solvent are markedly larger (0.2-0.4 eÅ) for the ethylene glycol than for dimethyl ether (discrepancies of less than 0.1 eÅ). In fact for dimethyl ether two of the classical dipole moments correctly match the quantum dipole moments, and with the exception of structure 4 and 5, the quantum dipole moments for dimethyl ether within solvent match the experimental value of 0.271 eÅ extremely well, whereas classical mechanics overestimates it. Another interesting comparison between the methods can be made with a TIP3P water molecule, where the quantum dipole moment in vacuum is calculated to be 0.380 eÅ and the classical dipole moment is 0.489 eÅ. The experimental value is 0.385 eÅ, showing again that the quantum value is extremely close and the classical value, again, much less accurate. Although, it should be noted that the dipole of an isolated TIP3P water molecule is higher in order to better match the bulk properties of water.

These results highlight the fact that only by using quantum corrections we can account for the explicit polarisation, as the force field can at best describe it in an implicit (average) manner, and indicate that the lower HMC acceptance for ethylene glycol may be a consequence of the large discrepancy in polarisation between the classical and quantum descriptions.

Integrating each energy vs $\lambda_{QM/MM}$ value curve gives the free energy, in accordance with equation 6.7. These free energy values are shown in Table 6.3 for each of the three runs.

Table 6.3: Free energy of changing from U_{MM} to $U_{MM+QM^{int}_{Coul}}$. All values shown are in kJ/mol, the difference is the largest difference between the three runs

	Run 1	Run 2	Run 3	Range
Ethanol	-11.29	-10.62	-10.45	0.84
Ethane	-2.78	-2.61	-2.35	0.43
Ethylene Glycol	-17.97	-19.82	-19.38	1.86
Dimethyl Ether	-8.60	-7.72	-8.99	1.27
Propane	-7.67	-7.45	-7.35	0.32

The magnitude of these free energies again follows the trend of the polarity of the molecules. The differences show the convergence between the free energies. In every case convergence has been achieved. The largest difference is present for ethylene glycol of 1.86 kJ/mol, however this value is still within thermal error (2.5 kJ/mol). By combining these free energies with the classical free energies shown in Table 6.4 under the Classical column, according to cycles I and II of the three-step thermodynamic cycle (Figure 6.1) we obtain the QM/MM corrected free energies shown in Table 6.4 under the $MM + QM^{int}_{Coul}$ corrected column.

The addition of the QM correction due to the MM to QM/MM transition to the classical relative free energies between ethanol and ethane shows a definite improvement of the free energy which was badly underestimated by the force field. The corrections for ethylene glycol and propane move the free energies in the direction of improvement although they overshoot the experimental values. The free energy for dimethyl ether is shifted further away from the experimental value, as the classical free energy was extremely close already. These results are encouraging and demonstrate the effect of QM/MM derived polarisation on the free energies as an intermediate step towards the final stage where we will introduce the full QM description.

6.3.3 Calculating the free energy of mutation from QM/MM to QM

Once a QM/MM ensemble of 500 structures was built up, the next step was the perturbation to the fully quantum system (see cycle II in figure 6.1). This was performed in two ways: Firstly by using the difference in total potential energies (equation 6.8) and secondly by using the difference in interaction energies (equation 6.9). In both cases, this perturbation was performed by using the Zwanzig equation (equation 3.1).

$$\Delta U^{total} = U_{QM} - U_{MM+QM^{int}_{Coul}} \tag{6.8}$$

$$\Delta U^{int} = U^{int}_{QM} - U^{int}_{MM+QM^{int}_{Coul}}. \tag{6.9}$$

Where U^{int}_{QM} is the interaction energy at the full QM level and $U^{int}_{MM+QM^{int}_{Coul}}$ is the interaction energy at the QM/MM level.

6.3.4 Total Energy Perturbation

The final calculated free energy when using total enegies (equation 6.8) shows no consistency between the two runs which demonstrates lack of convergence of the exponential average of the Zwanzig equation. This is demonstrated for three of the ligands in the left panel of Figure 6.4. Figure 6.4 shows the free energy as a function of the number of snapshots, i.e. a running exponential average as the number of snapshots is increased. These graphs show that the free energy is affected dramatically by individual snapshots. The graphs show large variation between the two runs for each ligand. For example, ethylene glycol a difference of 35.17 kJ/mol is present after 500 snapshots, while for dimethyl ether the difference between the runs is 0.33 kJ/mol. The small difference for dimethyl ether suggests convergence, however, if Figure 6.4 is examined it is clear that large "jumps" in the free energy are present and there is no guarantee that they won't affect the results if we were to run more than 500 snapshots. This is described as "sawtooth" sampling and is characteristic of a lack of overlap between the configuration space[99]. This observation has also been made when moving between a purely classical ensemble and a quantum ensemble by Cave-Ayland et al. [118]. They explain that this lack of configuration space overlap is due to differences in the intra-molecular degrees of freedom. Therefore by excluding all intra-molecular terms (i.e. using interaction

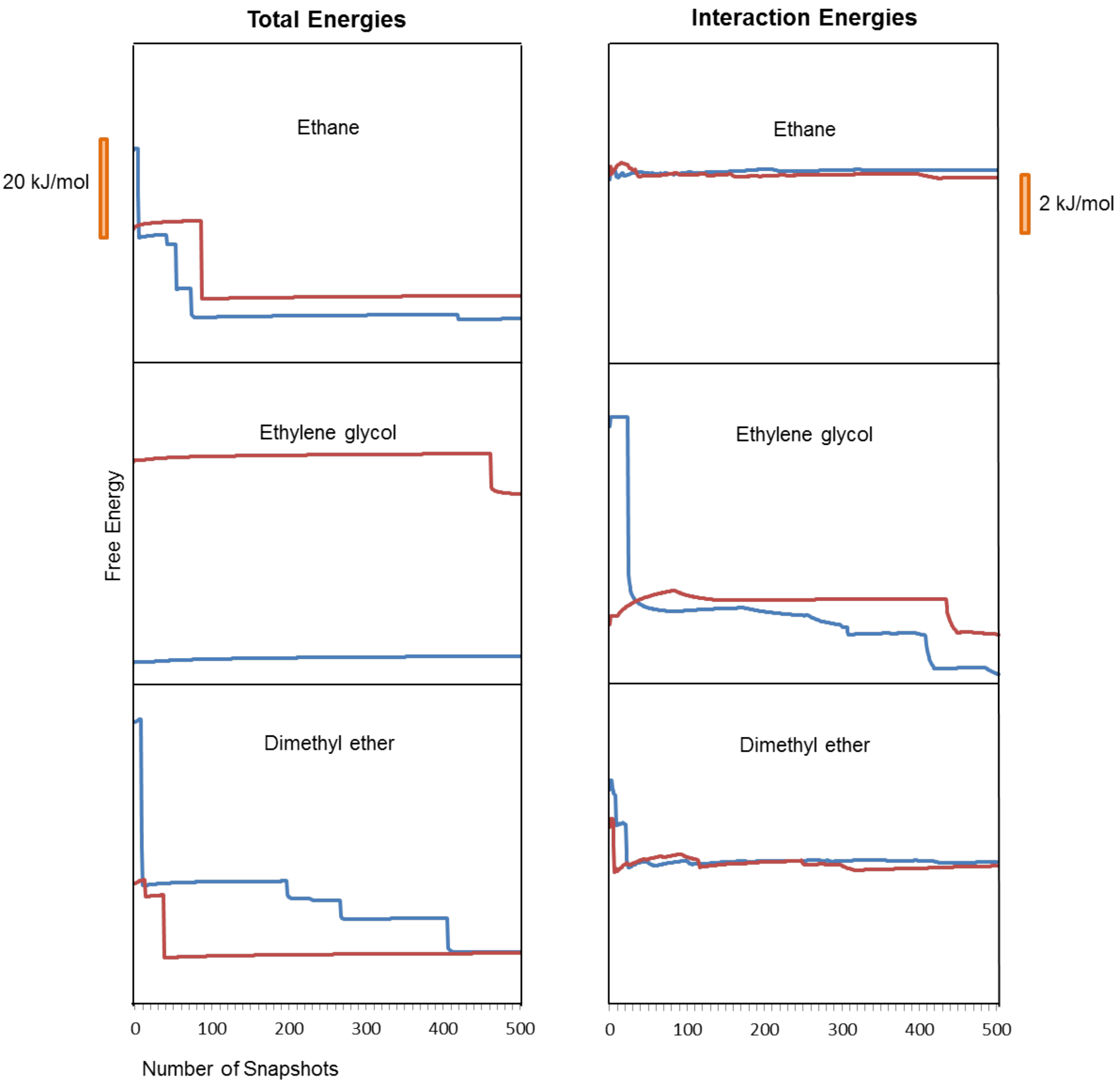


Figure 6.4: Free energy calculated with the Zwanzig equation as a function of the number of snapshots included when perturbing between the $MM + QM_{Coul}^{int}$ and QM ensembles for ethylene glycol, dimethyl ether and ethane. The blue line represents Run 1 and the red represents Run 2. For clarity, we don't provide an absolute energy scale for the y axis, as only the relative energies between each run are important. Three ligands are shown as a representative of the best and worst examples.

energy-based corrections) convergence can be achieved. Indeed this is what is observed here and will be covered in more detail in the next section.

To complete the three-step thermodynamic cycle when using total energies, an additional calculation must be performed of the ligand in vacuum. This is displayed in the lower half of cycle II and III in figure 6.1. However, the initial MM+QM$_{Coul}^{int}$ correction required for the ligand in vacuum (cycle II) would essentially be a standard classical MD as the interaction energy in vacuum is zero, and with the system size being relatively small, the fluctuation in the total Hamiltonian energy would be small, leading to a high acceptance. The ligand energy differences required for the mutation from MM to QM/MM in the Zwanzig equation (cycle III) were taken from values already calculated within the top section of cycle II. For each ligand the calculated free energies converge to less than 0.2 kJ/mol between the two runs. This result combined with the fact that our water molecules are rigid leads to the conclusion that the lack of overlap between the configuration space should be attributed to the inter-molecular interactions between the water molecules. While this appears to be the case for the small rigid ligands we use in this study for larger ligands, with more intra-molecular degrees of freedom, we would expect that the lack of configuration overlap would be caused also by ligand intra-molecular terms.

6.3.5 Interaction Energy Perturbation

Interaction energies are commonly used when applying the Zwanzig equation as it has been often observed that convergence with total energies is problematic[93]. The smaller magnitudes of interaction energies and the better overlap of the energies lead to better convergence. This can be understood by the notion that the free energy of binding is primarily delivered by the interactions between the ligand and receptor and these are improved by the quantum description.

When using interaction energies to go from QM/MM to full QM (equation 6.9), a much higher degree of convergence is obtained. This is shown in Figure 6.4. The largest difference between the two runs is 1.25 kJ/mol (ethylene glycol), well within thermal error (2.5 kJ/mol). These free energies can then be combined with the cycle II free energies from the three-step thermodynamic cycle (Figure 6.1). The final corrected free energies can be seen in Table 6.4 under the column labelled "Elstner". In every case the free energies do not improve. Dimethyl ether and ethylene glycol stay essentially unchanged. Thus, for the more polar ligands, where the main driving force behind the binding energies is electrostatic (e.g. ethylene glycol and dimethyl ether) the correction from QM/MM to QM is small. This is in contrast with the non-polar ligands, where dispersion is the prominent interaction, where the corrections increase the relative hydration free energies. Overall the cycle II corrections improve the correlation with the experimental free energies, however little to no improvement upon cycle II is achieved when applying the final cycle (cycle III) in figure 6.1.

We have further examined the method used to calculate the dispersion. In our first attempt to implement cycle III corrections the Elstner method was used. We have then also examined keeping classical dispersion (as calculated by the force field) and alternatively using Grimme's D2 correction. Where the Elstner and Grimme D2 corrections use slightly different damping functions, see reference [119] for more information. If the dispersion calculated by the force field is used, it is effectively cancelled out within the Zwanzig equation. This is because the interaction energy at the QM/MM level can be divided into two components, the electrostatics and dispersion.

$$\Delta U^{int} = \Delta U^{elec} + \Delta U^{disp} \tag{6.10}$$

The electrostatic term came from the QM/MM correction and the dispersion term comes from the force field. Similarly, the interaction energy at the full QM description can be split into the intrinsic DFT terms (e.g. the Coulomb and exchange) and the empirical dispersion correction. Thus, in the case where we retain the force field dispersion in the QM description, it will cancel out with the dispersion included in QM/MM as the two are the same.

The use of the force field dispersion significantly improves the free energies calculated for cycle II for every single ligand. Grimme's D2 correction causes the largest errors between the different dispersion approaches, to the point that overall it is a deterioration of accuracy compared to the classical and to the QM/MM results. We have also tried the more advanced Grimme D3 approach but the results obtained are essentially the same as with D2 method. In order to have a single measure of accuracy for each method, the RMS error for each method was calculated and is shown in Table 6.4. This was calculated with respect to the experimental values. It is clear that QM/MM and QM with force field dispersion improves upon the classical result with the QM with force field dispersion producing dramatic improvement with RMS error of only 2.05 kJ/mol.

TABLE 6.4: Relative hydration free energies using the experimental value for ethanol (-20.92 kJ/mol) as a reference [116]. The specified standard errors in the Classical calculation (Column 3) correspond to the hysteresis of independent forward and backwards classical TI calculations of the corresponding systems. The specified standard errors in the QM calculations (columns 4-8) correspond to the standard error of the mean resulting from the accepted HMC runs with QM corrections. The "RMS error" and "Max error" correspond to the respective differences between the calculated values and the experimental values.

	Free Energy (kJ/mol)						
Ligand	Experimental	Classical	MM+QM$^{int}_{Coul}$ corrected	Elstner	Force field disp	Grimme D2	Grimme D3
Ethane	7.66[116]	0.53 ± 0.23	8.71 ± 1.27	10.31 ± 0.18	7.52 ± 0.54	11.07 ± 0.58	10.79 ± 0.67
Ethylene Glycol	-38.91[117]	-35.91 ± 0.20	-44.18 ± 2.70	-43.59 ± 1.52	-42.61 ± 3.20	-44.09 ± 1.67	-44.14 ± 1.60
Dimethyl Ether	-7.99[116]	-7.93±0.02	-5.58 ± 2.11	-5.83 ± 0.26	-7.76 ± 0.76	-5.17 ± 0.54	-5.39 ± 0.42
Propane	8.20[116]	6.93 ± 1.18	10.22 ± 1.16	13.74 ± 0.16	6.47 ± 0.50	14.56 ± 0.74	14.06 ± 0.54
RMS error		3.93	3.11	4.04	2.05	4.66	4.42
Max Error		7.15	5.27	5.54	3.70	6.36	5.86

Finally we investigate how our method depends on the choice of exchange-correlation functional. For this purpose we have tried the LDA and BLYP exchange-correlation functionals in addition to the PBE functional that we have used up to now, for a single mutation (ethanol to ethane) and examined how these changes affect free energies at the QM/MM and at the full QM descriptions. The results are shown in Table 6.5.

Table 6.5: Free energy corrections with different exchange-correlation functionals applied to the relative hydration free energy between ethanol and ethane. The experimental relative hydration free energy is 7.66[116] kJ/mol and the classical free energy is 0.53 kJ/mol.

	Free Energy (kJ/mol)				
Functional	MM+QM$^{int}_{Coul}$ corrected	Elstner	Force field disp	Grimme D2	Grimme D3
PBE	8.71	10.31	7.52	11.07	10.79
LDA	9.46	15.48			
BLYP	8.70	9.07	7.13	9.81	9.24

These results indicate that within the GGA approximation, switching between PBE and BLYP plays a small role in obtaining accurate corrections. This is indicates similar behaviour between GGA functionals. For the LDA the result for the QM/MM description is 0.7 kJ/mol worse than the GGA functionals but really erroneous for the full QM description with an error of about 8 kJ/mol with respect to experiment. We should note that we have not used any dispersion correction for the LDA calculations as the LDA method intrinsically includes spurious attractive interactions that play a role of dispersion in this case or in any case make the empirical dispersion corrections not applicable to LDA.

Previously (Table 6.4), the results showed that using the force field dispersion yields the best corrections and here the same trend can be observed again for PBE and BLYP.

6.4 Conclusions

We have presented a "stepping stone" approach for computing QM corrections to MM free energies of binding that aims to overcome the convergence difficulties of similar approaches which are based on a single-step free energy perturbation from the classical to the quantum system. Our approach includes two stages: In the first stage we gradually mutate the MM system to QM/MM using thermodynamic integration (TI) on intermediate ensembles generated via hybrid Monte Carlo simulations. This stage accommodates most of the change in polarisation associated with the MM to QM mutation. As a result the second stage, which is a single-step QM/MM to full QM mutation actually converges well. Here, we validated our method on the calculation of hydration free energies for a set of ligands with different polarities. We found that the mutation from MM to QM/MM is a definite improvement in the relative hydration free energies with respect to classical TI results, and the stage 2 correction where we mutate to the full QM ensemble produces further and more substantial improvement reducing the classical max and RMS errors by a factor of 2. The approach is quite sensitive to the choice of dispersion model but less so in the choice of GGA exchange correlation functional. This method could be further tested on protein-ligand binding in the future.

Chapter 7

QM-PBSA Simulations on Trypsin with Benzamidine Derivatives

7.1 Introduction

The calculation of accurate binding free energies is of great importance within computational chemistry [120, 121, 122, 123]. As such there are many different methods used to calculate the binding affinity within protein-ligand complexes. These range from the statistically rigorous, like thermodynamic integration [124] and free energy perturbation [78] to other more computationally efficient, but less rigorous approaches such as MM-PBSA (molecular mechanics - Poisson-Boltzmann surface area) [79].

MM-PBSA is an example of a continuum method, where solvent effects are included by approximating the solvent as a polarisable continuum. One of the key features of PBSA approaches is that the linearised Poisson-Boltzmann equation (equation 7.1) is used to approximate the polar solvent effects.

$$\nabla\epsilon(\mathbf{r}) \cdot \nabla\phi(\mathbf{r}) - \kappa'\phi(\mathbf{r}) = -4\pi\rho(\mathbf{r}). \tag{7.1}$$

Where $\epsilon(\mathbf{r})$ is the dielectric permittivity, $\phi(\mathbf{r})$ is the electrostatic potential, κ' is the Debye-Hückel parameter and $\rho(\mathbf{r})$ is the charge density. The non-polar interactions are then approximated, and typically include the Van der Waals and the cavitation energy.

The combination of the polar and non-polar terms give the solvation free energy. This can be combined with the entropy and interaction energy to give the free energy of binding. These energies are obtained as averages over snapshots which require a molecular dynamics (MD) simulation in explicit solvent to be run, which is highly dependent on the parameterisation of the force field used. The development of force fields has inherent limitations which mean that ligands, especially when they contain non-standard groups that are difficult to parameterise, may not be accurately described. In addition to this, the widely-used fixed charge force fields cannot explicitly model quantum inherent interactions such as the electronic charge transfer and polarisation, although force fields with such capabilities are being developed. In an ideal world the classical MD would be replaced with *ab initio* MD, but this is far too computationally expensive to be feasible for the time scales required for free energy calculations. As a compromise between speed and accuracy, we can develop a QM-PBSA approach by replacing the classical interaction energy within the free energy of binding calculation with the quantum interaction energy. This work builds on the work of Fox et al. [125] who calculated the free energy by using QM-PBSA on the T4 lysozyme double mutant L99A/M102Q. This double mutant contains an artificial buried polar cavity within the protein, which was then used to bind to a variety of simple ligands. Here, we apply QM-PBSA on a more challenging and complex system, which is more biologically relevant and contains larger ligands bound within a solvent exposed cavity. In addition, we evaluate and compare different ways to obtain the most accurate classical results before applying the quantum correction.

To obtain accurate classical free energies of binding two of the most well parameterised force fields were tested, the ff99SB[21] and the ff14SB[22]. The ff14SB was further tested by restraining a section of the protein within the MD simulations. By allowing only the region around the binding pocket to move, the noise within the free energy is expected to be reduced. The idea of cancelling noise in this fashion is not novel. For example, Bradshaw et al. [126] used a noise-cancelling approach in MM-PBSA when applied to different mutants around protein-protein interfaces. When applying this method, only a region around the binding pocket could move independently, the rest of the protein is common between ligands, in a dual-topology inspired variant to MM-PBSA.

The calculation of the quantum energy on the whole complex would not be possible using conventional DFT approaches. These calculations can, however, be performed in a linear scaling code like ONETEP [54]. The solvation free energy is calculated by the implicit solvation model within ONETEP, more details can be found in reference [127].

For our study we have selected trypsin bound to benzamidine derivatives, which is a well studied system [128, 129, 130, 131]. Trypsin is a serine protease. Serine proteases are a group of enzymes whose main biological function is to cleave peptide bonds [132]. This is a key step in digestion and ensures that polypeptides are kept small enough to be absorbed by the small intestine. Although the binding affinity between trypsin and benzamidine derivatives has been investigated many times previously, these ligands are non-standard residues which means they are not parameterised as well as the amino acids present within the protein are. This can further compromise the accuracy of the calculated binding energies. Seven ligands were chosen for our study and are shown in figure 7.1, selected for the variety of chemical behaviours, including halogens, amines and nitrites, and wide range of experimental binding energies, from -4.7 kcal/mol to -7.6 kcal/mol.

Previous studies on this system have employed a variety of methods and achieved good correlation with experiment, in most cases predicting the binding energy to within 1 kcal/mol of the experimental values. In 1997 Essex et al.[133] used Monte Carlo simulations with free energy perturbation to calculate free energies and found that polarisation plays a big role in the binding between trypsin and benzamidine finding that the more polar the ligand is, the weaker the binding energy due to the more polar ligands being stabilised better in solvent. Radmer et al.[134] then used the Cornell force field with three different methods. These included using a single simulation to estimate the binding energies, using the first derivatives of the binding free energy and the PROFEC method

(Pictoral Representation of Free Energy Components). In 2008 Jiao et al.[135] investigated the effect of polarisation when calculating the binding free energies and found that it is of critical importance to include explicit polarisation by using the AMOEBA polarisable force field. They then used MM-PMPB/SA (molecular mechanics-polarisable multipole Poisson-Boltzmann/surface area), which uses an end-point continuum calculations paired with a polarisable force field to calculate binding free energies[136] . These results were then used in comparison to the results obtained when using BAR, a method which involved conformational sampling, using the AMOEBA polarisable force field, to a much higher degree[137]. They found that when using the alchemical mutation, the entropy has a positive effect on the results, where the opposite is true when sampling is based on MD only. They suggest that the alchemical mutation helps to capture more of the configurational entropy than a direct MD simulation. The driving force behind the binding was investigated in Shi et al.[138] and found to be electrostatics. Schwarzl et al.[129] introduced a new method based on MM-PBSA that achieves extremely good correlation with experiment which, however, relies on an empirical scaling factor that affects the Van der Waals interaction before the final free energy is calculated. Further investigation into the use of this method led to the development of applying it in combination with QM/MM[130] and treating the ligand only with QM with the rest of the system represented by MM. In a comparison of methods when using trypsin, including LIE and MM-PBSA, no single method came out as the definitive one to use[139], however, improvements can be found if several methods are combined; Ruiter et al. found that the combination of third power fitting and one step perturbation obtains the more accurate free energies than if either method is to be used in isolation[131]. Chen et al.[140] applied QM/MM MD to sample configurations in order to investigate the binding specificity of trypsin. As mentioned above, *ab initio* is far too computationally expensive, but by applying the method described by Fox et al. [125], where classical mechanics is used to obtain configurations quickly, these structures can then be used in QM calculations to replace the MM terms in MM-PBSA with QM terms.

We have applied QM-PBSA on trypsin with benzamidine derivatives that represent a variety of chemical behaviours. In the following section, we discuss how the systems were equilibrated and finally how the QM-PBSA method was applied.

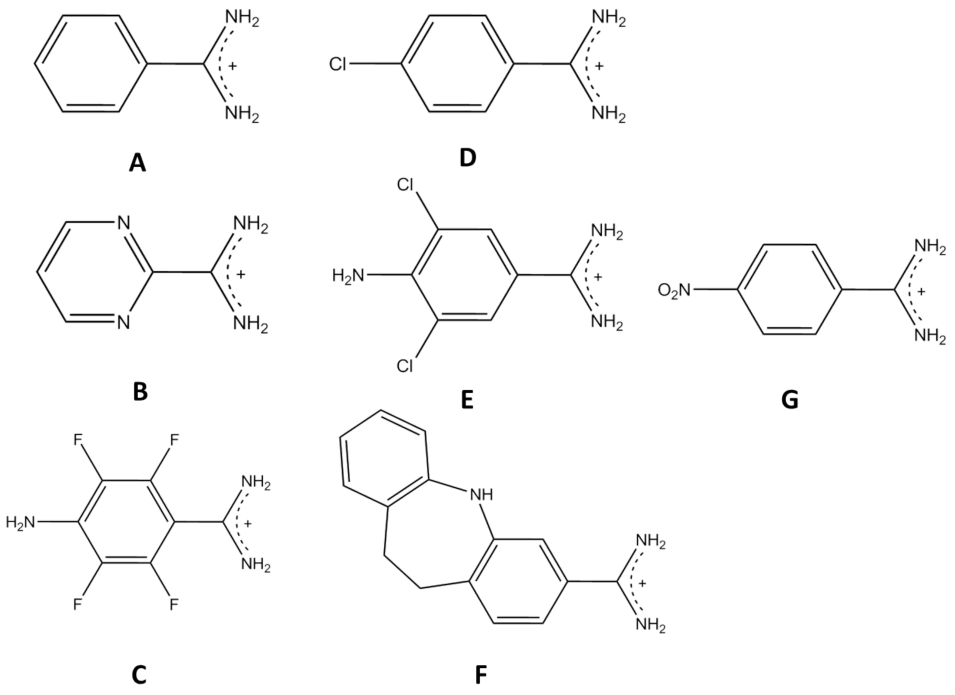


Figure 7.1: Benzamidine derivative ligands used as ligands for trypsin in this study

7.2 Methods

7.2.1 Trypsin simulations set-up

PDB code 1BTY [141] was used as an initial structure for the trypsin/benzamidine complex. This structure was then used as a starting point for all complex systems. The delta protonation state for histidine 57 (HID) was used, however, Schwarzl et al. [129] show that the protonation state will not affect the binding free energy by repeating the calculations with different protonation states. The system was then solvated with TIP3P water in a waterbox with water thickness of at least 10 Å from the protein and neutralised with nine Cl^- ions (eight to neutralise the protein and one to neutralise the positive charge on the ligand). The charges for all small molecules were obtained by using AM1-bcc charges and initial atom types were obtained from antechamber [97]. Atomtypes for the nitrogen within the amine groups (which were incorrectly assigned by Antechamber) were manually changed from 'nh' (amine connected to aromatic ring) to 'n' (sp2 amide) due to the delocalised bond present over the amidine group, as verified by B3LYP/cc-pVTZ calculations.

Each complex underwent a sixteen step equilibration process. First the structure was minimised by 5000 steps of steepest descent followed by 5000 steps of conjugate gradient with a 1000 kcal/mol-Å^2 restraint applied to all non-hydrogen atoms with a 9 Å nonbonded cutoff. This step was then repeated, removing the restraint from the oxygen within the water molecules. Following this the system was heated from 100K to 300K with the temperature increasing linearly over 500ps in the canonical ensemble using the Langevin thermostat [64] with a collision frequency of 3 ps^{-1}. The SHAKE algorithm was applied to allow a timestep of 2fs timestep and the same 1000 kcal/(molÅ^2) restraint as in the previous step was applied. A simulation within the isothermal-isobaric ensemble was then run for 200ps with the same restraints as previous steps still applied. The system was then cooled down to 100K in the canonical ensemble. The system was then minimised again, as in the second step, lowering the restraint to 500 kcal/(molÅ^2). This was repeated several times, lowering the restraint each time, to 100, 50, 20, 10, 5, 2, 0 kcal/(molÅ^2). The system was then heated smoothly to 300K with no restraints applied. Following this a further 500ps were run in the isothermal-isobaric ensemble.

This equilibration was performed for each ligand and for each method (ff99SB force field, ff14SB force field and applying restraints to the ff14SB force field). The process that involved restraining the backbone atoms throughout the production run (ff14SB restrained) followed the above equilibration procedure. However, whereas the restraints were removed within the other procedures, a 2 kcal/(molÅ^2) restraint was applied and the final structure from the 14th step (the final minimisation) was used as a reference. This simulation protocol was explored as a way of cancelling noise, in a similar manner to that described by Bradshaw et al. [126].

7.2.2 MM-PBSA calculations

Once equilibrated, a production calculation for each ligand was performed for 20ns within the canonical ensemble, again using the Langevin thermostat. The RMSD was then examined along with the fluctuation of the total energy to ensure equilibration had been achieved. 1000 snapshots were then extracted and the MM/PBSA technique [79] was applied using a dielectric constant of 1.0 within the protein. The PBSA energy for each snapshot was plotted over time to ensure no "drift" was present.

In addition to the PBSA calculations, 100 evenly distributed snapshots were used to calculate the entropy of the system via normal mode analysis.

7.2.3 QM calculations

Each QM calculation was performed in ONETEP [54]. 4 NGWFs were used to describe non-halogen heavy atoms, for halogens this was increased to 9 NGWFs and 1 NGWF was used for hydrogen. Each NGWF had a radius of 8.0 bohr. The psinc grid spacing was set to be 0.5 bohr in the x, y and z direction which is equal to 827 eV psinc kinetic energy cut off and the PBE exchange-correlation functional was used throughout. The parameters for the implicit solvation model have been validated by Dziedzic et al. [142]. These involved using a dielectric constant of 78.54 to represent the water, the smeared ion width of 0.8 bohr, a discretization order of 8 and a β value of 1.3. Initially calculations on 40 snapshots were performed and these were subsequently increased to 80 snapshots. The final 40 calculations were run by Dr J. Dziedzic. The initial 40 snapshots were taken from the MD simulation at regular intervals of 500ps. When the additional 40 calculations were required, these were taken in between the initial 40, such that an interval of 250ps between snapshots was used.

7.3 Results

7.3.1 Classical Force Field Results

Initially MM-PBSA was performed on all the aforementioned ligands with both the ff99SB and ff14SB force fields, as well as the ff14SB force field with restraints applied. The ff14SB force field is based on the ff99SB, with a corrections such as a lower dependence of side chain parameters on certain backbone conformations, because of this we would expect minimal differences between the values obtained from the unrestrained simulations, indeed, this is what was observed. These results can be seen in table 7.1.

Table 7.1: MM-PBSA values when using 1000 snapshots from the ff99SB and ff14SB (without restraints and with restraints) force fields. The values for each force field have been shifted by a constant so that ligand A matches experiment. In order to demonstrate how close the values are between the two force fields, the unshifted values are also shown in brackets. All values shown are in kcal/mol.

Ligand	Experimental	ff99SB	ff14SB	
			No restraints	Restraints
A	-6.3[129]	-6.30 (-12.22)	-6.30 (-11.39)	-6.30 (-13.32)
B	-4.7[129]	-3.65 (-8.20)	-2.21 (-7.30)	-3.29 (-10.93)
C	-4.8[129]	-4.28 (-10.20)	-4.46 (-9.55)	-9.77 (-17.41)
D	-5.6[143]	-6.23 (-12.15)	-6.68 (-11.77)	-6.24 (-13.88)
E	-7.6[143]	-4.83 (-10.75)	-5.58 (-10.67)	-1.28 (-8.92)
F	-5.7[130]	-8.56 (-14.48)	-9.26 (-14.35)	7.86 (0.22)
G	-4.8[143]	-6.57 (-12.49)	-7.70 (-12.79)	-4.61 (-12.25)

In each case, examining the difference between unshifted (bracketed) free energy of binding, the ff99SB and ff14SB unrestrained runs are within 1 kcal/mol. As explained in the table caption, each value (not in brackets) has been shifted so that ligand A has the value -6.30 kcal/mol. The ff14SB restrained energies, however, are very different to the energies calculated with the ff99SB and ff14SB. These values were subjected to the same convergence tests as the ff99SB and ff14SB, i.e. no drift was present, the values fluctuate around a central point. However, the final free energy values are considerably different than those calculated for the other approaches. Ligands A, B and D show similar results to free energies calculated when using the ff99SB or ff14SB, ligand G differs by around 2 kcal/mol, and all other ligands (C, E and F) differ by over 3.5 kcal/mol. The largest difference occurs for ligand F, which shows a positive free energy. These results show that this method for cancelling noise is not effective.

We observe that neither the ff99SB or the ff14SB match the experimental values consistently. Looking at the shifted values it appears that the ff99SB is an improvement on the ff14SB for ligand B, D, F and G and the ff14SB provides better results for ligand C and E. These shifted values have been affected by the difference within the values for ligand A. Once this is considered, there is no notable improvement between the ff99SB and ff14SB for these ligands.

When compared with the experimental free energies, ligand F provides the worst calculated free energy. However, the free energies within table 7.1 are shown without entropy, which is an approximation as we assume that the entropy component will cancel between ligands. As ligand F is much larger than any other ligand, the entropy approximation will not hold and must be included for this ligand. This will be covered later in this section. Ligand C matches the experimental free energy extremely well, which is a surprising result (and most likely due to cancellation of errors) when considering that four fluorines are present.

As mentioned previously, the entropy was initially calculated using 100 evenly distributed snapshots. The results can be seen in Table 7.2.

The entropy in Table 7.2 shows differences between the ff99SB and ff14SB for all the relative (bracketed) values and for the absolute values for ligand A, B and C. Ligands

Table 7.2: TΔS values (shown in kcal/mol) calculated using 100 snapshots using the mmpbsa nmode functionality within the AMBER tools. The error shown was calculated by bootstrapping with repeats 1000 times. The values in brackets are values shifted so that the TΔS for ligand A is 0 kcal/mol.

Ligand	ff99SB	ff14SB
A	-18.94 (0.00) ± 0.26	-17.66 (0.00) ± 0.29
B	-18.88 (0.06) ± 0.24	-16.98 (0.68) ± 0.25
C	-20.28 (-1.34) ± 0.33	-18.36 (-0.70) ± 0.33
D	-18.92 (0.02) ± 0.26	-18.55 (-0.89) ± 0.63
E	-20.64 (-1.70) ± 0.31	-20.71 (-3.05) ± 0.24
F	-19.20 (-0.26) ± 0.30	-19.43 (-1.77) ± 0.71
G	-20.95 (-2.01) ± 0.25	-20.43 (-2.77) ± 0.26

D, E, F, and G have similar energies between the force fields, with the largest difference of 0.52 kcal/mol. All the errors shown in Table 7.2 were calculated by bootstrapping with repeats, 1000 times. The largest error is present for ligand F ff14SB at 0.71 kcal/-mol, which is slightly higher than the thermal error (k_BT, 0.6 kcal/mol). To ensure convergence and to minimise errors, the number of snapshots included in the entropy was increased from 100 to 200 snapshots for all ligands when using the ff14SB. These values can be seen in Table 7.3.

Table 7.3: Comparison of TΔS (in kcal/mol) as when including 100 snapshots and 200 snapshots, using the ff14SB force field. Error calculated by bootstrapping 1000 times with repeats. The values in brackets have been shifted by a constant, so that the TΔS for ligand A is 0 kcal/mol.

Ligand	100	200
A	-17.66 (0.00) ± 0.29	-17.97 (0.00) ± 0.20
B	-16.98 (0.68) ± 0.25	-17.43 (0.54) ± 0.19
C	-18.36 (-0.70) ± 0.33	-18.62 (-0.64) ± 0.22
D	-18.55 (-0.89) ± 0.63	-18.37 (-0.40) ± 0.18
E	-20.71 (-3.05) ± 0.24	-20.82 (-2.85) ± 0.17
F	-19.43 (-1.77) ± 0.71	-18.87 (-0.89) ± 0.23
G	-20.43 (-2.77) ± 0.26	-20.14 (-2.16) ± 0.21

Although all the entropy values differ slightly, none of the values are drastically different. The largest difference is again present for ligand F, and is 0.56 kcal/mol. As expected the error in every case is lowered, most dramatically in the case of ligand F, where the error is decreased by 0.48 kcal/mol. Given the similarities between the entropy values for the inclusion of 100 and 200 snapshots, and the consistent lowering of errors, we are satisfied that the entropy is converged.

Table 7.4: Combining entropy with the energy values shown in Table 7.1. All values are in kcal/mol and the energies in brackets are from entropy calculations with 200 snapshots for each ligand.

Ligand	Experimental	ff99SB	ff14SB
A	-6.3[129]	-6.30	-6.30 (-6.30)
B	-4.7[129]	-2.34	-2.89 (-2.75)
C	-4.8[129]	-2.94	-3.76 (-3.82)
D	-5.6[143]	-6.25	-5.79 (-6.28)
E	-7.6[143]	-3.13	-2.53 (-2.73)
F	-5.7[130]	-8.3	-7.49 (-8.37)
G	-4.8[143]	-4.56	-4.93 (-5.54)

Table 7.4 shows the combination of the entropy with the energies in Table 7.1. The values for the ff99SB and ff14SB are again very similar, with each value being within 1 kcal/mol of one another. This is still the case when the 200 snapshots are included, shown in the brackets. The largest RMS error is found when entropy is included of 2.29, whereas when it is not included it is 2.16. However, if ligand E is ignored (as ligand E is made drastically worse by the inclusion of entropy), the RMS error lowers to 1.47 and the RMS error for the exclusion of entropy increases to 2.18. On most of the ligands the inclusion of entropy increases the correlation with the experimental binding free energies.

7.3.2 QM results

Minimal difference was found between the calculated free energies from the ff99SB and ff14SB force fields, however, due to the improvements made to the ff14SB on the ff99SB force field, the snapshots from the ff14SB force field were used within the QM calculations. These snapshots were selected as equally spaced throughout the MD simulation. The results can be seen in Table 7.5.

Table 7.5: QM-PBSA values (in kcal/mol) when using the ff14SB to sample. These values are shifted so that ligand A matches experiment, and the unshifted values are shown in brackets. 40 equally spaced snapshots were initially used for the QM-PBSA values for all ligands except ligand E, where one snapshot was ignored. This was later increased to 80 (79 for ligand E)

Ligand	Experimental	QM 40	QM 80	QM 80 Standard Deviation
A	-6.3[129]	-6.30 (-39.33)	-6.30 (-39.30)	2.61
B	-4.7[129]	-2.49 (-35.52)	-2.33 (-35.33)	2.46
C	-4.8[129]	-7.43 (-40.46)	-6.96 (-39.97)	2.34
D	-5.6[143]	-11.54 (-44.56)	-11.89 (-44.90)	2.36
E	-7.6[143]	-11.64 (-44.67)	-11.66 (-44.66)	2.49
F	-5.7[130]	-19.15 (-52.18)	-19.28 (-52.28)	2.43
G	-4.8[143]	-11.36 (-44.38)	-11.32 (-44.33)	2.70

80 equally spaced snapshots were used to generate the values shown in Table 7.5 for all ligands with the exception of ligand E. One of the 80 snapshots was discounted for ligand E due to a single QM-PBSA value being 497.79 kcal/mol, which has a massive effect on the results. The high value for this snapshot is caused by an anomalous complex energy. Both the host and ligand energies sit within the fluctuation of the other snapshots, the complex, however is 550.59 kcal/mol away from the average (excluding the anomalous snapshot). The high value of this single snapshot can only be attributed to a lack of overlap of the conformational space, as seen in the previous chapters. The fluctuations of the QM-PBSA energies can be seen in Figure 7.2. The problematic snapshot for ligand E can be seen moving off the scale of the graph, however, ignoring this snapshot, all the energies fluctuate around a central point for each ligand. This is reassuring, as it means that these energies should converge to a central value.

The energies in Table 7.5 differ significantly from the experimental energies. However, these values do not include entropy. Table 7.6 shows a comparison of the ff14SB and the QM-PBSA energies including entropy. These energies show no great improvement between the classical and quantum values. For example, for ligand B, we obtain similar results with the QM results showing marginal improvement. Improvement is also shown for ligand E, where the quantum results differ from the experimental by 1.21 kcal/mol, an improvement from the classical difference of 4.87 kcal/mol. For each of the other ligands (excluding ligand A) the correlation with experiment is made worse by including the quantum correction. In each case the quantum energies overestimate how favourable the interaction is and the binding free energies are made more negative. In fact, even

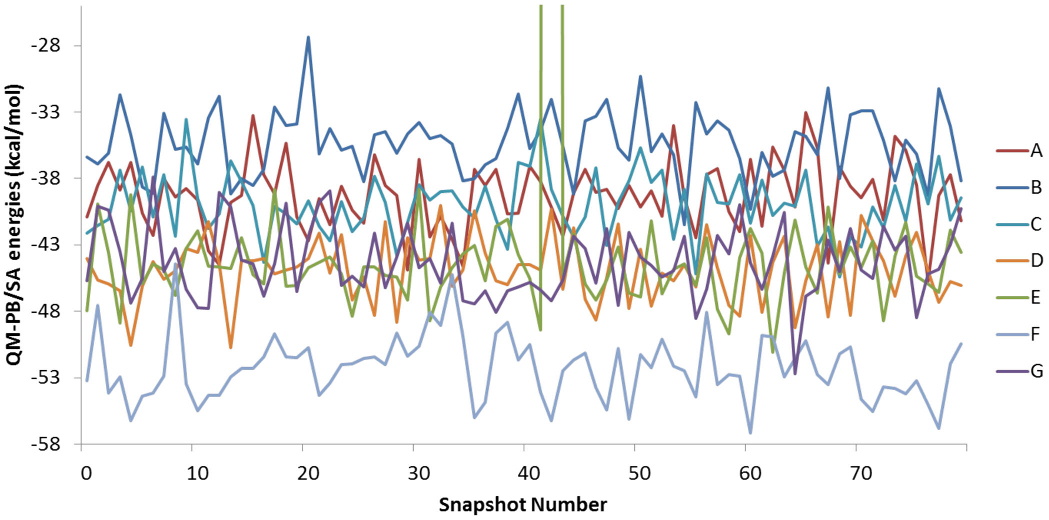


Figure 7.2: Fluctuations of the QM-PBSA binding energies for each of the 80 snapshots included for each of the ligands.

in the cases where the quantum results improve upon the classical results, they are more negative. This effect is most drastic for ligand F, with a difference between the classical and quantum energies of 10.02 kcal/mol. These difference in energies, along with the energy outlier for ligand E, could indicate that it is important to include quantum only interactions in the sampling process. This does not necessarily need to be a prohibitively expensive *ab initio* MD simulation, but could be a method similar to that described in the previous chapter, that includes polarisation at the sampling stage. Here, however, we see a poor correlation between the quantum and classical energies. This poor correlation has been shown in previous chapters, although here the free energies are based on interaction energies, so it is surprising how different these values are. This could be an indication of errors within the DFT calculations, that could be minimised by using a different exchange-correlation functional, this is something that should be investigated at a later stage.

Although the quantum results do not improve the classical results, there are also large errors in the MM results too. They also show a poor correlation with the experimental values.

Table 7.6: Combining entropy with the energy from the ff14SB and QM-PBSA values. Both 40 snapshots and 80 snapshots were used and are indicated by the number in the column heading. All values are in kcal/mol and the entropy is calculated when 200 snapshots are used.

Ligand	Experimental	ff14SB	QM-PBSA 40	QM-PBSA 80
A	-6.3[129]	-6.3	-6.3	-6.3
B	-4.7[129]	-2.89	-3.03	-2.87
C	-4.8[129]	-3.82	-6.79	-6.32
D	-5.6[143]	-6.28	-11.14	-11.50
E	-7.6[143]	-2.73	-8.79	-8.81
F	-5.7[130]	-8.37	-18.26	-18.39
G	-4.8[143]	-5.54	-9.20	-9.16

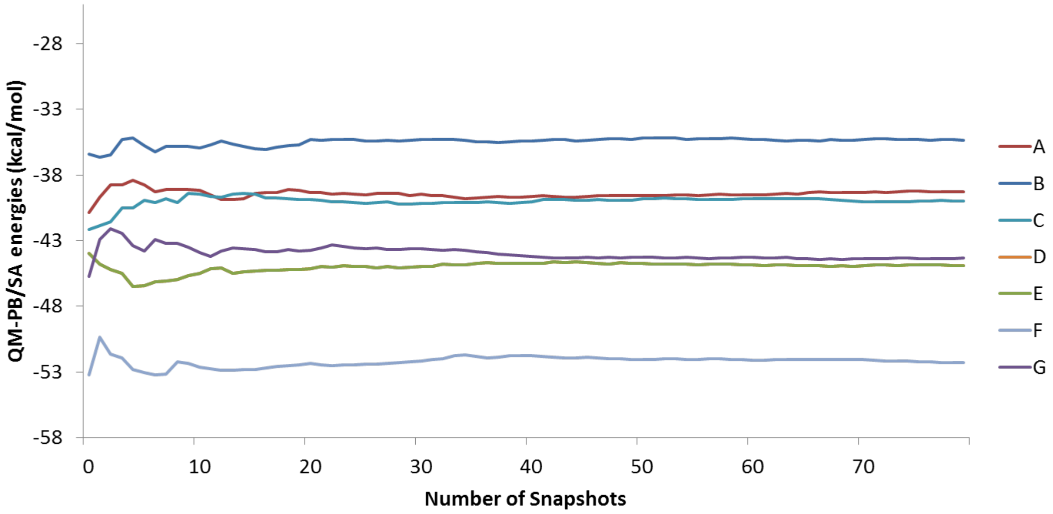


Figure 7.3: Running average of the QM-PBSA values when increasing the number of snapshots up to the total 80 snapshots. For ligand E only 79 snapshots are included, 1 snapshot has been discounted as discussed previously.

The difference in free energies when 80 snapshots are used to calculate the QM-PBSA values and when 40 snapshots are used is minimal. In the worst case the difference is 0.47 kcal/mol, which is within thermal error. Indeed, a running average of the QM-PBSA energies can be seen in figure 7.3, and the free energies look to converge very rapidly. For ligand E, the problematic snapshot discussed previously was discarded. However, in order to ensure convergence, a boostrapping approach was applied. By selecting 80 snapshots with replacements from the available snapshots and calculating the average, then repeating this step 1000 times, we can see if a single snapshot is having a large effect on the average energy. Once this average had been calculated, we compared them with the average energy from the 80 snapshots. These are shown in Table 7.7 and show very little difference between the bootstrap averaging and the standard average (taken from Table 7.6). This is a strong indicator to how converged these results are.

Table 7.7: Comparison of average energies with entropy included between the standard average and the bootstrapped average. All values shown are in kcal/mol.

Ligand	Experimental	Standard average	Bootstrap average
A	-6.3[129]	-6.3	-6.3
B	-4.7[129]	-3.03	-2.86
C	-4.8[129]	-6.79	-6.31
D	-5.6[143]	-11.14	-11.50
E	-7.6[143]	-8.79	-8.81
F	-5.7[130]	-18.26	-18.36
G	-4.8[143]	-9.20	-9.15

In chapter 6 we showed that the choice of dispersion model can have a drastic effect on the free energy. We also showed that the best choice for the systems studied was to use classical dispersion throughout the quantum correction. As such, we applied a similar correction to the QM-PBSA free energies when using both 40 and 80 snapshots. In practice, we subtract the QM dispersion correction from each snapshot and add the classical dispersion. The results can be seen in Table 7.8.

Table 7.8: Free energies when calculated with 40 and 80 equally spaced snapshots, when replacing the quantum dispersion correction with the classical dispersion. All values shown are in kcal/mol.

Ligand	Experimental	QM-PBSA 40	QM-PBSA 80
A	-6.3[129]	-6.3	-6.3
B	-4.7[129]	-3.69	-3.91
C	-4.8[129]	-11.03	-10.47
D	-5.6[143]	-13.49	-13.78
E	-7.6[143]	-15.32	-15.21
F	-5.7[130]	-33.63	-33.76
G	-4.8[143]	-13.74	-13.87

The results in Table 7.8 show consistent values regardless of when 40 or 80 snapshots are used. When using ligand A to shift the results, we see little improvement to the free energies presented in Table 7.6. Ligand B seems to be the only ligand that is not negatively effected by the inclusion of classical dispersion, where the calculated free energy is 0.88 kcal/mol closer to the experimental free energy. However, if the free energies were shifted using ligand D as the reference value, 3 of the ligand's (D, E and G) free energies are very close to the experimental values. Ligand C is much closer than it is when shifted by ligand A, but still around 2 kcal/mol away. Ligand A and B are made positive by this shift, thus the difference between the experimental and calculated free energies would widen. Ligand F is much more negative by the inclusion of the classical dispersion, this can only be attributed to its relative size when compared to the other ligands. The large change that occurs when including the classical dispersion energy for ligand F indicates that the issue of poor correlation with experiment could be down to the dispersion model used. However, in this case inclusion of the classical dispersion is not sufficient to improve the correlation. Table 7.9 shows these new shifted values with the results in table 7.6 along with the classical results for the ff14SB force field when using ligand D as a reference. However, only limited improvement is made on the classical free energies.

Table 7.9: Comparison of free energies when using ligand D as the reference ligand. All values shown are in kcal/mol.

Ligand	Experimental	ff14SB	QM-PBSA 80	QM-PBSA 80 (classical dispersion)
A	-6.3[129]	-5.62	-0.40	1.88
B	-4.7[129]	-2.07	3.03	4.27
C	-4.8[129]	-3.13	-0.42	-2.29
D	-5.6[143]	-5.60	-5.60	-5.60
E	-7.6[143]	-2.05	-2.92	-7.04
F	-5.7[130]	-7.68	-12.49	-25.59
G	-4.8[143]	-4.86	-3.27	-5.70

7.4 Conclusions

We have applied QM-PBSA on an important biologically relevant protein-ligand system. This involved running quantum calculations on an entire complex of around 3000 atoms. We have investigated the use of different force fields and a restraining method and replacing the quantum dispersion term with its classical counterpart.

We have shown that only a minimal difference is present when using the ff99SB or the ff14SB to calculate the free energies for trypsin with benzamidine derivatives. However, our approach to cancel noise shows no improvement on using a standard approach. The unrestrained values were improved by including entropy. When increasing the number of snapshots from 100 to 200 for the ff14SB force field, very little difference was observed, other than a lowering of error.

80 equally spaced snapshots from the ff14SB force field were then used in quantum calculations. However, no overall improvement was observed. The classical results, although better than the quantum results, still did not match the experimental results. This highlights an important issue when using QM-PBSA, in that, as accurate as the QM calculations are, they are still limited by what structures can be sampled by the force field. This means that unfavourable QM snapshots can be sampled and given equal weight to other snapshots, which can lead to drastic effects in the free energy (see chapter 3). Additionally in order to ensure that the error in the DFT calculations are minimal, calculations could be rerun with a different exchange-correlation functional and the results compared. Further testing of the classical results could also be performed, testing different methods to calculate the free energy to try to improve the classical free energies. Such methods include, for example, thermodynamic integration (TI). This will increase the amount of conformational sampling, therefore, should improve the result. The increased cost of TI when compared to MM-PBSA is a disadvantage, but still negligible when compared to the cost of the QM calculations.

The inclusion of classical dispersion in place of the quantum dispersion improved the results in the previous chapter, however, within this chapter, such an approach did not improve the results. This could be due to the different environments present for the systems in the previous chapter. In the previous chapter all the ligands were surrounded by water only, which can be thought of as largely uniform, whereas within the binding pocket of trypsin, the interactions are more varied. When the reference ligand was changed from ligand A to ligand D an improvement upon the previous quantum method was achieved, but overall the force field produced free energies closer to the experimental values.

Chapter 8

Conclusions

We have presented a way to correct classical binding free energies and the steps leading up to a method that produced accurate free energies of hydration. These steps have shown several important difficulties when applying quantum corrections. Chapter 3 shows the issue of using a single step perturbation (SSP) approach to correcting free energies with total potential energies, namely that large "jumps" are present within the free energy. These "jumps" occur as a result of energy outliers at the low energy tail of the energy difference distribution. The energy difference distributions within this thesis are primarily for simple systems, such as ethanol, ethane, propane, dimethyl ether and ethylene glycol in water. Nevertheless, these distributions all have a large range. If further examination is performed on the outliers within the energy difference distribution, it is found that they are not outliers in either the classical or quantum energy distributions. However, they are on different sides of the distributions. This highlights the inexact overlap of the configurational space.

The issue of inexact overlap was then further investigated by perturbing between two classical potentials with different charges. By adjusting the charges the level of overlap between the guiding and target potential can be controlled. It was found that, as the level of overlap was decreased (i.e. the difference in the charges was increased) the worse the convergence of the free energy between subsequent runs. This was then further investigated by running the "reverse" calculations, where the MD is performed using the higher charged potential and a single step perturbation is performed back to a standard charged potential. No convergence was found between the "forward" and "reverse" process. This showed the sensitivity of the Zwanzig equation and led to the examination of post-processing methods with an aim to "smooth" over the low energy tail. First a Gaussian distribution was fitted to the energy difference. Numerical integration showed much improved convergence on the direct SSP approach, but still yielded a difference of over 6 kcal/mol between runs. Additionally an analytical approach was applied, which emphasized the sensitivity of the exponents on the convergence of the free energy. Specifically, the variance or α exponent has the largest effect on the free energy. Again this is due to the low energy tail of the distribution, by only allowing data from ten thousandth height of the Gaussian the free energy converges. However, there is no valid scientific reason to ignore part of the data, which shows that such a post-processing method cannot be applied to increase the convergence. This demonstrated the importance of finding an accurate representation of the QM configurational space.

We then examined the accuracy of a method that applied an acceptance test in order to validate each snapshot's relevance in a QM/MM ensemble. However, the issue of inexact overlap of configurational space affected the acceptance rate again. For simple systems, such as N_2 in a vacuum, where there is a high degree of overlap, the acceptance was high. By changing the bond length, such that the overlap between the ensembles was made worse, the acceptance lowered. By including additional λ steps for this simple system, it was clear that, as the bond length was moved further away from the true equilibrium value, the PMF would become more "bent". A single step perturbation approach cannot model all the nuances within the PMF, only the start and (if the overlap is high) the end point, therefore as the PMF moves from a straight line shape a single step perturbation approach becomes less accurate, thus highlighting the issue with a single step procedure.

Following the promising results of N_2 in vacuum, systems of increasing complexity were utilised in order to locate the poor areas of configurational space overlap between the MM and QM. The results showed that, for simple systems, as the main interaction between the solute and solvent became more electrostatically driven, the overlap became much worse. For larger ligands the number of degrees of freedom present had a negative effect on the overlap, this was the case even when an alanine dipeptide (made up from standard residues) was used. Various techniques were tried in order to increase the overlap between the ensembles, with very limited success. Therefore the next step in the process was to find a method that could discount the snapshots that are energy outliers within the QM/MM distribution.

Examination of the potential energy surfaces showed that there is overlap present between the MM and QM at the minima. Therefore a bias was applied to only sample around this region. A number of systems were tested, including alanine dipeptide, which previously provided no acceptance to the QM/MM ensemble. With the bias in place, the acceptance was good and duplicate runs with the same bias parameters applied provided the same free energy. However, if the bias parameters were allowed to change dynamically, and different bias parameters were found to be optimal, the free energy would differ. In order to sample more, for a fraction of the cost, a numerical model was designed. The results showed if the energy difference distribution (QM-MM) had a standard deviation of just 5 kJ/mol, then consecutive runs would not converge the free energy. This was then compared to the energy difference distribution in chapter 3, which has a standard deviation of approximately 144 kJ/mol.

This led to the method described in chapter 6, where a "stepping stone" approach was used to obtain the quantum free energies. This "stepping stone" was a QM/MM ensemble, built up from the classical energy with quantum polarisation included. The first step was to transition from a purely classical ensemble to the QM/MM ensemble using total energies. The second step was to go from the QM/MM ensemble to the full QM. Unfortunately, the QM/MM ensemble still lacked overlap of the configurational space with the full QM ensemble, such that when total energies were used in a single step perturbation, convergence could not be achieved. The reason for this lack of convergence, for the relatively small ligands used within the validation, was found to be the intermolecular interactions between the water molecules.

When using parts of or the whole interaction energies to perturb from the generated QM/MM ensemble to QM, we see an overall improvement on the classical results when compared with the experimental values. The largest improvement comes when using the classical dispersion throughout, instead of switching to a more quantum dispersion model. Differing the exchange-correlation functional was also tested, but it was found that results between GGA functionals are very similar and a vast improvement if compared to using an LDA functional.

A future extension of the work could involve the application of the "stepping stone" approach to a more realistic test system. This should be much closer to a protein-ligand binding site, but much smaller, for example, cyclodextrin bound to a variety of small organic ligands. This would provide a non-uniform polarity around the ligand and should show the real benefit this method can have on complex systems.

The final results chapter examined the QM-PBSA method when applied to trypsin bound to several different benzamidine derivatives. This involved ONETEP DFT calculations on the entire protein-ligand complex, which consisted of around 3000 atoms. Previous work in this area by Fox et al. [125] showed an improvement of QM-PBSA on MM-PBSA binding free energies. However, this study was performed on a simpler system, here we applied the QM-PBSA method to a more complex system. Neither the classical nor quantum results cannot replicate the experimental values. The quantum results do not improve the classical results for most of the ligands and in each case the binding free energies are predicted to be lower than the classical calculation. The method is limited by the sampling performed at the classical level, this shows the importance of including quantum only interactions in the sampling. It would be an interesting study to obtain the free energies by using a method like the one described in chapter 6 (the "stepping stone" approach) and compare the free energies with the QM-PBSA energies. By using an approach like the "stepping stone" approach, polarisation would be included at the sampling stage, which would form a more accurate ensemble of structures to use within the QM-PBSA method. The limiting factor would be whether the acceptance is high enough to provide converged properties and the time taken to calculate the quantum interaction energy used within the acceptance criterion. However, this could be affected by adjusting how much of the protein is represented by quantum mechanics.

This thesis has developed and evaluated a variety of methods to correct free energies and the limitations associated with these methods. Several conclusions can be drawn from the work presented here, however, the key points can be summed up like this: The MM and QM configurational spaces have extremely poor overlap, and because of this single step perturbations are likely to only have a limited, if any, success; This lack of overlap increases as the degrees of freedom increase, and is exacerbated by the intermolecular interactions within the solvent; In order to obtain accurate binding free energies it is imperative to work with the quantum potential energy surface in the sampling stage; And finally, although formally not correct, interaction energies can provide converged free energies of binding, where total energies cannot.

Bibliography

[1] D.A. Case and J.W. Ponder. Force Fields For Protein Simulations. *Journal of Advanced Protein Chemistry*, 66:27–85, 2003.

[2] C.J. Cramer. *Essentials of Computational Chemistry, Theories and Models*. Wiley, 2nd edition, 2007.

[3] M.R. Shirts, D.L. Mobley, J.D. Chodera, and V.S. Pande. Accurate and Efficient Corrections for Missing Dispersion Interactions in Molecular Simulations. *Journal of Physical Chemistry B*, 111:13052–13063, 2007.

[4] P. Ewald. Die berechnung optischer und elektrostatischer gitterpotentiale. *Annals of Physics*, 369:253–287, 1921.

[5] A.R. Leach. *Molecular Modeling Principles and Applications*. Pearson Education Limited, 2 edition, 1996.

[6] Notes on Ewald summation techniques. `users.wfu.edu/natalie/s11phy752/lecturenote/Ewald.pdf`.

[7] M.E. Tuckerman. *Statistical Mechanics: Theory and Molecular Simulation*. Oxford Graduate Texts, 2010.

[8] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren, A. DiNola, and J.R. Haak. Molecular dynamics with coupling to an external bath. *The Journal of Chemical Physics*, 81(8):3684–3690, 1984.

[9] T. Darden, D. York, and L. Pedersen. Particle Mesh Ewald: An N $\cdot$ log(n) method for Ewald sums in large systems. *The Journal of Chemical Physics*, 98:10089, 1993.

[10] H.J.C. Berendsen, J.P.M. Postma, W.F. Van Gunsteren, and J. Hermans. *Interaction Models for Water in Relation to Protein Hydration.* Reidel, Dordrecht, 1981.

[11] W.L. Jorgensen, J. Chandrasekhar, J.D. Madura, R.W. Impey, and M.L. Klein. Comparison of simple potential functions for simulating liquid water. *The Journal of Chemical Physics*, 79(2):926–935, 1983.

[12] M.W. Mahoney and W.L. Jorgensen. A five-site model for liquid water and the reproduction of the density anomaly by rigid, nonpolarizable potential functions. *The Journal of Chemical Physics*, 112(20):8910–8922, 2000.

[13] J.D. Jackson. *Classical Electrodynamics.* Wiley, 1998.

[14] D.J. Griffiths. *Introduction to Electrodynamics.* Prentice Hall, 1999.

[15] H.J.C. Berendsen. *Simulating the Physical World.* Cambridge University Press, 2007.

[16] A.R. Leach. *Molecular Modeling Principles and Applications*, volume 2. Pearson Education Limited, 2001.

[17] D.A. Case, T.A. Darden, T.E. Cheatham III, C.L. Simmerling, J. Wang, R.E. Duke, R. Luo, M. Crowley, R.C. Walker, W.Zhang, K.M. Merz, B. Wang, S. Hayik, A. Roitberg, G. Seabra, I. Kolossvary, K.F. Song, F. Paesani, J. Vanicek, X. Wu, S.R. Brozell, T. Steinbrecher, H. Gohlke, L. Yang, C. Tan, J. Mongen, V. Hornak, G. Cui, D.H. Mathews, M.G. Seetin, C. Sagui, V. Babin, and P.A. Kollman. *AMBER 10.* University of California, San Fransico, 2008.

[18] W.D Cornell, P. Cieplak, C.I. Bayly, I.R. Gould, K.M.J. Merz, D.M. Ferguson, D.C. Spellmeyer, T. Fox, J.W. Caldwell, and P.A. Kollman. A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules. *Journal of the America Chemical Society*, 117(19):5179–5197, 1995.

[19] W.F. Van Gunsteren, P.K. Weiner, and A.J. Wilkinson. *Computer Simulation of Biomolecular Systems: Theoretical and Experimental Applications v. 3.* ESCOM Science Publishers, The Netherlands, 1997.

[20] P. Cieplak, J.W. Caldwell, and P.A. Kollman. Molecular mechanical models for organic and biological systems going beyond the atom centered two body additive approximation: aqueous solution free energies of methanol and N-methyl acetamide, nucleic acid base, and amide hydrogen bonding and chloroform/water partition coefficients of the nucleic acid bases. *Journal of Computational Chemistry*, 22(10), 1048-1057.

[21] V. Hornak, R. Abel, A. Okur, B. Strockbine, A. Roitberg, and C.L. Simmerling. Comparison of Multiple Amber Force Fields and Development of Improved Protein Backbone Parameters. *PROTEINS: Structure, Function and Bioinformatics*, 65:712–725, 2006.

[22] D.A. Case, V. Babin, J.T. Berryman, R.M. Betz, Q. Cai, D.S. Cerutti, T.E. Cheatham III, T.A. Darden, R.E. Duke, H. Gohlke, A.W. Goetz, S.Gusarov, N. Homeyer, P. Janowski, J. Kaus, I. Kolossváry, A. Kovalenko, T.S. Lee, S. LeGrand, T. Luchko, R. LuoR. Luo. Madej, K.M. Merz, F. PaeF. Paesani.R. Roe, A. Roitberg, C. Sagui, R. Salomon-Ferrer, G. Seabra, C.L. Simmerling, W. Smith, J. Swails, R.C. Walker, J. Wang, R.M. Wolf, X. Wu, and P.A. Kollman. *AMBER 14*. University of California, San Fransico, 2014.

[23] J. Wang, R.M. Wolf, J.W. Caldwell, P.A. Kollman, and D.A. Case. Development and testing of a general amber force field. *Journal of Computational Chemistry*, 25(9):1157–1174, 2004.

[24] L. De Broglie. *Recherches sur la théorie des quanta*. PhD thesis, 1924.

[25] M. Planck. Ueber das Gesetz der Energieverteilung im Normalspectrum. *Annals of Physics*, 309:553–563, 1901.

[26] W. Heisenberg. Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik. *Zeitschrift für Physik*, 43:172–198, 1927.

[27] M.J. Winter. *Chemical Bonding*. Oxford Science Publications, 1994.

[28] L. Susskind. Modern Physics: Quantum Mechanics; Lecture 1.

[29] E. Schrödinger. An Undulatory Theory of the Mechanics of Atoms and Molecules. *Physical Review*, 28:1049–1070, 1926.

[30] M. Born and R. Oppenheimer. Zur quantentheorie der molekeln. *Annalen der Physik*, 389:457–464, 1927.

[31] I.N. Levine. *Quantum Chemistry*. Prentice Hall, 5 edition, 2000.

[32] A. Szabo and N.S. Ostlund. *Modern Quantum Chemistry*. Dover, 1996.

[33] C.-K. Skylaris. Computational Quantum Chemistry; Lecture 4.

[34] G.H. Grant and W.G. Richards. *Computational Chemistry*. Oxford University Press, 1998.

[35] P. Hohenburg and W. Kohn. Inhomogeneous Electron Gas. *Physical Review*, 136(3B):B864–B871, 1964.

[36] N.M. Harrison. An Introduction to Density Functional Theory. http://www.ch.ic.ac.uk/harrison/Teaching/DFT_NATO.pdf.

[37] W. Kohn and L.J. Sham. Self-Consistent Equations Including Exchange and Correlation Effects. *Physical Review*, 140(4A):A1133–A1138, 1965.

[38] F. Jensen. *Introduction to Computational Chemistry*. Wiley, 2 edition, 2007.

[39] S. H. Vosko, L. Wilk, and M. Nusair. Accurate spin-dependent electron liquid correlation energies for local spin density calculations: a critical analysis. *Canadian Journal of Physics*, 58:1200–1211, 1980.

[40] A.D. Becke. Density functional calculations of molecular bond energies. *Journal of Physical Chemistry*, 84:4524–4529, 1986.

[41] J.P. Perdew, K. Burke, and M. Ernzerhof. Generalized Gradient Approximation Made Simple. *Physical Review Letters*, 77:3865–3868, Oct 1996.

[42] J. P. Perdew and Y. Wang. Accurate and simple analytic representation of the electron-gas correlation energy. *Physical Review B*, 45:13244, 1992.

[43] C. Lee, W. Yang, and R. G. Parr. Development of the Colle-Salvetti Correlation-Energy Formula into a Functional of the Electron Density. *Physical Review B*, 37:785–789, 1988.

[44] A.D. Becke. A new mixing of hartree-fock and local density-functional theories. *Journal of Chemical Physics*, 98:1372–1377, 1993.

[45] J.C. Slater. Atomic Sheilding Constants. *Physical Reviews*, 36:57–64, 1930.

[46] S.F. Boys. Electronic Wave Functions. I. A General Method of Calculation for the Stationary States of Any Molecular System. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 200(1063):542–552, 1950.

[47] S.F. Boys and F. Bernardi. The calculation of small molecular interactions by the differences of separate total energies. some procedures with reduced errors. *Mol. Phys.*, 19:553–566, 1970.

[48] P. Schwerdtfeger. The pseudopotential approximation in electronic structure theory. *ChemPhysChem*, 12:3143–3155, 2011.

[49] M.P. Teter, D.C. Allan, T.A. Arias, J.D. Joannopoulos, and M.C. Payne. Iterative minimisation techniques for ab initio total-energy calculations- molecular-dynamics and conjugate gradients. *Reviews of Modern Physics*, 64:1045–1097, 1992.

[50] W. Kohn. Density functional and density matrix method scaling linearly with the number of atoms. *Physical Review Letters*, 76:3168–3171, 1996.

[51] A.A. Mostoi, C.-K. Skylaris, P.D. Haynes, and M.C. Payne. Total-energy calculations on a real space grid with localized functions and a plane-wave basis. *Computer Physics Communications*, 147:788–802, 2002.

[52] C.-K. Skylaris, A.A. Mostofi, P.D. Haynes, O.Diéguez, and M.C. Payne. The Non-orthogonal Generalised Wannier Function pseudopotential plane-wave method. *Physical Review B*, 66:035119, 2002.

[53] ONETEP Developers. http://www2.tcm.phy.cam.ac.uk/onetep/Intro/Node4.

[54] C.-K. Skylaris, P.D. Haynes, A.A. Mostofi, and M.C. Payne. Introducing ONETEP: Linear-scaling density functional simulations on parallel computers. *The Journal of Chemical Physics*, 122:084119, 2005.

[55] A.A. Mostofi, P.D. Haynes, C.-K. Skylaris, and M. C. Payne. Preconditioned iterative minimization for linear-scaling electronic structure calculations. *Journal of Physical Chemistry*, 119:8842–8848, 2003.

[56] ONETEP Developers. http://www2.tcm.phy.cam.ac.uk/onetep/Intro/Node6.

[57] Q. Hill and C.-K. Skylaris. Including dispersion interactions in the ONETEP program for linear-scaling density functional theory calculations. *Proceedings of the Royal Society A*, 465:669–683, 2009.

[58] S. Grimme. Semiempirical GGA-Type Density Functional Constructed with a Long-Range Dispersion Correction. *Journal of Computational Chemistry*, 27:1787–1799, 2006.

[59] M. Elstner, P. Hobza, T. Frauenheim, S. Suhai, and E. Kaxiras. Hydrogen bonding and stacking interactions of nucleic acid base pairs: A density-functional-theory based treatment. *The Journal of Chemical Physics*, 114:5149–5155, 2001.

[60] B.J. Alder and T.E. Wainwright. Studies in Molecular Dynamics. I. General Method. *The Journal of Chemical Physics*, 31:459, 1959.

[61] D.A. McQuarrie. *Statistical Mechanics*. University Science Books, 2000.

[62] L. Verlet. Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Physical Review*, 159:98–103, 1967.

[63] W.C. Swope, H.C. Andersen, P.H. Berens, and K.R. Wilson. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *The Journal of Chemical Physics*, 76(1):637–649, 1982.

[64] S.A. Adelman and J.D. Doll. Generalized Langevin equation approach for atom/solid-surface scattering: General formulation for classical scattering off harmonic solids. *The Journal of Chemical Physics*, 64(6):2375–2388, 1976.

[65] V. Barsegov. http://faculty.uml.edu/vbarsegov/teaching/bioinformatics/lectures/MDEnsemblesModified.pdf.

[66] T. Morishita. Fluctuation formulas in molecular-dynamics simulations with the weak coupling heat bath. *The Journal of Chemical Physics*, 113:2976–2982, 2000.

[67] J. Finnerty. http://www.grs-sim.de/cms/upload/Carloni/Tutorials/FMCP/Thermostats_and_Barostats.pdf.

[68] J.P. Ryckaert, G. Ciccotti, and H.J.C. Berendsen. Numerical Integration of the Cartesian Equations of Motion of a System with Constraints: Molecular Dynamics of n-Alkanes. *Journal of Computational Physics*, 23(3):327–341, 1977.

[69] B. Hess, H. Bekker, H.J.C. Berendsen, and J.G.E.M. Fraaije. Lincs: A Linear Constrain Solver for Molecular Simulations. *Journal of Computational Chemistry*, 18:1463–1472, 1997.

[70] D.J. Tildesley M.P. Allen. *Computer Simulation of Liquids*. Oxford Science Publications, 1987.

[71] M.S. Shell. engineering.ucsb.edu/~shell/che210d/Monte_Carlo.pdf.

[72] W.K. Hastings. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97–109, 1970.

[73] D.C. Handscomb J.M. Hammersley. *Monte Carlo Methods*. Methuen, 1964.

[74] T.L. Hill. *An Introduction to Statistical Thermodynamics*. Dover, 1986.

[75] M. Glazer and J. Wark. *Statistical Mechanics: A Survival Guide*. Oxford University Press, 2009.

[76] D.A. Case T. Steinbrecher, I.S. Joung. Soft-Core Potentials in Thermodynamic Integration. Comparing One- and Two-Step Transformations. *Journal of Computational Chemistry*, 32:3253–3263, 2011.

[77] A. Pohorille C. Chipot. *Free Energy Calculations*. Springer, 2007.

[78] R.W. Zwanzig. High-Temperature Equation of State by a Perturbation Method. I. Nonpolar Gases. *The Journal of Chemical Physics*, 22(8):1420–1426, 1954.

[79] J. Srinivasan, J. Miller, P.A. Kollman, and D.A. Case. Continuum solvent studies of the stability of RNA hairpin loops and helices. *Journal of Biomolecular Structure & Dynamics*, 16:671–682, 1998.

[80] D.W. Wright, B.A. Hall, O.A. Kenway, S. Jha, and P.V. Coveney. Computing Clinically Relevant Binding Free Energies of HIV-1 Protease Inhibitors. *Journal of Chemical Theory and Computation*, 10:1288–1241, 2014.

[81] C.H. Robert. http://www-sop.inria.fr/manifestations/algoSB/slides/crobert-monday_lecture_part2.pdf.

[82] Holger Gohlke. *Protein-ligand Interactions*. Wiley-VCH, 2012.

[83] W.L. Jorgensen. Free Energy Calculations: A Breakthrough for Modeling Organic Chemistry in Solution. *Accounts of Chemical Research*, 22:184–189, 1989.

[84] P. Kollman. Free Energy Calculations: Applications to Chemical and Biochemical Phenomena. *Chemical Reviews*, 93:2395–2417, 1993.

[85] N. Galkin A. Illarionov M. Olevanov V. Ozrin C. Queen O. Khoruzhii, A.G. Donchev and V. Tarasov. Application of a Polarizable Force Field to Calculations of Relative Protein-Ligand Binding Affinities. *Proceedings of the National Academy of Sciences*, 105:10378–10383, 2008.

[86] R.A. Friesner E. Harder, B. Kim and B.J. Berne. Efficient Simulation Method for Polarizable Protein Force Fields: Application to the Simulation of BPTI in Liquid Water. *Journal of Chemical Theory and Computation*, 1:169–180, 2005.

[87] M. Levitt A. Warhsel. Theoretical Studies of Enzymatic Reactions: Dielectric, Electrostatic and Steric Stabilization of the Carbonium Ion in the Reaction of Lysozyme. *Journal of Molecular Biology*, 103:227–249, 1976.

[88] Q. Peng G. Lu Y. Zhao, C. Wang. Error Analysis and Applications of a General QM/MM Approach. *Computational Materials Science*, 50:714–719, 2010.

[89] W. E E. Kaxiras N. Choly, G. Lu. Multiscale Simulations in Simple Metals: A desnity-functional-based methodology. *Physical Review B*, 71:094101, 2005.

[90] M. Štrajbl, G. Hong, and A. Warshel. Ab Initio QM/MM Simulation with Proper Sampling: "First Principle" Calculations of the Free Energy of the Autodissociation of Water in Aqueous Solution. *Journal of Physical Chemistry B*, 106:13333–13343, 2002.

[91] T.H. Rod and U. Ryde. Accurate QM/MM Free Energy Calculations of Enzyme Reactions: Methylation by Catechol O-Methyltransferase. *Journal of Chemical Theory and Computation*, 1:1240–1251, 2005.

[92] F.R. Beierlein, J. Michel, and J.W. Essex. A Simple QM/MM Approach for Capturing Polarization Effects in Protein-Ligand Binding Free Energy Calculations. *Journal of Physical Chemistry B*, 115:4911–4926, 2011.

[93] S.J. Fox, C.Pittock, C.S. Tautermann, T. Fox, C. Christ, N.O.J. Malcolm, J.W. Essex, and C.-K. Skylaris. Free Energies of Binding from Large-Scale First-Principles Quantum Mechanical Calculation: Application to Ligand Hydration Energies. *Journal of Physical Chemistry B*, 117:9478–9485, 2013.

[94] C.J. Woods, F.R. Manby, and A.J. Mulholland. An efficient method for the calculation of quantum mechanics/molecular mechanics free energies. *J. Chem. Phys.*, 128, 2008.

[95] B.A. Berg. Multicanonical Simulations Step by Step. *Computer Physics Communications*, 153:397–406, 2003.

[96] Chemical Computing Group Inc.: Montreal. *MOE2009.10*, 2009.

[97] P.A. Kollman D.A. Case J. Wang, W. Wang. Automatic atom Type and Bond Type Perception in Molecular Mechanical Calculations. *Journal of Molecular Graphics and Modelling*, 25:247–260, 2006.

[98] S.J. Fox, C. Pittock, T. Fox, C.S. Tautermann, N. Malcolm, and C.-K. Skylaris. Electrostatic Embedding in Large-Scale First Principles Quantum Mechanical Calculations on Biomolecules. *The Journal of Chemical Physics*, 135:224107, 2011.

[99] C. Chipot A. Pohorille, C. Jarzynski. Good practices in free-energy calculations. *The Journal of Physical Chemistry B*, 114:10235–10253, 2010.

[100] S. Park, F. Khalili-Araghi, E. Tajkhorshid, and K. Schulten. Free Energy Calculation from Steered Molecular Dynamics Simulations using Jarzynski's Equality. *The Journal of Chemical Physics*, 119:3559–3566, 2003.

[101] G. Hummer, L.R. Pratt, and A.E. García. Multistate Gaussian Model for Electrostatic Solvation Free Energies. *Journal of the American Chemical Society*, 119:8523–8527, 1997.

[102] S. Duane, A.D. Kennedy, B.J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987.

[103] B. Mehlig, D.W. Heermann, and B.M. Forrest. Hybrid Monte Carlo method for condensed-matter systems. *Physical Review B*, 45(2):679–685, 1992.

[104] R. Iftimie, D. Salahub, D. Wei, and J. Schofield. Using a classical potential as an efficient importance function for sampling from ab initio potential. *The Journal of Chemical Physics*, 113:4852–4862, 2000.

[105] G. Marsaglia and T.A. Bray. A Convenient Method for Generating Normal Variables. *Society for Industrial and Applied Mathematics Review*, 6(3):260–264, 1964.

[106] K.S. Byun K. Morokuma M.J. Frisch S. Dapprich, I. Komáromi. A new ONIOM implementation in Gaussian98. Part I. The Calculation of energies, gradient, vibrational frequencies and electric field derivatives. *Journal of Molecular Structure: THEOCHEM*, 461-462:1–21, 1999.

[107] B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl. GROMACS 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation. *Journal of Chemical Theory and Computation*, 4:435–447, 2008.

[108] J.A. White. Lennard-Jones as a model for argon and test of extended renormalization group calculations. *The Journal of Chemical Physics*, 111:9352–9356, 1999.

[109] R.D. Johnson III. NIST Computational Chemistry Comparison and Benchmark Database, NIST Standard Reference Database Number 101, Release 16a, August 2013. http://cccbdb.nist.gov/.

[110] J.P. Valleau G.M. Torrie. Nonphysical Sampling Distributions in Monte Carlo Free-Energy Estimation: Umbrella Sampling. *Journal of Computational Physics*, 23:187–199, 1977.

[111] G. König, P.S. Hudson, S. Boresch, and H.L. Woodcock. Multiscale Free Energy Simulations: An Efficient Method for Connecting Classical MD Simulations to QM or QM/MM Free Energies Using Non-Bolztmann Bennett Reweighting Schemes. *Journal of chemical theory and computation*, 10:1406–1419, 2014.

[112] J. A. Garate and C. Oostenbrink. Free Energy Differences between States with Different Conformational Ensembles. *Journal of Computational Chemistry*, 34:1398–1408, 2013.

[113] N. Ota and A.T. Brünger. Overcoming barriers in macromolecular simulations: non-Boltzmann thermodynamic integration. *Theoretical Chemistry Accounts*, 98:171–181, 1997.

[114] C. Sampson, T. Fox, C.S. Tautermann, C. Woods, and C.-K. Skylaris. A "stepping stone" approach for obtaining quantum free energies of hydration. *Journal of Physical Chemistry B*, 119:7030–7040, 2015.

[115] S. Grimme, J. Antony, S. Ehrlich, and S. Krieg. A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu. *Journal of Chemical Physics*, 132:154104, 2010.

[116] M.H. Abraham, G.S. Whiting, R. Fuchs, and E.J. Chambers. Thermodynamics of solute transfer from water to hexadecane. *Journal of the Chemical Society, Perkin Transactions 2*, 1:291–300, 1990.

[117] C.C. Chambers, G.D. Hawkins, C.J. Cramer, and D.G. Truhlar. Model for aqueous solvation based on class IV atomic charges and first solvation shell effects. *Journal of Physical Chemistry*, 100:16385–16398, 1996.

[118] C. Cave-Ayland, C.-K. Skylaris, and J.W. Essex. Direct Validation of the Single Step Classical to Quantum Free Energy Perturbation. *Journal of Physical Chemistry B*, 119:1017–1025, 2014.

[119] M. Phipps. Empirical Dispersion Correction Within ONETEP.

[120] D.L. Beveridge and F.M. DiCapua. Free Energy via Molecular Simulation: Applications to Chemical and Biomolecular Systems. *Annual Review of Biophysics and Biophysical Chemistry*, 18:431–492, 1989.

[121] T.P. Straatsma and J.A. McCammon. Computational alchemy. *Annual Reviews of Physical Chemistry*, 43:407–435, 1992.

[122] J.C. Gumbart, B. Roux, and C. Chipot. Standard Binding Free Energies from Computer Simulations: What is the Best Strategy. *Journal of chemical theory and computation*, 9:794–802, 2013.

[123] Y. Deng and B. Roux. Computations of Standard Binding Energies with Molecular Dynamics. *Journal of Physical Chemistry B*, 113:2234–2246, 2009.

[124] J.G. Kirkwood. Statistical Mechanics of Fluid Mixtures. *Journal of Chemical Physics*, 3:300–313, 1935.

[125] S.J. Fox, J. Dziedzic, T. Fox, C.S. Tautermann, and C.-K. Skylaris. Density functional theory calculations on entire proteins for free energies of binding: Application to a model polar binding site. *Proteins: Structure, Function, and Bioinformatics*, 2014.

[126] R.T. Bradshaw, P.G.A. Aronica, E.W. Tate, R.J. Leatherbarrow, and I.R. Gould. Mutational Locally Enhanced Sampling (MULES) for quantitative prediction of the effects of mutations at protein-protein interfaces. *Journal of Chemical Science*, 3:1503–1511, 2012.

[127] J. Dziedzic, H.H. Helal, C. K. Skylaris, A.A. Mostofi, and M.C. Payne. Minimal parameter implicit solvent model for ab initio electronic structure calculations. *Europhysics Letters*, 95:43001, 2011.

[128] A. Villa, R. Zangi, G. Pieffet, and A.E. Mark. Sampling and convergence in free energy calculations of protein-ligand interactions: The binding of triphenoxypyridine derivatives to factor Xa and trypsin. *Journal of Computer-Aided Molecular Design*, 17:673–686, 2003.

[129] S.M. Schwarzl, T.B. Tschopp, J.C. Smith, and S. Fischer. Can the Calculation of Ligand Binding Free Energies Be Improved with Continuum Solvent Electrostatic and an Ideal-Gas Entropy Correction? *Journal of Computational Chemistry*, 23:1143–1149, 2002.

[130] F. Gräter, S.M. Schwarzl, A. Dejaegere, S. Fischer, and J.C. Smith. Protein/Ligand binding free energies calculated with quantum mechanics/molecular mechanics. *Journal of Physical Chemistry B*, 109:10474–10483, 2005.

[131] A. de Ruiter and C. Oostenbrink. Efficient and Accurate Free Energy Calculations on Trypsin Inhibitors. *Journal of Chemical Theory and Computation*, 8:3686–3695, 2012.

[132] L. Polgár. The catalytic triad of serine peptidases. *Journal of Cellular and Molecular Life Sciences*, 62:2161–2172, 2005.

[133] J.W. Essex, D.L. Severence, J. Tirado-Rives, and W.L. Jorgensen. Monte Carlo Simulations for Proteins: Binding Affinities for Trypsin-Benzamidine Complexes via Free-Energy Perturbations. *Journal of Physical Chemistry B*, 101:9663–9669, 1997.

[134] R.J. Radmer and P.A. Kollman. The application of three approximate free energy calculations methods to structure based ligand design: Trypsin and its complex with inhibitors. *Journal of Computer-Aided Molecular Design*, 12:215–227, 1998.

[135] D. Jiao, P.A. Golubkov, T.A. Darden, and P. Ren. Calculation of protein-ligand binding free energy by using a polarizable potential. *Proceedings of the National Academy of Sciences*, 105:6290–6295, 2008.

[136] D. Jiao, J. Zhang, R.E. Duke, G. Li, M.J. Schnieders, and P. Ren. Trypsin-Ligand Binding Free Energies from Explicit and Implicit Solvent Simulations with Polarizable Potential. *Journal of Computational Chemistry*, 30:1701–1711, 2009.

[137] T. Yang, J.C. Wu, C. Yan, Y. Wang, R. Luo, M.B. Gonzales, K.N.Dalby, and P. Ren. Virtual screening using molcular simulations. *Proteins*, 79:1940–1951, 2011.

[138] Y. Shi, D. Jiao, M.J. Schnieders, and P. Ren. Trypsin-Ligand Binding Free Energy Calculation with AMOEBA. *Conference Proceedings: IEEE Engineering in Medicine and Biology Society*, 1:2328, 2009.

[139] P. Mikulskis, S. Genheden, P. Rydberg, L. Sandberg, L. Olsen, and U. Ryde. Binding affinities in the SAMPL3 trypsin and host-guest blind ttest estimated with the MM/PBSA and LIE methods. *Journal of Computer Aided Molecular Design*, 26:527–541, 2012.

[140] J. Chen., J. Wang, Q. Zhang, K. Chen, and W. Zhu. A comparitive study of trypsin spespecific based on QM/MM molecular dynamics simulation and QM/MM GBSA calculation. *Journal of Biomolecular Structure & Dynamics*, 1003146, 2015.

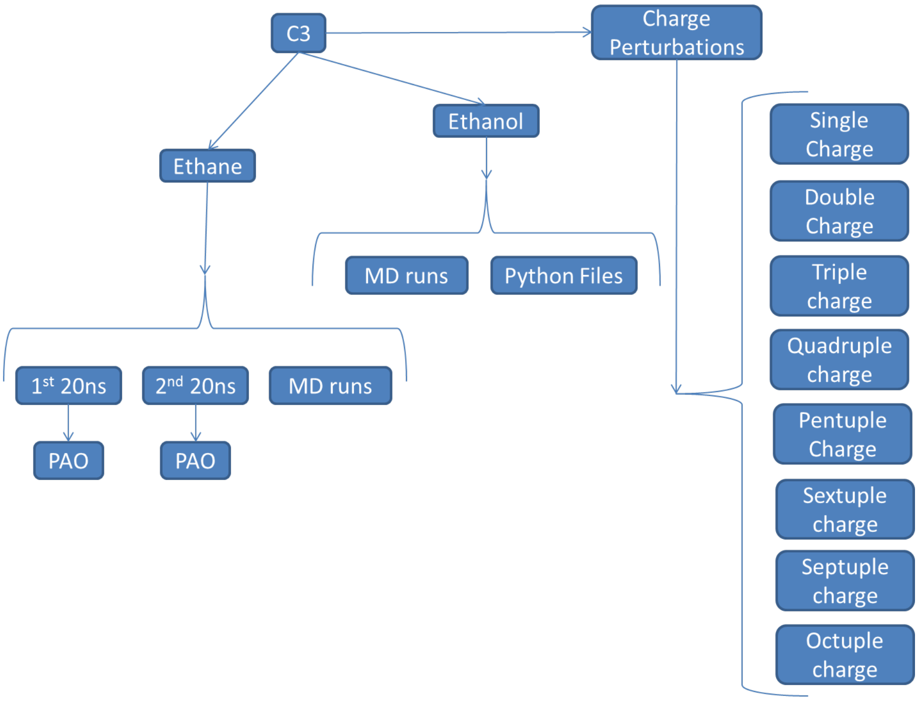
[141] B.A. Katz, J. Finer-Moore, R. Mortezaei, D.H. Rich, and R.M. Stroud. Episelection: novel Ki approximately nonomolar inhibitors of serine proteases selected by binding or chemistry on an enzyme surface. *Biochemistry*, 34:8264–8280, 1995.

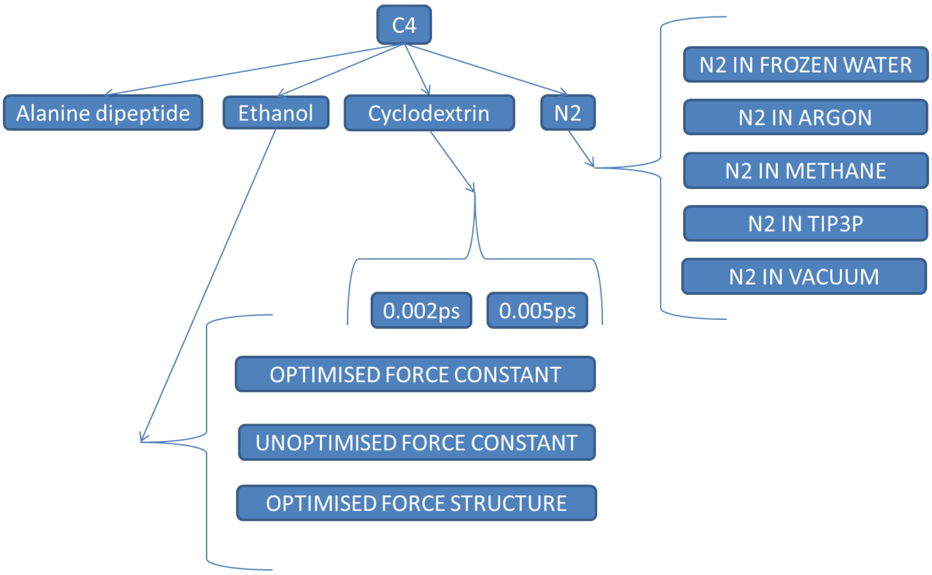
[142] J. Dziedzic, S.J. Fox, T. Fox, C.S. Tautermann, and C.-K. Skylaris. Large-scale DFT Calculatons in Implicit Solvent - A Case Study on the T4 Lysozyme L99A/M102Q Protein. *International Journal of Quantum Chemistry*, 113:771–785, 2013.

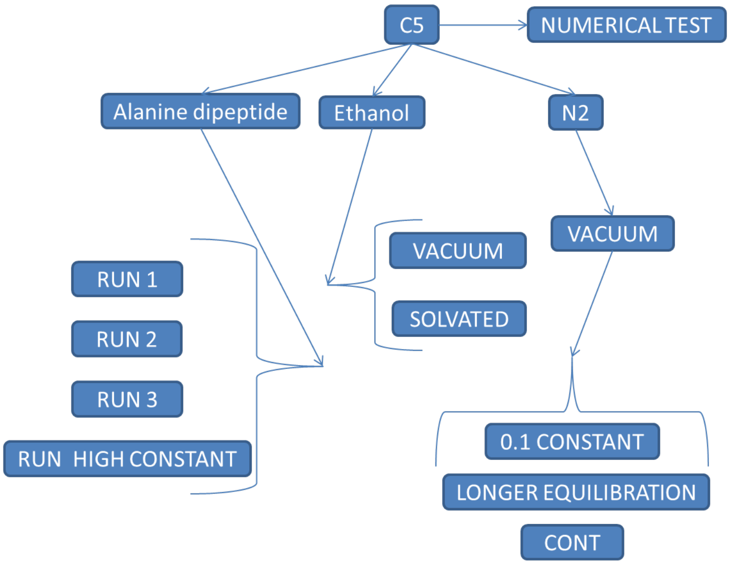
[143] M. Mares-Guia, D.L. Nelson, and E. Rogana. Electronic Effects in the Interaction of Para-Substituted Benzamidines with Trypsin: the Involvement of the π-Electronic Density at the Central Atom of the Substituent in Binding. *Journal of American Society*, 99:2331–2336, 1977.

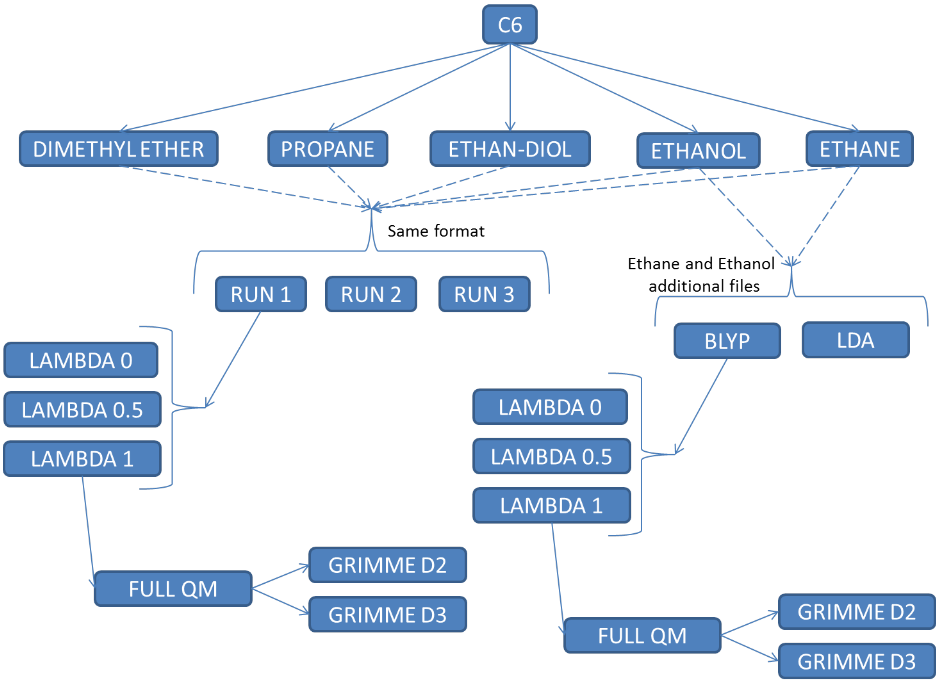
Apendix A

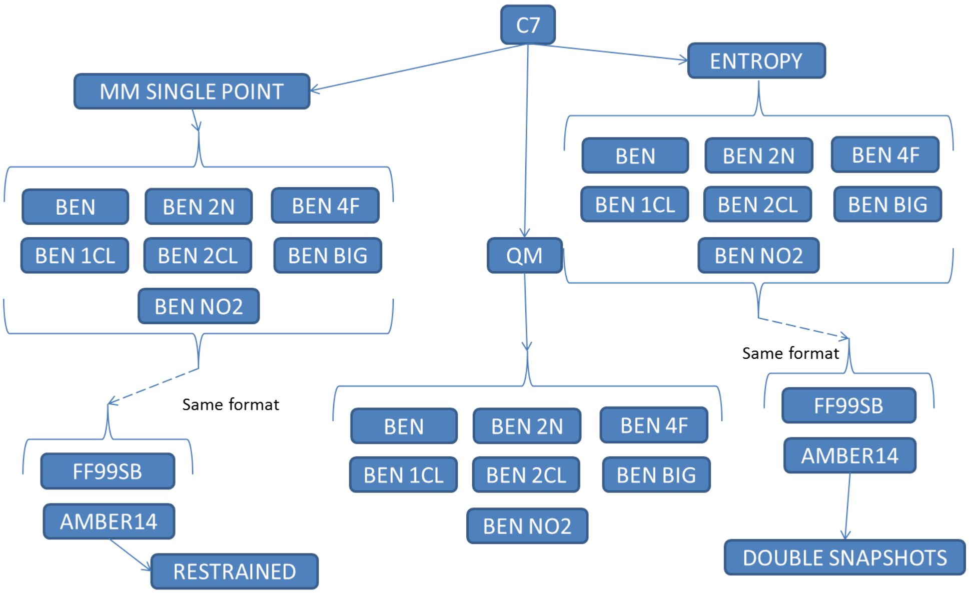
DVD Format:




C4
Alanine dipeptide
Ethanol
Cyclodextrin
N2
N2 IN FROZEN WATER
N2 IN ARGON
N2 IN METHANE
N2 IN TIP3P
N2 IN VACUUM
0.002ps
0.005ps
OPTIMISED FORCE CONSTANT
UNOPTIMISED FORCE CONSTANT
OPTIMISED FORCE STRUCTURE


C5
NUMERICAL TEST
Alanine dipeptide
Ethanol
N2
VACUUM
VACUUM
SOLVATED
RUN 1
RUN 2
RUN 3
RUN HIGH CONSTANT
0.1 CONSTANT
LONGER EQUILIBRATION
CONT


C6
DIMETHYL ETHER
PROPANE
ETHAN-DIOL
ETHANOL
ETHANE
Same format
RUN 1
RUN 2
RUN 3
Ethane and Ethanol additional files
BLYP
LDA
LAMBDA 0
LAMBDA 0.5
LAMBDA 1
FULL QM
GRIMME D2
GRIMME D3
LAMBDA 0
LAMBDA 0.5
LAMBDA 1
FULL QM
GRIMME D2
GRIMME D3


C7
MM SINGLE POINT
ENTROPY
BEN
BEN 2N
BEN 4F
BEN 1CL
BEN 2CL
BEN BIG
BEN NO2
QM
BEN
BEN 2N
BEN 4F
BEN 1CL
BEN 2CL
BEN BIG
BEN NO2
Same format
FF99SB
AMBER14
RESTRAINED
BEN
BEN 2N
BEN 4F
BEN 1CL
BEN 2CL
BEN BIG
BEN NO2
Same format
FF99SB
AMBER14
DOUBLE SNAPSHOTS

Apendix B

Code example:

```
#Last edited 22/08/13
#V2.0 04/08/13
#python gromacs...py inputFile lambda

#
    ---------------------------------------------------------------------------------------

#
    ---------------------------------------------------------------------------------------
    SETUP
    ---------------------------------------------------------------------------------------

#
    ---------------------------------------------------------------------------------------


#required modules
import random
import os
import sys
import re
import numpy
from decimal import Decimal
import math
#
    ---------------------------------------------------------------------------------------

#Constants
nm2bohr=numpy.float64(018.897161643)
hartrees2kj=numpy.float64(2625.49962)
atomicMass2kg=numpy.float64(1.66053892173e-27)
kBKcal=numpy.float64(0.001987204118)
```

```
kBJoules=numpy.float64(1.380648813e-23)
avagadro=numpy.float64(6.0221412927e+23)
kBJoulesMol=numpy.float64(0.008314462175)
T=300.0
#
    ----------------------------------------------------------------------------------------------------


foutOutput=open("script_output.txt", 'w')

numberOfSteps=400000
timeStep=0.00025

#Writes Gromacs NVE input file

fout=open("nve.mdp", 'w')
fout.write("integrator      = md-vv\ngen_vel         = no\nnsteps          = %s\
    ndt             = %s\nnstlist         = 10\nrlist           = 1.1\
    ncoulombtype     = PME\ncoulomb-modifier = Potential-shift\nrcoulomb        =
     0.8\nvdw-type        = cutoff\nvdw-modifier    = Potential-shift\nrvdw
          = 0.6\ntcoupl          = no\npcoupl          = no\nnstenergy       = 1\
    nnstxtcout       = 1000\nxtc-precision   = 100000\ncontinuation    = no\
    nconstraint-algorithm = Lincs\nconstraints = h-bonds" % (numberOfSteps,
    timeStep))

fout.close()


#
    ----------------------------------------------------------------------------------------------------

#
    ----------------------------------------------------------------------------------------------------
    MODULES
    ----------------------------------------------------------------------------------------------------

#
    ----------------------------------------------------------------------------------------------------



#Add in dictionaries here!!
dicETH={"C1":"C", "H11":"H", "H12":"H", "H13":"H", "C2":"C", "H21":"H", "H22":"H"
    , "H23":"H"}
```

```
dicWAT={"O":"O", "H1":"H", "H2":"H"}
dicINT={"N1":"N", "N2":"N", "N":"N"}
dicEOH={"C1":"C", "H11":"H", "H12":"H", "H13":"H", "C":"C", "H2":"H", "H3":"H", "
    H4":"H", "O12":"O"}
dicPCL={"C":"C", "C1":"C", "C2":"C", "H":"H", "H1":"H", "C3":"C", "H2":"H", "C4":
    "C", "H3":"H", "C5":"C", "H4":"H", "Cl":"Cl"}
dic4GA={"O4":"O", "H10":"H", "C6":"C", "H8":"H", "H9":"H", "C5":"C", "O3":"O", "
    H7":"H", "C4":"C", "O5":"O", "H6":"H", "C3":"C", "O2":"O", "H5":"H", "H4":"H"
    , "C2":"C", "O1":"O", "H3":"H", "H2":"H", "C1":"C", "H1":"H", "O30":"O", "C34
    ":"C", "C35":"C", "O28":"O", "C36":"C", "O29":"O", "H59":"H", "H57":"H", "H58
    ":"H", "H56":"H", "H55":"H", "C33":"C", "O27":"O", "H54":"H", "H53":"H", "C32
    ":"C", "O26":"O", "H52":"H", "H51":"H", "C31":"C", "H50":"H", "O25":"O", "C28
    ":"C", "C29":"C", "O23":"O", "C30":"C", "O24":"O", "H49":"H", "H47":"H", "H48
    ":"H", "H46":"H", "H45":"H", "C27":"C", "O22":"O", "H44":"H", "H43":"H", "C26
    ":"C", "O21":"O", "H42":"H", "H41":"H", "C25":"C", "H40":"H", "O20":"O", "C22
    ":"C", "C23":"C", "O18":"O", "C24":"C", "O19":"O", "H39":"H", "H37":"H", "H38
    ":"H", "H36":"H", "H35":"H", "C21":"C", "O17":"O", "H34":"H", "H33":"H", "C20
    ":"C", "O16":"O", "H32":"H", "H31":"H", "C19":"C", "H30":"H", "O15":"O", "C16
    ":"C", "C17":"C", "O13":"O", "C18":"C", "O14":"O", "H29":"H", "H27":"H", "H28
    ":"H", "H26":"H", "H25":"H", "C15":"C", "O12":"O", "H24":"H", "H23":"H", "C14
    ":"C", "O11":"O", "H22":"H", "H21":"H", "C13":"C", "H20":"H", "O10":"O", "C10
    ":"C", "H16":"H", "C9":"C",  "O7":"O",  "H15":"H", "H14":"H", "C8":"C",  "O6"
    :"O", "H13":"H", "H12":"H", "C7":"C",  "H11":"H", "O8":"O",  "C11":"C", "H17"
    :"H", "C12":"C", "H18":"H", "H19":"H", "O9":"O",  "H60":"H" }
dicALA={"N":"N", "H":"H", "CA":"C", "HA":"H", "CB":"C", "HB1":"H", "HB2":"H", "
    HB3":"H", "C":"C", "O":"O"}
dicACE={"HH31":"H", "CH3":"C", "HH32":"H", "HH33":"H", "C":"C", "O":"O"}
dicNME={"N":"N", "H":"H", "CH3":"C", "HH31":"H", "HH32":"H", "HH33":"H"}

def generateVelocities(ligandAtoms, solventAtoms):
        """Input: ligand and solvent atoms
        Output: new random velocities"""

        kineticEnergy=0

        randomVelocities=[]

        #Process for ligand atoms
        #velocity needs to be in m/s to cancel out 1/kBT (in Joules)
        for i in range(len(ligandAtoms)):
                tempVelo=[]
                if ligandAtoms[i]=='H':
                        atomicMass=1
                        mass=1.008*atomicMass2kg
```

```
            elif ligandAtoms[i]==’C’:
                    atomicMass=12
                    mass=12.000*atomicMass2kg
            elif ligandAtoms[i]==’O’:
                    atomicMass=16
                    mass=16.000*atomicMass2kg
            elif ligandAtoms[i]==’N’:
                    atomicMass=14
                    mass=14.000*atomicMass2kg
            elif ligandAtoms[i]==’Cl’:
                    atomicMass=35.500
                    mass=35.500*atomicMass2kg

            stdev=numpy.sqrt(kBJoules*T*mass)

            while len(tempVelo)<3:
                    z=100
                    while z>=1:
                            pseudoRandomNumber1=random.randrange(-1000000,
1000000, 1)/1000000.
                            pseudoRandomNumber2=random.randrange(-1000000,
1000000, 1)/1000000.

                            z=(pseudoRandomNumber1**2)+(pseudoRandomNumber2
**2)

                    w=numpy.sqrt((-2*numpy.log(z))/z)

                    var1=(pseudoRandomNumber1*w*stdev)/mass
                    var2=(pseudoRandomNumber2*w*stdev)/mass



                    #0.01 converts from m/s to Ang/ps
                    #Dviding through by 1000 to get units of km/s, needed for
 gromacs
                    if len(tempVelo)<3:
                            tempVelo.append((var1)/1000)

                    if len(tempVelo)<3:
                            tempVelo.append((var2)/1000)



            for vel in tempVelo:
                    randomVelocities.append(vel)
```

```
        #Repeats process for solvent atoms
        for i in range(len(solventAtoms)):
                tempVelo=[]
                if solventAtoms[i]=='H':
                        atomicMass=1
                        mass=1.008*atomicMass2kg
                elif solventAtoms[i]=='C':
                        atomicMass=12
                        mass=12.000*atomicMass2kg
                elif solventAtoms[i]=='O':
                        atomicMass=16
                        mass=16.000*atomicMass2kg

                stdev=numpy.sqrt(kBJoules*T*mass)

                while len(tempVelo)<3:
                        z=100
                        while z>=1:
                                pseudoRandomNumber1=random.randrange(-1000000,
    1000000, 1)/1000000.
                                pseudoRandomNumber2=random.randrange(-1000000,
    1000000, 1)/1000000.

                                z=(pseudoRandomNumber1**2)+(pseudoRandomNumber2
    **2)

                        w=numpy.sqrt((-2*numpy.log(z))/z)

                        var1=(pseudoRandomNumber1*w*stdev)/mass
                        var2=(pseudoRandomNumber2*w*stdev)/mass

                        #0.01 converts from m/s to Ang/ps
                        #Dividing through by 20.455 converts Ang/ps to Ang
    /(1/20.455)ps - Units Amber uses
                        if len(tempVelo)<3:
                                tempVelo.append((var1)/1000)

                        if len(tempVelo)<3:
                                tempVelo.append((var2)/1000)

                for vel in tempVelo:
                        randomVelocities.append(vel)
        return(randomVelocities)


def writeVelocitiesG96(file, velocities):
        """Input G96 file
        Output: .g96 file with new velocities"""

        fin=open(file, 'r').readlines()

        fileContent=[]

        for line in fin:
                for value in line.split():
                        fileContent.append(value)



        numberOfAtoms=fileContent[1]
        boxInfo=fileContent[-4:-1]

        fout=open("temp_file", 'w')

        fout.write("TITLE\n%s system\nEND\nPOSITION" % (numberOfAtoms))

        for coordinate in re.findall('POSITION(.*?)END', open(file).read(), re.S)
    :
                fout.write(coordinate)

        fout.write("END\nVELOCITY\n")

        velocityInfo=[]

        for velocity in re.findall('VELOCITY(.*?)END', open(file).read(), re.S):
                velocityInfo.append(velocity.split())

        roundedVelocities=[]

        for generatedVelocity in velocities:
                value=Decimal(str(generatedVelocity))
                roundedVelocities.append(value.quantize(Decimal(10)**-9))

        for i in range(len(velocityInfo[0])):
                try:

                        a=(5-len(velocityInfo[0][i*7]))*" "
                        b=(12-len(velocityInfo[0][i*7+2])-len(velocityInfo[0][i
    *7+3]))*" "
                        c=(15-len(str(roundedVelocities[i*3])))*" "
                        d=(15-len(str(roundedVelocities[i*3+1])))*" "
                        e=(15-len(str(roundedVelocities[i*3+2])))*" "

                        fout.write("%s%s %s   %s%s%s%s%s%s%s%s%s\n" % (a,
    velocityInfo[0][i*7], velocityInfo[0][i*7+1], velocityInfo[0][i*7+2], b,
    velocityInfo[0][i*7+3], c, roundedVelocities[i*3], d, roundedVelocities[i
    *3+1], e, roundedVelocities[i*3+2]))
                except IndexError:
                        pass

        fout.write("END\nBOX\n    %s    %s    %s\nEND" % (boxInfo[0], boxInfo[1],
     boxInfo[2]))
        os.system("mv temp_file %s" % file)



def runGromacs(file, outFile, structureNumber):
        """input: .gro file, output file
        output: runs gromacs"""
        foutOutput.write( "Running GROMACS....\n")
        os.system(" grompp -f nve.mdp -c structure_%s.g96 -p %s -po mdout_%s.mdp
     -o %s >> gromacs_output.txt" % (structureNumber, topologyFile,
    structureNumber, outFile))
        os.system(" mdrun_d -nt 4 -s %s -c structure_%s.g96 -e structure_%s_end.
    edr -o structure_%s_end.trr -x structure_%s_end.xtc -g structure_%s_end.log
    &> gromacs_output.txt" % (outFile, structureNumber+1, structureNumber,
    structureNumber, structureNumber, structureNumber))

#22/08/13 16:11
def readG96FileNames(file):
        """Input: .g96 file
        Output: ligand atoms, solvent atoms, box size"""

        atomNames=[]
        residueNames=[]
        boxInfo=[]

        fin=open(file, 'r').readlines()

        boxInfo.append(fin[-2].split())

        fin=fin[4:-4]
        velocityCorrection=0

        for line in fin:
                if "VEL" in line:
                        velocityCorrection=1
                try:
                        residueNames.append(line.split()[1])
                        atomNames.append(line.split()[2])
                except IndexError:
                        pass

        if velocityCorrection==1:

#               print "Residue Names", len(residueNames)
                residueNames=residueNames[0:(len(residueNames)/2)]
#               print len(residueNames)

        correctLigandNames=[]
        correctSolventNames=[]

#Add in dictionaries link here
        for i in range(len(residueNames)):
                residue=residueNames[i]
                if "ETH" in residue:
                        correctLigandNames.append(dicETH[atomNames[i]])
                elif "WAT" in residue:
                        correctSolventNames.append(dicWAT[atomNames[i]])
                elif "OHH" in residue:
                        correctSolventNames.append(dicWAT[atomNames[i]])
                elif "INT" in residue:
                        correctLigandNames.append(dicINT[atomNames[i]])
                elif "EOH" in residue:
                        correctLigandNames.append(dicEOH[atomNames[i]])
                elif "4GA" in residue:
                        correctLigandNames.append(dic4GA[atomNames[i]])
                elif "PCL" in residue:
                        correctLigandNames.append(dicPCL[atomNames[i]])
                elif "ALA" in residue:
                        correctLigandNames.append(dicALA[atomNames[i]])
                elif "ACE" in residue:
                        correctLigandNames.append(dicACE[atomNames[i]])
                elif "NME" in residue:
                        correctLigandNames.append(dicNME[atomNames[i]])
                else:
                        print 60*"-"
                        print "RESIDUES NOT ADDED TO DICTIONARIES, PLEASE ADD
    DICTIONARY, EDIT MODULE ’readG96FileNames’ AND EDIT ’writeOnetepFile’ TO
    INCLUDE PSEUDOPOTENTIALS AND ATOM TYPES"
                        print 60*"-"
        return(correctLigandNames, correctSolventNames, boxInfo)



def g96Reader(file):
        """Input: g96 file
        Output: x, y, z coordinates, box Information"""

        coordinates=[]

        #Reads in the coordinates
        for coordinate in re.findall(’POSITION(.*?)END’, open(file).read(), re.S)
    :
                coordinates.append(coordinate.split())



        coordinates=coordinates[0]



        x=[]
        y=[]
        z=[]

        #Divides the coordinates into x, y and z components
        for value in range(len(coordinates)/7):
                x.append(float(coordinates[value*7+4]))
                y.append(float(coordinates[value*7+5]))
                z.append(float(coordinates[value*7+6]))

        fin=open(file, ’r’).readlines()

        fileContent=[]

        #Reads the box information
        for line in fin:
                for value in line.split():
                        fileContent.append(value)

        boxInfo=fileContent[-4:-1]
        return(boxInfo, x, y, z)

def extractEnergy(file, keyWord):
        energies=[]
        for energyValue in re.findall('Writing(.*?)<==', open(file).read(), re.S)
    :
                energies=energyValue.split()

        c=0
        energy_numbers=[]
        strings=[]
        #Splits the numbers and strings
        for i in energies:
                try:
                        energy_numbers.append(float(i))
                except:
                        strings.append(i)

        #Removes (SR) from strings
        while "(SR)" in strings:
                strings.remove("(SR)")

        #Removes all the start unimportant information
        strings=strings[8:]

        #Removes other units etc.. to get the strings and energies the same
    length
        if "(bar)" in strings:
                strings.remove("(bar)")
        if "recip." in strings:
                strings.remove("recip.")
        if "En." in strings:
                strings.remove("En.")
        if "Energy" in strings:
                strings.remove("Energy")

        energy_numbers=energy_numbers[3:]

        #Finds the energies related
        for i in range(len(strings)):
                if strings[i]==keyWord:
                        return energy_numbers[i]
```

```
def scaleVelocities(ligandAtoms, solventAtoms, velocities):
        """Scales the vlocities so the center of mass movement is 0"""

        totalMass=0
        xMomenta=0
        yMomenta=0
        zMomenta=0

        for i in range(len(ligandAtoms)):
                tempVelo=[]
                if ligandAtoms[i]==’H’:
                        atomicMass=1
                        mass=1.008*atomicMass2kg
                elif ligandAtoms[i]==’C’:
                        atomicMass=12
                        mass=12.010*atomicMass2kg
                elif ligandAtoms[i]==’O’:
                        atomicMass=16
                elif ligandAtoms[i]==’N’:
                        atomicMass=14
                        mass=16.000*atomicMass2kg

                totalMass+=atomicMass

                xMomenta+=atomicMass*velocities[3*i]
                yMomenta+=atomicMass*velocities[3*i+1]
                zMomenta+=atomicMass*velocities[3*i+2]

        velocities=velocities[(len(ligandAtoms)*3):]

        for i in range(len(solventAtoms)):
                tempVelo=[]
                if solventAtoms[i]==’H’:
                        atomicMass=1
                        mass=1.008*atomicMass2kg
                elif solventAtoms[i]==’C’:
                        atomicMass=12
                        mass=12.010*atomicMass2kg
                elif solventAtoms[i]==’O’:
                        atomicMass=16
                        mass=16.000*atomicMass2kg

                totalMass+=atomicMass
                xMomenta+=atomicMass*velocities[3*i]
                yMomenta+=atomicMass*velocities[3*i+1]
                zMomenta+=atomicMass*velocities[3*i+2]

        return(xMomenta/totalMass, yMomenta/totalMass, zMomenta/totalMass)



def removeCenterOfMass(velocities, xVAll, yVAll, zVAll):
        """Scales all velocities by vAll"""

        newVelocities=[]
        c=1
        for value in velocities:
                if c==1:
                        newVelocities.append(value-xVAll)
                        c+=1
                elif c==2:
                        newVelocities.append(value-yVAll)
                        c+=1
                elif c==3:
                        newVelocities.append(value-zVAll)
                        c=1

        return newVelocities



def kineticEnergyCalc(file, keyWord):
        energies=[]
        for energyValue in re.findall('Energies (.*?)Step', open(file).read(), re
    .S):
                for i in energyValue.split():
                        energies.append(i)



        c=0
        energy_numbers=[]
        strings=[]
        #Splits the numbers and strings
        for i in energies:
                try:
                        energy_numbers.append(float(i))
                except:
                        strings.append(i)
        #Removes (SR) from strings
        while "(SR)" in strings:
                strings.remove("(SR)")

        #Removes all the start unimportant information
#       strings=strings[8:]

        #Removes other units etc.. to get the strings and energies the same
    length
        while "(bar)" in strings:
                strings.remove("(bar)")
        while "recip." in strings:
                strings.remove("recip.")
        while "En." in strings:
                strings.remove("En.")
        while "Energy" in strings:
                strings.remove("Energy")
        while "(kJ/mol)" in strings:
                strings.remove("(kJ/mol)")
        while "Energies" in strings:
                strings.remove("Energies")
        while "rmsd" in strings:
                strings.remove("rmsd")

#       energy_numbers=energy_numbers[3:]

        values=[]

        #Finds the energies related
        for i in range(len(strings)):
                if strings[i]==keyWord:
                        try:
                                values.append(energy_numbers[i])
                        except: pass
#       print values

        return(values[0])

def acceptanceCriteriaHybridMC(totalEnergy1, totalEnergy2):
        """Input: Initial and final classical energies
        Output: 1 if structure accepted, 0 if not"""

        deltaE=(float(totalEnergy2)-float(totalEnergy1))
        foutOutput.write("ACCEPTANCE VALUE = %s\n" % (deltaE))

        probAcceptance=numpy.exp(-(1/(kBJoulesMol*T))*deltaE)

        foutOutput.write("prob acceptance = %s\n" % (probAcceptance))

        if probAcceptance>=1:
                return 1
        else:
                randomNumber=random.random()
                foutOutput.write("%s\n" % (randomNumber))
                if probAcceptance>=randomNumber:
                        return 1
                else:
                        return 0

def writeClassicalStructureFile(structureNumber):
        """Writes 2 files, one for the ligand and one for the host"""
        fin=open("structure_%s.g96" % (structureNumber+1), ’r’).readlines()

        foutClassicalLigand=open("structure_%s_LIGAND.g96" % (structureNumber+1),
     ’w’)

        foutClassicalLigand.write("TITLE\n1359 system\nEND\nPOSITION\n")

        c=0
        for i in fin:
                if "VELOCITY" in i:
                        x=c
                c+=1

        fin2=fin[:x]

        for line in fin2:
                if "EOH" in line:
                        foutClassicalLigand.write(line)

        foutClassicalLigand.write("END\nBOX\n")
        foutClassicalLigand.write(fin[-2])
        foutClassicalLigand.write("END")

        foutClassicalHost=open("structure_%s_HOST.g96" % (structureNumber+1), ’w’
    )
```

```
        foutClassicalHost.write("TITLE\n1359 system\nEND\nPOSITION\n")



        for line in fin2:
                if "WAT" in line:
                        foutClassicalHost.write(line)
        foutClassicalHost.write("END\nBOX\n")
        foutClassicalHost.write(fin[-2])
        foutClassicalHost.write("END")

def calculateInteractionEnergy(structureNumber):
        """Calculates the electrostatic term for the MM"""

        complexElectrostatics=float(extractEnergy("structure_%s_end.log" % (
    structureNumber), "Coulomb")) + float(extractEnergy("structure_%s_end.log" %
    (structureNumber), "Coul."))

        #RUN THE CALCULATIONS HERE!!!

        os.system("mdrun_d -s 450_waters.tpr -rerun structure_%s_HOST.g96 -g
    structure_%s_end_HOST.log -e structure_%s_end_HOST.edr -o structure_%
    s_end_HOST.trr &> gromacs_output.txt" % ( structureNumber+1, structureNumber,
     structureNumber, structureNumber))
        os.system("mdrun_d -s ethanol_vacuum.tpr -rerun structure_%s_LIGAND.g96 -
    g structure_%s_end_LIGAND.log -e structure_%s_end_LIGAND.edr -o structure_%
    s_end_LIGAND.trr &> gromacs_output.txt" % ( structureNumber+1,
    structureNumber, structureNumber, structureNumber))



        for result in re.findall('Energies(.*?)Constr.', open("structure_%
    s_end_LIGAND.log" % (structureNumber), 'r').read(), re.S):
                ligandElectrostatics=float(result.split()[21]) + float(result.
    split()[22])

        for result in re.findall('Energies(.*?)Constr.', open("structure_%
    s_end_HOST.log" % (structureNumber), 'r').read(), re.S):
               hostElectrostatics=float(result.split()[11]) + float(result.split
    ()[12])

        return(complexElectrostatics - ligandElectrostatics - hostElectrostatics)


#22/08/13 16:39
```

```
def move(ligandAtoms, solventAtoms, structureNumber,
    initialPotentialComplexEnergy, initialPotentialHostEnergy,
    initialPotentialLigandEnergy, initialClassicalComplexEnergy,
    initialMMElectrostatics, failCount):
        """Controls the movement step with the Monte Carlo"""
        #Generate new velocities
        (velocities)=generateVelocities(ligandAtoms, solventAtoms)
        #Calculate the values the velocities need to be scaled by
        (xVAll, yVAll, zVAll)=scaleVelocities(ligandAtoms, solventAtoms,
    velocities)
        #Scale the new velocities
        newVelocities=removeCenterOfMass(velocities, xVAll, yVAll, zVAll)
        #Write the velocities to file
        foutOutput.write("WRITING TO FILE structure_%s.g96" % (structureNumber) )
        writeVelocitiesG96("structure_%s.g96" % (structureNumber), newVelocities)
        #Run gromacs
        runGromacs("structure_%s.g96" % (structureNumber), "structure_%s_end.tpr"
     % (structureNumber), structureNumber)
        #Extract energies
        initialKineticEnergy=kineticEnergyCalc("structure_%s_end.log" % (
    structureNumber), "Kinetic")
        finalKineticEnergy=extractEnergy("structure_%s_end.log" % (
    structureNumber), "Kinetic")
        finalClassicalComplexEnergy=extractEnergy("structure_%s_end.log" % (
    structureNumber), "Potential")
        finalLJEnergy=extractEnergy("structure_%s_end.log" % (structureNumber), "
    LJ")

#HMC edit
        (boxInfo, x, y, z)=g96Reader("structure_%s.g96" % (structure+1))
        (newX, newY, newZ)=wrapCoordinates(x, y, z, boxInfo)
        writeClassicalStructureFile(structureNumber)
        finalMMElectrostatics=calculateInteractionEnergy(structureNumber)
        writeOnetepFile(newX, newY, newZ, boxInfo, ligandAtoms, solventAtoms)
        #Make this so 3 energies from thisfunction
        (finalPotentialComplexEnergy, finalPotentialHostEnergy,
    finalPotentialLigandEnergy)=runOnetep(structure)

        initialCorrectedInteractionPotential=((initialClassicalComplexEnergy -
    initialMMElectrostatics) + (initialPotentialComplexEnergy -
    initialPotentialHostEnergy - initialPotentialLigandEnergy))
        finalCorrectedInteractionPotential=((finalClassicalComplexEnergy -
    finalMMElectrostatics) + (finalPotentialComplexEnergy -
    finalPotentialHostEnergy - finalPotentialLigandEnergy))
```

```
        print initialCorrectedInteractionPotential,
    finalCorrectedInteractionPotential

        initialQMLambda=(((1-lambdaWindow)*float(initialClassicalComplexEnergy))
    +((lambdaWindow)*initialCorrectedInteractionPotential))
        finalQMLambda=(((1-lambdaWindow)*float(finalClassicalComplexEnergy))+((
    lambdaWindow)*finalCorrectedInteractionPotential))

        totalEnergy1=float(initialQMLambda)+float(initialKineticEnergy)
        totalEnergy2=float(finalQMLambda)+float(finalKineticEnergy)

        acceptance=acceptanceCriteriaHybridMC(totalEnergy1, totalEnergy2)
#Return comp host lig for MM and QM
        return(acceptance, float(initialKineticEnergy), float(finalKineticEnergy)
    , float(finalPotentialComplexEnergy), finalPotentialHostEnergy,
    finalPotentialLigandEnergy, float(finalMMElectrostatics), float(
    finalClassicalComplexEnergy), failCount)

#23/1/14 11:16
def classicalMove(structureNumber, initialClassicalComplexEnergy):
        """Controls the classical hybrid Monte Carlo movement"""

        (velocities)=generateVelocities(ligandAtoms, solventAtoms)
        #Calculate the values the velocities need to be scaled by
        (xVAll, yVAll, zVAll)=scaleVelocities(ligandAtoms, solventAtoms,
    velocities)
        #Scale the new velocities
        newVelocities=removeCenterOfMass(velocities, xVAll, yVAll, zVAll)
        #Write the velocities to file
        foutOutput.write("WRITING TO FILE structure_%s.g96" % (structureNumber) )
        writeVelocitiesG96("structure_%s.g96" % (structureNumber), newVelocities)
        #Run gromacs
        runGromacs("structure_%s.g96" % (structureNumber), "structure_%s_end.tpr"
     % (structureNumber), structureNumber)
        #Extract energies
        initialKineticEnergy=kineticEnergyCalc("structure_%s_end.log" % (
    structureNumber), "Kinetic")
        finalKineticEnergy=extractEnergy("structure_%s_end.log" % (
    structureNumber), "Kinetic")
        finalClassicalComplexEnergy=extractEnergy("structure_%s_end.log" % (
    structureNumber), "Potential")
```

```
        finalLJEnergy=extractEnergy("structure_%s_end.log" % (structureNumber), "
    LJ")

        totalEnergy1=float(initialKineticEnergy)+float(
    initialClassicalComplexEnergy)
        totalEnergy2=float(finalKineticEnergy)+float(finalClassicalComplexEnergy)

        acceptance=acceptanceCriteriaHybridMC(totalEnergy1, totalEnergy2)

        return(acceptance, initialKineticEnergy, finalKineticEnergy,
    finalLJEnergy, finalClassicalComplexEnergy)

def writeOnetepFile(x, y, z, boxSize, ligandAtoms, solventAtoms):
       """Input: coordinates
       Output: Onetep dat file"""
       from decimal import Decimal

       fout=open("qmElec_embed_coords_%d.dat" % int(structure), 'w')

       fout.write("cutoff_energy       : 800 eV  \n")
       fout.write("ngwf_threshold_orig : 0.000002\n")
       fout.write("kernel_cutoff       : 1000    \n")
       fout.write("k_zero              : 3.5     \n")
       fout.write("write_xyz true                \n")
       fout.write("write_tightbox_ngwfs false    \n")
       fout.write("write_denskern       false    \n")
       fout.write("                              \n")
       fout.write("elec_cg_max 5                 \n")
       fout.write("occ_mix 1.0                   \n")
       fout.write("                              \n")
#       fout.write("threadsmax 4                  \n")
#       fout.write("threadsperfftbox 1            \n")
#       fout.write("threadsnumfftboxes 4          \n")
#       fout.write("threadspercellfft 4           \n")
#       fout.write("threadsnummkl 4               \n")
       fout.write("                              \n")
#       fout.write("comms_group_size 4            \n")
       #Change to 10 if PAOs are used
       fout.write("minit_lnv 5                   \n")
       fout.write("maxit_lnv 5                   \n")
       fout.write("                              \n")
       fout.write("maxit_pen 0                   \n")
       fout.write("                              \n")
       fout.write("dispersion 1                  \n")
```

```
        fout.write("                              \n")
        fout.write("                              \n")
        fout.write("lnv_threshold_orig 1.0e-7     \n")
        fout.write("                              \n")
        fout.write("output_detail VERBOSE         \n")
        fout.write("                              \n")
        fout.write("xc_functional PBE             \n")
        fout.write("                              \n")
        fout.write("maxit_ngwf_cg 100             \n")
        #fout.write("maxit_ngwf_cg 0              \n")
        fout.write("                              \n")
#Add additional atom types here
        fout.write("%block species                \n")
#        fout.write("N   N    7 4 8.0              \n")
        fout.write("H   H    1 1 8.0              \n")
        fout.write("C   C    6 4 8.0              \n")
        fout.write("O   O    8 4 8.0 \n")
#        fout.write("Cl  Cl  17 8 8.0 \n")
        fout.write("%endblock species             \n")
        fout.write("                              \n")
#        fout.write("%block species_atomic_set     \n")
#        fout.write('H "SOLVE conf=1s1"            \n')
#        fout.write('C "SOLVE conf=2s2 2p4"        \n')
#        fout.write("N AUTO\n")
#        fout.write("%endblock species_atomic_set  \n")
        fout.write("                              \n")
#        fout.write("initial_dens_realspace F\n")
        fout.write("\n")
#Add additional pseudopotentials here
        fout.write("%block species_pot            \n")
#        fout.write("N   N_00.recpot           \n")
        fout.write("H   H_04.recpot          \n")
        fout.write("C   C_01.recpot            \n")
        fout.write("O   O_01.recpot\n")
#        fout.write("Cl  Cl_00.recpot\n")
        fout.write("%endblock species_pot         \n")
        fout.write("                              \n")
        fout.write("%block lattice_cart           \n")

        #Sorts out the box sizes here

        roundedX=str(Decimal(str(float(boxSize[0])*nm2bohr)).quantize(Decimal(10)
    **-3))
```

```
       roundedY=str(Decimal(str(float(boxSize[1])*nm2bohr)).quantize(Decimal(10)
    **-3))
       roundedZ=str(Decimal(str(float(boxSize[2])*nm2bohr)).quantize(Decimal(10)
    **-3))


       fout.write((7-len(roundedX))*" "+roundedX+"   0.000   0.000\n")
       fout.write("  0.000"+(8-len(roundedY))*" "+roundedY+"   0.000\n")
       fout.write("  0.000   0.000"+(8-len(roundedY))*" "+roundedZ+"\n")

       fout.write("%endblock lattice_cart        \n\n")


       fout.write("%block positions_abs\n")

       #Writes the standard full atom section here
       for i in range(len(ligandAtoms)):
               #Rounds values to 8 decimal places after converting them from Ang
    to Bohr
                roundedX=str(Decimal(str(x[i]*nm2bohr)).quantize(Decimal(10)**-8)
    )
                roundedY=str(Decimal(str(y[i]*nm2bohr)).quantize(Decimal(10)**-8)
    )
                roundedZ=str(Decimal(str(z[i]*nm2bohr)).quantize(Decimal(10)**-8)
    )
                try:
                        fout.write(ligandAtoms[i]+(17-len(roundedX))*" "+str(
    roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(roundedZ))*" "+str(
    roundedZ)+"\n")
                except: pass
#                       fout.write(solventAtoms[i-len(ligandAtoms)]+(17-len(
    roundedX))*" "+str(roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(
    roundedZ))*" "+str(roundedZ)+"\n")

       fout.write("%endblock positions_abs\n\n")

       fout.write("%block classical_info\n")



       #Writes the electrostatic embedding section here
       for i in range( len(ligandAtoms), (len(solventAtoms)+len(ligandAtoms))):
               roundedX=str(Decimal(str(x[i]*nm2bohr)).quantize(Decimal(10)**-8))
               #Decimal module will round to 0E-8 if the value is 0,
               #Below converts back to standard format
               if roundedX=="0E-8":
                       roundedX="0.00000000"

               roundedY=str(Decimal(str(y[i]*nm2bohr)).quantize(Decimal(10)**-8))
               if roundedY=="0E-8":
                       roundedY="0.00000000"
               roundedZ=str(Decimal(str(z[i]*nm2bohr)).quantize(Decimal(10)**-8))
               if roundedZ=="0E-8":
                       roundedZ="0.00000000"

               if solventAtoms[i-len(ligandAtoms)]=='O':
                        #These Charges are consistent with the flexible SPC model
                       charge=-0.82
               elif solventAtoms[i-len(ligandAtoms)]=='H':
                       charge=0.41
               else:
                       print "SOLVENT ATOM COULD NOT BE FOUND"
               fout.write(solventAtoms[i-len(ligandAtoms)]+(17-len(roundedX))*" "
    +str(roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(roundedZ))*" "+
    str(roundedZ)+(15-len(str(charge)))*" "+str(charge)+"\n")

       fout.write("%endblock classical_info\n")

#############################################################################
#################################### HOST ###################################
#############################################################################

       foutHost=open("qmElec_embed_coords_%d_HOST.dat" % int(structure), 'w')

       foutHost.write("cutoff_energy       : 800 eV  \n")
       foutHost.write("ngwf_threshold_orig : 0.000002\n")
       foutHost.write("kernel_cutoff       : 1000    \n")
       foutHost.write("k_zero              : 3.5     \n")
       foutHost.write("write_xyz true                \n")
       foutHost.write("write_tightbox_ngwfs false    \n")
       foutHost.write("write_denskern       false    \n")
       foutHost.write("                              \n")
       foutHost.write("elec_cg_max 5                 \n")
       foutHost.write("occ_mix 1.0                   \n")
       foutHost.write("                              \n")
       foutHost.write("                              \n")
       foutHost.write("minit_lnv 0                   \n")
       foutHost.write("maxit_lnv 0                   \n")
```

```
   foutHost.write("                              \n")
   foutHost.write("maxit_pen 0                   \n")
   foutHost.write("                              \n")
   foutHost.write("dispersion 0                  \n")
   foutHost.write("                              \n")
   foutHost.write("                              \n")
   foutHost.write("lnv_threshold_orig 1.0e-7     \n")
   foutHost.write("                              \n")
   foutHost.write("output_detail VERBOSE         \n")
   foutHost.write("                              \n")
   foutHost.write("xc_functional PBE             \n")
   foutHost.write("                              \n")
   foutHost.write("maxit_ngwf_cg 0             \n")
   foutHost.write("                              \n")
   foutHost.write("%block species                \n")
   foutHost.write("H   H    1 1 8.0              \n")
   foutHost.write("C   C    6 4 8.0              \n")
   foutHost.write("O   O    8 4 8.0 \n")
   foutHost.write("%endblock species             \n")
   foutHost.write("                              \n")
   foutHost.write("                              \n")
   foutHost.write("\n")
   foutHost.write("%block species_pot            \n")
   foutHost.write("H   null.recpot           \n")
   foutHost.write("C   null.recpot             \n")
   foutHost.write("O   null.recpot\n")
   foutHost.write("%endblock species_pot         \n")
   foutHost.write("                              \n")
   foutHost.write("%block lattice_cart           \n")

   #Sorts out the box sizes here

   roundedX=str(Decimal(str(float(boxSize[0])*nm2bohr)).quantize(Decimal(10)
**-3))
   roundedY=str(Decimal(str(float(boxSize[1])*nm2bohr)).quantize(Decimal(10)
**-3))
   roundedZ=str(Decimal(str(float(boxSize[2])*nm2bohr)).quantize(Decimal(10)
**-3))



   foutHost.write((7-len(roundedX))*" "+roundedX+"   0.000   0.000\n")
   foutHost.write("  0.000"+(8-len(roundedY))*" "+roundedY+"   0.000\n")
   foutHost.write("  0.000   0.000"+(8-len(roundedY))*" "+roundedZ+"\n")
```

```
       foutHost.write("%endblock lattice_cart        \n\n")

       foutHost.write("%block positions_abs\n")

       #Writes the standard full atom section here
       for i in range(len(ligandAtoms)):
               #Rounds values to 8 decimal places after converting them from Ang to Bohr
                roundedX=str(Decimal(str(x[i]*nm2bohr)).quantize(Decimal(10)**-8))
                roundedY=str(Decimal(str(y[i]*nm2bohr)).quantize(Decimal(10)**-8))
                roundedZ=str(Decimal(str(z[i]*nm2bohr)).quantize(Decimal(10)**-8))
                try:
                        foutHost.write(ligandAtoms[i]+(17-len(roundedX))*" "+str(roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(roundedZ))*" "+str(roundedZ)+"\n")
                except: pass
#                       fout.write(solventAtoms[i-len(ligandAtoms)]+(17-len(roundedX))*" "+str(roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(roundedZ))*" "+str(roundedZ)+"\n")

       foutHost.write("%endblock positions_abs\n\n")



       foutHost.write("%block classical_info\n")




       #Writes the electrostatic embedding section here
       for i in range( len(ligandAtoms), (len(solventAtoms)+len(ligandAtoms))):
               roundedX=str(Decimal(str(x[i]*nm2bohr)).quantize(Decimal(10)**-8))
               #Decimal module will round to 0E-8 if the value is 0,
               #Below converts back to standard format
               if roundedX=="0E-8":
                       roundedX="0.00000000"

               roundedY=str(Decimal(str(y[i]*nm2bohr)).quantize(Decimal(10)**-8))
               if roundedY=="0E-8":
                       roundedY="0.00000000"
               roundedZ=str(Decimal(str(z[i]*nm2bohr)).quantize(Decimal(10)**-8))
               if roundedZ=="0E-8":
                       roundedZ="0.00000000"
                if solventAtoms[i-len(ligandAtoms)]=='O':
                        #These Charges are consistent with the flexible SPC model
                        charge=-0.82
                elif solventAtoms[i-len(ligandAtoms)]=='H':
                        charge=0.41
                else:
                        print "SOLVENT ATOM COULD NOT BE FOUND"
                foutHost.write(solventAtoms[i-len(ligandAtoms)]+(17-len(roundedX))
    *" "+str(roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(roundedZ))*"
    "+str(roundedZ)+(15-len(str(charge)))*" "+str(charge)+"\n")

        foutHost.write("%endblock classical_info\n")

#############################################################################
################################### LIGAND ##################################
#############################################################################


        foutLigand=open("qmElec_embed_coords_%d_LIGAND.dat" % int(structure), 'w')

        foutLigand.write("cutoff_energy       : 800 eV  \n")
        foutLigand.write("ngwf_threshold_orig : 0.000002\n")
        foutLigand.write("kernel_cutoff       : 1000    \n")
        foutLigand.write("k_zero              : 3.5     \n")
        foutLigand.write("write_xyz true                \n")
        foutLigand.write("write_tightbox_ngwfs false    \n")
        foutLigand.write("write_denskern       false    \n")
        foutLigand.write("                              \n")
        foutLigand.write("elec_cg_max 5                 \n")
        foutLigand.write("occ_mix 1.0                   \n")
        foutLigand.write("                              \n")
        foutLigand.write("                              \n")
        foutLigand.write("minit_lnv 5                   \n")
        foutLigand.write("maxit_lnv 5                   \n")
        foutLigand.write("                              \n")
        foutLigand.write("maxit_pen 0                   \n")
        foutLigand.write("                              \n")
        foutLigand.write("dispersion 1                  \n")
        foutLigand.write("                              \n")
        foutLigand.write("                              \n")
        foutLigand.write("lnv_threshold_orig 1.0e-7     \n")
        foutLigand.write("                              \n")
        foutLigand.write("output_detail VERBOSE         \n")
```

```
   foutLigand.write("                            \n")
   foutLigand.write("xc_functional PBE           \n")
   foutLigand.write("                            \n")
   foutLigand.write("maxit_ngwf_cg 100           \n")
   foutLigand.write("                            \n")
   foutLigand.write("%block species              \n")
   foutLigand.write("H  H    1 1 8.0             \n")
   foutLigand.write("C  C    6 4 8.0             \n")
   foutLigand.write("O  O    8 4 8.0 \n")
   foutLigand.write("%endblock species           \n")
   foutLigand.write("                            \n")
   foutLigand.write("                            \n")
   foutLigand.write("\n")
   foutLigand.write("%block species_pot          \n")
   foutLigand.write("H  H_04.recpot          \n")
   foutLigand.write("C  C_01.recpot            \n")
   foutLigand.write("O  O_01.recpot\n")
   foutLigand.write("%endblock species_pot       \n")
   foutLigand.write("                            \n")
   foutLigand.write("%block lattice_cart         \n")

   #Sorts out the box sizes here

   roundedX=str(Decimal(str(float(boxSize[0])*nm2bohr)).quantize(Decimal(10)
**-3))
   roundedY=str(Decimal(str(float(boxSize[1])*nm2bohr)).quantize(Decimal(10)
**-3))
   roundedZ=str(Decimal(str(float(boxSize[2])*nm2bohr)).quantize(Decimal(10)
**-3))



   foutLigand.write((7-len(roundedX))*" "+roundedX+"   0.000   0.000\n")
   foutLigand.write("  0.000"+(8-len(roundedY))*" "+roundedY+"   0.000\n")
   foutLigand.write("  0.000   0.000"+(8-len(roundedY))*" "+roundedZ+"\n")

   foutLigand.write("%endblock lattice_cart       \n\n")



   foutLigand.write("%block positions_abs\n")

   #Writes the standard full atom section here
   for i in range(len(ligandAtoms)):
           #Rounds values to 8 decimal places after converting them from Ang
to Bohr
```

```
                roundedX=str(Decimal(str(x[i]*nm2bohr)).quantize(Decimal(10)**-8)
    )
                roundedY=str(Decimal(str(y[i]*nm2bohr)).quantize(Decimal(10)**-8)
    )
                roundedZ=str(Decimal(str(z[i]*nm2bohr)).quantize(Decimal(10)**-8)
    )
                try:
                        foutLigand.write(ligandAtoms[i]+(17-len(roundedX))*" "+
    str(roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(roundedZ))*" "+str
    (roundedZ)+"\n")
                except: pass
#                       fout.write(solventAtoms[i-len(ligandAtoms)]+(17-len(
    roundedX))*" "+str(roundedX)+(15-len(roundedY))*" "+str(roundedY)+(15-len(
    roundedZ))*" "+str(roundedZ)+"\n")

        foutLigand.write("%endblock positions_abs\n\n")



#22/08/13 16:45
def runOnetep(QMStructure):
        """Input: None
        Output: energy"""
        foutOutput.write("Running ONETEP...\n")

        os.system( "mpirun -np 5 -hostfile nodes.host ./onetep.iridis3
    qmElec_embed_coords_%s_HOST.dat > qmElec_embed_coords_%s_HOST.out &" % (int(
    QMStructure), int(QMStructure)))

        os.system( "mpirun -np 5 -hostfile nodes.lig ./onetep.iridis3
    qmElec_embed_coords_%s_LIGAND.dat > qmElec_embed_coords_%s_LIGAND.out &" % (
    int(QMStructure), int(QMStructure)))

        os.system( "mpirun -np 4 -hostfile nodes.comp ./onetep.iridis3
    qmElec_embed_coords_%s.dat > qmElec_embed_coords_%s.out" % (int(QMStructure),
     int(QMStructure)))

        looper=0

        while looper==0:
                try:
                        for result in re.findall('Epredicted(.*?)<--', open("
    qmElec_embed_coords_%s.out" % (QMStructure), 'r').read(), re.S):
                                if float(result.split()[-2])<0.000002:
                                        complexEnergy=result.split()[-1]
```

```
                                       foutOutput.write(60*"-"+"ONETEP ENERGY
    CONVERGED"+60*"-"+"\n")

                       for result in re.findall('Ewald Energy(.*?)===', open("
    qmElec_embed_coords_%s_HOST.out" % (QMStructure), 'r').read(), re.S):
                                       hostEnergy=result.split()[1]

                       for result in re.findall('Epredicted(.*?)<--', open("
    qmElec_embed_coords_%s_LIGAND.out" % (QMStructure), 'r').read(), re.S):
                               if float(result.split()[-2])<0.000002:
                                       ligandEnergy=result.split()[-1]
                                       foutOutput.write(60*"-"+"ONETEP LIGAND
    ENERGY CONVERGED"+60*"-"+"\n")

                       looper=1
               except:
                       looper=0



       return float(complexEnergy)*hartrees2kj, float(hostEnergy)*hartrees2kj,
    float(ligandEnergy)*hartrees2kj

def MetropolisHastingsAcceptance(initialQuantMEnergy, finalQuantMEnergy,
    initialCLEnergy, finalCLEnergy):
        """Input: start & end QM energy, start & end classical energy
        Output: whether the structure is accepted or not"""

        deltaDeltaE=((float(finalQuantMEnergy)-float(finalCLEnergy))-(float(
    initialQuantMEnergy)-float(initialCLEnergy)))

        print "QUANTUM ACCEPTANCE VALUE = ", deltaDeltaE

        probAcceptance=numpy.exp(-(1/(kBJoulesMol*T))*deltaDeltaE)

        print "prob acceptance = ", probAcceptance

        if probAcceptance>=1:
                return(1, deltaDeltaE)
        else:
                randomNumber=random.random()
                foutOutput.write("randomNumber %s\n" % (randomNumber))
                if probAcceptance>=randomNumber:
                        return(1, deltaDeltaE)
                else:
                        return(0, deltaDeltaE)

def wrapCoordinates(xCoordinates, yCoordinates, zCoordinates, boxSize):
       """Input: coordinates
       Output: Wrapped coordinates"""

       boxSizeX=float(boxSize[0])
       boxSizeY=float(boxSize[1])
       boxSizeZ=float(boxSize[2])

       wrappedXCoordinates=[]
       wrappedYCoordinates=[]
       wrappedZCoordinates=[]

       #Subtracting a box length from all coordinates that are outside the box
       for value in xCoordinates:
               if value>(boxSizeX/2):
                       wrappedXCoordinates.append(value-boxSizeX)
               else:
                       wrappedXCoordinates.append(value)

       for value in yCoordinates:
               if value>(boxSizeY/2):
                       wrappedYCoordinates.append(value-boxSizeY)
               else:
                       wrappedYCoordinates.append(value)

       for value in zCoordinates:
               if value>(boxSizeZ/2):
                       wrappedZCoordinates.append(value-boxSizeZ)
               else:
                       wrappedZCoordinates.append(value)

       #translates the box so the left hand corner is at the origin
       wrappedXCoordinates=numpy.array(wrappedXCoordinates)-min(
    wrappedXCoordinates)
       wrappedYCoordinates=numpy.array(wrappedYCoordinates)-min(
    wrappedYCoordinates)
       wrappedZCoordinates=numpy.array(wrappedZCoordinates)-min(
    wrappedZCoordinates)

       return wrappedXCoordinates, wrappedYCoordinates, wrappedZCoordinates
```

```
#22/08/13 16:04
def inputReader(file):
        """Input: file specifying all parameters for run
        Output: parameters used for run in python variables"""

        fin=open(file, 'r').readlines()

        error=0
        for line in fin:
                if "-QM" in line.split()[0]:
                        numberOfQM=int(line.split()[1])
                elif "-MM" in line.split()[0]:
                        numberOfMM=int(line.split()[1])
                elif "-TOP" in line.split()[0]:
                        topologyFile=line.split()[1]
                elif "-LJ" in line.split()[0]:
#CHANGE code at later date
                        lJParam=0
                elif "-G96" in line.split()[0]:
                        g96File=line.split()[1]
                elif "-EQ" in line.split()[0]:
                        equilibrationNumber=int(line.split()[1])
                elif "-CONT" in line.split()[0]:
                        continuation=int(line.split()[1])

                else:
                        print 120*"-"
                        print "                                       UNKNOWN
    FLAG IN INPUT FILE", line.split()[0]
                        print 120*"-"
                        error=1

        if error==0:
                try:
                        return(topologyFile, g96File, numberOfQM, numberOfMM,
    lJParam, equilibrationNumber, continuation)
                except UnboundLocalError:
                        print  120*"-"
                        print "                                       CHECK ALL
    FLAGS PRESENT IN INPUT FILE"
                        print 120*"-"
```

```
#
    ----------------------------------------------------------------------------------------------------------

#
    ----------------------------------------------------------------------------------------------------------
    CONTROL
    ----------------------------------------------------------------------------------------------------------

#
    ----------------------------------------------------------------------------------------------------------


#23/08/13 09:20
if __name__=='__main__':

        (topologyFile, g96File, numberOfQM, numberOfMM, lJParam,
    equilibrationNumber, CONT)=inputReader(sys.argv[1])



        print "Using %s as topology file and %s for coordinates" % (topologyFile,
     g96File)

        fout=open("run_information.txt", 'w')
        fout.write("Structure   Start Ekin      Qpot    Cpot    LJ      End Ekin
            Qpot    Cpot    LJ      Epot\n")

        #Sets the value of lambda
        lambdaWindow=float(sys.argv[2])

        #Start structure number
        structure=0

        #Read in gro file for ligand and solvent atoms
        (ligandAtoms, solventAtoms, boxSize)=readG96FileNames(g96File)

        if len(solventAtoms)==0:
                print 120*"-"
                print "WARNING, YOU ARE RUNNING IN A VACUUM AND WILL NEED TO MAKE
     SMALL CHANGES TO THE CODE IN MODULE 'writeOnetepFile'"
                print 120*"-"

        #Copys the original file to structure_0.g96 -- Change this?
        os.system('cp %s structure_%s.g96' % (g96File, structure))
        print("copying %s to structure_%s.g96" % (g96File, structure))
        if CONT==0:

                #initialise the average Energy variables
                averageEnergyDifference=-245743.865632
                averageLength=0

                #runs gromacs
                runGromacs("structure_%s.g96" % (structure), "structure_%s_end.
    tpr" % (structure), structure)

                initialClassicalComplexEnergy=extractEnergy("structure_%s_end.log
    " % (structure), "Potential")
                initialLJEnergy=extractEnergy("structure_%s_end.log" % (structure
    ), "LJ")

                #Equilibration Steps

                initialPotentialComplexEnergy="N/A"
                finalPotentialComplexEnergy="N/A"
                counterMM=0
                while counterMM<equilibrationNumber:
                        fout.flush()
                        foutOutput.flush()
                        structure+=1

                        (acceptance, initialKineticEnergy, finalKineticEnergy,
    finalLJEnergy, finalClassicalComplexEnergy)=classicalMove(structure,
    initialClassicalComplexEnergy)

                        #Write classical move

                        if acceptance==1:
                                fout.write("%s  %s      %s      %s      %s      %
    s      %s      %s      %s      Accepted\n" % (structure, initialKineticEnergy
    , initialPotentialComplexEnergy, initialClassicalComplexEnergy,
    initialLJEnergy, finalKineticEnergy, finalPotentialComplexEnergy,
    finalClassicalComplexEnergy, finalLJEnergy))
                                fout.write("structure %s accepted!!!!!\n" % (
    structure))
                                initialPotentialComplexEnergy=
    finalPotentialComplexEnergy
```

```
                                initialClassicalComplexEnergy=
    finalClassicalComplexEnergy
                                initialLJEnergy=finalLJEnergy
                                counterMM+=1
                                acceptedStructure=structure
                        else:
                                fout.write("%s  %s      %s      %s      %s      %
    s      %s      %s      %s      Failed\n" % (structure, initialKineticEnergy,
    initialPotentialComplexEnergy, initialClassicalComplexEnergy, initialLJEnergy
    , finalKineticEnergy, finalPotentialComplexEnergy,
    finalClassicalComplexEnergy, finalLJEnergy))
                                fout.write("Structure FAILED\n")
                                os.system("cp structure_%s.g96 structure_%s.g96"
    % ( structure, structure+1))
                                counterMM+=1



                fout.write("Structure   Start Ekin      Qcomp   Qhost   Qlig
    Ccomp   Celec   End Ekin        Qpot    Qhost   Qlig    Cpot    Celec   \n")

                #initialise the force constant
                forceConst=0

                (boxInfo, x, y, z)=g96Reader("structure_%s.g96" % (structure+1))
                (newX, newY, newZ)=wrapCoordinates(x, y, z, boxInfo)
#               print("length of newZ=", len(newZ))

                writeClassicalStructureFile(structure)
                initialMMElectrostatics=calculateInteractionEnergy(structure)
                writeOnetepFile(newX, newY, newZ, boxInfo, ligandAtoms,
    solventAtoms)
                (initialPotentialComplexEnergy, initialPotentialHostEnergy,
    initialPotentialLigandEnergy)=runOnetep(structure)

                failCount=0
                counterQM=0
                while counterQM<numberOfQM:
                        fout.flush()
                        foutOutput.flush()
                        #Add 1 to the structure
                        structure+=1

                        #Makes a move
```

```
                    (acceptance, initialKineticEnergy, finalKineticEnergy,
finalPotentialComplexEnergy, finalPotentialHostEnergy,
finalPotentialLigandEnergy, finalMMELectrostatics,
finalClassicalComplexEnergy, failCount)=move(ligandAtoms, solventAtoms,
structure, initialPotentialComplexEnergy, initialPotentialHostEnergy,
initialPotentialLigandEnergy, initialClassicalComplexEnergy,
initialMMElectrostatics, failCount)

                    if acceptance==1:
                            fout.write("%s  %s  %s      %s  %s      %s      %
s      %s      %s      %s      %s      %s      %s      Accepted    %s\n" % (
structure, initialKineticEnergy, initialPotentialComplexEnergy,
initialPotentialHostEnergy, initialPotentialLigandEnergy,
initialClassicalComplexEnergy, initialMMElectrostatics, finalKineticEnergy,
finalPotentialComplexEnergy, finalPotentialHostEnergy,
finalPotentialLigandEnergy, finalClassicalComplexEnergy,
finalMMELectrostatics, failCount))
                            fout.write("structure %s accepted!!!!!\n" % (
structure))
                            initialPotentialComplexEnergy=
finalPotentialComplexEnergy
                            initialClassicalComplexEnergy=
finalClassicalComplexEnergy
                            initialPotentialHostEnergy=
finalPotentialHostEnergy
                            initialPotentialLigandEnergy=
finalPotentialLigandEnergy
                            initialMMElectrostatics=finalMMELectrostatics
                            counterQM+=1
                            acceptedStructure=structure
                            failCount=0
                    else:
                            fout.write("%s  %s      %s  %s  %s      %s      %
s      %s      %s      %s      %s      %s      %s      Failed      %s\n" % (
structure, initialKineticEnergy, initialPotentialComplexEnergy,
initialPotentialHostEnergy, initialPotentialLigandEnergy,
initialClassicalComplexEnergy, initialMMElectrostatics, finalKineticEnergy,
finalPotentialComplexEnergy, finalPotentialHostEnergy,
finalPotentialLigandEnergy, finalClassicalComplexEnergy,
finalMMELectrostatics, failCount))
                            fout.write("Structure FAILED\n")
                            os.system("cp structure_%s.g96 structure_%s.g96"
% ( structure, structure+1))
                            counterQM+=1
```

```
                                failCount+=1


        elif CONT==1:


                finContinuation=open("../run_information.txt", 'r').readlines()

                for line in finContinuation:
                        if "Accepted" in line:
                                try:
                                        structure=int(line.split()[0])
                                        initialClassicalComplexEnergy=float(line.
    split()[7])
                                        initialLJEnergy=float(line.split()[8])
                                        averageEnergyDifference=float(line.split
    ()[9])
                                        forceConst=float(line.split()[-1])
                                        averageLength=0
                                        failCount=0
                                        initialPotentialComplexEnergy=float(line.
    split()[6])

                                except: pass



                #the count will start at 1, required for the correct qmElecEmbed
    numbering
                os.system("cp structure_0.g96 structure_%s.g96" % (structure+1))
                print "copying structure_0.g96 structure_%s.g96" % (structure+1)

                (boxInfo, x, y, z)=g96Reader("structure_%s.g96" % (structure+1))
                (newX, newY, newZ)=wrapCoordinates(x, y, z, boxInfo)

                writeOnetepFile(newX, newY, newZ, boxInfo, ligandAtoms,
    solventAtoms)
#               (initialPotentialComplexEnergy)=runOnetep(structure)

                counterQM=0
                while counterQM<numberOfQM:
                        fout.flush()
                        foutOutput.flush()
```

```
                    #Add 1 to the structure
                    structure+=1




                    (acceptance, initialKineticEnergy, finalKineticEnergy,
finalPotentialComplexEnergy, finalLJEnergy, finalClassicalComplexEnergy,
averageEnergyDifference, failCount)=move(ligandAtoms, solventAtoms, structure
, initialPotentialComplexEnergy, initialLJEnergy, averageEnergyDifference,
averageLength, failCount, forceConst)

                    if acceptance==1:
                            fout.write("%s  %s  %s      %s      %s      %s
   %s      %s      %s      %s      %s      Accepted    %s\n" % (structure,
initialKineticEnergy, initialPotentialComplexEnergy,
initialClassicalComplexEnergy, initialLJEnergy, finalKineticEnergy,
finalPotentialComplexEnergy, finalClassicalComplexEnergy, finalLJEnergy,
averageEnergyDifference, failCount, forceConst))
                            fout.write("structure %s accepted!!!!!\n" % (
structure))
                            initialPotentialComplexEnergy=
finalPotentialComplexEnergy
                            initialClassicalComplexEnergy=
finalClassicalComplexEnergy
                            initialPotentialHostEnergy=
finalPotentialHostEnergy
                            initialPotentialLigandEnergy=
finalPotentialLigandEnergy
                            counterQM+=1
                            acceptedStructure=structure
                            failCount=0
                    else:
                            fout.write("%s  %s  %s      %s      %s      %s
   %s      %s      %s      %s      %s      Failed      %s\n" % (structure,
initialKineticEnergy, initialPotentialComplexEnergy,
initialClassicalComplexEnergy, initialLJEnergy, finalKineticEnergy,
finalPotentialComplexEnergy, finalClassicalComplexEnergy, finalLJEnergy,
averageEnergyDifference, failCount, forceConst))
                            fout.write("Structure FAILED\n")
                            os.system("cp structure_%s.g96 structure_%s.g96"
% ( structure, structure+1))
                            counterQM+=1
```

```
                                failCount+=1
fout.close()
```