uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Sébastien Osborne
AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Chemistry)
GRADE / DEGREE

Department of Chemistry
FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

A Fast Linear Scaling Analytical Xα Method &
an Adaptive Importance Function in Monte Carlo Simulations

TITRE DE LA THÈSE / TITLE OF THESIS

Alaint St-Amant
DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Paul Mayer

Tommy Woo

Raymond Poirier

James Wright

Gary W. Slater
Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

A Fast Linear Scaling Analytical Xα Method

&

an Adaptive Importance Function in Monte Carlo Simulations

Sébastien Osborne

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of Doctorate in Chemistry

The Ottawa-Carleton Chemistry Institute
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-51820-5
Our file Notre référence
ISBN: 978-0-494-51820-5

Table of contents

List of figures and tables

Acknowledgments

First and foremost, I would like to thank Dr. Alain St-Amant. We have worked together for more than eight years now. During that period of time you have been more than my research supervisor and I consider you my friend. Other members of our research group, particularly Delphine Courmier and Daniel Matusek, have always been good company and helpful when I requested their assistance. The High Performance Computing Virtual Laboratory provided computing resources that were very useful for the duration of my graduate studies. I also thank Dr. Tom Woo for allowing me to attend and participate in his group meetings. The simple fact that I discussed the progress of my research to fellow students helped me in many ways.

My parents have been supportive of my choice to come back to the University of Ottawa and obtain my Ph. D. degree and have helped me in more ways that they realize. I am grateful to my friends for putting up with my staying in school for five more years.

Last but not least, I have spent a decade of my life studying at the University of Ottawa. I have always been treated with respect and courtesy, by teachers, administrative employees and fellow students. The generous admission scholarship program of the Faculty of Graduate and Post-Doctorate Studies made it possible for me to obtain my degree. I also thank the government of the province of Ontario and the Ontario Graduate Scholarship program for financial support.

Abstract

The need for more accurate theoretical data pushes computational chemists to perform calculations at the quantum mechanical level on systems of ever increasing size and for comprehensive explorations of their potential energy surface. Fast and reliable ab initio methods must be developed in order to take advantage of the readily available and increasingly powerful computing resources.

A grid-free analytical Xα functional was implemented, focusing on speed and scaling behaviour, within our own density functional theory software, DeFT. Appropriate basis sets for the requisite fits of $\rho^{1/3}$ and $\rho^{2/3}$ are constructed so as to have a method that is both fast and reliable. Our analytical Xα functional is tested against a conventional numerical approach that uses a grid to integrate the exchange-correlation contribution to the Kohn-Sham matrix. The analytical method reproduces the electronic energy and optimized geometrical parameters obtained with a sophisticated numerical grid. More importantly, compared to the numerical approach, it is 4 to 8 times faster for the evaluation of the Xα energy and 40 to 60 times faster for the evaluation of the Xα energy gradient on systems of medium to large sizes. The formal cubic scaling factor for the analytical Xα functional is improved to 1.31, ensuring that the method will remain the method of choice for larger systems. Finally, the utility of the analytical Xα functional as a low level method within hybrid schemes is demonstrated while taking advantage of its adaptive nature by treating α as an atom-dependent parameter.

In a separate project, a novel adaptive semi-empirical importance function has been developed to better reproduce a quantum mechanical potential and improve the sampling efficiency of Monte Carlo simulations at the quantum mechanical level. The efficiency of our adaptive importance function is demonstrated in a proof of concept application focusing on the ring opening of cyclobutene. Using the adaptive importance function leads to a significant improvement in the sampling of all areas of the potential energy surface, especially near the transition state, when compared with a traditional single chain or when using a rigid importance function.

Introduction

Over the years, the field of computational chemistry has progressed into a significant component of various research areas such as drug design, catalysis and reaction mechanisms. The results of theoretical calculations can be used to predict the behaviour of various chemical systems or to assist in better understanding a wide range of experimental data. One particular branch of computational chemistry focuses on the development of tools, in the form of algorithms and computer programs, to allow scientists to include computational chemistry as part of their research. In an effort to produce theoretical data that is more pertinent to real life experiments, computational chemists need to develop efficient algorithms to study large systems using accurate levels of theory. Ab initio methods are more expensive than semi-empirical methods or classical potentials but often necessary for accuracy purposes. Additionally, the shape of the potential energy surface of large molecular systems is often such that single point calculations are not sufficient. For that reason, the potential energy surface must be adequately sampled to include contributions from the thermal energy that is available in a real situation.

The objective of the present research project is two-fold. The first part of the project involves the implementation of a fast ab initio method within our own Density Functional Theory software package. Secondly, to complement the implementation of the fast ab initio method, an efficient technique must be developed to successfully sample the potential energy surface of molecular systems.

The simpler ab initio methods found today are fast enough to be used for calculations on molecular systems containing hundreds of atoms. In addition to being the simplest and fastest exchange-correlation functional, the Xα functional has the advantage of providing a complete analytical solution and benefits from the complete removal of the numerical grid typically used in Density Functional Theory calculations. The analytical Xα functional will be implemented within our own software using a minimalist approach (minimal orbital basis set, charge fitting basis set, exchange fitting basis set) in an effort to speed up the calculations, at the expense of a loss of accuracy. The accuracy lost by this approach can

be partly recuperated by treating the α parameter of the functional as a flexible atom dependent, atom-type dependent or even basis function dependent parameter rather than a fixed constant. Unfortunately, the straightforward implementation of ab initio methods is typically hindered by a poor scaling behaviour with respect to the size of the system, prohibiting its usage for very large systems even when implemented in a very efficient manner. However, matrices and vectors found in the intermediate stages of ab initio calculations often contain a large proportion of negligible elements for large systems. Mathematical simplifications aiming to improve upon the formal cubic scaling behaviour of the $X\alpha$ functional can be made by taking advantage of the sparse vectors and matrices. Striving for linear scaling ensures that the implementation of the algorithm will remain faster than other algorithms with poorer scaling behaviours for systems of all sizes.

The sampling of the potential energy surface of molecular systems is typically performed with molecular simulations. With the computing resources available today, these simulations can be performed at the ab initio level for small to medium systems. The metropolis-Monte-Carlo algorithm generates a chain of configurations by randomly perturbing the position of individual nuclei or groups of nuclei. Each configuration of the chain is strongly correlated to the previous configuration, in effect sampling the potential energy surface in an inefficient manner since a very large number of configurations are required. It has been shown that sampling efficiency can be improved by more than an order of magnitude if the configurations of the primary ab initio chain are separated by short length secondary chains using a classical potential. Because calculations using the classical potential are extremely fast compared to the ab initio method, the additional time spent on the secondary chains is negligible. Unfortunately, readily available classical potentials are not well-suited for these applications because they generally do not perform well when chemical bonds are broken or formed. Different classical potentials are usually developed on a case by case basis.

Nothing prevents the usage of a semi-empirical potential for the secondary chains to circumvent the creation of a new classical potential each time a new system is studied. Although calculations using semi-empirical potentials are much slower than calculations

using classical potentials, the difference in speed between semi-empirical and ab initio methods is still large enough to proceed with semi-empirical secondary chains. The improvement in the sampling when using a secondary chain, regardless of the type of secondary potential used, is directly related to the similarity between the primary and secondary potentials. Difficulties arise in areas of the potential energy surface where significant discrepancies exist between the semi-empirical and ab initio potentials. Such areas are usually found in regions away from energy minima since semi-empirical methods are not parameterized to perform well in those regions. Instead of using a rigid secondary semi-empirical potential, we developed a novel scheme in which the secondary semi-empirical potential can adapt to the primary ab initio potential on the fly. A selected set of parameters from a semi-empirical potential are treated as variables to reproduce the energy gradient of the ab initio potential of the new accepted configurations of the primary chain. When using semi-empirical secondary chains, the adaptive potential can better mimic the ab initio potential in all regions of the potential energy surface, thus further improving the sampling efficiency when compared to the rigid potential.

The implementation of a fast, linear scaling analytical Xα functional and our novel adaptive secondary potential for Monte Carlo simulations represent important steps toward the goal of producing theoretical data that is more pertinent to real chemistry.

Chapter 1 – Density Functional Theory

The ingenuity behind Density Functional Theory (DFT) allows one to work with the electronic density $\rho(r)$, a simple three-dimensional function, instead of the far more complex n-electron wave function of Hartree-Fock (HF) and correlated post-HF methods. DFT has been popular in solid state physics for decades and gained its place in the field of computational chemistry in the 1990s. DFT calculations partially include electronic exchange and correlation effects in a fraction of the time required for correlated post-HF methods such as Moller-Plesset perturbation theory, Configuration Interaction or Coupled Cluster. Numerous DFT methods of varying complexity and accuracy have been developed and extensive research focuses on improving upon the existing exchange-correlation functionals. Today, DFT is generally the method of choice for calculations involving medium to large molecular systems, metal-containing compounds often used in catalysis and periodic systems such as condensed phases. This brief introductory chapter aims to demonstrate how DFT evolved from a simplistic model to the powerful technique that is now routinely being used for electronic calculations in chemistry.

1.1. The Thomas-Fermi model and Hohenberg-Kohn theorems

In wave function methods such as Hartree-Fock and post Hartree-Fock methods, the time-independent Schrödinger equation is solved in order to obtain the wave function. Once the wave function is known, any observable property of the given system can be evaluated. The wave function, symbolized by the greek letter Ψ, is a $4n$-dimensional function of the position and spin of n individual electrons in motion under the influence of an external potential created by N nuclei.

$$H\Psi = E\Psi \qquad (1.1)$$

The Hamiltonian used in the Schrödinger equation accounts for the kinetic energy of the electrons, the potential energy of electron-nucleus interactions and the potential energy of electron-electron interactions. The following equation represents the Hamiltonian when atomic units are used.

$$H = \sum_{i=1}^{n} -\frac{1}{2}\nabla_i^2 + \sum_{i=1}^{n}\sum_{j=1}^{N} -\frac{Z_j}{|r_i - R_j|} + \sum_{i=1}^{n}\sum_{j=i+1}^{n} \frac{1}{|r_i - r_j|} \qquad (1.2)$$

The Born-Oppenheimer approximation states that the movement of the nuclei can be neglected when compared to the movement of the electrons because nuclei are much heavier than electrons. There is no term for the kinetic energy of the nuclei in the Hamiltonian since they are considered to be stationary. Relativistic effects are usually neglected as well, which is a valid approximation for typical organic molecules made of first and second row elements. The Schrödinger equation has an exact analytical solution for single electron systems only (hydrogen atom, helium cation, etc...). For many-electronic systems, there is no analytical solution to the Schrödinger equation. One must then resort to approximate solutions using a variety of mathematical techniques. The wave function quickly becomes very cumbersome as the number of electrons increases, making it difficult and expensive to obtain an approximate solution to the Schrödinger equation. HF theory and several post-HF correlated methods have been developed in order to obtain an approximation of the wave function[1].

In contrast with DFT, the ground state electronic energy and all other ground-state properties are evaluated as functionals of the ground state electronic density. In mathematical terms, a functional is simply a function which uses another function as its independent variable. A functional is usually written with its independent function in square brackets, as opposed to parentheses. For electronic calculations, the electronic energy is a function of the electronic density, which is itself a function of three-dimensional space coordinates.

$$E = E[\rho(r)] \qquad (1.3)$$

DFT is attractive because it always works with a three-dimensional function, regardless of the number of electrons, as opposed to describing the behaviour of every electron with the *4n*-dimensional wave function. Early attempts to use DFT as a method for electronic calculations go back to 1927 with the Thomas-Fermi model. Just as the electronic Hamiltonian can be separated into its components, the total electronic energy functional can be divided into a kinetic energy functional, electron-nucleus interaction functional and electron-electron interaction functional. The classical Coulomb electrostatic energy resulting from the attraction between n negatively charged electrons and N positively charged nuclei can explicitly be expressed as a functional of the electronic density. A similar explicit functional exists for the energy resulting from the repulsion of n negatively charged electrons.

$$V_{eN}[\rho(r)] = \sum_{A=1}^{N} \int -\frac{Z_A \rho(r)}{|r - R_A|} dr \tag{1.4}$$

$$V_{ee}[\rho(r)] = \frac{1}{2} \int\int \frac{\rho(r)\,\rho(r')}{|r - r'|} dr\, dr' \tag{1.5}$$

However, there is no such explicit functional for the kinetic energy functional. By definition, potential energy depends on positions alone while kinetic energy depends on momentum. Intuitively, it is more difficult to think of kinetic energy as a functional of the electronic density. Thomas[2] and Fermi[3] independently developed an approximation of the kinetic energy functional based on the uniform electron gas model and applied it to atomic electronic calculations. According to Thomas: "Electrons are distributed uniformly in the six-dimensional phase space for the motion of an electron at the rate of two for each h^3 of volume and the effective potential is itself determined by the nuclear charge and this distribution of electrons"[2]. In the uniform electron gas model, a positive charge is spread uniformly throughout space to neutralize the negative charge of the electrons. The reality of an atom or molecule is obviously much different, having highly localized positive

charges at the nuclei. Nonetheless, Thomas and Fermi derived an approximate kinetic energy functional from this crude model.

$$T_{TF}[\rho(r)] = \frac{3}{10}\left(3\pi^2\right)^{2/3}\int \rho^{5/3}(r)dr = C_F\int \rho^{5/3}(r) \tag{1.6}$$

The complete energy functional of the Thomas-Fermi model combines this approximate kinetic energy functional with the classical coulomb electrostatic attraction and repulsion functionals.

$$E_{TF}[\rho(r)] = T_{TF}[\rho(r)] + V_{eN}[\rho(r)] + V_{ee}[\rho(r)] \tag{1.7}$$

Although there was no rigorous proof at the time, it was assumed that this electronic energy could be minimized according to the variational principle. The technique of unknown Lagrange multipliers can be used to ensure that the electronic density integrates to the total number of electrons.

$$\int \rho(r)\, dr = n \tag{1.8}$$

Equation 1.9 must be satisfied in order to obey the variational principle. The unknown Lagrange multiplier physically represents the chemical potential. Equation 1.10, known in mathematics as the Euler-Lagrange equation, can be solved with the constraint resulting from the unknown Lagrange multiplier.

$$\partial\left\{E_{TF}[\rho(r)] - \mu_{TF}\left(\int \rho(r)dr - n\right)\right\} = 0 \tag{1.9}$$

$$\mu_{TF} = \frac{\partial E_{TF}[\rho(r)]}{\partial\rho(r)} = \frac{5}{3}C_F\rho^{2/3}(r) + \sum_{A=1}^{N} -\frac{Z_A}{|r-R_A|} + \int \frac{\rho(r')}{|r-r'|}dr' \tag{1.10}$$

Once the Euler-Lagrange equation is solved, the resulting electronic density is inserted into equation 1.7 in order to calculate the total electronic energy. Unfortunately, the Thomas-Fermi model is not very accurate for atomic calculations and even less accurate for molecular systems. It lacked a solid theoretical foundation at the time and was mainly viewed as a simplistic model without much potential for electronic calculations. The flaws of the Thomas-Fermi model originate from two sources: the approximate kinetic energy functional and the neglect of electronic exchange as well as correlation effects, quantum effects that were not considered in this model. In 1930, Dirac addressed the latter flaw in part by developing an exchange functional based on the same uniform electron gas approximation[4]. Dirac's exchange is the basis of the Local Density Approximation, which will be discussed in more detail in the next section. The Thomas-Fermi-Dirac model simply adds Dirac's exchange functional to the original functionals of the Thomas-Fermi model.

Still, Teller showed that the Thomas-Fermi-Dirac model fails miserably when applied to molecules. He demonstrated that diatomic molecules will always have higher energy than that of the separated atoms when using the Thomas-Fermi-Dirac model[5]. In other words, he showed that chemical bonds would not happen with the Thomas-Fermi-Dirac model. The fact that molecules cannot exist even when exchange effects are included identifies the approximate kinetic energy as the main culprit. In 1935, Weizsacker tried to improve the kinetic energy functional by adding a gradient correction to the original kinetic energy functional of the Thomas-Fermi model[6].

$$T_W[\rho(r)] = \frac{1}{8}\int \frac{|\nabla\rho(r)|^2}{\rho(r)}\,dr \tag{1.11}$$

$$T_{TFW}[\rho(r)] = T_{TF}[\rho(r)] + \lambda T_W[\rho(r)] \tag{1.12}$$

The Thomas-Fermi-Dirac-Weizsacker model combines the kinetic energy functional of equation 1.12 with the classical Coulomb functionals and Dirac's exchange functional. The λ parameter was originally assigned a value of 1 by Weizsacker himself, but it was later

discovered that its value is 1/9 for a correct gradient-expansion. The Thomas-Fermi-Dirac-Weizsacker model fairs much better than its predecessor, improving the behaviour of the electronic density and allowing the existence of stable chemical bonds.

In 1964, Hohenberg and Kohn published a pair of theorems[7] that made DFT a legitimate method for electronic calculations, as opposed to the approximate model it was conceived to be. Just as the number of electrons and the external potential determine all the properties of the ground state in HF theory, the first Hohenberg-Kohn theorem states that: "The external potential v(r) is determined, within a trivial additive constant, by the electronic density $\rho(r)$". There is one-to-one mapping between the ground state wave function and the ground state electronic density. Because of this one-to-one mapping, the electronic density alone can determine the wave function and all the other electronic properties of the ground state, including the electronic energy. The first theorem validates the electronic density as a proper variable to calculate the electronic energy. The exact ground state energy can be calculated with the exact energy functional, which exists but sill remains unknown.

$$E_0 = E[\rho(r)] \tag{1.13}$$

The total energy functional can always be broken down into a kinetic energy functional, a nucleus-electron attraction functional and electron-electron repulsion functional. The kinetic energy and electron-electron repulsion are often lumped together into what is called a universal functional, F_{HK}, because they do not depend on the position of the nuclei. The universal functional, being independent of the external potential, will have the same form for any system.

$$E[\rho(r)] = T[\rho(r)] + V_{ee}[\rho(r)] + V_{ne}[\rho(r)] = F_{HK}[\rho(r)] + \int \rho(r)v(r)dr \tag{1.14}$$

The second Hohenberg-Kohn theorem is the equivalent of the variational principle for wave function methods: "For a trial density $\rho_{trial}(r)$, such that $\rho_{trial}(r) \geq 0$ and $\int \rho_{trial}(r)dr = n$,

$$E_0 \leq E[\rho_{trial}(r)] \tag{1.15}$$

where $E[\rho(r)]$ is the energy functional of equation 1.14". For any reasonable but inexact electronic density, the resulting energy will always be higher than the true ground state energy. Obviously, for the density to be reasonable, it cannot be negative at any point in space and must yield the total number of electrons when integrated over all space. The stationary point that is equation 1.16 results from the variational principle under the constraint that the integral of the density equals the number of electrons. The Euler-Lagrange equation is obtained from the functional derivative of the energy functional.

$$\partial\left\{E[\rho(r)]-\mu\left(\int \rho(r)dr - n\right)\right\}=0 \tag{1.16}$$

$$\mu=\frac{\partial E[\rho(r)]}{\partial\rho(r)}=v(r)+\frac{\partial F_{HK}[\rho(r)]}{\partial\rho(r)} \tag{1.17}$$

If the exact form of the universal functional F_{HK} was known, then DFT would be an exact theory. Nonetheless, any approximate, explicit form of F_{HK} will suffice to apply the method to any system. Although the Hohenberg-Kohn theorems do not give any information on the manner in which to proceed in obtaining this exact functional, they provide a sound theoretical foundation for the usage of DFT for practical electronic calculations. Knowing that the theory can be exact if given the right functional is a great incentive to push forward and improve upon the early models. In that sense, the Hohenberg-Kohn theorems demonstrate that the Thomas-Fermi model is an approximation of an exact theory.

1.2. The Kohn-Sham method

The Thomas-Fermi model and other models derived from it are very simple because they involve the electronic density alone by using an explicit, approximate functional for the

kinetic energy. However, as mentioned in the previous section, they do not produce acceptable results for practical chemical applications. Approximations made to the kinetic energy functional negatively impact the end results to the extent that molecules cannot be stable. In 1965, Kohn and Sham came up with an indirect approach to the kinetic energy functional problem[8]. Taking a page from HF theory, they introduced orbitals into DFT to help deal with the kinetic energy problem. For a system containing *n* electrons, the electronic density and kinetic energy can easily be expressed in terms of *n* occupied spin orbitals.

$$\rho(r) = \sum_{i=1}^{n} |\psi_i(r)|^2 \tag{1.18}$$

$$T_s[\rho(r)] = \sum_{i=1}^{n} \left\langle \psi_i \left| -\frac{1}{2}\nabla^2 \right| \psi_i \right\rangle \tag{1.19}$$

Unfortunately, equations 1.18 and 1.19 are valid only for the exact determinant wave function of *n* non-interacting electrons. Kohn and Sham invoked a reference system of non-interacting electrons under the effect of an external potential. Such a system is described by the Hamiltonian of equation 1.20, resulting in the wave function of equation 1.21.

$$H_s = \sum_{i=1}^{n} -\frac{1}{2}\nabla_i^2 + \sum_{i=1}^{n} v_s \tag{1.20}$$

$$\Psi_s = \frac{1}{\sqrt{n!}} \det[\psi_1 \psi_2 ... \psi_n] \tag{1.21}$$

For that reference system, the wave function can be expressed exactly as a Slater determinant while the electronic density is expressed exactly by equation 1.18 and the kinetic energy is expressed exactly by equation 1.19. Kohn and Sham solved the kinetic energy problem by providing an exact solution to a reference system of non-interacting

electrons rather than an approximate solution to the real system. The universal functional is partitioned into the kinetic energy functional of the reference system, the classical electron-electron repulsion functional of the real system and the newly defined exchange-correlation (XC) functional of the real system.

$$F[\rho(r)] = T_s[\rho(r)] + J[\rho(r)] + E_{XC}[\rho(r)] \tag{1.22}$$

As its name implies, the XC energy functional includes the non-classical electron-electron interactions of exchange and correlation. Additionally, the difference between the kinetic energy of the real system and the kinetic energy of the reference system is included. In a sense, approximating the true kinetic energy with that of the reference system is accounted for explicitly in the XC functional.

$$E_{XC}[\rho(r)] = T[\rho(r)] - T_s[\rho(r)] + V_{ee}[\rho(r)] - J[\rho(r)] \tag{1.23}$$

The complete energy functional of Kohn-Sham (KS) DFT is the combination of equation 1.22 and the external potential caused by the nuclei.

$$E[\rho(r)] = T_s[\rho(r)] + J[\rho(r)] + E_{XC}[\rho(r)] + \int \rho(r)v(r)dr \tag{1.24}$$

Once again, taking the functional derivative of equation 1.24, the Euler equation can be written for KS-DFT.

$$\mu = v_{eff}(r) + \frac{\partial T_s[\rho(r)]}{\partial \rho(r)} \tag{1.25}$$

$$v_{eff}(r) = v(r) + \frac{\partial J[\rho(r)]}{\partial \rho(r)} + \frac{\partial E_{XC}[\rho(r)]}{\partial \rho(r)} = v(r) + \int \frac{\rho(r')}{|r - r'|} dr' + v_{XC}(r) \tag{1.26}$$

If the orbitals are constrained to be orthonormal, which is necessary for the kinetic energy of the reference system, then the entire problem of minimizing the energy functional is solved via a set of one-electron equations, namely the canonical KS spin orbital equations.

$$\left[-\frac{1}{2}\nabla^2 + v_{eff}\right]\psi_i = \varepsilon_i\psi_i \tag{1.27}$$

The KS orbital equations are very similar to the HF equations. The *n* one-electron operators of equation 1.27 depend on the orbitals from the effective potential. An iterative process is necessary to solve the problem since the solution (the set of orbitals) is required to build the equations. An initial set of orbitals is used to obtain a set of operators, which in turn produce a new set of orbitals. This cycle is repeated until consistency is observed between two successive sets of orbitals. In most software packages, the KS orbital equations are solved by expanding the KS molecular orbitals into linear combinations of atomic orbitals (LCAO), often called basis functions. A given set of basis functions is referred to as an orbital basis set.

$$\psi_i(r) = \sum_{\mu=1}^{NAO} c_{\mu i}\varphi_\mu(r) \tag{1.28}$$

In equation 1.28, each KS molecular orbital is expressed as the linear combination of *NAO* basis functions with the coefficients c. The basis functions are usually atom-centered Gaussian functions or atom-centered Slater orbitals. By using the LCAO approach, the problem can be recast with matrices.

$$KC = SC\varepsilon \tag{1.29}$$

In equation 1.29, K is the Kohn-Sham matrix, C is the coefficient matrix of the LCAO expansions, S is the overlap matrix of the basis functions and ε is a vector containing the eigenvalues (the KS orbital energies) of equation 1.27. The K matrix is very similar to the

Fock matrix in HF theory. It is obtained by applying the one-electron KS operator on the basis functions.

$$K_{\mu\nu} = \int \varphi_\mu(r)\left\{-\frac{1}{2}\nabla^2 + v_{eff}(r)\right\}\varphi_\nu(r)\,dr \tag{1.30}$$

$$S_{\mu\nu} = \int \varphi_\mu(r)\,\varphi_\nu(r)\,dr \tag{1.31}$$

Equation 1.29 is nothing more than a set of linear equations that can be solved with conventional linear algebra techniques. The electronic density, and by association the electronic energy, become a function of the LCAO coefficients expansions which make up the molecular orbitals. With help from the variational principle, a solution to the problem is provided by the set of coefficients which minimizes the electronic energy.

In theory, KS-DFT can fully include the effect of electronic exchange and correlation. If the exact form of the XC functional was known, KS-DFT would yield the exact electronic density, and in turn the exact electronic energy. The XC functional is the only inexact portion of the method. However, the XC energy is only a small fraction of the total electronic energy, which partly explains the success of KS-DFT even when crude approximations to the XC functional are used. It is implied that modern DFT applications in chemistry use the KS approach.

1.3. The exchange-correlation functional

It is not surprising that a great deal of research focuses on the development of more accurate XC functionals since the XC functional is the only inexact component of the total energy functional of KS-DFT. Many XC functionals have been developed through the years, ranging in complexity, accuracy and efficiency. Kohn and Sham proposed the local density approximation (LDA) to complete their KS-DFT method. In the LDA, the XC

energy of an atom or molecule at a given point in space is assumed to be equal to that of a uniform electron gas with the same electronic density. This type of XC functional is termed local because the XC energy at a given point in space is a function of the electronic density at that specific point alone. For the more sophisticated functionals, which will be discussed later, the XC energy can be a function of the electronic density and the derivatives of the electronic density. It is customary to separate the XC functional into an exchange functional and a correlation functional.

$$E_{XC}^{LDA}[\rho] = E_{X}^{LDA}[\rho] + E_{C}^{LDA}[\rho] = \int \rho(r)\,\varepsilon_{X}(\rho)\,dr + \int \rho(r)\,\varepsilon_{C}(\rho)\,dr \qquad (1.32)$$

Dirac derived an expression for the $\varepsilon_X(\rho)$ term of equation 1.32 based on the uniform electron gas model in 1930[4]. Although it was originally developed to introduce the exchange energy into the Thomas-Fermi model, Kohn and Sham use the exact same formula in their LDA functional.

$$E_{X}^{LDA}[\rho] = -\frac{3}{4}\left(\frac{3}{\pi}\right)^{1/3} \int \rho^{4/3}(r)\,dr \qquad (1.33)$$

Unfortunately, no such simple, explicit functional exists for the correlation energy component to accompany equation 1.33 and complete the approximation. Early calculations simply ignored the correlation energy, which can still give acceptable results since the correlation energy is approximately an order of magnitude smaller than the exchange energy. The LDA is valid for spin-compensated cases only. The Local Spin Density Approximation (LSDA), an extension of the LDA, was developed in the early 1970s to treat polarized systems[9], such as open-shell systems. Equation 1.33 can be rewritten with the total electronic density separated into α spin and β spin to obtain the LSDA equivalent. The LSDA exchange energy is equal to the LDA exchange energy if ρ_α is equal to ρ_β, which is the case for an even number of paired electrons.

$$E_{X}^{LSDA}[\rho] = -\left(2^{1/3}\right)\frac{3}{4}\left(\frac{3}{\pi}\right)^{1/3} \left\{\int \rho_{\alpha}^{4/3}(r)\,dr + \int \rho_{\beta}^{4/3}(r)\,dr\right\} \qquad (1.34)$$

It was not until 1980 that a respectable correlation functional was developed. Ceperley and Alder determined the "exact" XC energy of numerous homogeneous gases of varying total density with Monte Carlo simulations[10]. Vosko, Wilk and Nusair then made interpolations using the data of Alder and Ceperley. They obtained an analytic form for the correlation energy functional. The VWN functional[11] is the result of the combination of the exchange functional of equation 1.33 with the correlation energy functional of Vosko, Wilk and Nusair. VWN is generally accepted to be as accurate as the uniform electron gas permits since it is interpolated from "exact" quantities. Although the LSDA has been applied to numerous systems with surprising success, its most notable flaw is the consistent overestimation of binding energies, which are especially problematic for weakly bound systems[12]. Weak hydrogen bonds are commonly found in biomolecular systems and the LSDA is notorious for its gross overestimation of the strength of hydrogen bonds, in some cases producing very poor results.

The next step taken to go beyond the LSDA and the uniform electron gas model is to include the gradient of the electronic density in the functional. The generalized gradient approximation (GGA) forms a new class of gradient-corrected functionals. GGA functionals are more expensive but more accurate than LSDA functionals. Rather than starting from scratch, most of the GGA functionals start from the LSDA and simply add a gradient-correction.

$$E_{XC}^{GGA}[\rho] = \int \rho(r)\, \varepsilon_{XC}(\rho, \nabla\rho)\, dr \qquad (1.35)$$

Without going into the exact formulation of the more notable GGA functionals, it is advisable to briefly discuss them. In 1986, Perdew introduced a gradient-corrected correlation functional, abbreviated P86[13]. In 1988, Becke added a gradient-correction to Slater's exchange and obtained the famous B88 exchange functional[14]. The most recognized correlation functional in the GGA class is probably that of Lee, Yang and Parr, developed in 1988 and commonly referred to as the LYP correlation functional[15]. Any exchange functional can be matched to any correlation functional. For example, the

popular BLYP combination matches the B88 exchange functional with the LYP correlation functional.

Common to all the previous GGA functionals is that fact that they contain empirical parameters in their formulae. The parameters are obtained by fitting the functional's output (electronic energy) to experimental data such as atomization energies or other theoretical data. These functionals are not necessarily founded on physical principles and must rely on their parameters to mimic real physical systems. Although they are successful, the ab initio aspect of DFT is partially removed and the quality of the functional will depend on the quality of the fitting procedure. The alternative route is to develop functionals that are entirely based on tangible quantum mechanics principles. For example, a functional can be developed to obey certain rules such as scaling relations, limits for high or low densities or LSDA limits for slowly varying densities. Because of their theoretical foundation, such functionals do not rely on parameters to be accurate. The Perdew-Wang 1991 (PW91)[16] and the Perdew-Burke-Ernzerhof (PBE)[17] XC functionals result from this type of functional development. Because of the fitting procedure, it is not surprising that fitted functionals are usually more accurate for calculations such as atomization energies or molecular reactions. However, the parameter-free functionals outperform the fitted functionals for solid-state applications[18].

In 1993, Becke developed his three-parameter "hybrid" XC functional, B3[19]. The B3 functional mixes the exact HF exchange energy with LDA and GGA XC. For example, the popular B3LYP combination is a mixture of HF exchange, VWN XC, B88 exchange and LYP correlation.

$$E_{XC}^{B3LYP} = a\,E_X^{VWN} + (1-a)\,E_X^{HF} + b\,\Delta E_X^{B88} + E_C^{VWN} + c\,\Delta E_C^{LYP} \qquad (1.36)$$

The HF exchange is obtained by applying the HF operator to the Kohn-Sham orbitals. The parameters a, b and c were obtained by fitting the results to experimental data and have different values depending on the chosen correlation functional. Hybrid functionals are obviously slightly more expensive to use than standard GGA functionals. However, with

the computing power that is easily accessible today, the gain in accuracy makes up for the somewhat longer calculations. The B3LYP functional constitutes the method of choice for moderately sized molecular applications.

The final class of XC functionals are referred to as meta-generalized-gradient-approximation (meta-GGA) functionals. They improve upon the GGA by adding a dependence on the Laplacian of the density or the kinetic energy density. Obtaining second derivatives of the density is quite expensive and therefore using the Laplacian is not a practical improvement to the GGA. However, the kinetic energy density is available from the evaluation of the kinetic energy. By using the kinetic energy density, meta-GGA functionals are not much more time consuming than their GGA counterpart. Few of the meta-GGA functionals have found their way into common applications. In 1998, Van Voorhis and Scuseria released their VSXC functional[20], using the kinetic energy density as an additional variable. The TPSS functional[21] of Tao, Perdew, Staroverov and Scuseria, released in 2003, builds upon the PBE functional by adding a kinetic energy density dependence.

As demonstrated with this brief discussion of functionals, one important setback of DFT is that there is no clearly defined way to improve the XC functional and ultimately converge to the exact result, which is guaranteed to exist. The development of new and improved functionals relies on physical principles, intuition and parameterization with calculated or experimental values. This is a contrast to wave function methods, where there is a clear path (increasing the order of the perturbation for perturbation theory or including a greater number of determinants in the wave function for configuration interaction) leading to exact results if one can afford the computational requirements.

1.4. The computational advantage of DFT over HF

The formalism of HF theory and KS-DFT is remarkably similar. In both cases, the problem of calculating the total electronic energy is solved via a set of one electron operators acting on occupied molecular orbitals, which are usually expressed as linear combinations of atom-centered basis functions. Why then is DFT generally more efficient than HF for large systems? The increase in efficiency is related to the scaling behaviour of the respective methods.

The time required to perform HF calculations formally scale as $O(NAO^4)$ where *NAO* is the number of basis functions in the orbital basis set. This scaling factor is due to the complicated two-electron four-center Coulomb integrals, J, and exchange integrals, K. The following equations represent the Coulomb and exchange integral between two molecular orbitals i and j.

$$J_{ij} = \int\int \frac{\psi_i(r_1)\psi_i^*(r_1)\psi_j^*(r_2)\psi_j(r_2)}{r_{12}} dr_1 dr_2 \qquad (1.37)$$

$$K_{ij} = \int\int \frac{\psi_i^*(r_1)\psi_j(r_1)\psi_i(r_2)\psi_j^*(r_2)}{r_{12}} dr_1 dr_2 \qquad (1.38)$$

Since each molecular orbital is a linear combination of atomic orbitals, an exchange integral or Coulomb integral for a HF calculation could involve four basis functions centered on four different atoms. Needless to say, a very large number of double integrals must be computed since every possible permutation of basis functions is considered. In fact, NAO^4 primitive double integrals are implicated in the evaluation of exchange and Coulomb integrals.

$$J_{ij} = \sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO}\sum_{\theta=1}^{NAO}\sum_{\sigma=1}^{NAO} c_{\mu i}c_{\nu i}c_{\theta j}c_{\sigma j}\int\int \frac{\varphi_\mu(r_1)\,\varphi_\nu(r_1)\,\varphi_\theta(r_2)\,\varphi_\sigma(r_2)}{r_{12}} dr_1\, dr_2 \qquad (1.39)$$

$$K_{ij} = \sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO}\sum_{\theta=1}^{NAO}\sum_{\sigma=1}^{NAO} c_{\mu i}c_{\nu j}c_{\theta i}c_{\sigma j}\int\int \frac{\varphi_\mu(r_1)\,\varphi_\nu(r_1)\,\varphi_\theta(r_2)\,\varphi_\sigma(r_2)}{r_{12}}\,dr_1\,dr_2 \quad (1.40)$$

Conventional DFT calculations will encounter the same type of two-electron four-center integrals when the classical Coulomb electronic repulsion is calculated. The Coulomb electronic repulsion energy involves the integral of a product of electronic densities, which are themselves expressed as a product of basis functions.

$$\int\int \frac{\rho(r)\,\rho(r')}{|r-r'|}\,dr\,dr' = \sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO}\sum_{\theta=1}^{NAO}\sum_{\sigma=1}^{NAO} P_{\mu\nu}P_{\theta\sigma}\int\int \frac{\varphi_\mu(r)\,\varphi_\nu(r)\,\varphi_\theta(r')\,\varphi_\sigma(r')}{|r-r'|}\,dr\,dr' \quad (1.41)$$

A density matrix element, $P_{\mu\nu}$, is obtained from the expansion coefficients of these two particular basis functions in the occupied molecular orbitals.

$$P_{\mu\nu} = 2\sum_{i=1}^{Nocc} c_{\mu i}c_{\nu i} \quad (1.42)$$

Once again, NAO^4 two-electron four-center primitive integrals are required. As such, conventional DFT calculations will also scale formally as $O(NAO^4)$ if no special care is taken to resolve this issue. However, in many DFT applications, the computational load is greatly reduced be eliminating the costly two-electron four-center integrals.

A possible option to eliminate these integrals is to approximate the true electronic density by a linear combination of functions rather than a product of molecular orbitals.

$$\rho(r) \approx \overline{\rho(r)} = \sum_{\omega=1}^{NAUX} c_\omega \chi_\omega(r) \quad (1.43)$$

The true electronic density is fitted onto NAUX auxiliary functions. These auxiliary functions are usually of the same type as the functions that make up the orbital basis set (Gaussian functions or Slater functions). This new set of functions is often referred to as the auxiliary charge fitting basis set. The auxiliary charge fitting basis set can then be used to approximate the electronic repulsion energy.

$$\iint \frac{\rho(r)\,\rho(r')}{|r-r'|}\,dr\,dr' \approx 2\iint \frac{\rho(r)\,\overline{\rho(r')}}{|r-r'|}\,dr\,dr' - \iint \frac{\overline{\rho(r)}\,\overline{\rho(r')}}{|r-r'|}\,dr\,dr' \qquad (1.44)$$

$$\iint \frac{\rho(r)\,\overline{\rho(r')}}{|r-r'|}\,dr\,dr' = \sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO}\sum_{\omega=1}^{NAUX} P_{\mu\nu}c_{\omega}\iint \frac{\varphi_{\mu}(r)\,\varphi_{\nu}(r)\,\chi_{\omega}(r')}{|r-r'|}\,dr\,dr' \qquad (1.45)$$

$$\iint \frac{\overline{\rho(r)}\,\overline{\rho(r')}}{|r-r'|}\,dr\,dr' = \sum_{\omega=1}^{NAUX}\sum_{\kappa=1}^{NAUX} c_{\omega}c_{\kappa}\iint \frac{\chi_{\omega}(r)\,\chi_{\kappa}(r')}{|r-r'|}\,dr\,dr' \qquad (1.46)$$

It is necessary to use equation 1.44 to eliminate first order errors associated with the incompleteness of the auxiliary charge fitting basis set. The four-center integrals are replaced with three-center integrals and two-center integrals. There are fewer primitive integrals required ($NAO^4 > NAO^2NAUX + NAUX^2$) and they are much simpler to evaluate. The computational cost is greatly reduced, even though an additional step is necessary in order to obtain the expansion coefficients of the auxiliary charge fitting basis set. This holds especially true for larger systems because with the auxiliary charge fitting basis set, the evaluation of the Coulomb repulsion energy now scales as $O(NAO^2NAUX)$ instead of $O(NAO^4)$. The expansion coefficients of the fitted electronic density in equation 1.43 are determined by a least-squares fitting procedure[22]. Our own DFT code, named DeFT[23], proceeds in this fashion.

A second alternative to the evaluation of four-center integrals is to solve Poisson's equation to obtain the Coulomb potential, V_C.

$$\nabla^2 V_C(r) = -4\pi\rho(r) \tag{1.47}$$

The expression for the Coulomb repulsion energy can be rewritten in a form that includes the Coulomb potential.

$$\iint \frac{\rho(r)\,\rho(r')}{|r-r'|}\,dr\,dr' = \int \rho(r)\left\{\int \frac{\rho(r')}{|r-r'|}\,dr'\right\}dr = \int \rho(r)\,V_C(r)\,dr \tag{1.48}$$

Once the Coulomb potential is known, it is much easier to evaluate the Coulomb repulsion energy from equation 1.48 rather than using the four-center integrals. The Coulomb potential can be obtained by solving Poisson's equation numerically using a grid or analytically using fitting functions.

For example, in the DMol[24] software package, Poisson's equation is solved numerically and the Coulomb potential is expressed on a grid made of NGRID points positioned at R_I with quadrature weight W_I. The primitive integrals required to evaluate the Coulomb repulsion energy are then approximated numerically by summing contributions from the individual grid points.

$$\int \varphi_\mu(r)\,\varphi_\nu(r)\,V_C(r)\,dr \approx \sum_{I=1}^{NGRID} \varphi_\mu(R_I)\,\varphi_\nu(R_I)\,V_C(R_I)\,W_I \tag{1.49}$$

Alternatively, in the ADF[25] software package, Poisson's equation is solved by approximating the Coulomb potential with a set of NFP fitting functions. In this case, Slater functions are used to approximate the Coulomb potential since ADF uses Slater functions in its orbital basis sets.

$$V_C(r) \approx \sum_{i=1}^{NFP} c_i f_i(r) \tag{1.50}$$

The primitive integrals required to calculate the Coulomb repulsion energy can be evaluated analytically with the approximate Coulomb potential.

$$\int \varphi_\mu(r)\varphi_\nu(r)V_C(r)\,dr \approx \sum_{i=1}^{NFP} c_i \int \varphi_\mu(r)\varphi_\nu(r)f_i(r)dr \qquad (1.51)$$

Not all DFT software packages use approximations to evaluate the Coulomb repulsion energy. For example, although charge fitting is available in the popular Gaussian[26] electronic structure program, the default option is to directly evaluate the Coulomb repulsion energy from equation 1.41, with the explicit four-center integrals. The fact that a large portion of Gaussian users perform B3LYP calculations may explain the reasoning behind this choice of not resorting to a fitted density approximation by default. The popular hybrid B3LYP exchange-correlation functional uses the HF energy as one of its parameters. Since the four-center primitive integrals are computed in the exchange energy of the HF part of the calculation, the Coulomb repulsion energy can easily be calculated with the same four-center primitive integrals.

In theory, the simplification of the Coulomb repulsion energy techniques could be applied to HF theory as well as DFT. However, in the case of HF theory, even if the Coulomb integrals are simplified, the same set of two-electron four-center primitive integrals is still explicitly required for the evaluation of the exact exchange energy[27]. It would be pointless to apply the approximations to the Coulomb energy in HF theory because the method would still scale as $O(NAO^4)$ and the primitive integrals are already evaluated for the exchange energy. On the other hand, in DFT (given that the XC functional is not of the hybrid type which sometimes includes the HF energy), there are no explicit exchange integrals since the exchange energy is approximated by the XC functional.

Chapter 2 – A Fast Linear Scaling Analytical Xα Method

Because much contemporary research in chemistry focuses on large systems (proteins, condensed phases, biomolecular systems, etc.), computational chemists need to develop adequate methods to study such systems. With the ever increasing computing resources that are readily available today for the common scientist, larger and larger systems are studied at the ab initio level of theory. In addition to pure ab initio calculations, ab initio methods of varying degrees of complexity are commonly mixed with other ab initio methods, semi-empirical methods or force fields within hybrid QM/QM and QM/MM frameworks. However, problems can arise when the potential energy surface described by the two schemes within the coupled approach differ substantially. A very fast quantum method with the ability to easily adapt itself to higher levels of theory would certainly be ideal to serve as a low level method for such hybrid applications.

For reasons discussed in chapter 1, DFT is generally the method of choice to study large systems at the ab initio level of theory. The Xα functional is the simplest of all the exchange-correlation functionals that are available today. Its simplicity translates into obvious deficiencies over the superior, more complex exchange-correlation functionals. However, advantages also stem from its simplicity. Furthermore, it can easily be adapted to provide a purely analytical evaluation of the exchange energy rather than the numerical grid-based solution required by other functionals which incorporate correlation or a more elaborate treatment of exchange. When solving the problem analytically, it is trivial to assign different values of α for each atom type or basis function in the orbital basis set. Treating α as a flexible parameter rather than a fixed constant improves the accuracy of the functional without any additional cost.

This chapter goes through the complete formulation of the analytical Xα functional while providing insights into its performance in terms of accuracy, efficiency and scaling behaviour. Its performance and comparisons with the numerical Xα functional are studied using benchmark calculations on common organic molecules and extended peptides made of glycine units. It is our intent to develop an implementation of the analytical Xα

functional that will perform much faster than its numerical counterpart with similar accuracy for systems of all sizes. The loss of accuracy that may result from the necessary approximations made to improve the efficiency and scaling behaviour can be recuperated by treating α as an adjustable parameter. This is very similar in philosophy to semi-empirical QM methods.

2.1. The analytical Xα functional

In 1951, prior to the work of Kohn and Sham, Slater introduced his simplification of the HF method[28], commonly referred to as the Hartree-Fock-Slater method. Slater's idea was to replace the non-local exchange operator in HF theory with a local operator. Instead of accounting for explicit interactions between electrons, each electron feels the effect of an average field. Slater's method is completed by approximating the average field with that of a uniform electron gas of the same electronic density. Slater used the average Dirac exchange energy.

$$E_X^{Slater} = -\frac{9}{8}\left(\frac{3}{\pi}\right)^{1/3}\int \rho^{4/3}(r)\,dr \qquad (2.1)$$

A few years later, Gáspár used the variational principle and obtained 2/3 of Slater's expression[29]. Kohn and Sham also realized that 2/3 of Slater's exchange and a neglect of correlation is equivalent to their LDA[8]. The two different constants (2/3 for Gáspár, Kohn, Sham and 1 for Slater) arise from applying the uniform electron gas approximation differently. Slater used the uniform electron gas to approximate the exchange potential while Gáspár, Kohn and Sham used it to approximate the exchange energy. This ambiguity prompted the introduction of the α parameter and the formal definition of the Xα functional.

$$E^{X\alpha}[\rho(r)] = C_\alpha \int \rho^{4/3}(r)\,dr \qquad (2.2)$$

$$C_\alpha = -\frac{9}{8}\alpha\left(\frac{3}{\pi}\right)^{1/3} \tag{2.3}$$

The Xα potential, which will enter the one-electron Kohn-Sham operators, is simply the functional derivative of equation 2.2.

$$v_{X\alpha} = \frac{4}{3}C_\alpha \rho^{1/3}(r) \tag{2.4}$$

The Xα functional approximates the exchange energy as a constant multiplying the electronic density to the power of 4/3, while completely ignoring the correlation energy. A value of α=2/3 gives an exchange energy equivalent to that of the LSDA, while a value of α=1 leads to the HF-Slater exchange energy. When the Xα functional is used on its own (without a correlation functional) for atoms and molecules, an intermediary value of $\alpha \approx 0.75$ seems to perform better than either of the limiting cases. Schwarz performed a series of Xα calculations on a single neutral atom for the first 41 elements of the periodic table[30]. His objective was to optimize the value of α for each element to reproduce the HF energy since the exact exchange energy is intrinsic to HF theory. In every case except for the hydrogen atom, which had an optimal value of α=0.978, the optimal value for α lied between 0.7 and 0.8.

In the vast majority of DFT codes, the LCAO approximation is used to construct molecular orbitals from atomic orbitals and the electronic density can be expressed with the basis functions. The electronic density is equal to the sum of density matrix elements multiplying atomic basis functions.

$$\rho(r) = \sum_{\mu=1}^{NAO}\sum_{\upsilon=1}^{NAO} P_{\mu\upsilon}\varphi_\mu(r)\,\varphi_\upsilon(r) \tag{2.5}$$

The density matrix element of two given basis functions, $P_{\mu\upsilon}$, is obtained by adding the product of the expansion coefficients of the two given basis functions for all occupied molecular orbitals, as represented in equation 1.42 .

When the electronic density is expanded into basis functions, which are often Gaussian functions, the integral of equation 2.2 cannot be solved as a sum of analytical integrals. The integral is instead evaluated numerically using a three-dimensional grid of points. Each atom is surrounded by sets of angular grid points at successive radii. For example, a 110-point angular mesh used at 25 different radii would make up a modest grid of 2750 points. The molecular grid is made of overlapping atom-centered grids. The fuzzy-weighting scheme was designed to smooth the overlapped areas[31] where the grids of two atoms intersect.

Very fine grids of thousands of points per atom are required to numerically integrate with acceptable accuracy, leading to a computationally expensive approach. In addition to the high cost of this numerical endeavour, there are many other disadvantages associated with the grid-based integration technique. There will always be a grid error associated with the incomplete discrete nature of the grid. Unfortunately, this grid error is not constant over the potential energy surface of a molecular system because the coordinates of the grid points depend on the nuclear coordinates, creating grid noise[32]. Grid noise makes it very difficult to perform analyses that require the exact energy gradient. The exact energy gradient becomes very difficult to evaluate and often does not vanish properly at minima or saddle points. Because of the grid, calculations are no longer rotationally invariant. Nonetheless, the grid-based numerical integration technique is a necessary evil in DFT calculations. The grids used today are very sophisticated and produce acceptable numerical errors for most applications.

In the majority of cases, the integrand is directly evaluated on the grid points. The integral is numerically approximated by adding the individual result of each grid point. The Simpson and trapezoid integration techniques are two-dimensional analogues of the grid-based numerical integration. It is a very simple, purely numerical, approach. However, as

previously mentioned, the exact analytical energy gradient is quite difficult to obtain with this approach, because of small forces resulting from the grid points.

$$\int f(r)dr \approx \sum_{i=1}^{NGRID} f(r_i)W_i \tag{2.6}$$

In equation 2.6, the integral of the function *f* is approximated with the help of *NGRID* points, each having its own predetermined weight. Several weighting schemes and quadratures have been developed to create accurate grids for molecular systems[33].

As an alternative to the purely numerical integration, the integrand can be evaluated on the grid points and the results fit to a function which can easily be integrated analytically. In computational chemistry, linear combinations of Gaussian functions are routinely used for fitting. A simple least-squares fit allows the numerical data to be projected onto the linear combination of Gaussian functions in order to determine the set of expansion coefficients.

$$\sum_{i=1}^{NGRID}\left(f(r_i)W_i - \sum_{j=1}^{NFUNC} c_j G_j(r_i)\right)^2 = \min \tag{2.7}$$

$$\int f(r)dr \approx \sum_{j=1}^{NFUNC} c_j \int G_j(r)dr \tag{2.8}$$

The integral of the function *f* is approximated by a sum of *NFUNC* analytical integrals of Gaussian functions (G_j) whose expansion coefficients have been determined from the fitting procedure. It is much easier to obtain analytical expressions for the energy gradient when using the grid-based fitting approach, as opposed to the pure numerical approach.

In 1986, Dunlap proposed a formalism that would allow an exchange-only local functional with $\rho^{1/3}$ potential, such as the Xα functional, to be solved purely analytically[34]. Instead of fitting numerical data onto analytical fitting functions, Dunlap devised a way to obtain

the expansion coefficients by solving a set of non-linear equations, thus completely removing the numerical grid. Two distinct set of fitting functions are required, a first to fit $\rho^{1/3}$ and another to fit $\rho^{2/3}$. The fits would be exact for an infinitely large fitting basis set. In practice, a finite linear combination of Gaussian functions, regardless of how many functions are included, is incomplete. *NX* and *NY* Gaussian functions could be used to approximate $\rho^{1/3}$ and $\rho^{2/3}$ respectively. The set of functions used to fit $\rho^{1/3}$ and $\rho^{2/3}$ will be referred to simply as the fitting basis sets, as opposed to the orbital basis set or the density fitting basis set.

$$\rho^{1/3}(r) = X(r) + \Delta X = \sum_{i=1}^{NX} a_i x_i(r) + \Delta X \tag{2.9}$$

$$\rho^{2/3}(r) = Y(r) + \Delta Y = \sum_{i=1}^{NY} b_i y_i(r) + \Delta Y \tag{2.10}$$

ΔX and ΔY represent the error due to the incompleteness of the fitting basis sets. An approximation to $\rho^{4/3}$ can be written from X and Y without first order errors associated with the incompleteness of the fitting basis sets. A first order error consists of having either ΔX or ΔY appear in the final expression for the energy while higher order errors contain products of ΔX or ΔY only. Although the removal of first order errors is not strictly required for a proper variational fit, it is a necessity when striving for acceptable results with fitting procedures in quantum chemistry[35]. All the terms containing second and third order errors due to the incompleteness of the fitting basis sets are ignored when calculating the energy.

$$\rho^{4/3} = \frac{4}{3}\rho(X+\Delta X) - \frac{2}{3}(X+\Delta X)^2(Y+\Delta Y) + \frac{1}{3}(Y+\Delta Y)^2$$

$$\rho^{4/3} = \left\{ \begin{array}{l} \frac{4}{3}\rho X + \frac{4}{3}\rho(\Delta X) \\ -\frac{2}{3}X^2Y - \frac{2}{3}X^2(\Delta Y) - \frac{4}{3}X(\Delta X)Y - \frac{4}{3}X(\Delta X)(\Delta Y) - \frac{2}{3}(\Delta X)^2Y - \frac{2}{3}(\Delta X)^2(\Delta Y) \\ +\frac{1}{3}Y^2 + \frac{2}{3}Y(\Delta Y) + \frac{1}{3}(\Delta Y)^2 \end{array} \right\} \quad (2.11)$$

$$\rho^{4/3} = \frac{4}{3}\rho X - \frac{2}{3}X^2Y + \frac{1}{3}Y^2 + \text{second order and higher terms}$$

In equation 2.11, the sum of the three remaining terms equals one (4/3 – 2/3 + 1/3) and the sum of the exponents in each term equals 4/3. The Xα exchange energy and potential can be expressed with the fits by inserting equation 2.11 into equation 2.2.

$$E^{X\alpha}[\rho(r)] = C_\alpha \left\{ \frac{4}{3}\int \rho(r)X(r)dr - \frac{2}{3}\int X(r)X(r)Y(r)dr + \frac{1}{3}\int Y(r)Y(r)dr \right\} \quad (2.12)$$

$$v_{X\alpha} = \frac{4}{3}C_\alpha X(r) \quad (2.13)$$

The integrals implicated in the evaluation of the Xα energy are simply two-center or three-center overlap integrals. These overlap integrals are easily evaluated when the fits are made of standard *s*, *p* or *d* Gaussian atomic orbitals.

$$E^{X\alpha}[\rho(r)] = C_\alpha \left\{ \begin{array}{l} +\frac{4}{3}\sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO}\sum_{i=1}^{NX} P_{\mu\nu}a_i\langle \varphi_\mu \varphi_\nu x_i \rangle \\ -\frac{2}{3}\sum_{i=1}^{NX}\sum_{j=1}^{NX}\sum_{k=1}^{NY} a_i a_j b_k \langle x_i x_j y_k \rangle \\ +\frac{1}{3}\sum_{i=1}^{NY}\sum_{j=1}^{NY} b_i b_j \langle y_i y_j \rangle \end{array} \right\} \quad (2.14)$$

The expansion coefficients of X and Y are the unknown that must be solved for. A variational minimization of equation 2.12 with respect to the expansion coefficients a_k of X and b_k of Y yields a pair of non-linear equations.

$$\begin{aligned}&\sum_{\mu=1}^{NAO}\sum_{\upsilon=1}^{NAO}P_{\mu\upsilon}\langle\varphi_\mu\varphi_\upsilon x_k\rangle-\sum_{i=1}^{NX}\sum_{j=1}^{NY}a_i b_j\langle x_k x_i y_j\rangle=\lambda\sum_{j=1}^{NY}b_j\langle y_k y_j\rangle\\&\sum_{i=1}^{NX}\sum_{j=1}^{NX}a_i a_j\langle x_i x_j y_k\rangle-\sum_{i=1}^{NY}b_i\langle y_k y_i\rangle\end{aligned}\qquad(2.15)$$

$$\lambda=\frac{\sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO}\sum_{i=1}^{NX}P_{\mu\nu}a_i\langle\varphi_\mu\varphi_\nu x_i\rangle-\sum_{i=1}^{NX}\sum_{j=1}^{NX}\sum_{m=1}^{NY}a_i a_j b_m\langle x_i x_j y_m\rangle}{n}\qquad(2.16)$$

The λ of equation 2.15 represent an implicitly determined Lagrange multiplier. Its role is to constrain the fits to ensure that they integrate to the correct number of electrons. Setting λ to zero is equivalent to unconstrained fitting.

$$\sum_{i=1}^{NFIT}\sum_{j=1}^{NFIT}a_i b_j\langle x_i y_j\rangle=n\qquad(2.17)$$

The two sets of non-linear equations are solved with an iterative Newton-Raphson algorithm[36]. For this particular algorithm to work, the number of fitting functions in X must be equal to the number of fitting functions in Y (*NX* = *NY* = *NFIT*). Computationally, it makes sense to have an equal number of fitting functions for X and Y since they are usually constructed from the same set of functions. A single cycle of the Newton-Raphson algorithm involves many steps. Throughout the iterative process of the algorithm, only the coefficients of X are altered. The coefficients of Y are obtained afterward. Variations to the coefficients of X result from each Newton-Raphson cycle.

$$\Delta a_i = \sum_{j=1}^{NFIT} W_{ij}^{-1}\left(\langle f\, x_j\rangle - V_j + \lambda Q_j\right)$$
$$f(r) = \sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO} P_{\mu\nu}\,\varphi_\mu(r)\,\varphi_\nu(r) \qquad (2.18)$$

The Lagrange multiplier λ is implicitly determined by satisfying the constraint of equation 2.19 (the formal definition of W, V and Q will be presented shortly).

$$\sum_{i=1}^{NFIT} \Delta a_i Q_i = \langle f\rangle - \sum_{i=1}^{NFIT} a_i Q_i \qquad (2.19)$$

It is worth noting that our applications use unconstrained fits. The advantages of unconstrained fits are two-fold. Firstly, each cycle of the Newton-Raphson algorithm is faster since fewer computations are required. Secondly, the Newton-Raphson algorithm generally converges to the solution in fewer cycles. We found that the unconstrained approach yields fits that integrate to within a few hundredths of the correct number of electrons, thus justifying the approximation. There is no significant difference between the constrained and unconstrained results as far as electronic energy and optimized molecular geometry are concerned.

W is a square matrix of dimension *NFIT* and must be inverted several times. V and Q are vectors of dimension *NFIT*. The elements of W, V and Q are defined with the following equations.

$$V_i = 4\sum_{j=1}^{NFIT}\sum_{k=1}^{NFIT}\langle x_i X y_j\rangle\langle y_j y_k\rangle^{-1}\langle XX y_k\rangle \qquad (2.20)$$

$$Q_i = \sum_{j=1}^{NFIT}\sum_{k=1}^{NFIT}\langle x_i y_j\rangle\langle y_j y_k\rangle^{-1}\langle XX y_k\rangle \qquad (2.21)$$

$$W_{ij} = 2\sum_{k=1}^{NFIT}\sum_{m=1}^{NFIT}\langle x_i x_j y_k\rangle\langle y_k y_m\rangle^{-1}\langle XXy_m\rangle + 4\sum_{k=1}^{NFIT}\sum_{m=1}^{NFIT}\langle Xx_i y_k\rangle\langle y_k y_m\rangle^{-1}\langle Xx_j y_m\rangle \quad (2.22)$$

It is important to distinguish capital letters from small letters in the expressions of V, Q and W. Small letters have indices and represent an individual function within a fitting basis set, without its expansion coefficient. Capital letters represent the entire fitting basis set and thus do not have indices. The cycle of Newton-Raphson steps is repeated until self-consistency is achieved, that is until the coefficients of X have converged to a proper solution. Once the iterations have ended and the coefficients of X are known, the coefficients of Y are easily calculated in a single step.

$$b_i = \sum_{j=1}^{NFIT}\langle y_i y_j\rangle^{-1}\langle XXy_j\rangle \quad (2.23)$$

With the coefficients of X and Y, the exchange energy is easily calculated with equation 2.14. The exchange energy gradient is quickly evaluated once the coefficients are known since it, too, merely involves sums of overlap integrals. The hard work, solving for the coefficients of the fitting basis sets does not have to be repeated for the evaluation of the energy gradient. Taking the derivative of equation 2.14 with respect to a given cartesian coordinate of atom Z yields the following equation.

$$\left.\frac{\partial E^{X\alpha}[\rho(r)]}{\partial Z_\gamma}\right|_{\gamma=x,y,z} = C_\alpha\left\{\begin{array}{l} +\frac{4}{3}\sum_{\mu=1}^{NAO}\sum_{\nu=1}^{NAO}\sum_{i=1}^{NX}P_{\mu\nu}a_i\left(2\left\langle\varphi_\mu\frac{\partial\varphi_\nu}{\partial Z_\gamma}x_i\right\rangle+\left\langle\varphi_\mu\varphi_\nu\frac{\partial x_i}{\partial Z_\gamma}\right\rangle\right) \\ -\frac{2}{3}\sum_{i=1}^{NX}\sum_{j=1}^{NX}\sum_{k=1}^{NY}a_i a_j b_k\left(2\left\langle x_i\frac{\partial x_j}{\partial Z_\gamma}y_k\right\rangle+\left\langle x_i x_j\frac{\partial y_k}{\partial Z_\gamma}\right\rangle\right) \\ +\frac{1}{3}\sum_{i=1}^{NY}\sum_{j=1}^{NY}2\,b_i b_j\left\langle y_i\frac{\partial y_j}{\partial Z_\gamma}\right\rangle \end{array}\right\} \quad (2.24)$$

The exchange energy gradient is especially easy to evaluate when Gaussian atomic function are used in the orbital basis set and the fitting basis sets. The derivative of a Gaussian-type

orbital is simply another Gaussian-type orbital of higher angular momentum, centered at the same point in space. For example, the derivative of a *s* orbital in the *x* direction is a p_x orbital and the derivative of a p_y orbital in the *z* direction would be a d_{yz} orbital.

In the present description of the algorithm, it is assumed that α keeps a single, uniform value throughout the calculation. The same Newton-Raphson algorithm needs only minor adjustments if one wishes to use different values of α in the same calculation. The α parameter can be included in equation 2.18 rather than appear explicitly in the C_α constant of the Xα energy expression.

$$f(r) = \sum_{\mu=1}^{NAO} \sum_{\nu=1}^{NAO} (\alpha_\mu \alpha_\nu)^{3/8} P_{\mu\nu}\, \varphi_\mu(r)\, \varphi_\nu(r) \qquad (2.25)$$

The values of α in equation 2.25 can be fixed at 2/3, fixed at 1, atom-dependent or even basis function-dependent. If a fixed value is chosen, the same energy will result by using either equation 2.25 or equation 2.18. The variable α parameter is trivial to implement and greatly improves the accuracy of the method without any additional computational cost compared to a fixed value implementation[37]. Specific applications in which the Xα functional must reproduce the results of more complex methods greatly benefit from allowing α to take multiple values, as will be demonstrated in a subsequent section.

2.2. The fitting basis sets

To this point, no information has been given on the nature of the X and Y fitting basis sets required in the Newton-Raphson algorithm, other than the fact that they are often made of linear combinations of Gaussian functions. It is customary to create the fitting basis sets from the orbital basis set. Such an approach substantially simplifies the implementation of the algorithm. Like most software packages available, DeFT uses Gaussian functions in its

orbital basis set. Unnormalized *s*, *p* and *d* Gaussian functions, centered at the cartesian coordinates R_x, R_y, R_z, are expressed by the following equations.

$$d^2 = (x - R_x)^2 + (y - R_y)^2 + (z - R_z)^2 \tag{2.26}$$

$$G^s(x,y,z) = e^{-\zeta d^2} \tag{2.27}$$

$$G^p_{L=x,y,z}(x,y,z) = (L - R_L)\, e^{-\zeta(r-R)^2} \tag{2.28}$$

$$G^d_{\substack{L1=x,y,z \\ L2=x,y,z}}(x,y,z) = (L1 - R_{L1})(L2 - R_{L2})\, e^{-\zeta(r-R)^2} \tag{2.29}$$

The most common orbital basis sets are made of numerous contracted Gaussian functions. A contracted Gaussian function is nothing more than a linear combination of primitive Gaussian functions with fixed expansion coefficients. Fitting functions for X are obtained by uncontracting the orbital basis set and scaling the orbital exponents, ζ, of the primitive functions by 2/3. The orbital exponents are multiplied by two because the fitting functions are used to fit the electronic density, which itself is the sum of products of functions, and divided by three because the fit is for the electronic density to the power of one third. Similarly, the fitting functions for Y are obtained by scaling the orbital exponents of the same functions by 4/3 (the exponents of Y are doubled because Y is the square of X).

Since our implementation of the Xα functional focuses primarily on speed, we intend to use Pople's minimal STO-3G orbital basis set[38]. The contracted Gaussian functions in the STO-3G basis set are made of three primitive functions. For example, consider the case of single hydrogen atom. For hydrogen, the STO-3G orbital basis set consists of a single contracted *s* Gaussian function.

$$\varphi^{H/STO-3G}(x,y,z) = 0.154\, e^{-3.425d^2} + 0.535\, e^{-0.624d^2} + 0.445\, e^{-0.169d^2} \tag{2.30}$$

The X and Y fitting basis sets for that hydrogen atom consist of three *s* Gaussian functions with appropriately scaled orbital exponents.

$$X^{H}(x,y,z) = a_1\left(0.154\,e^{-2.283d^2}\right) + a_2\left(0.535\,e^{-0.416d^2}\right) + a_3\left(0.445\,e^{-0.113d^2}\right) \quad (2.31)$$

$$Y^{H}(x,y,z) = b_1\left(0.154\,e^{-4.566d^2}\right) + b_2\left(0.535\,e^{-0.832d^2}\right) + b_3\left(0.445\,e^{-0.226d^2}\right) \quad (2.32)$$

Unfortunately, our attempt to use the uncontracted, scaled functions of STO-3G as fitting basis sets did not produce satisfactory results. Although this approach is favoured in the literature, it is never used with a minimal orbital basis set. It would seem this method for the creation of a fitting basis set is better suited for medium to large orbital basis sets. Alternatively, our code already uses an auxiliary charge fitting basis set for the evaluation of the electronic repulsion energy. Since the charge fitting basis set's obvious role is to fit the charge density, it is only natural to use it as a template for X and Y. The standard auxiliary charge fitting basis set of DeFT is quite extensive. Table 1 summarizes the total number of Gaussian functions for hydrogen and heavy atoms found in this standard auxiliary basis set, taking into account that a *p* orbital consists of three functions while a *d* orbital consists of six functions (a redundant *s* function is produced by the cartesian Gaussians).

Table 1. Standard charge fitting basis set of DeFT.

	s orbitals	*p* orbitals	*d* orbitals	# of functions
heavy atoms	10	4	4	46
hydrogen	6	1	1	15

It would be unreasonable to use the entire charge fitting basis set for the Xα algorithm. The number of fitting functions would be far too great even for relatively small molecules. Keeping in mind that many matrices of dimension *NFIT* are inverted during the iterative

process, it is advantageous to limit the number of fitting functions. Since *d* orbitals are not present in the STO-3G basis set of first row elements, they were not considered for the creation of the fitting basis sets. To further reduce the number of functions, a maximum of one *p* orbital per atom is allowed. Functions with larger exponents have a more abrupt decay. They are often referred to as "tight" functions because their purpose is to describe the electronic density in proximity of the nucleus. We found that these tight functions have a lesser impact on the exchange energy. In fact, eliminating the four tightest *s* functions for heavy atoms and the three tightest *s* functions for hydrogen atoms had an insignificant impact on the Xα exchange energy.

Two different fitting basis sets are created. A smaller fitting basis set, labelled *S*, consists only of the remaining *s* orbitals after removal of the tighter functions. A second, larger fitting basis set, labelled *PS+*, consists of all the functions contained in *S* with an additional *p* function per atom and a bond-centered *s* function per bond. The arbitrary values of 0.5 and 1.0 were chosen for the exponents of the bond-centered *s* functions in X and Y respectively. The additional *p* function is the second tightest of the four original *p* functions in the standard charge fitting basis set. An exception is made for hydrogen since better results are obtained with an exponent of 0.25 for the *p* function instead of the prescribed value of 0.80. Table 2 summarizes the number of basis functions involved in each fitting basis set.

Table 2. Two Xα fitting basis sets.

	s orbitals [a]	*p* orbitals	*d* orbitals	# of functions
heavy atoms (S)	6	0	0	6
hydrogen (S)	3	0	0	3
heavy atoms (PS+)	7	1	0	10
hydrogen (PS+)	4	1	0	7

[a] approximation of one bond for every atom

The number of fitting functions in *S* and *PS+* is small enough that they can be used for large systems. Bond-centered functions can be useful when trying to limit the number of atom-centered functions of higher angular momentum (*p* and *d* functions). Special care must be taken in the evaluation of the energy gradient when bond-centered functions are present. With the chain rule of derivation, it can be proven that the contribution of a bond-centered function is equally distributed to the two implicated nuclei if that bond-centered function is positioned exactly at the midpoint between the nuclei.

2.3. The accuracy of our S and PS+ fitting basis set

The limitations of the Xα functional have been known for a long time[39]. It is not the intent of the present research project to test the accuracy of the analytical Xα functional against other functionals or experimental data. It is unrealistic to expect the simplest of all functionals to perform at the level of the more sophisticated functionals described in the first chapter. At this stage, we are interested in the development of an analytical formulation of the Xα functional using a minimal fitting basis set which reproduces the results of the numerical grid-based Xα functional. Since an efficient and fast algorithm is of premiere importance for our prospected applications, it is to our advantage to limit the size of the fitting basis set. The results of the analytical Xα functional, using the *S* and *PS+* fitting basis sets, are compared to the results of the numerical Xα functional already implemented in DeFT. It is safe to assume that the standard grid of DeFT is accurate enough to yield results that are similar to those obtained with an infinitely fine grid. On the other hand, our minimal fitting basis sets are far from complete. As such, the discrepancies between the analytical and numerical results will arise mainly from the incompleteness of the fitting basis sets. All of the calculations, numerical or analytical, use the STO-3G orbital basis set and a fixed value of 2/3 for α. The choice of orbital basis set and value of α is irrelevant, as long as the same parameters are shared by the numerical and analytical calculations.

42 chemical reactions involving 40 different molecules made of the first row elements hydrogen, carbon, nitrogen and fluorine are included. A wide variety of molecules (cyclic, linear, double bonds, triple bonds, highly substituted, etc...) is found amongst the set, which was inspired from a paper by Andzelm and Wimmer[40]. For the three different approaches (numerical, analytical with *S* and analytical with *PS+*), the gas phase geometry of each molecule is optimized prior to the evaluation of the electronic energy. Ideally, the fitting basis sets will reproduce the numerical electronic energy of the 40 molecules and the numerical variation of electronic energy of the 42 reactions. The average absolute deviation between the analytical and numerical results will determine the quality of the fitting basis sets. Table 3 displays the analytical and numerical results for the total electronic energy of each molecule. Table 4 displays the same information for the variation of total electronic energy of each reaction.

Table 3. Accuracy of *S* and *PS+* fitting basis sets against numerical data when calculating the total electronic energy of 40 molecules.

molecule	Analytical – Numerical (kcal / mol)		molecule	Analytical – Numerical (kcal / mol)	
	S	PS+		S	PS+
hydrogen (H_2)	< 0.1	0.3	imine (CH_3N)	22.1	0.8
methane (CH_4)	0.5	1.2	hydrogen cyanide (HCN)	16.5	0.3
ethane (C_2H_6)	2.3	2.1	methyl cyanide (C_2H_3N)	18.8	< 0.1
propane (C_3H_8)	4.1	3.2	water (H_2O)	25.6	2.2
isobutane (C_4H_{10})	5.6	4.2	hydrogen peroxide (H_2O_2)	84.7	3.3
neopentane (C_5H_{12})	6.8	4.6	methanol (CH_4O)	26.0	3.6
cyclopropane (C_3H_6)	13.9	3.9	ethanol (C_2H_6O)	30.9	4.7
ethylene (C_2H_4)	4.8	1.8	dimethyl ether (C_2H_6O)	25.9	5.8
propene (C_3H_6)	7.3	3.6	oxacyclopropane (C_2H_4O)	46.8	2.7
but-2-ene (C_4H_8)	10.0	5.9	formaldehyde (CH_2O)	21.9	2.5
cyclopropene (C_3H_4)	13.3	1.2	acetaldehyde (C_2H_4O)	26.9	2.8
acetylene (C_2H_2)	-0.1	-1.4	ketene (C_2H_2O)	26.0	-0.4
nitrogen (N_2)	64.4	-3.1	carbon monoxide (CO)	55.3	-0.4
ammonia (NH_3)	21.3	-1.9	nitroxyl (HNO)	78.3	0.5
hydrazine (N_2H_4)	79.8	-3.7	fluorine (F_2)	70.5	3.4
diimide (N_2H_2)	77.4	-3.8	hydrogen fluoride (HF)	13.2	1.4
methylamine (CH_5N)	22.3	1.0	fluoromethane (CH_3F)	16.6	2.1
ethylamine (C_2H_7N)	28.3	1.8	difluoromethane (CH_2F_2)	22.3	1.5
dimethylamine (C_2H_7N)	22.1	4.4	trifluoromethane (CHF_3)	23.3	0.1
trimethylamine (C_3H_9N)	21.2	7.9	tetrafluoromethane (CF_4)	22.6	-1.5

It is clear from tables 3 and 4 that the *S* fitting basis set is not adequate for common applications in chemistry. The exchange energy of polar species involving oxygen, nitrogen and fluorine is largely overestimated. Not surprisingly, the situation is especially troublesome for species involving double or triple bonds.

Table 4. Accuracy of *S* and *PS+* fitting basis sets against numerical data when calculating the variation of total electronic energy of 42 chemical reactions.

reaction	Analytical – Numerical (kcal / mol)		reaction	Analytical – Numerical (kcal / mol)	
	S	PS+		S	PS+
$C_2H_6 + H_2 \rightarrow 2\ CH_4$	-1.2	0.1	$CO + 2CH_4 + 2H_2O \rightarrow 3\ CH_4O$	-29.5	4.6
$CH_5N + H_2 \rightarrow CH_4 + NH_3$	-0.5	-2.0	$N_2 + 4\ NH_3 \rightarrow 3\ N_2H_4$	89.9	-0.5
$CH_4O + H_2 \rightarrow CH_4 + H_2O$	< 0.1	-0.6	$C_3H_8 + CH_4 \rightarrow 2\ C_2H_6$	< 0.1	-0.2
$CH_3F + H_2 \rightarrow CH_4 + HF$	-2.9	0.3	$C_4H_{10} + 2\ CH_4 \rightarrow 3\ C_2H_6$	0.1	-0.4
$N_2H_4 + H_2 \rightarrow 2\ NH_3$	-37.3	-0.3	$C_5H_{12} + 3\ CH_4 \rightarrow 4\ C_2H_6$	0.7	0.1
$H_2O_2 + H_2 \rightarrow 2\ H_2O$	-33.6	0.8	$EtNH_2 + CH_4 \rightarrow C_2H_6 + CH_5N$	-4.3	0.1
$F_2 + H_2 \rightarrow 2\ HF$	-44.2	-0.8	$Met_2NH + NH_3 \rightarrow 2\ CH_5N$	1.2	-0.4
$C_2H_4 + 2\ H_2 \rightarrow 2\ CH_4$	-3.8	0.1	$C_3H_9N + 2\ NH_3 \rightarrow 3\ CH_5N$	3.0	-1.1
$CH_3N + 2\ H_2 \rightarrow CH_4 + NH_3$	-0.3	-2.0	$EtOH + CH_4 \rightarrow C_2H_6 + CH_4O$	-3.1	-0.2
$CH_2O + 2\ H_2 \rightarrow CH_4 + H_2O$	4.2	0.3	$MetOMet + H_2O \rightarrow 2\ CH_4O$	0.6	-0.7
$N_2H_2 + 2\ H_2 \rightarrow 2\ NH_3$	-34.9	-0.5	$CH_2F_2 + CH_4 \rightarrow 2\ CH_3F$	10.3	1.4
$HNO + 2\ H_2 \rightarrow NH_3 + H_2O$	-31.5	-0.8	$CHF_3 + 3\ CH_4 \rightarrow 3\ CH_3F$	25.3	3.8
$HCN + 3\ H_2 \rightarrow NH_3 + CH4$	5.2	-1.7	$CF_4 + 4\ CH_4 \rightarrow 4\ CH_3F$	42.0	6.2
$CO + 3\ H_2 \rightarrow CH_4 + H_2O$	-29.3	2.9	$C_3H_6{}^* + CH_4 \rightarrow C_2H_6 + C_2H_4$	-0.7	-1.0
$N_2 + 3\ H_2 \rightarrow 2\ NH_3$	-21.9	-1.4	$C_2H_4O^{\#} + CH_4 \rightarrow C_2H_6 + CH_2O$	-3.2	0.6
$C_2H_4 + 2\ CH_4 \rightarrow 2\ C_2H_6$	-1.3	< 0.1	$C_2H_3N + CH_4 \rightarrow C_2H_6 + HCN$	-0.5	1.2
$CH_3N + CH_4 + NH_3 \rightarrow 2\ CH_5N$	0.6	2.0	$C_2H_2O + CH_4 \rightarrow C_2H_4 + CH_2O$	0.2	3.4
$CH_2O + CH_4 + H_2O \rightarrow 2\ CH_4O$	4.1	1.4	$C_4H_8 + 2CH_4 \rightarrow C_2H_4 + 2\ C_2H_6$	-1.6	2.4
$N_2H_2 + 2\ NH_3 \rightarrow 2\ N_2H_4$	39.7	0.1	$C_3H_6{}^{**} + 3\ CH_4 \rightarrow 3\ C_2H_6$	-8.7	-1.2
$C_2H_2 + 4\ CH_4 \rightarrow 3\ C_2H_6$	4.8	2.8	$C_2H_4O^{\#\#} + 2CH_4 + H_2O \rightarrow C_2H_6 + 2CH_4O$	-19.1	2.2
$HCN + 2\ CH_4 + 2\ NH_3 \rightarrow 3\ CH_3NH_2$	6.6	4.1	$C_3H_4 + 3CH_4 \rightarrow 2\ C_2H_6 + C_2H_4$	-5.5	1.1

* propene **cyclopropane #acetaldehyde ##oxacyclopropane

As expected, a finite set of atom-centered *s* functions simply cannot account for the resulting electronic density of π-bonds. Additionally, many of the optimized geometries obtained with the *S* fitting basis set differ significantly from the ones obtained numerically, again reflecting the inability of *S* to reflect the correct electronic density. Optimized bond lengths differ by as much as 0.05 angstrom while optimized bond angles differ by as much as 10 degrees. We found that we could not significantly improve the *S* fitting basis set with additional atom-centered *s* functions. On the other hand, the *PS+* fitting basis set fares

much better, reducing the average deviation with the numerical data by roughly an order of magnitude. The difference in optimized geometrical parameters between the analytical and numerical optimizations is also reduced by roughly an order of magnitude when *PS+* is used.

The *PS+* fitting basis set could be improved with additional atom-centered *p* functions, atom-centered *d* functions or bond-centered *s* functions. However, we feel that *PS+*, as detailed in this section, represents a viable compromise between efficiency and accuracy. It gets rid of the gross overestimations of the exchange energy seen with the simpler *S* fitting basis set and optimizes the geometry of molecules correctly. *PS+* performs accurately enough with a variety of chemical systems and is small enough to be used on larger systems.

Table 5. Performance of S and PS+ fitting basis sets against numerical data.

	Average absolute deviation (kcal / mol)	
	S	*PS+*
electronic energy of 40 molecules	27.0	2.5
variation of electronic energy of 42 reactions	13.3	1.4

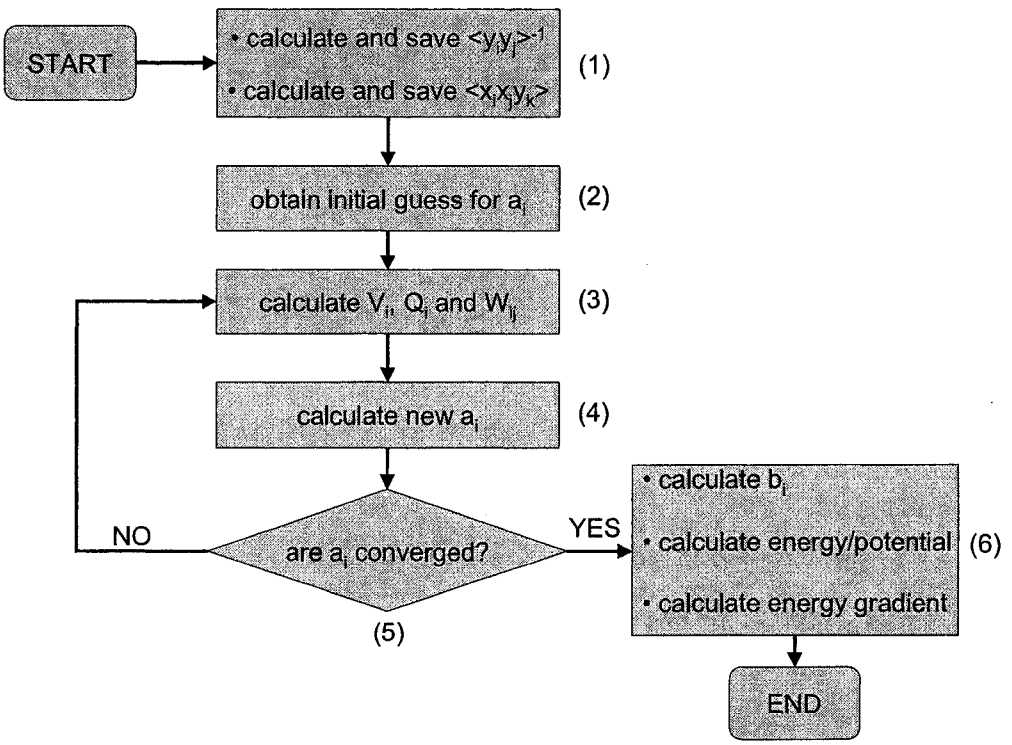
Naturally, subsequent benchmark calculations involved in the study of the efficiency and scaling behaviour of our analytical Xα functional, as well as other applications of the functional, will use the *PS+* fitting basis set.

2.4. Beyond the equations: a look at the code

Making the transition from a list of equations on paper to an efficient computer program is often more challenging than it seems. The algorithm must obviously provide the right answers. It must also be coded in an efficient manner in order to obtain those right answers

as quickly as possible. Additionally, a programmer's options are always limited by the amount of memory and disk space available. The analytical Xα functional was implemented within DeFT, our own DFT software package, which is written in the Fortran77 language[41]. Users can request an analytical Xα calculation when selecting the exchange-correlation functional in the standard DeFT input file. When the analytical Xα functional is chosen, a call to the xalpha.f subroutine is made and subroutines responsible for the generation of the grid are bypassed.

Figure 1. General flow chart of the xalpha.f subroutine.



The xalpha.f subroutine requires the electronic density (in the form of atomic orbitals and the associated expansion coefficients for occupied molecular orbitals) and the fitting basis sets as input. The output of the xalpha.f subroutine is the Xα energy, Xα potential and Xα energy gradient in the case of a geometry optimization. The purpose of the xalpha.f subroutine is to solve equation 2.15, in the process obtaining the expansion coefficients a_i

of X and b_i of Y. Once the expansion coefficients are known, calculating the Xα energy, potential and energy gradient is trivial, involving sums of simple analytical two-center and three-center overlap integrals.

As shown in the flow chart of figure 1, the Newton-Raphson algorithm is iterative. Starting from an initial, arbitrary, set of coefficients a_i, a new set of coefficients is obtained after a series of mathematical operations. The loop is repeated until convergence is achieved, which in this case means the newly generated set of coefficients is identical, to some chosen degree of numerical precision, to the previous set of coefficients. This iterative algorithm must be repeated at each ab initio SCF cycle since the electronic density changes throughout the ab initio calculation. In essence, the Newton-Raphson (NR) algorithm that we use to calculate the Xα energy is an iterative process within another iterative process that is the SCF procedure of ab initio calculations. To prevent unnecessary confusion, a NR cycle will refer to one cycle of the Newton-Raphson algorithm used to calculate the Xα energy (boxes 3-5 in figure 1) while a SCF cycle will refer to one cycle of the ab initio procedure. Details of the mathematical operations required in each box of figure 1, as well as a general discussion of overlap integrals and the manner in which matrices are inverted, will now be presented.

Two-center and three-center overlap integrals

Two-center and three-center overlap integrals involving unnormalized Gaussian functions can be evaluated efficiently with the recursive equations of Obara and Saika[42]. Refer to equations 2.27, 2.28 and 2.29 for the mathematical definition of a Gaussian *s*, *p* and *d* function respectively. The overlap integral between a *s* Gaussian function A, centered at the cartesian coordinates (A_x, A_y, A_z) with exponent ζ_A, and a second *s* Gaussian function B, centered at the cartesian coordinates (B_x, B_y, B_z) with exponent ζ_B, can be calculated with the following equation.

$$\langle s_A | s_B \rangle = \left(\frac{\pi}{\zeta_A + \zeta_B} \right) \exp\left[-\frac{\zeta_A \zeta_B}{\zeta_A + \zeta_B} \left\{ (A_x - B_x)^2 + (A_y - B_y)^2 + (A_z - B_z)^2 \right\} \right] \quad (2.33)$$

Obara and Saika's equations being recursive, all other types of two-center overlap integrals are calculated from the $\langle s_A | s_B \rangle$ integral.

$$P_{i=x,y,z} = \frac{\zeta_A A_i + \zeta_B B_i}{\zeta_A + \zeta_B} \quad (2.34)$$

$$\langle p_{A,i} | s_B \rangle_{i=x,y,z} = (P_i - A_i) \langle s_A | s_B \rangle \quad (2.35)$$

$$\langle p_{A,i} | p_{B,j} \rangle_{i,j=x,y,z} = (P_j - B_j) \langle p_{A,i} | s_B \rangle + \frac{\delta_{ij}}{2(\zeta_A + \zeta_B)} \langle s_A | s_B \rangle \quad (2.36)$$

The fitting basis set *PS+* does not contain *d* functions. No overlap integral involving *d* functions is ever required during the evaluation of the Xα energy or potential. However, integrals involving a single *d* function will appear during the evaluation of the Xα energy gradient, as will be discussed later in this subsection.

$$\langle d_{A,ij} | s_B \rangle_{i,j=x,y,z} = (P_j - A_j) \langle p_{A,i} | s_B \rangle + \frac{\delta_{ij}}{2(\zeta_A + \zeta_B)} \langle s_A | s_B \rangle \quad (2.37)$$

$$\langle d_{A,ij} | p_{B,k} \rangle_{i,j,k=x,y,z} = (P_k - B_k) \langle d_{A,ij} | s_B \rangle + \frac{\delta_{ik}}{2(\zeta_A + \zeta_B)} \langle p_{A,j} | s_B \rangle + \frac{\delta_{jk}}{2(\zeta_A + \zeta_B)} \langle p_{A,i} | s_B \rangle \quad (2.38)$$

From the two-center overlap integral $\langle s_A | s_B \rangle$ it is possible to calculate a three-center overlap integral when a third Gaussian function is considered. The three-center overlap integral between a first *s* Gaussian function A, centered at the cartesian coordinates (A_x, A_y,

A_z) with exponent ζ_A, a second *s* Gaussian function B, centered at the cartesian coordinates (B_x, B_y, B_z) with exponent ζ_B, and a third *s* Gaussian function C, centered at the cartesian coordinates (C_x, C_y, C_z) with exponent ζ_C, can be calculated with the following equation.

$$\langle s_A|s_B|s_C\rangle = \left(\frac{\zeta_A+\zeta_B}{\zeta_A+\zeta_B+\zeta_C}\right)^{3/2}\langle s_A|s_B\rangle \exp\left[-\frac{(\zeta_A+\zeta_B)\zeta_C}{\zeta_A+\zeta_B+\zeta_C}\{(P_x-C_x)^2+(P_y-C_y)^2+(P_z-C_z)^2\}\right] \quad (2.39)$$

As was the case for two-center overlap integrals, all other types of three-center overlap integrals are calculated from the $\langle s_A|s_B|s_C\rangle$ integral.

$$G_{i=x,y,z} = \frac{\zeta_A A_i + \zeta_B B_i + \zeta_C C_i}{\zeta_A+\zeta_B+\zeta_C} \quad (2.40)$$

$$\langle p_{A,i}|s_B|s_C\rangle_{i=x,y,z} = (G_i - A_i)\langle s_A|s_B|s_C\rangle \quad (2.41)$$

$$\langle p_{A,i}|p_{B,j}|s_C\rangle_{i,j=x,y,z} = (G_j - B_j)\langle p_{A,i}|s_B|s_C\rangle + \frac{\delta_{ij}}{2(\zeta_A+\zeta_B+\zeta_C)}\langle s_A|s_B|s_C\rangle \quad (2.42)$$

$$\langle p_{A,i}|p_{B,j}|p_{C,k}\rangle_{i,j,k=x,y,z} = (G_k - C_k)\langle p_{A,i}|p_{B,j}|s_C\rangle + \frac{\delta_{ik}}{2(\zeta_A+\zeta_B+\zeta_C)}\langle s_A|p_{B,j}|s_C\rangle + \frac{\delta_{jk}}{2(\zeta_A+\zeta_B+\zeta_C)}\langle p_{A,i}|s_B|s_C\rangle \quad (2.43)$$

$$\langle d_{A,ij}|s_B|s_C\rangle_{i,j=x,y,z} = (G_j - A_j)\langle p_{A,i}|s_B|s_C\rangle + \frac{\delta_{ij}}{2(\zeta_A+\zeta_B+\zeta_C)}\langle s_A|s_B|s_C\rangle \quad (2.44)$$

$$\langle d_{A,ij}|p_{B,k}|s_C\rangle_{i,j,k=x,y,z} = (G_j - A_j)\langle p_{A,i}|p_{B,k}|s_C\rangle + \frac{\delta_{ij}}{2(\zeta_A+\zeta_B+\zeta_C)}\langle s_A|p_{B,k}|s_C\rangle + \frac{\delta_{jk}}{2(\zeta_A+\zeta_B+\zeta_C)}\langle p_{A,i}|s_B|s_C\rangle \quad (2.45)$$

$$\left\langle d_{A,ij}\middle|p_{B,k}\middle|p_{C,l}\right\rangle_{i,j,k,l=x,y,z} = (G_j - A_j)\left\langle p_{A,i}\middle|p_{B,k}\middle|p_{C,l}\right\rangle + \frac{\delta_{ij}}{2(\zeta_A+\zeta_B+\zeta_C)}\left\langle s_A\middle|p_{B,k}\middle|p_{C,l}\right\rangle + \frac{\delta_{jk}}{2(\zeta_A+\zeta_B+\zeta_C)}\left\langle p_{A,i}\middle|s_B\middle|p_{C,l}\right\rangle + \frac{\delta_{jl}}{2(\zeta_A+\zeta_B+\zeta_C)}\left\langle p_{A,i}\middle|p_{B,k}\middle|s_C\right\rangle \quad (2.46)$$

Overlap integrals involving two *d* functions never appear in our algorithm (due to the usage of the *PS+* fitting basis set), even during the evaluation of the Xα energy gradient. The equations for such integrals are not presented here but they are available in the paper of Obara and Saika[42].

The process of inverting a matrix

Cholesky decompositions[43] refer to a particular type of LU (lower-upper) decomposition of matrices. They are often used to ease the inversion of square, symmetric and positive definite matrices, properties that are all found in an overlap matrix. Cholesky decompositions are advantageous because they require roughly half of the operations necessary in other types of LU decompositions. The entire process is actually quite simple and can be done by hand for smaller matrices. It is best to use a concrete example to demonstrate how a matrix inversion really works with Cholesky decompositions. I encourage the reader to go through the following example on a piece of paper, getting a feel for the inversion process. Matrix elements will be rounded off to the thousandth to lighten the notation. However, rounding off numbers in the actual intermediate mathematical operations will obviously cause numerical errors.

Suppose that the following fictitious three by three overlap matrix S must be inverted. The objective of the problem is obviously to build a matrix S^{-1} which will satisfy the equation $SS^{-1}=I$, where I is the identity matrix.

$$S = \begin{bmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.4 \\ 0.2 & 0.4 & 1 \end{bmatrix} \tag{2.47}$$

Even with a small 3x3 matrix, one would be hard pressed to invert it directly on a piece of paper without using some type of LU decomposition. However, with a decomposition scheme, the inversion is done quickly by solving a series of linear equations, each with a single unknown. The matrix to be inverted is first decomposed into the product of a lower triangular matrix, L, and its conjugate transpose, L*, which are guaranteed to exist and be unique for square, symmetric, positive definite matrices. In a lower triangular matrix, elements above the diagonal are equal to zero. The transpose of a matrix is simply obtained by mutating columns into rows and rows into columns. By definition, the transpose of a lower triangular matrix is an upper triangular matrix. The problem to solve is summarized by the following matrix multiplication, which requires six individual equations to be solved.

$$L\,L^* = S, \begin{bmatrix} x & 0 & 0 \\ y & \alpha & 0 \\ z & \beta & \gamma \end{bmatrix} \begin{bmatrix} x & y & z \\ 0 & \alpha & \beta \\ 0 & 0 & \gamma \end{bmatrix} = \begin{bmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.4 \\ 0.2 & 0.4 & 1 \end{bmatrix} \tag{2.48}$$

$$\begin{aligned} x^2 + 0 + 0 &= 1 \\ xy + 0 + 0 &= 0.5 \\ xz + 0 + 0 &= 0.2 \\ y^2 + \alpha^2 + 0 &= 1 \\ yz + \alpha\beta + 0 &= 0.4 \\ z^2 + \beta^2 + \gamma^2 &= 1 \end{aligned} \tag{2.49}$$

The six unknowns can be progressively solved for, starting with x and followed by y, z, α, β and finally γ in that order.

$$S = LL^* = \begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 0.866 & 0 \\ 0.2 & 0.346 & 0.917 \end{bmatrix} \begin{bmatrix} 1 & 0.5 & 0.2 \\ 0 & 0.866 & 0.346 \\ 0 & 0 & 0.917 \end{bmatrix} \tag{2.50}$$

The inverse of the lower and upper triangular matrices are then individually evaluated. As previously stated, inverting a full matrix is mathematically demanding. However, the inversion of a triangular matrix again merely involves solving linear equations, each with a single unknown. Interestingly, the inverse of a lower/upper triangular matrix is also a lower/upper triangular matrix. L^{-1} is obtained by solving a matrix multiplication problem via six individual linear equations.

$$LL^{-1} = I, \begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 0.866 & 0 \\ 0.2 & 0.346 & 0.917 \end{bmatrix} \begin{bmatrix} x & 0 & 0 \\ y & \alpha & 0 \\ z & \beta & \gamma \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.51}$$

$$\begin{aligned} &x + 0 + 0 = 1 \\ &0.5x + 0.866y + 0 = 0 \\ &0.2x + 0.346y + 0.917z = 0 \\ &0 + 0.866\alpha + 0 = 1 \\ &0 + 0.346\alpha + 0.917\beta = 0 \\ &0 + 0 + 0.917\gamma = 1 \end{aligned} \tag{2.52}$$

In addition to reducing the total number of operations during the actual decomposition, only one of the two triangular matrices needs to be inverted when using Cholesky decompositions. The inverse of L* is simply the transpose of the inverse of L. Regardless of the size of the original matrix, the decomposition and inversion of the triangular matrices will always be performed with similar simple linear equations.

$$L^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -0.577 & 1.155 & 0 \\ 0.000 & -0.436 & 1.091 \end{bmatrix} \text{ and } (L^*)^{-1} = \begin{bmatrix} 1 & -0.577 & 0.000 \\ 0 & 1.155 & -0.436 \\ 0 & 0 & 1.091 \end{bmatrix} \tag{2.53}$$

The inverse of the overlap matrix S is simply equal to the product of the inverted upper triangular matrix by the inverted lower triangular matrix. Notice that the inverse of the symmetric matrix S is also symmetric. Additional time can be saved during the final multiplication of inverted triangular matrices, knowing the result will be symmetric.

$$S^{-1} = (L^*)^{-1}\,(L^{-1}) = \begin{bmatrix} 1 & -0.577 & 0.000 \\ 0 & 1.155 & -0.436 \\ 0 & 0 & 1.091 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -0.577 & 1.155 & 0 \\ 0.000 & -0.436 & 1.091 \end{bmatrix} = \begin{bmatrix} 1.333 & -0.667 & 0.000 \\ -0.667 & 1.524 & -0.476 \\ 0.000 & -0.476 & 1.190 \end{bmatrix} \quad (2.54)$$

It can easily be verified that the inverted matrix is correct by multiplying S by its inverse and obtaining the identity matrix.

$$SS^{-1} = \begin{bmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.4 \\ 0.2 & 0.4 & 1 \end{bmatrix} \begin{bmatrix} 1.333 & -0.667 & 0.000 \\ -0.667 & 1.524 & -0.476 \\ 0.000 & -0.476 & 1.190 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.55)$$

Matrix inversions performed in conjunction with Cholesky decompositions are efficient and numerically stable. Nonetheless, the time required to perform a standard Cholesky decomposition will scale to the third power of the dimension of the matrix. For that reason, the inversion of large matrices is often impossible or at the very least not practical. Our implementation of the Cholesky decomposition algorithm is modified to take advantage of the large proportion of elements that are inherently equal to zero in the large overlap matrices to be inverted.

Box 1 – the once-and-done calculations

Many of the calculations required by the Newton-Raphson algorithm are repeated numerous times in each NR cycle of each SCF cycle. If a particular mathematical operation does not depend on the electronic density or the expansion coefficients of the fitting basis sets, then it will always give the same result, regardless of the NR cycle or SCF

cycle at which it is performed during the algorithm. Provided that the means of storing the results, either on memory or on disk, are available then there is no point in repeating such a calculation. The first step taken to speed up the Newton-Raphson algorithm is to extract and perform all the once-and-done calculations a single time during the first SCF cycle. These results are stored in memory and remain available throughout the rest of the algorithm whenever necessary.

The inverse of the overlap matrix between the fitting functions of Y without the expansion coefficients is required numerous times in a single cycle of the Newton-Raphson algorithm, more precisely in equations 2.20, 2.21, 2.22 and 2.23. Before calculating the inverse of the overlap matrix, it must first be constructed. Two-center overlap matrices are symmetric ($\langle y_i y_j \rangle = \langle y_j y_i \rangle$), hence only the upper triangle and the diagonal are evaluated. If the value of the term in the exponential of equation 2.33 is smaller than -30 then the overlap integral between these two *s* functions is automatically set to zero. This approximation is quite reasonable since the exponential of -30 is approximately equal to 1×10^{-13}. The likelihood of such an occurrence increases quickly as the distance between the centers of the two concerned functions increases (square dependence on the distance in the exponential). Tight functions (large exponents) are also more prone to fall under this automatic cutoff.

Applying such an approximation is advantageous for three reasons. Firstly, it obviously diminishes the number of mathematical operations required for the evaluation of the overlap matrix itself. This is significant since the evaluation of an exponential is costly when compared to other mathematical operations. More importantly, if the $\langle s_A | s_B \rangle$ integral is equal to zero as determined by equation 2.33, then all other integrals involving *p* or *d* functions are also equal to zero, drastically reducing the load of mathematical operations. Secondly, the sparsity created in the matrices (large proportion of elements that equal zero) can be taken advantage of when trying to improve the scaling factor of an algorithm. Finally, setting the integrals to zero as opposed to insignificantly small numbers can improve the numerical stability of the algorithm. Once the overlap matrix is completely built, it is inverted and the $\langle y_i y_j \rangle^{-1}$ matrix elements are stored in memory. This matrix will be called upon on numerous occasions during the algorithm.

The three-center overlap integrals $\langle x_i x_j y_k \rangle$ are also evaluated and stored in memory. Storing this three-dimensional matrix requires much more memory than the simpler two-center overlap matrices. For example, assuming 1000 fitting functions and a requirement of 8 bytes of memory per matrix element, storing the entire matrix would necessitate 8Gb of memory (1000 x 1000 x 1000 x 8 bytes), compared to only 8Mb (1000 x 1000 x 8 bytes) for the two-dimensional equivalent. It is not a viable option to blindly evaluate and store the three-center overlap matrix, using a ridiculously large amount of memory in the process. However, the algorithm slows down dramatically if the three-center overlap integrals are not stored but rather recalculated every single time they are required. Fortunately, for medium to large systems, a three-center overlap matrix will naturally contain an exceedingly large proportion of negligible elements as three-center overlaps are significant only if the three functions considered are relatively close to each other. The odds of having all three functions in any one region of space rapidly diminish as the size of the system increases. There is no reason to waste memory space to store all the matrix elements that are, for all intent and purposes, equal to zero. Symmetry arguments also help in reducing memory requirements since only half of the elements need to be evaluated and stored ($\langle x_i x_j y_k \rangle = \langle x_j x_i y_k \rangle$). By using the symmetry and storing non-zero values only, the three-center overlap matrix of a system with 1000 fitting functions easily fits into 500Mb of memory instead of the gloomy projection of 8Gb.

A quick look at the three-center overlap integrals recursive equations reveals why so many of these integrals are equal to zero. If $\langle s_A | s_B \rangle$ equals zero, which in itself is quite likely for a large system, then all three-center overlap integrals involving s_A and s_B will automatically equal zero, regardless of the identity of the third function. In the few cases where functions s_A and s_B do overlap significantly, then the third function must be in proximity of s_A and s_B in order to obtain a significant three-center overlap as per equation 2.39. For the same reasons it was done for two-center overlap integrals, the three-center overlap integral is automatically set to zero if the value of the term in the exponential of equation 2.39 is smaller than -30.

Box 2 – the initial guess

As is the case for any iterative process, an initial guess of the solution is required to get the cycle started. For the present Newton-Raphson algorithm, an initial set of expansion coefficients a_i for the X fitting basis set is updated during each NR cycle until convergence is achieved. No initial guess of the expansion coefficients b_i for the Y fitting basis set is required since the final coefficients b_i are calculated directly from the converged coefficients a_i. Choosing a smart initial guess will reduce the number of iterations that are necessary to achieve the wanted level of convergence. Most of these iterative algorithms are not foolproof. Using a really illogical initial guess can cause numerical failures or prevent the algorithm from converging to the proper solution.

Starting the Newton-Raphson algorithm from scratch at every SCF cycle would be pointless. The initial guess is necessary for the first NR cycle of the first SCF cycle only. For subsequent SCF cycles, the starting point of the Newton-Raphson algorithm is the end point (converged a_i coefficients) of the previous SCF cycle. While testing different sets of coefficients as initial guesses, we found that the initial guess had very little impact on the efficiency of the Newton-Raphson algorithm.

The best choice for the initial guess is obviously the one that is closer to the final answer for most systems. The determination of the best initial guess is not as simple as it sounds since the chemical environment around an atom will obviously change from one system to the next. Hydrogen atoms in methane are quite different from hydrogen atoms in water. How can a single set of initial coefficients for hydrogen be close to the final solution of a methane molecule and a water molecule? Ideally, the starting coefficients would adjust to the chemical environment. Although that is not the case in our implementation, we are satisfied with the simplicity and performance of the following approach.

If the object is to reproduce an exact final solution, then why not start with coefficients that are known to be exact in a given situation? The geometry of a diatomic hydrogen molecule

was optimized with the analytical Xα functional, using an arbitrary set of expansion coefficients as an initial guess for the *PS+* fitting basis set. At the end of the geometry optimization, the converged expansion coefficients of X for atom-centered *s* functions were recorded. These coefficients now become the initial guess for *s* fitting functions of X centered on a hydrogen atom in any molecule. The same logic was applied for carbon, nitrogen, oxygen and fluorine with a similar procedure using a methane molecule, a diatomic nitrogen molecule, a water molecule and a diatomic fluorine molecule. For bond-centered *s* functions, the starting coefficient is simply the normalization constant of that function, regardless of the two atoms involved in the bond. The *p* type functions found in *PS+* are not always useful (depending on the system). They are assigned a starting coefficient of zero. That way, if a *p* function is not helpful then its coefficient will remain zero, which increases the efficiency of the algorithm since the starting point is closer to the solution.

Box 3 – intermediate calculations

The objective of a single NR cycle is to update the set of expansion coefficients of X. Several intermediate calculations are necessary to achieve this objective. More precisely, the vector V of equation 2.20, vector Q of equation 2.21 and matrix W of equation 2.22 must be evaluated. Q is actually required for constrained fits only. Our fits being unconstrained, there is no need to evaluate this vector. The manner in which Q is evaluated will not be explicitly discussed but the reader can refer to the evaluation of V below, as they are calculated in an almost identical fashion.

Before going into the details of these intermediate calculations, it is important to remember the distinction between a small letter and a capital letter in the next equations. A capital X or Y represents the complete fitting basis set with the expansion coefficients, as depicted in equation 2.9. A small x or y is always accompanied by a subscript and represents an individual fitting function without its expansion coefficient. As such, $\langle XXy_i \rangle$ is a given element of a vector consisting of the sum of all three-center overlap integrals implicating

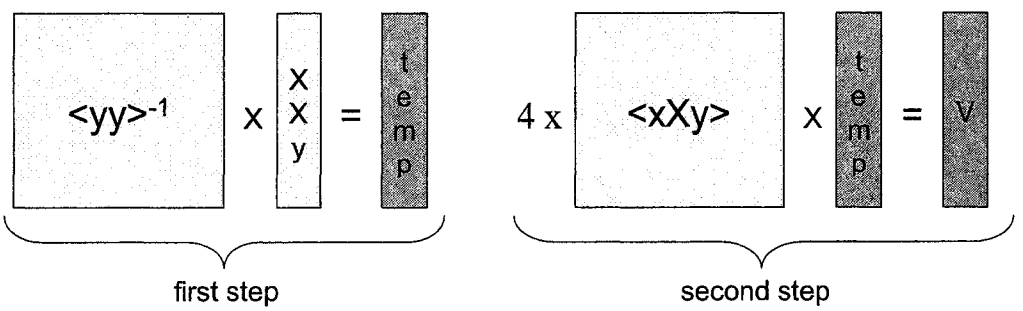
the function y_i. Similarly, $\langle x_i X y_j \rangle$ is a given element of a square matrix consisting of the sum of all three-center overlap integrals implicating the functions x_i and y_j. Prior to actually evaluating V and W, the vector elements $\langle XXy_i \rangle$ and matrix elements $\langle x_i X y_j \rangle$, which are nothing more than sums of three-center overlap integrals, are computed with equations 2.56 and 2.57. This step is quick since all the three-center overlap integrals were previously calculated and stored in memory. All that remains to be done is simply to go through the summations with the expansion coefficients.

$$\langle XXy_i \rangle = \sum_{j=1}^{NFIT} \sum_{k=1}^{NFIT} a_j a_k \langle x_j x_k y_i \rangle \tag{2.56}$$

$$\langle x_i X y_j \rangle = \sum_{k=1}^{NFIT} a_k \langle x_i x_k y_j \rangle \tag{2.57}$$

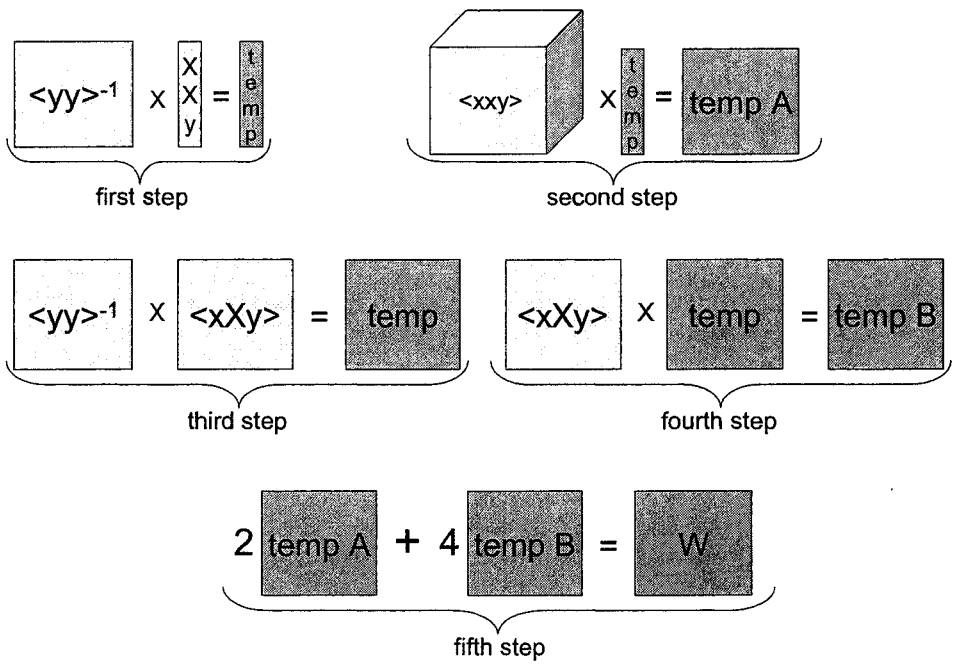
The intermediate calculations of equations 2.20 and 2.22 are nothing more than series of matrix-matrix and matrix-vector multiplications written in the form of multiple summations. V is obtained in a two-step process consisting of consecutive matrix-vector multiplications. In the first step, the $\langle yy \rangle^{-1}$ matrix, previously evaluated and stored in memory, is multiplied by the newly created $\langle XXy \rangle$ vector. A square matrix of dimensions NFIT x NFIT multiplying a vector of dimension NFIT yields a new temporary vector of the same dimension. In the second step, the $\langle xXy \rangle$ matrix is multiplied by the temporary vector that resulted from the first step. The vector resulting from this last multiplication is then multiplied by four to finally obtain V.

Figure 2. Diagram of the matrix algebra required for the calculation of V.



W is obtained through a more demanding five-step process. Two temporary matrices, both of which resulting from consecutive multiplications, are created and then added together to form the matrix W. The first step is done by multiplying the $\langle yy \rangle^{-1}$ matrix by the <XXy> vector (identical to the first step of the evaluation of V). Obviously, the multiplication between $\langle yy \rangle^{-1}$ and <XXy> is not repeated for the purpose of evaluating W. The temporary vector obtained after the same operation during the calculation of V is used again. During the second step, a slice of the three dimensional matrix <xxy>, which was previously evaluated and stored in memory, is multiplied by the vector that resulted from the first step, in effect building the first temporary matrix. In the third step, the $\langle yy \rangle^{-1}$ matrix is multiplied by the <xXy> matrix. The resulting matrix then multiplies the <xXy> matrix in the fourth step, creating the second temporary matrix. Two times the first temporary matrix added to four times the second temporary matrix yields W.

Figure 3. Diagram of the matrix algebra required for the calculation of W.



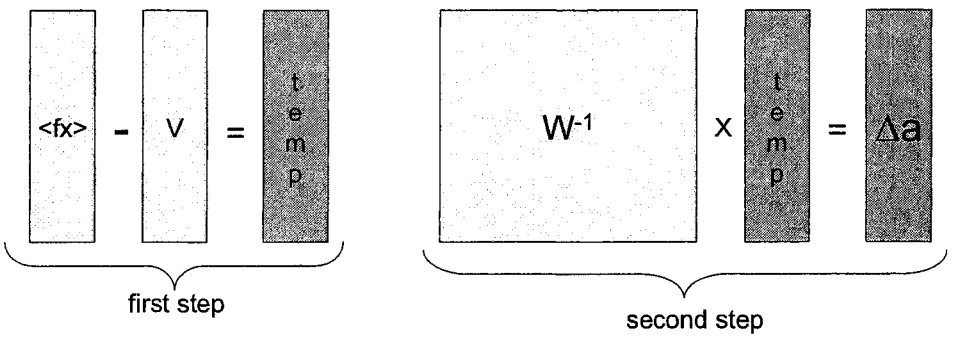
Box 4 – new expansion coefficients

The values of the new coefficients are calculated with equation 2.18. Our fits being unconstrained, λ is set to zero and the λQ term can simply be ignored. A new vector appears, labelled $\langle fx_i \rangle$, which is similar in form to the <XXy> vector. $\langle fx_i \rangle$ elements consist of sums of three-center overlap integrals between functions in the orbital basis set and functions in the fitting basis set X.

$$\langle f\, x_i \rangle = \sum_{\mu=1}^{NAO} \sum_{\nu=1}^{NAO} P_{\mu\nu} \langle \varphi_\mu \varphi_\nu x_i \rangle \qquad (2.58)$$

This vector must be revaluated at each SCF cycle since it is a function of the electronic density and the electronic density varies from one SCF cycle to the next. A two-step process leads to the vector required for the evaluation of new coefficients. In the first step, the difference between the vectors <fx> and V is calculated to create a temporary vector. In the second step, the inverse of W is multiplied by the temporary vector of the first step to obtain a new vector, Δa.

Figure 4. Diagram of the matrix algebra required for the calculation of Δa.



The vector elements Δa_i represent the variation that must be added to the old coefficients in order to obtain the new coefficients. However, we found that adding the whole variations

sometimes resulted in chaotic behaviour where the jump to the next set of coefficients overshoots the correct answer, resulting in a new set of coefficients that is actually worse than the previous set. This situation is more frequent in early NR cycles where the coefficients are further from their correct values. To prevent such numerical instability from derailing the calculation, a less aggressive approach in which only half of the variations are added to the previous coefficients, as demonstrated in equation 2.59, is taken.

$$a_i^{new} = a_i^{old} + \frac{1}{2}\Delta a_i \tag{2.59}$$

The matrix W is inverted with the same modified Cholesky decomposition scheme used to invert the <yy> matrix. If W was inverted at every NR cycle of every SCF cycle, the total time required for the entire Xα calculation would be heavily dominated by matrix inversions. To reduce the amount of time spent on matrix inversions, we exploit the fact that W needs not be inverted within each NR cycle of each SCF cycle since it does not change appreciably as convergence is being achieved. This kind of approximation is not uncommon in computational chemistry. For example, the inverse of the Hessian matrix (second derivatives of the energy with respect to nuclear coordinates) is necessary in the Newton optimization algorithm. However, for systems of appreciable size, the evaluation and inversion of the Hessian matrix is too costly to be repeated more than once during a calculation. In Quasi-Newton methods, the inverted Hessian matrix is updated with the analysis of the gradient vectors rather than being explicitly evaluated each time. Several update formulas have been proposed. The BFGS update scheme, independently developed by Broyden[44], Fletcher[45], Goldfarb[46] and Shanno[47], is commonly found in many computational chemistry software packages.

In our case, we do not use an updating technique for W^{-1} but rather simply skip some of the inversions. For particular NR cycles, instead of revaluating and inverting the true W, we use the W^{-1} of an earlier NR cycle. We experimented in a trial and error fashion with various frequencies at which W should be evaluated and inverted. If the inversion is performed at every NR cycle, then the algorithm will perform optimally and require fewer

NR cycles. If, however, the inversion is performed less frequently then a larger number of NR cycles will be required to achieve convergence.

The objective is to find the right balance to actually minimize the total time spent for the entire evaluation of the Xα energy. We observed that it is sufficient to perform the matrix inversion of W solely on the first NR cycle of the first, second and other SCF cycles that are multiples of four (SCF cycles 1, 2, 4, 8, 12, etc.). The matrix inversions are more frequent early in the calculation since the electronic density changes more significantly at that point. Skipping the matrix inversions on given NR cycles does not alter the number of SCF cycles that are required for a particular calculation. The only undesirable effect is the requirement of a additional NR cycles. Since the inversion of matrices is much more costly than a complete NR cycle without the inversion, a considerable amount of time is saved by trading matrix inversions for additional NR cycles without any inversion.

Box 5 – has the cycle converged?

By nature, iterative algorithms converge toward the correct answer cycle by cycle. As with any iterative process, the user defines convergence with his desired level of numerical precision. The criterion can be tight (closer to the true answer) or loose, depending on the application's needs. Obviously, a tight convergence criterion will require a larger number of cycles to achieve convergence. Too tight a convergence criterion can sometimes lead to endless loops where convergence is impossible to reach for numerical reasons.

Many examples of convergence criteria are found in ab initio calculations. For example, the SCF procedure will stop when the electronic density at a particular cycle is similar enough to the electronic density of the previous cycle. In order to quantify the convergence into a criterion, one usually looks at the elements of the density matrix. From one SCF cycle to the next, a convergence criterion can be established with the maximal variation of a single matrix element, the root mean square deviation of all the matrix elements or a combination of the two. A similar situation arises during geometry optimizations. When is

the geometry of a molecule optimized? If performed in cartesian coordinates, the convergence criteria usually include the maximal value of the energy gradient for any atom in any direction, the average energy gradient for all atoms in all three directions, the maximal displacement during an optimization step for any atom in any direction, the average displacement during an optimization step for all atoms in all three directions or a combination of the four.

In our case, the proper variable to verify the convergence is obviously the set of expansion coefficients a_i since they are the ones that change at every NR cycle. At the end of each NR cycle, a total of NFIT new coefficients are obtained and the maximal absolute value of Δa_i (see equation 2.59) encountered is recorded. Our convergence criterion is based on this maximal absolute value. Prior to choosing the value of the convergence criterion, we established that a variation of $1x10^{-8}$ Hartree for the Xα energy was small enough to be considered as converged. We then investigated what order of variations in the coefficients would produce such a variation in the Xα energy. We found that a maximal absolute value of $1x10^{-4}$ for Δa_i produces variations in the Xα energy that are less than $1x10^{-8}$ Hartree. However, there is no need to achieve such a tight level of convergence on early SCF cycles since the electronic density will change significantly at this stage of the calculation. In order to reduce the total number of NR cycles and ultimately save time, we start with a looser convergence criterion in early SCF cycles and progressively tighten it to $1x10^{-4}$.

Table 6. Convergence criterion as a function of the SCF cycle.

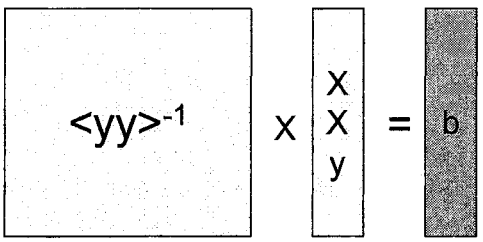
SCF cycle	Convergence criterion (maximal $\lvert\Delta a_i\rvert$)
1 and 2	1
3 and 4	$1x10^{-1}$
5 and 6	$1x10^{-2}$
7 and 8	$1x10^{-3}$
> 8	$1x10^{-4}$

Using such a progressive convergence criterion does not add to the number of SCF cycles required in the calculation. In addition to the obvious advantage of reducing the total number of NR cycles, the progressive convergence criterion provides the indirect benefit of reducing the total number of matrix inversions, which is significant since matrix inversions are the most costly operation within the algorithm.

Box 6 – final touches

Once the expansion coefficients of the fitting basis set X have been determined through the Newton-Raphson iterative algorithm, the expansion coefficients of the fitting basis set Y, the Xα energy and Xα energy gradient are easily computed. The expansion coefficients of Y are directly calculated with a matrix-vector multiplication involving the converged expansion coefficients of X, as demonstrated in equation 2.23.

Figure 5. Diagram of the matrix algebra required for the calculation of b.



Once the expansion coefficients of X and Y have been evaluated, the Xα energy is easily calculated with equation 2.14, involving sums of integrals multiplied by the appropriate expansion coefficients. If required, the Xα energy gradient is calculated very rapidly, again with sums of overlap integrals multiplied by expansion coefficients as demonstrated in equation 2.24.

The derivative of a Gaussian function with respect to a cartesian coordinate is simply another Gaussian function centered at the same position but with higher angular momentum. The equations required to calculate these overlap integrals have been detailed in the earlier parts of this subsection. However, special care must be taken when taking the derivative of bond-centered functions in order to calculate the correct forces on the nuclei. Consider a bond-centered function, φ, that is placed exactly at the midpoint, M, of a line connecting nucleus A and nucleus B. The cartesian coordinates of M can easily be calculated from the cartesian coordinates of A and B.

$$M_{i=x,y,z} = \frac{A_i + B_i}{2} \tag{2.60}$$

Using the chain rule of derivation, the derivative of that function, centered at M, is equally divided in two contributions going to atom A and atom B respectively.

$$\begin{aligned} \frac{d\varphi}{dA_i} &= \frac{d\varphi}{dM_i}\frac{dM_i}{dA_i} = \frac{d\varphi}{dM_i}\left(\frac{1}{2}\right) \\ \frac{d\varphi}{dB_i} &= \frac{d\varphi}{dM_i}\frac{dM_i}{dB_i} = \frac{d\varphi}{dM_i}\left(\frac{1}{2}\right) \end{aligned} \tag{2.61}$$

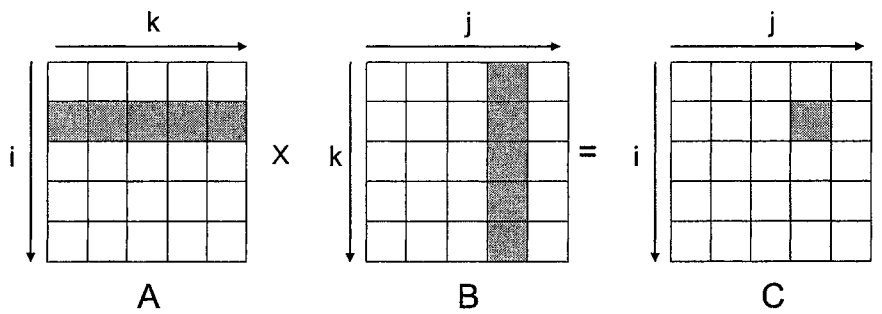
Half of the derivative of the bond-centered function with respect to the midpoint coordinate is attributed to nucleus A and the other half is attributed to nucleus B. Bond-centered functions are adjusted to remain exactly at the midpoint between the nuclei at each step of the geometry optimization.

2.5. Improving the scaling factor by taking advantage of matrix sparsity

It has been mentioned numerous times in the previous sections that parts of the code were modified to take advantage of matrix sparsity. The overlap matrices and other

vectors/matrices contain a large proportion of negligible elements. For example, for an extended peptide of 20 glycine units, the absolute value of only 8% of the elements of W (matrix of dimension 1318x1318 for 1737124 elements in total) is larger than 10^{-10}. It is possible to take advantage of all those negligible elements whether that matrix is involved in a sum, difference, multiplication or inversion. The same technique is used regardless of the kind of mathematical operation considered. The following example will demonstrate this technique applied to the multiplication of two square matrices. Consider the multiplication of a pair of 5x5 matrices A and B, leading to the matrix C. The matrix element C(i,j) is obtained by the dot product of row i in matrix A by column j in matrix B.

Figure 6. Schematisation of a matrix multiplication.



A computer program written to perform this multiplication of matrices would consist of summations and scalar multiplications distributed among three loops. The indices in figure 7 are arranged in this fashion to intuitively recognize the dot product between the rows in A and the columns in B.

Figure 7. The intuitive multiplication of 5x5 matrices.

```
      do 1000 i=1,5
      do 1000 j=1,5
      do 1000 k=1,5
1000  C(i,j)=C(i,j)+A(i,k)*B(k,j)
```

A total of 250 mathematical operations (5x5x5x2), 125 sums and 125 multiplications, are required to completely evaluate the matrix C. If the matrices involved were ten times larger then 250000 mathematical operations (50x50x50x2) would be required. The number of operations increases as the cube of the matrices' dimensions (10 times the size leads to 10^3 times the operations). Poor scaling behaviour such as this absolutely must be eradicated if an algorithm is to be efficient for calculations on large systems. Fortunately, in computational chemistry, large matrices often have a very large proportion of elements that are equal to zero. It is possible take advantage of this matrix sparsity with a very simple technique, in the process improving the scaling behaviour of matrix multiplications or other mathematical operations. Each running of the inner loop involving either A(i,k)=0 or B(k,j)=0 is wasteful. The objective is to eliminate every operation that would either multiply a matrix element by zero or add zero to an existing matrix element.

Although figure 7 represents the intuitive way of programming a matrix multiplication, it is not the most effective. Other than being intuitive, there is no reason to have k as the inner-most loop. As a first step, by simply interchanging loops j and k, it becomes possible to remove scalar multiplications in which A(i,k)=0 with a logical statement.

Figure 8. A less intuitive multiplication of 5x5 matrices.

```
      do 1000 i=1,5
      do 1000 k=1,5

      if (A(i,k).ne.zero) then
                          do 1001 j=1,5
1001                      C(i,j)=C(i,j)+A(i,k)*B(k,j)
      endif

1000  continue
```

Notice that we are no longer proceeding via the intuitive dot product between row i of A and column j of B to give element C(i,j). Instead of completely calculating a given C(i,j)

element and moving on to the next, a single element of A individually multiplies all the elements in a row of B and the products are distributed as partial contributions to the correct C(i,j) element.

For the sake of simplicity, consider a situation where each row or column contains exactly three elements that are not equal to zero, regardless of the size of the matrix. This might seem like a silly supposition but it is not that far fetched. For example, A and B could be overlap matrices in which each function can only significantly overlap with itself and two other functions because of the distances separating the functions. If a given function can only reach two other functions for a system of a given size, it will still only reach the same two functions when the system grows. By using the less intuitive approach to the problem in figure 8, the number of mathematical operations is reduced from 250 to 150 (5x3x5x2) when multiplying 5x5 matrices. More importantly, the number would drastically be reduced from 250000 to 15000 (50x3x50x2) when multiplying 50x50 matrices. This simple interchange in the order of the loops fixes half the problem of poor scaling behaviour as the number of operations now increases as the square of the matrices' dimensions (10 times the size leads to 10^2 times the operations).

Unfortunately, wasteful mathematical operations still occur if B(k,j)=0. Prior to the actual matrix multiplication, the rows of B must be scanned in order to record how many elements within a particular row are not equal to zero and the position of these elements in this row. A vector icountB counts how many elements in row k are not equal to zero while a matrix iposB records the position of such elements within row k.

Figure 9. Scanning for elements that are not equal to zero within a column.

```
      do 1000 k=1,5
      do 1000 j=1,5
      if (B(k,j).ne.zero) then
                        icountB(k)=icountB(k)+1
                        iposB(k,icountB(k))=j
      endif
 1000 continue
```

Obviously, additional time is required to scan the matrix and record the pertinent information. However, for the sake of this demonstration, consider that, particularly for a big matrix, the scanning time is insignificant compared to the time required for the actual multiplication. Additionally, since we have to build the matrices rather than simply read them, the scanning and recording endeavour is made on the fly as the matrices are built. The scanning and recording of the value and position of significant matrix elements only needs to be performed the first time this matrix is involved in a mathematical operation. For subsequent operations, the matrix is already prepped and ready. In our case, many of the matrices, such as the $<yy>^{-1}$ matrix, are multiplied numerous times during the algorithm but scanned only once.

Armed with the very useful information contained in icountB and iposB, the matrix elements A(i,k) that are not equal to zero will now strictly multiply matrix elements B(k,j) that are also not equal to zero. Not a single mathematical operation is wasted by multiplying by zero or adding zero.

Figure 10. The multiplication of sparse 5x5 matrices.

```
      do 1000 i=1,5
      do 1000 k=1,5

      if (A(i,k).ne.zero) then
                        do 1001 j=1,icountB(k)
1001                    C(i,iposB(k,j)=C(i,iposB(k,j))+A(i,k)*B(k,iposB(k,j))
      endif

1000  continue
```

In this simple example, icountB is always equal to three since exactly three elements by column or row are not equal to zero, regardless of the size of the matrix. The number of mathematical operations required to multiply 5x5 matrices is further reduced to 90 (5x3x3x2) compared to earlier results of 150 (figure 8) and 250 (figure 7). 900 (50x3x3x2)

mathematical operations would be required to multiply 50x50 matrices, a drastic reduction when compared to 15000 (figure 8) or 250000 (figure 7). The number of operations now increases at the same rate as the matrices' dimensions (10 times the size leads to 10 times the operations), which is theoretically perfect linear scaling.

Practical applications do not necessarily follow all the assumptions that were made in this theoretical exercise. Therefore, perfect linear scaling will not be achieved. The benefit of using this approach is obviously proportional to the extent of the sparsity in the matrices. For small matrices or large matrices that are not sparse enough, this alternative scheme might even be slower than a standard, intuitive matrix multiplication. In addition to matrix-matrix multiplications, the same logic can easily be applied to any other linear algebra operation such as matrix-vector multiplications, vector-vector multiplications and operations detailed in the matrix inversion section.

2.6. Scaling and efficiency of our analytical Xα functional

Ideally, when compared with the numerical Xα algorithm, the analytical Xα algorithm would be as accurate, be more efficient (faster) and scale equally or better when the size of the system increases (faster for systems of all sizes). The accuracy is simple enough to measure and depends on how well the fitting basis set performs compared to a numerical grid of points. We have already stated that we are satisfied with the accuracy provided by the *PS+* fitting basis set.

It is important to distinguish between efficiency and scaling behaviour since the two concepts do not necessarily go hand in hand. If the analytical Xα algorithm was implemented in a straightforward fashion, without any concern for its scaling behaviour, then the time required for a calculation would increase with the third power of the number of fitting functions (scaling factor of three). Doubling the size of the system (twice the number of fitting functions) would result in a calculation that is eight times longer. Many

operations in each NR cycle, such as the evaluation of the three-center overlap integrals, matrix multiplications and matrix inversions are responsible for such a scaling behaviour. On the other hand, the time required for a calculation with the numerical version of the Xα algorithm naturally scales linearly with the number of grid points (scaling factor of one). In that case, doubling the size of the system (doubling the number of grid points) results in a calculation that is only two times longer. No matter how fast the analytical algorithm is compared to its numerical counterpart, it would quickly become the slower of the two methods as the size of the system increases because of the scaling factors. Consider a hypothetical situation where the analytical Xα algorithm, with its scaling factor of three, is twenty times faster than the numerical Xα algorithm, with its scaling factor of one, for a given system.

Table 7. Comparisons of efficiency for algorithms with different scaling behaviours.

Size	Time required (time units)	
	straightforward Analytical Xα	Numerical Xα
X	1	20
2X	8	40
3X	27	60
4X	64	80
5X	125	100
6X	216	120
7X	343	140
nX	n^3	20n

Even if the analytical algorithm had an overwhelming speed advantage for a system of size X, the trend would be reversed for a system that is five times larger, with the numerical algorithm now being the more efficient method. The situation worsens as the size of the system increases to 6X with the numerical algorithm now being roughly twice as fast. It is a necessity to improve upon the natural scaling behaviour of the analytical Xα algorithm to remain competitive with the numerical algorithm for larger systems. That is why much

time and effort was spent toward not only writing a code that is efficient, but also writing a code with a good scaling behaviour.

To study the efficiency and scaling behaviour of the analytical and numerical Xα algorithms, benchmark calculations were performed on fully extended peptides made of 0 (capping fragments only), 1, 2, 3, 4, 6, 8, 10, 12, 14, 16, 18 and 20 glycine units respectively. The peptides are capped with the usual terminal fragments, as shown in figure 11. They were generated using the XLEAP module of the AMBER7[48] software. The atomic coordinates were then extracted and inserted into a DeFT input file. All the benchmark calculations ran on a single dedicated PC with an Intel Xeon 3.06 GHz processor (512K of cache memory) and 2 Gb of RAM.

Figure 11. Systems used for benchmark calculations with n glycine units

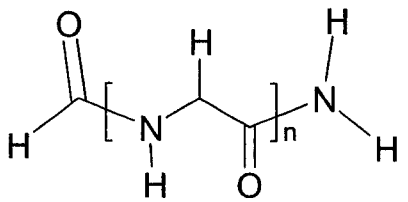


Table 8 summarizes the number of fitting functions, the number of SCF cycles required, the total number of NR cycles required and the total number of matrix inversions required for the complete evaluation of the Xα energy of each peptide. The scaling factor of a given algorithm (or section of algorithm) is calculated as the slope of a logarithm versus logarithm plot. In this case, the time required to perform the entire calculation (or a section of the entire calculation) can be expressed as a constant multiplying the number of fitting functions to the power of the scaling factor, which will be symbolized as SF.

$$\text{time} = K \times (\text{NFIT})^{SF} \qquad (2.62)$$

A linear equation is obtained by taking the logarithm on both side of equation 2.62. If the scaling factor is not size-dependent then a plot of the logarithm of the time versus the logarithm of the number of fitting functions will be a straight line of slope SF. The scaling factor can easily be obtained via a standard linear regression.

$$\log(\text{time}) = \log(K) + SF \times \log(NFIT) \tag{2.63}$$

Table 8. Requirements for analytical Xα calculations on glycine peptides.

n	NFIT	# SCF cycles	# NR cycles	# matrix inversions
0	98	13	30	6
1	159	14	33	6
2	220	15	34	6
3	281	16	36	7
4	342	16	39	7
6	464	16	44	7
8	586	16	43	7
10	708	16	45	7
12	830	16	45	7
14	952	15	44	6
16	1074	15	44	6
18	1196	15	44	6
20	1318	15	44	6

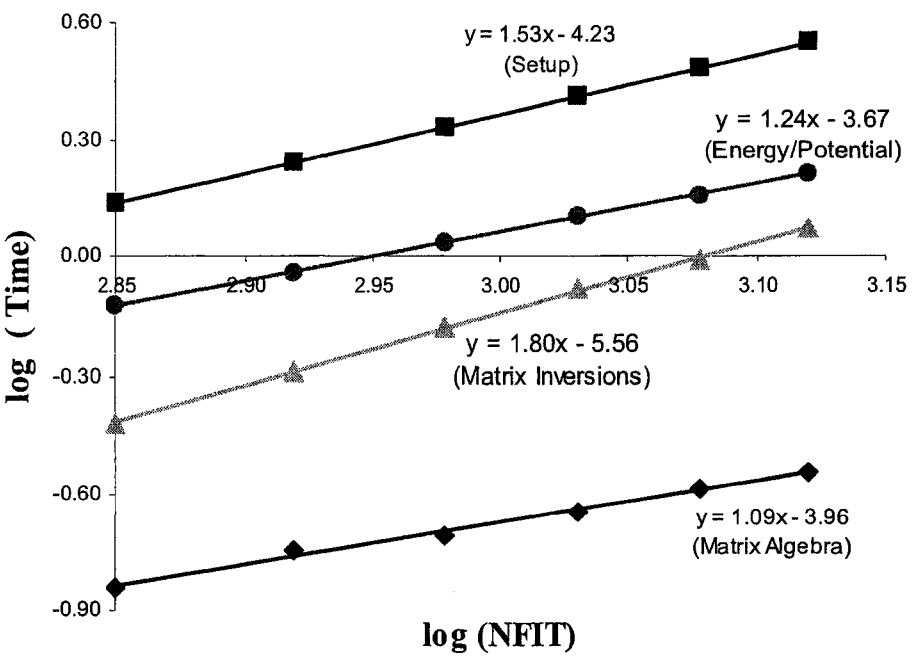
Improvements to the scaling factor are made possible by taking advantage of the sparsity of the various matrices throughout the algorithm. The sparsity becomes significant for large systems only. It is unrealistic to hope to improve upon the inherent scaling factor of three of the analytical Xα algorithm for small systems. For that reason, only the peptides containing in excess of eight glycine units are considered for the measurement of the scaling factor. However, timings related to peptides of all sizes are still presented to

demonstrate that the analytical algorithm is faster than the numerical algorithm for peptides of any size.

Our implementation of the analytical Xα algorithm is divided into many subsections, each with their own set of mathematical operations. The algorithm was broken down into four independent phases: the setup, the matrix inversions, calculation of the Xα energy/potential and the general matrix algebra. The efficiency and scaling behaviour of the overall algorithm depends on the performance of its individual phases. Just as a chain is as strong as its weakest link, the scaling behaviour of an algorithm is as good as that of its worst phase.

The linear regression obtained for the logarithm of the time versus logarithm of the number of fitting function is displayed for each of the four phases in figure 12. The data related to the plots and the manner in which the individual scaling factors were calculated will now be discussed.

Figure 12. Scaling factor of four phases of the analytical Xα algorithm.



Setup phase

The setup phase comprises all the calculations performed inside box 1 in figure 1, except for the inversion of the <yy> matrix, which is included in the matrix inversion phase. These calculations include the evaluation of two-center overlap integrals $\langle y_i y_j \rangle$, the evaluation of three-center overlap integrals $\langle x_i x_j y_k \rangle$ and the storage of matrix elements that are not equal to zero in a fashion that will allow us to take advantage of matrix sparsity.

Table 9. Timings recorded and scaling factor for the setup.

n	log NFIT	Time (s)	Fraction of Total time (%)	log Time	Scaling factor
0		0.06	9.7		
1		0.14	7.7		
2		0.24	7.0		
3	N/A	0.35	6.0	N/A	
4		0.49	6.3		
6		0.75	6.0		
8		1.05	6.1		
10	2.850	1.37	6.1	0.137	
12	2.919	1.74	6.2	0.241	
14	2.979	2.14	6.9	0.330	1.53
16	3.031	2.59	7.1	0.413	
18	3.078	3.04	7.2	0.483	
20	3.120	3.55	7.4	0.550	

Approximately 7% of the total time required for the complete evaluation of the Xα energy is spent on the setup phase. This fraction slightly increases as the peptides get bigger,

illustrating that the scaling factor of this particular phase is poorer than the overall scaling factor. The high scaling factor of 1.53 is primarily explained by the building, scanning and recording of overlap matrices. In order to build and prepare the overlap matrices for the rest of the algorithm, the code inherently needs to evaluate each individual matrix element and record its value and position if it is not equal to zero. The simple fact that all matrix elements need to be scanned at least once leads to part of the code that inherently have a scaling factor of two for the two-center overlap matrix and three for the three-center overlap matrix.

Matrix inversions phase

Two different matrices are inverted at various instants during the algorithm. The <yy> overlap matrix is inverted once (box 1 of figure 1) and W is inverted at predetermined NR cycles in order to calculate the new set of expansion coefficients (box 4 of figure 1). As can be seen from table 8, the smaller peptides require fewer NR cycles and matrix inversions than the larger peptides. To be able to fairly compare between different peptides, the timings presented in table 10 represent the average time spent on a single matrix inversion (total time for matrix inversions divided by total number of matrix inversions) instead of the total time spent on matrix inversions for a complete evaluation of the Xα energy.

Table 10. Timings recorded and scaling factor for the matrix inversions.

n	log NFIT	Time (s / # Inv)	Fraction of Total time (%)	log Time	Scaling factor
0		< 0.01	1.6		
1		0.01	2.2		
2		0.01	2.1		
3	N/A	0.02	2.6	N/A	
4		0.05	4.5		
6		0.14	7.8		
8		0.26	10.4		
10	2.850	0.38	11.9	-0.417	
12	2.919	0.52	12.9	-0.288	
14	2.979	0.66	12.9	-0.178	1.80
16	3.031	0.84	13.7	-0.078	
18	3.078	0.99	14.1	-0.007	
20	3.120	1.18	14.8	0.072	

The fraction of the total time required for a complete evaluation of the Xα energy spent on matrix inversions steadily increases from a negligible value for the smaller peptides to approximately 15% for the largest peptide. This steady increase is a sign that the scaling factor of the matrix inversions phase is significantly larger than that of the overall algorithm. That is one of the reasons why we sought to limit the number of matrix inversions during the algorithm. Still, a scaling factor of 1.80 is quite an improvement over the natural scaling factor of three that is inherent for standard Cholesky decompositions. Improving the scaling factor of the standard Cholesky decomposition scheme proved to be more challenging than doing so in other phases of the Xα algorithm, especially when inverting W. The absolute value of each overlap matrix element $\langle y_i y_j \rangle$ is smaller or equal to one. On the other hand, W, while still satisfying the conditions required for a Cholesky decomposition that were earlier mentioned, is not as well behaved.

Pushing the approximations to their limit while aggressively trying to take advantage of the sparsity of W had two negative repercussions. In extreme cases, numerical failures occur when the algorithm tries to calculate the square root of a negative number during the inversion of the triangular matrices. In less severe cases, the algorithm would go through the entire matrix inversion, but the inverted matrix was of poor quality. The quality of the inversion can be substantiated by verifying if the original matrix multiplied by its inverse is really equal to the identity matrix. This second effect is similar to inverting matrices by hand and rounding off the numbers to a given decimal place at every intermediate step. Consequently, the lack of exactitude in the inverted matrices led to difficulties when trying to achieve convergence in the Newton-Raphson algorithm. For some systems, we are able to be much more aggressive and significantly lower the scaling factor for the matrix inversions. However, we had to account for the failures observed in other systems and use an approach that was stable for all cases, resulting in a less aggressive approach and a higher than initially hoped for scaling factor of 1.80.

Calculation of Xα energy/potential phase

The calculation of the Xα energy and potential is straightforward once the expansion coefficients of X and Y are known (box 6 of figure 1). It simply involves the evaluation and summation of two-center and three-center overlap integrals. The Xα energy is calculated from equation 2.14 while the Xα potential is added via contributions to the Kohn-Sham matrix by evaluating the integral of the product of the electronic density and equation 2.13. The Xα energy and potential are evaluated once per SCF cycle because they depend on the electronic density. In order to fairly compare calculations that required different numbers of SCF cycles, the timings presented in table 11 represent the average time required for a complete evaluation of the Xα energy and potential (total time for the evaluation of Xα energy and potential divided by the total number of SCF cycles).

A significant fraction of the total time required for a complete evaluation of the Xα energy is spent toward the evaluation of the Xα energy and potential, primarily because the three-center overlap integrals involving functions in the orbital basis set, $\langle\varphi_i\varphi_j x_k\rangle$, are recalculated at each SCF cycle. It is not feasible to store these integrals in memory as it was done for the $\langle x_i x_j y_k\rangle$ integrals. Because of memory constraints, we decided to store only one of these two sets of integrals in memory. We chose the $\langle x_i x_j y_k\rangle$ integrals for two reasons. Firstly, there are fewer $\langle x_i x_j y_k\rangle$ integrals than $\langle\varphi_i\varphi_j x_k\rangle$ integrals. Even with the minimal STO-3G orbital basis set, the number of primitive Gaussian function in the orbital basis set easily exceeds the number of fitting functions. Secondly, and more importantly, the $\langle x_i x_j y_k\rangle$ integrals are required numerous times within each NR cycle of each SCF cycle while the $\langle\varphi_i\varphi_j x_k\rangle$ integrals are required only twice in each SCF cycle. A greater load of mathematical operations and more time is saved while storing the $\langle x_i x_j y_k\rangle$ integrals rather than the $\langle\varphi_i\varphi_j x_k\rangle$ integrals.

Table 11. Timings recorded and scaling factor for the Xα energy/potential.

n	log NFIT	Time (s / SCF)	Fraction of Total time (%)	log Time	Scaling factor
0		0.03	67.7		
1		0.09	68.3		
2		0.16	68.8		
3	N/A	0.23	64.2	N/A	
4		0.30	60.3		
6		0.44	56.8		
8		0.59	54.7		
10	2.850	0.75	53.3	-0.125	
12	2.919	0.91	52.0	-0.042	
14	2.979	1.08	52.3	0.033	1.24
16	3.031	1.26	51.9	0.102	
18	3.078	1.43	51.1	0.156	
20	3.120	1.63	51.1	0.212	

The scaling factor of 1.24 for that phase is quite satisfactory and is another reason why we feel it is acceptable to recalculate the $\langle\varphi_i\varphi_j x_k\rangle$ integrals during each SCF cycle. As the peptides gain size, the fraction of the total time spent on the calculation of the Xα energy and potential steadily decreases. That trend would continue for peptides in excess of 20 glycine units.

The general matrix algebra phase

All mathematical operations that are not covered by the three previous phases are included in the general matrix algebra phase. Schematically, these mathematical operations include calculations inside boxes 2 through 5 of figure 1, except for the inversion of W, which is

already covered by the matrix inversions phase. Theoretically, this phase should have the best scaling behaviour since all the preparation work is already done to perform matrix-matrix and matrix-vector multiplications. To accurately compare peptides that required different numbers of NR cycles, the timings presented in table 12 represent the average time spent on general matrix algebra per NR cycle (total time spent on general matrix algebra divided by total number of NR cycles).

Table 12. Timings recorded and scaling factor for the general matrix algebra.

n	log NFIT	Time (s / NR)	Fraction of Total time (%)	log Time	Scaling factor
0		< 0.01	22.6		
1		0.01	20.8		
2		0.02	22.2		
3	N/A	0.04	27.1	N/A	
4		0.06	28.7		
6		0.08	29.3		
8		0.11	28.5		
10	2.850	0.14	28.7	-0.842	
12	2.919	0.18	28.9	-0.745	
14	2.979	0.20	27.8	-0.709	1.09
16	3.031	0.23	27.1	-0.647	
18	3.078	0.26	27.4	-0.582	
20	3.120	0.29	26.5	-0.540	

The fraction of the total time required for a complete evaluation of the Xα energy spent on general matrix algebra increases abruptly for small peptides, then plateaus at approximately 29% and finally decreases for larger peptides. This trend is consistent with the fact that larger matrices have a greater proportion of negligible elements. The smaller peptides cannot benefit from matrix sparsity as much as the larger peptides. Not surprisingly, this

portion of the code almost scales in a perfectly linear fashion by using the simple technique previously described to take advantage of matrix sparsity.

Analytical versus Numerical: Xα energy calculation

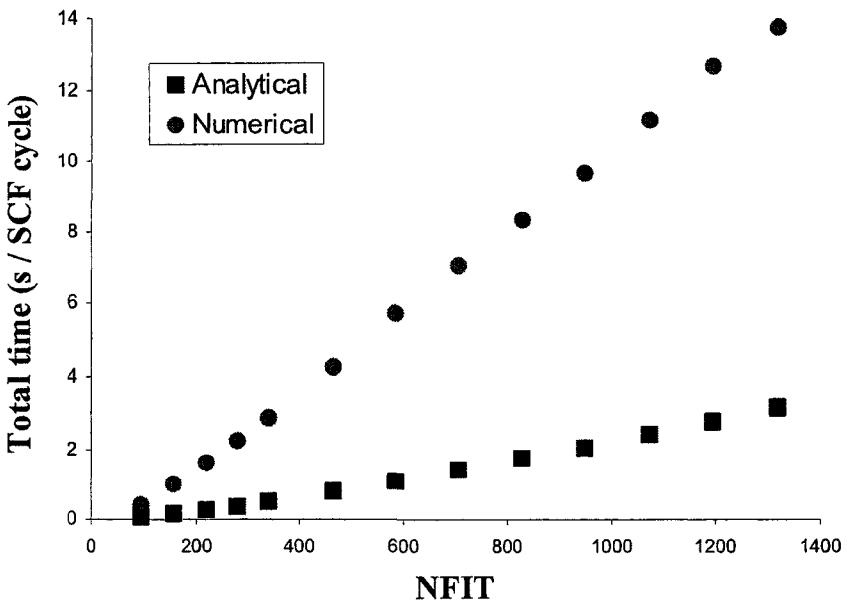
The overall scaling factor of the analytical algorithm depends on that of the four individual phases. A comparison between the overall performance of the analytical Xα algorithm against its numerical counterpart is in order. The same benchmark calculations were repeated using the numerical version of the Xα algorithm. Most of the numerical and analytical calculations converged in the same number of SCF cycles. Exceptions are noted for the peptides n = 0, 3, 4, 8, 10 and 12, for which the analytical calculation requires one extra SCF cycle compared with the numerical calculation. For that reason, comparisons between the analytical and numerical calculations will be made using the average time per SCF cycle (total time for a complete evaluation of the Xα energy divided by total number of SCF cycle) rather than the total time for a complete evaluation of the Xα energy.

The results in table 13 clearly indicate that the analytical algorithm is much more efficient than its numerical counterpart. The analytical calculation is approximately eight times faster for the smallest peptide and a little better than four times faster for the largest peptide.

Table 13. Data for the evaluation of the analytical and numerical Xα energy.

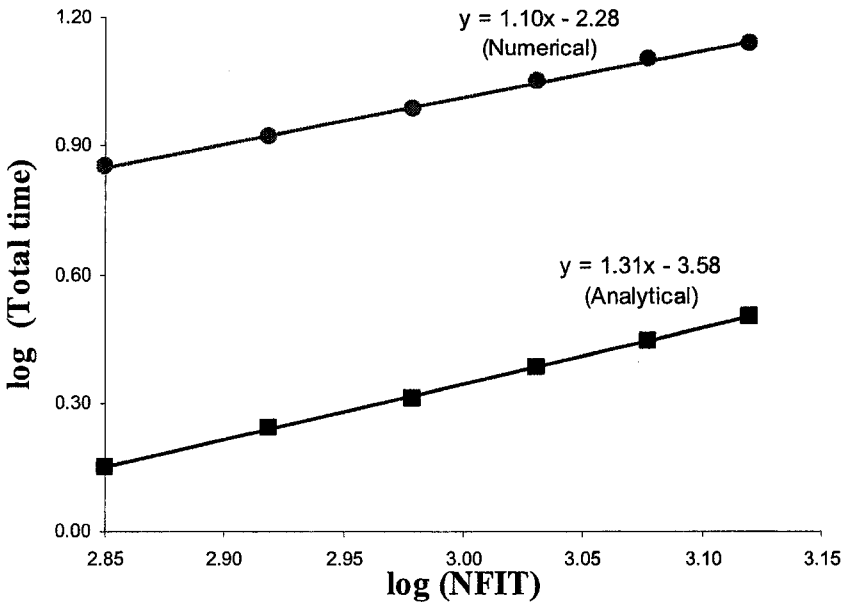
n	log NFIT	Total time (s / SCF)		log Total time		Scaling factor	
		analytical	numerical	analytical	numerical	analytical	numerical
0		0.05	0.41				
1		0.13	0.99				
2		0.23	1.63				
3	N/A	0.36	2.26		N/A		
4		0.49	2.89				
6		0.78	4.28				
8		1.08	5.72				
10	2.850	1.41	7.06	0.149	0.849		
12	2.919	1.75	8.30	0.243	0.919		
14	2.979	2.06	9.64	0.315	0.984	1.31	1.10
16	3.031	2.44	11.16	0.387	1.048		
18	3.078	2.80	12.66	0.447	1.102		
20	3.120	3.19	13.76	0.503	1.139		

Figure 13. Time required for the Xα energy versus number of fitting functions.



As the peptides increase in size, the gap between the analytical and numerical algorithms narrows, indicating a better scaling behaviour for the numerical algorithm. The overall scaling factor for the analytical algorithm is 1.31 compared to 1.10 for the numerical algorithm. Considering that the numerical method is well established, it is hoped that further work on the analytical approach can be done to further reduce the difference in the scaling behaviour.

Figure 14. Scaling factors for the analytical and numerical Xα energy.



Unfortunately, the overall scaling factor obtained for the analytical algorithm is valid for this range of sizes only. As it was mentioned earlier, the overall scaling factor of an algorithm is only as good as that of its poorest phase. In our case, the matrix inversions phase has a scaling factor of 1.80. If we were to go beyond peptides of 20 units, the fraction of the total time spent on matrix inversions would constantly grow and eventually dominate the total time. As more and more time is spent on matrix inversions, the overall

scaling factor will approach 1.80. However, even for the largest peptide of 20 glycine units, only 15% of the time required for a complete evaluation of the Xα energy is spent on matrix inversions. The domination by matrix inversions will not occur until exceedingly large systems are studied, well beyond the size of the peptides studied here. Other considerations such as memory limitations could arise before the size of the system reached the point where total time is dominated by matrix inversions.

Analytical versus Numerical: Xα energy gradient calculation

The evaluation of the Xα energy gradient in the analytical algorithm is very simple. With the expansion coefficients of X and Y already determined from the energy calculation, the evaluation of the Xα energy gradient simply involves summations of a new set of two-center and three-center overlap integrals, as demonstrated by equation 2.24. After each optimization step, the positions of the nuclei and their associated fitting functions change. The overlap integrals of equation 2.24 obviously depend on these positions. There is no sense in storing the integrals because the overlap between two given functions will not remain constant throughout the optimization process. For that reason, the overlap integrals are revaluated during each optimization cycle. Nonetheless, as was observed in the setup phase of the Xα energy calculation, the evaluation of overlap integrals is fast and scales well.

The evaluation of the Xα energy gradient itself in the numerical algorithm is equally simple. As was the case for the numerical Xα energy, the numerical Xα energy gradient simply involves the summation of contributions from each grid point. However, additional forces result from the grid. These forces are small compared to other forces but must be considered to obtain the exact energy gradient. Additionally, during each optimization step, the grid points are displaced with the atoms. The positions and weights of the grid points must be adjusted accordingly. Adjustments to the grid are time consuming but necessary. As such, an indirect advantage of using the analytical algorithm is the opportunity to completely remove the grid and rid the calculation of these time consuming calculations.

In the comparison between the performances of the analytical and numerical Xα energy gradient, the time spent on grid-related operations is added to the actual time spent on the numerical energy gradient. It is only fair to proceed in this manner since the analytical algorithm is not subjected to the costly grid-related adjustments necessary during the evaluation of the numerical energy gradient. In order to test the efficiency and scaling behaviour of the evaluation of the analytical and numerical Xα energy gradient, the time required for a single evaluation of the Xα energy gradient is measured for the same previously used set of peptides.

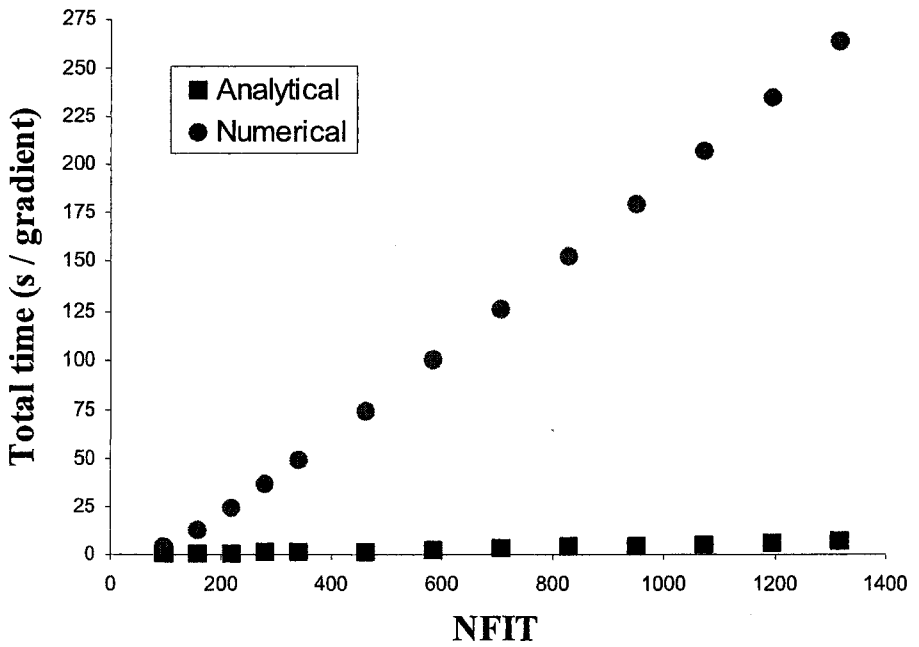
Table 14. Data for the evaluation of the analytical and numerical Xα energy gradient.

n	log NFIT	Time (s / grad)		log Time		Scaling factor	
		analytical	numerical	analytical	numerical	analytical	numerical
0		0.1	3.7				
1		0.2	12.8				
2		0.4	24.0				
3	N/A	0.6	36.0		N/A		
4		0.8	48.5				
6		1.3	73.8				
8		1.9	99.4				
10	2.850	2.6	125.5	0.415	2.099		
12	2.919	3.4	152.5	0.531	2.183		
14	2.979	4.2	179.5	0.623	2.254	1.61	1.19
16	3.031	5.1	206.7	0.708	2.315		
18	3.078	6.1	235.0	0.785	2.371		
20	3.120	7.1	263.1	0.851	2.420		

The analytical algorithm is remarkably more efficient than the numerical algorithm for the evaluation of the Xα energy gradient. For the smaller peptides, the analytical algorithm is approximately 60 times faster than the numerical algorithm. The speed advantage remains

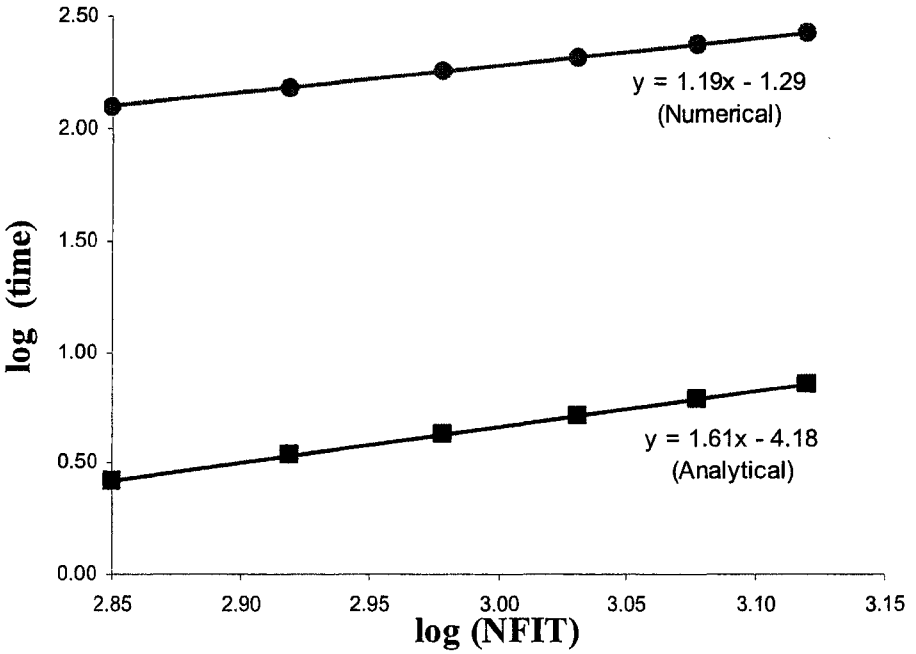
throughout the benchmark calculations but steadily diminishes as the size of the peptides increases. However, even for the largest peptide considered, the analytical algorithm is still 37 times faster than the numerical algorithm. For the largest peptide, a little over four minutes by optimization step would be saved by using the analytical algorithm rather than the numerical algorithm.

Figure 15. Time required for the Xα energy gradient versus number of fitting functions.



Scaling factors of 1.19 and 1.61 are measured for the evaluation of the numerical and analytical energy gradient respectively. Although more efficient, the evaluation of the analytical energy gradient does not scale as well as the numerical equivalent. Still, with the difference seen here between the two scaling factors, the analytical evaluation of the energy gradient should remain the more efficient option for systems that are well beyond the size of the bigger systems commonly studied with DFT.

Figure 16. Scaling factors for the analytical and numerical Xα energy gradient.



2.7. Application within a QM/QM framework

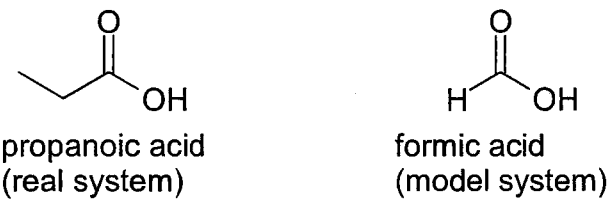
The analytical Xα functional is fast and can easily be adjusted (the parameter α) to better suit the needs of a particular type of calculation. The speed and versatility of the analytical Xα functional make it an interesting candidate to accompany various sophisticated quantum mechanical methods in a hybrid quantum mechanical / quantum mechanical (QM/QM) framework.

Hybrid methods are generally used for calculations on systems whose size is such that is it impossible to study them with a sophisticated high level of theory. One popular hybrid method is the ONIOM (Own N-layer Integrated molecular Orbital molecular Mechanics) method developed by Morokuma[49]. In ONIOM calculations, the "critical" area of a large system is identified as the model system. This critical area can be anything from a solvated

compound within a solvent, the active site of an enzyme, the surface of a metal or simply a functional group within an organic molecule. An expensive QM calculation (referred thereafter as the high level of theory) is performed on the model system, while the entire system is studied with a cheaper QM calculation (referred thereafter as the low level of theory). As the name of the method implies, the calculation can be divided into any number of layers. Our discussion will focus on two layers (high level / low level and model system / real system). Three layers (high level / mid level / low level and small model system / intermediate model system / real system) are common, with the outer most layer usually being semi-empirical or molecular mechanics[50]. Any combination of methods can be used. The difference between the levels of theory can be severe, such as a sophisticated DFT method and a semi-empirical method, or as subtle as a difference in orbital basis sets alone. As with all hybrid schemes, one wishes to accurately quantify the important interactions of the model system using the expensive high level of theory and hopes that the cheap low level of theory is sufficient to account for interactions outside of the model system.

For example, formic acid would be a logical choice for the model system of any carboxylic acid. The acid part (-COOH) of the carboxylic acid is where most of the computing efforts should be concentrated. For most applications, there would be no purpose in modelling the carbon chain with a very sophisticated level of theory since the essential chemistry will happen at the carboxylic acid end of the chain.

Figure 17. A real system and its model system.



A two-layer ONIOM calculation involves three distinct individual calculations: a high level calculation on the model system, a low level calculation on the model system and a low

level calculation on the real system. Significant time savings occur since no high level calculation is performed on the real system. If such a calculation was possible, or practical, then there would be no need to use the ONIOM method, as the objective of ONIOM is to approximate this otherwise unattainable result. The final ONIOM energy is calculated from the energies of the three individual calculations.

$$E_{ONIOM} = E_{high}^{model} + \left(E_{low}^{real} - E_{low}^{model}\right) \tag{2.64}$$

The ONIOM energy approximates the real energy at the high level of theory by calculating the energy of the model system at the high level of theory and adding a correction for interactions outside of the model system, at the low level of theory. If the same level of theory was used for the high and low levels, then it follows from equation 2.64 that the ONIOM result would be the same as that of a standard calculation on the entire system with that given level of theory. The ONIOM result will converge to the high level calculation on the real system as the size of the model increases and the low level of theory improves. Equation 2.64 is not limited to the calculation of electronic energies. Any given property, for example the NMR chemical shielding, can be calculated with the ONIOM method[51].

The quality of a low level method within ONIOM calculations is not necessarily related to its inherent stand-alone quality. The role of the low level method is not to be accurate on its own, but rather to be able to mimic the high level method. The S-value test is generally a good indicator of whether or not the chosen combination of methods will be successful for a given ONIOM calculation[52]. The S-value is simply the difference between the energy of the real system and the model system for a given level of theory.

$$S_{level} = E_{level}^{real} - E_{level}^{model} \tag{2.65}$$

If the high and low levels of theory have similar S-values, then the quality of the ONIOM approximation is expected to be good. In a perfect situation, the high and low levels of

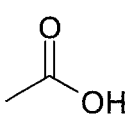
theory have identical S-values and the ONIOM result is exactly the same as the high level of theory on the real system.

The variable α parameter of the analytical Xα functional can easily be parameterized to minimize the difference between the Xα S-value and that of a higher level of theory. To demonstrate the effectiveness of a parameterized Xα functional within ONIOM calculations, the gas phase deprotonation energies of various carboxylic acids were calculated. The deprotonation energy of a carboxylic acid is equal to the variation of electronic energy for the following chemical reaction.

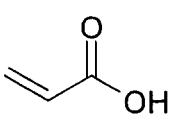
$$XCOOH_{(g)} \rightarrow H^{+}_{(g)} + XCOO^{-}_{(g)} \qquad (2.66)$$

By definition, the electronic energy of a proton in the gas phase is equal to zero since the proton is stationary and neither attracted by electrons or repulsed by other nuclei. The gas phase deprotonation energy is simply equal to the electronic energy of the conjugate base minus the electronic energy of the carboxylic acid. Three carboxylic acids were chosen for the parameterization process: acetic acid, acrylic acid and benzoic acid. For all the reactions considered in the following ONIOM calculations, the model system of the carboxylic acid is formic acid and the model system of the conjugate base is the formate anion. The high level of theory is always B3LYP/6-311++G(3df,2pd).

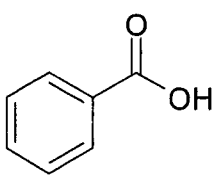
Figure 18. Carboxylic acids used for the parameterization of α.



acetic acid



acrylic acid



benzoic acid

As a first step, the B3LYP/6-311++G(3df,2pd) S-value was evaluated for the three carboxylic acids of the parameterization set. The S-value for a given acid and a given level

of theory is simply the difference between the deprotonation energy of that acid and the deprotonation energy of formic acid (the model system).

$$S_{acid}^{level} = \Delta E_{deprot\ (acid)}^{level} - \Delta E_{deprot\ (formic)}^{level} \tag{2.67}$$

The objective of the parameterization process is to determine the atom-dependant value of α for hydrogen, carbon and oxygen that will minimize the difference between the S-value of B3LYP/6-311++G(3df,2pd) and the S-value of Xα/STO-3G for the three carboxylic acid of the parameterization set.

$$\sum_{i=1}^{3} \left| S_i^{B3LYP/6-311++G(3df,2pd)} - S_i^{X\alpha/STO-3G} \right| = \min \tag{2.68}$$

The entire parameterization process is quick. Although several hundreds individual Xα calculations are necessary, only four deprotonation energies at the B3LYP/6-311++G(3df,2pd) level are required during the entire parameterization process. Xα calculations on acetic acid/acetate and acrylic acid/acrylate take less than two seconds while Xα calculations on benzoic acid/benzoate take approximately eight seconds. The optimized parameters obtained after minimization are α=0.701 for hydrogen, α=0.638 for carbon and α=0.850 for oxygen.

Table 15. S-Value for the three acids in the parameterization set.

acid	B3LYP/6-311++G(3df,2pd)		Xα/STO-3G [a]	
	ΔE (kcal/mol) [b]	S-value	ΔE (kcal/mol) [b]	S-value
formic acid (model)	349.60	N/A	420.23	N/A
acetic acid	353.91	4.30	424.34	4.11
acrylic acid	350.70	1.10	422.01	1.78
benzoic acid	347.12	-2.48	417.29	-2.94

[a] $\alpha_H = 0.701$, $\alpha_C = 0.638$, $\alpha_O = 0.850$

[b] as per the reaction depicted in equation 2.66

Once the optimal values for the α parameter of hydrogen, carbon and oxygen were obtained, we proceeded to calculate the deprotonation energy of seven other carboxylic acids (propanoic acid, 2-butenoic acid, sorbic acid, 2-methylbenzoic acid, 3-methylbenzoic acid, 4-methylbenzoic acid and cyclohexanoic acid) with the same ONIOM setup. The ONIOM deprotonation energy is compared to the true B3LYP/6-311++G(3df,2pd) for each of the ten carboxylic acids and the root mean square deviation between the ONIOM results and the true results is calculated. The root mean square deviation is calculated with the following equation.

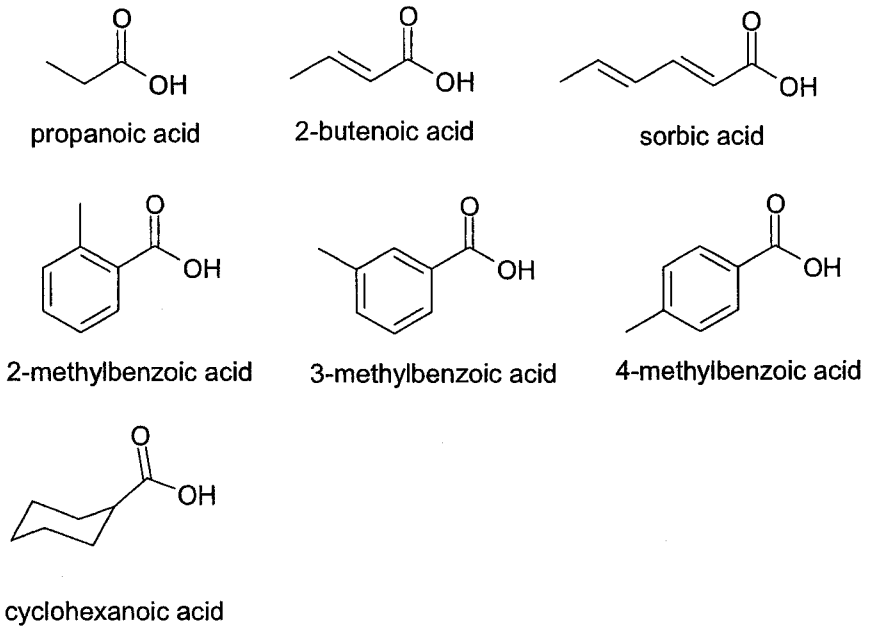
$$RMSD = \sqrt{\frac{\sum_{i=1}^{N} \left(\Delta E_{i,true} - \Delta E_{i,ONIOM}\right)^2}{N}} \qquad (2.69)$$

The calculated root mean square deviation for the entire set of ten carboxylic acids is 0.39 kcal/mol. The root mean square deviation for the parameterization set only (acetic, acrylic, benzoic) is 0.49 kcal/mol. The fact that the root mean square deviation for the parameterization set is larger than that of the entire set reflects the good transferability of the α parameters to carboxylic acids that were not included in the parameterization process.

Table 16. Accuracy of ONIOM calculations with Xα/STO-3G as low level

acid	ΔE (kcal/mol)	
	B3LYP/6-311++G(3df,2pd)	ONIOM (ΔΔE)
acetic acid	353.91	353.71 (-0.20)
acrylic acid	350.70	351.38 (+0.68)
benzoic acid	347.12	346.66 (-0.45)
propanoic acid	353.68	353.76 (+0.08)
2-butenoic acid	353.26	353.67 (+0.41)
sorbic acid	351.84	351.97 (+0.13)
2-methylbenzoic acid	347.93	348.02 (+0.09)
3-methylbenzoic acid	347.95	347.46 (-0.49)
4-methylbenzoic acid	348.43	347.82 (-0.61)
cyclohexanoic acid	351.74	351.57 (-0.17)

Figure 19. Additional carboxylic acids used to test the optimized α parameters.



In order to measure the effectiveness of our Xα functional in this particular application, the same series of ONIOM calculations were repeated using B3LYP with various orbital basis sets as the low level of theory. Table 17 summarizes how the B3LYP low levels performed by listing the root mean square deviation between the ONIOM deprotonation energies and the pure B3LYP/6-311++G(3df,2pd) deprotonation energies of the same set of ten carboxylic acids.

Table 17. Accuracy of ONIOM calculations with various low level methods.

Low level method	Root mean square deviation (kcal/mol)
B3LYP/STO-3G	10.80
B3LYP/3-21G	6.39
B3LYP/3-21+G	2.43
B3LYP/6-31+G	2.67
B3LYP/6-31+G(d)	0.74
B3LYP/6-31+G(d,p)	0.54
B3LYP/6-311+G(d)	0.72
Xα/STO-3G	**0.39**
B3LYP/6-311+G(d,p)	0.23
B3LYP/6-311++G(d,p)	0.29
B3LYP/6-311++G(2d,p)	0.11
B3LYP/6-311++G(3df,2pd)	0

Surprisingly, the analytical Xα/STO-3G functional outperforms many of the B3LYP low levels of theory. One would never expect Xα/STO-3G to outperform B3LYP, regardless of the orbital basis set, as a low level of theory whose role is to mimic the B3LYP/6-311++G(3df,2pd). B3LYP with basis sets that do not include polarization functions perform particularly poorly with root mean square deviations between 6 and 28 times worse than Xα/STO-3G. A large and expensive orbital basis set, 6-311+G(d,p), is required to finally surpass the Xα/STO-3G functional in terms of accuracy when coupled with the

B3LYP/6-311++G(3df,2pd) as a high level of theory in these ONIOM calculations. Obviously, the computational demands of Xα/STO-3G are negligible compared to B3LYP/6-311+G(d,p). The systems considered in this small scale application are quite small. If larger acids were implicated, such as for example stearic acid ($C_{18}H_{36}O_2$), the benefits of using Xα/STO-3G instead of B3LYP with cheap basis sets as a low level method would be even greater.

2.8. Afterthoughts

The three objectives that were initially set for our implementation of the analytical Xα functional have been met. The analytical version of the algorithm was expected to match the numerical version in terms of accuracy, outperform the numerical version in terms of efficiency and rival the numerical version in terms of scaling behaviour.

As far as accuracy is concerned, the standard grid used to numerically evaluate the Xα energy can be substituted by a minimal set of fitting Gaussian functions to analytically calculate the Xα energy. Our *PS+* fitting basis set, made of atom-centered *s* functions, atom-centered *p* functions and bond-centered *s* functions, reproduces the electronic energy of 40 molecules calculated numerically with an average absolute deviation of 2.5 kcal/mol. The variation of electronic energy for 42 chemical reactions calculated numerically was reproduced with an average absolute deviation of 1.4 kcal/mol, again using the *PS+* fitting basis set. Even if the *PS+* fitting basis set is incomplete and leads to calculations that may not be as accurate as the ones obtained numerically, chances are the premium was not on accuracy if the Xα functional was chosen for a particular application.

For the medium to large systems that were considered in this study, our implementation of the analytical Xα functional easily outperforms the traditional numerical implementation in terms of efficiency, being approximately four to eight times faster for the evaluation of the Xα energy and 40 to 60 times faster for the evaluation of the Xα energy gradient. The

scaling factors of the analytical Xα algorithm are 1.31 and 1.61 for the evaluation of the energy and energy gradient respectively. Although linear scaling was not achieved, the scaling behaviour of the analytical algorithm is vastly improved considering its inherent scaling factor of three. The scaling factors of the analytical algorithm are close enough to the scaling factors of 1.10 and 1.19 obtained for the numerical version of the algorithm to conclude that the analytical Xα algorithm will retain its speed advantage over the numerical algorithm for systems of sizes that are well beyond the ones considered in this project.

Additionally, the adaptive nature of the analytical Xα algorithm was demonstrated in a series of ONIOM calculations involving the deprotonation energy of various carboxylic acids in the gas phase. By allowing α to take a different value for hydrogen, carbon and oxygen atoms, the analytical Xα algorithm played the role of low level of theory to the high level of theory B3LYP/6-311++G(3df,2pd) with surprising success. In fact, the quality of the analytical Xα algorithm as a low level of theory in the ONIOM application surpassed that of B3LYP/6-311+G(d).

For all these reasons, the choice should be clear between the analytical and numerical Xα method. The advantages gained by completely removing the grid significantly outweigh the disadvantages, especially for applications requiring the energy gradient. Our implementation of the analytical Xα method is ideal to study large systems at the ab initio level of theory on its own or coupled with more expensive methods in a QM/QM framework. Future work aiming to improve the performance of the analytical algorithm should focus primarily on the matrix inversion part of the code, either by working on the inversions themselves or by reducing the actual number of inversions required in a given calculation.

Chapter 3 – Adaptive Importance Function in Monte Carlo Simulations

Typical single point calculations in computational chemistry yield the electronic energy of a molecular system in a particular configuration at absolute zero temperature. The thermal energy (translation, rotation, vibration) and entropy must be considered for systems at a finite temperature. Although the electronic energy can be very useful, free energy is the one variable a chemist ultimately needs to make decisions about real life occurrences from calculations. These decisions can be as simple as to whether or not a reaction is spontaneous at a given temperature, or as complex as to compare how strongly different drugs bind to a receptor. The estimation of the partition function of a molecular system is necessary to make the transition from electronic energy to meaningful thermodynamic quantities. From statistical mechanics, all the relevant thermochemistry such as enthalpy, entropy, free energy or heat capacity can be calculated with the partition function. Thermodynamic quantities of molecular systems are usually calculated in one of two ways.

In a first approach, the partition function is broken down into fragments (translation, rotation, vibration, electronic) that are evaluated individually. The translational and rotational partition functions are approximated with simple integrals provided the high temperature limit is valid. The vibrational partition function is approximated from the frequencies of all the vibrational degrees of freedom. Vibrational frequencies are obtained by calculating the second derivative of the electronic energy with respect to the nuclear cartesian coordinates (the Hessian matrix). All the vibrational modes are usually treated as harmonic vibrations. The electronic partition function is easily calculated from the output of a standard single point calculation (energy levels and their occupancy). The global partition function is equal to the product of the four individual partition functions (translational, rotational, vibrational and electronic). Unfortunately, frequency calculations are valid for stationary points (minima or saddle points) on the potential energy surface only and are quite expensive for large systems. Additionally, the harmonic approximation breaks down when weak interactions such as hydrogen bonds or other weakly bounded complexes are involved. This approach is ideally suited to study small, rigid molecule in fixed configurations.

A second approach focuses on molecular simulations to estimate the partition function directly or other thermodynamic quantities. In contrast to the previous approach, molecular simulations generate multiple configurations of the same molecular system and average the results obtained from individual calculations. Rather than studying the molecular system within a single configuration (either sitting in a minimum or on a saddle point), the molecular system is allowed to move about the potential energy surface. Two different methods of generating configurations, each with its advantages and shortcomings, are routinely utilized in chemistry: molecular dynamics and Monte Carlo. In molecular dynamics simulations, the nuclei of the molecular system are given initial positions and velocities. The trajectory of the system is then followed by integrating Newton's equation of motion in small, discrete steps. In Monte Carlo simulations, rather than following a deterministic trajectory, a chain of configurations is generated by randomly perturbing the positions of the nuclei. Whether the simulation is of the molecular dynamics or the Monte Carlo variety, results from standard single point calculations performed on the individual configurations are stored and averaged at the end of the simulation.

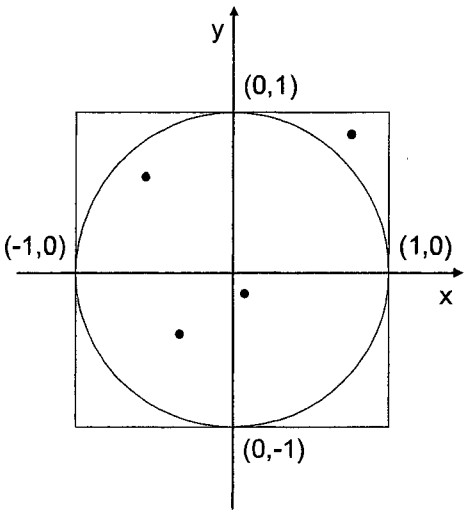
The objective of these molecular simulations is to properly sample the potential energy surface in an effort to better understand the movement of the system on the surface when thermal energy is available. This chapter will focus primarily on Monte Carlo simulations applied to molecular systems. Issues with the efficiency of sampling the potential energy surface and the development of a novel adaptive importance function to alleviate these issues will be discussed. The performance of the adaptive importance function will be tested on a gas phase chemical reaction (ring opening of cyclobutene) in a proof of concept application.

3.1. Monte Carlo simulations applied to chemistry

Molecular simulations, whether they be at the molecular mechanics, semi-empirical or ab initio level of theory, are increasingly popular for the study of molecular systems. The various types of simulations and the range of their application are too wide to extensively review in this document but can be found elsewhere in the literature[53]. In a general sense, Monte Carlo (MC) simulations refer to a type of computational algorithm that relies on random numbers to reach a solution to a given problem. They are most useful to solve a variety of problems for which a deterministic solution is either impossible or impractical to compute. The development of the MC method applied to the physical sciences can be traced back to 1949 and the work of Metropolis[54]. Metropolis presented a statistical approach to the solution of differential equations. In all MC simulations, random inputs are generated inside a predefined domain. Deterministic computations are performed on the randomly generated inputs and the individual results are combined to produce the final result.

For example, consider the classic demonstration of approximating the value of π with a MC simulation. A circle of unit radius, centered at the origin of a cartesian system of coordinates, is inscribed inside a square with sides of two units. In this case, the domain is defined as the area inside the square.

Figure 20. Approximation of π with a Monte Carlo simulation.



Using a random number generator, data points are positioned inside the domain ($x,y \in [1,1]$). The manner in which random numbers or pseudo-random numbers are generated will be addressed later. The value of π can be determined by evaluating the ratio of the area of the circle over the area of the square.

$$\frac{A_{circle}}{A_{square}} = \frac{\pi(1)^2}{2x2} = \frac{\pi}{4} \tag{3.1}$$

The ratio of areas can be estimated by the ratio of points that fall within the two geometrical shapes. All the points will obviously fall within the square. On the other hand, only a fraction of the points will fall within the circle. Points whose distance from the origin is less than one ($x^2+y^2<1$) fall inside the circle.

$$\pi = 4\left(\frac{A_{circle}}{A_{square}}\right) \approx 4\left(\frac{\#\,\text{points inside circle}}{\text{total}\,\#\,\text{points}}\right) \tag{3.2}$$

For this approach to be successful, the points must be generated uniformly throughout the square in an unbiased fashion. If the chain of random numbers is truly unbiased then the approximation of π will converge toward the real value of π as the number of points increases. This method of estimating π is quite inefficient but it clearly demonstrates how the statistical approach envisioned by Metropolis can be applied to problems that are not related to statistics at first view.

This simple application translates into a useful simulation technique in chemistry. It is generally acceptable to assume that a small, rigid molecular system simply sits in a particular energy well on its potential energy surface. The absolute minimum is usually much lower in energy than other minima and minima are separated by large energy barriers. Even with a given amount of thermal energy, the shape of the potential energy surface of such a system is restricting the movement along the surface. The probability of finding the molecular system in the absolute energy minimum well is almost 100%. A simple single point calculation is sufficient to accurately study the behaviour of the small, rigid molecular system. Such is not the case for all molecular systems. For example, the potential energy surface of large, floppy molecular systems or condensed phases is littered with numerous energy wells that are very close to each other energetically and separated by low energy barriers. By acquiring thermal energy, such systems will flow from one energy well to the next with ease. A single point calculation is no longer sufficient since the probability of finding the molecular system at any given point on the potential energy surface is distributed among the energy wells. The potential energy surface must be sampled to accurately study the behaviour of the molecular system since it is no longer acceptable to think of the system as simply sitting in a particular well.

Statistical mechanics makes use of probability theory to calculate the average of a certain property[55]. Rather than proceeding with an in-depth discussion of statistical mechanics at this point, the key concepts relating to the study of a potential energy surface will be quickly covered. The average of any thermodynamic property, $< A >$, depends on how that property is distributed among available states and the probability of residing into the different states. If the property is distributed into discrete states then the average value is

equal to the sum of the value of the property in a given state, A_i, multiplied by the probability of residing into this state, P_i, for all states. If, on the other hand, the property is distributed continuously with respect to one or many variables then the average value is obtained by an integral rather than a summation.

$$\langle A \rangle = \sum_i A_i P_i \tag{3.3}$$

$$\langle A \rangle = \int A(x) P(x)\, dx \tag{3.4}$$

The potential or electronic energy of a molecular system containing N atoms is a continuous function of 3N nuclear coordinates. Since the energy is a continuous, multivariable function, its average is calculated from an integral over all 3N variables.

$$\langle E \rangle = \int\int \ldots \int E(r_1, r_2, \ldots, r_{3N})\, P(r_1, r_2, \ldots, r_{3N})\, dr_1 dr_2 \ldots dr_{3N} \tag{3.5}$$

This integral cannot be evaluated analytically or by standard numerical techniques such as the trapezoid rule or Simpson's rule because of its high dimensionality (known as the curse of dimensionality). Even for a small system containing as little as 10 atoms, the integral would be of dimension 30. Consider a situation where each dimension is divided into 100 equal parts in order to perform a standard numerical integration with the trapezoid rule or Simpson's rule. In total, 100^{30} individual calculations would be required to numerically estimate the integral with substandard accuracy due to the rather coarse dividing of the dimensions. Performing such a large number of calculations in a reasonable amount of time is impossible, even if each individual calculation only takes a fraction of a second.

MC provides an alternative way of numerically evaluating integrals that cannot be evaluated otherwise. Integrals are nothing more than areas under curves. For example, equation 3.2 can be rewritten in the form of a ratio of integrals.

$$\frac{\pi}{4} = \frac{A_{circle}}{A_{square}} = \frac{4\int_0^1 \sqrt{1-x^2}\,dx}{4\int_0^1 \sqrt[\infty]{1-x^{\infty}}\,dx} \tag{3.6}$$

Counting the proportion of points that fall within the circle is equivalent to approximating the ratio of integrals of the cartesian function of a circle over the cartesian function of a square. In contrast to other numerical integration techniques, the efficiency of the MC integration technique does not depend on the dimension of the integral to be evaluated. For that reason, MC is better suited than other numerical techniques to evaluate integrals with high dimensionality such as equation 3.5. At sufficiently high temperatures, the probability of finding the molecular system at a particular point on the potential energy surface is obtained from a Boltzmann distribution.

$$P(r_1, r_2, \ldots, r_{3N}) = \frac{e^{-E(r_1, r_2, \ldots, r_{3N})/k_B T}}{\int\int \ldots \int e^{-E(r_1, r_2, \ldots, r_{3N})/k_B T} dr_1 dr_2 \ldots dr_{3N}} \tag{3.7}$$

The denominator of equation 3.7 acts as a normalization constant since the probability of finding the molecular system throughout the potential energy surface must be equal to one. It also represents the partition function for a continuous distribution. As expected, lower energy configurations are more probable than higher energy configurations. By inserting the Boltzmann distribution function into the expression for the average energy, the problem of evaluating the average energy is rewritten as a ratio of integrals. The average energy can be seen as a weighted average in which lower energy configurations have more weight than higher energy configurations.

$$\langle E \rangle = \frac{\int\int \ldots \int E(r_1, r_2, \ldots, r_{3N})\, e^{-E(r_1, r_2, \ldots, r_{3N})/k_B T} dr_1 dr_2 \ldots dr_{3N}}{\int\int \ldots \int e^{-E(r_1, r_2, \ldots, r_{3N})/k_B T} dr_1 dr_2 \ldots dr_{3N}} \tag{3.8}$$

A MC simulation can numerically approximate this ratio of integrals, just as it can numerically approximate the ratio of areas of the circle over the square. A chain of NCONF molecular configurations can be generated randomly with random number generators and their energy evaluated. After each individual calculation, a contribution to the numerator and denominator of equation 3.8 is added.

$$\langle E \rangle \approx \frac{\sum_{i=1}^{NCONF} E_i(r_1, r_2, \ldots, r_{3N}) e^{-E_i(r_1, r_2, \ldots, r_{3N})/k_B T}}{\sum_{i=1}^{NCONF} e^{-E_i(r_1, r_2, \ldots, r_{3N})/k_B T}} \tag{3.9}$$

Depending on the potential energy surface of the system, any number from thousands to millions of individual configurations may be required to reach a respectable level of accuracy. No matter how many configurations are required, that number is still much smaller than what would be required for a traditional numerical integration (100^{3N} individual calculations if each degree of freedom was divided into 100 equal parts). Theoretically, an infinitely long simulation will yield the exact average energy.

$$\langle E \rangle = \lim_{NCONF \to \infty} \frac{\sum_{i=1}^{NCONF} E_i(r_1, r_2, \ldots, r_{3N}) e^{-E_i(r_1, r_2, \ldots, r_{3N})/k_B T}}{\sum_{i=1}^{NCONF} e^{-E_i(r_1, r_2, \ldots, r_{3N})/k_B T}} \tag{3.10}$$

In practice, simulations are of finite length and the average energy can only be approximated. Unfortunately, generating truly random configurations will not allow the MC numerical approximation of the average energy to converge in a reasonable number of configurations. For example, the odds of generating a respectable configuration for a molecular system from a random assortment of nuclei are practically nonexistent. The energy of a molecular system is extremely sensitive to bond lengths, bond angles and

torsion angles. The same can be said of the simulation of a liquid phase in which molecules are replaced with Lennard-Jones spheres. Chances are that by randomly positioning the individual spheres, numerous spheres will be too close to each other, driving the energy upwards. High energy configurations, because of their low probabilities, do not contribute as much as low energy configurations in the weighted average. It does not make sense to simply generate configurations that for the most part will be wasted because of a negligible contribution due to high energy and low probability. Ideally, the sampling of the potential energy surface should focus primarily on the lower energy configurations.

Metropolis adapted his original MC algorithm to better suit the needs of molecular simulations. Rather than simply choosing configurations at random and calculating the weighted average, Metropolis proposed to select configurations from a Boltzmann probability distribution and then calculate a simple arithmetic average[56]. This technique is also recognized as importance sampling or Metropolis-MC. Nowadays, it is implied that MC simulations of molecular systems use importance sampling in one way or another. Instead of randomly positioning the nuclei of a molecular system, new configurations are generated by slightly perturbing an existing, reasonable configuration. This is easily achieved by adding small displacements to the nuclear coordinates in order to obtain a new configuration. The new nuclear coordinates are calculated using a random number between zero and one (χ) and a predetermined maximal displacement (δ_{max}).

$$A^{new}_{i=x,y,z} = A^{old}_{i=x,y,z} + (2\chi - 1)\delta_{max} \qquad (3.11)$$

In equation 3.11, the nuclear coordinates of atom A are updated from the same nuclear coordinates of the previous configuration. There is some flexibility concerning the manner in which random displacements are applied. For example, the position of all nuclear coordinates in all three directions can be altered each time the configuration is updated. Alternatively, a single nuclear coordinate or a group of coordinates, chosen at random or sequentially, could be altered to update the configuration. Regardless of how the configurations are updated, a trial configuration is accepted or rejected in such a way that a

correct Boltzmann distribution is obtained. The probability of accepting a trial configuration depends on the variation of energy with respect to the previous configuration and the temperature, which in MC simulations is a fixed parameter.

$$P_{i\to i+1} = \min\left\{1, e^{-\Delta E_{i\to i+1}/k_B T}\right\} \tag{3.12}$$

If the trial configuration is lower in energy than the previously accepted configuration then the exponential of equation 3.12 is larger than one. Downhill moves (in terms of energy) are always accepted since the probability of accepting the new configuration is equal to one. If, however, the trial configuration is higher in energy than the previously accepted configuration then the probability of accepting the new configuration is equal to the exponential of equation 3.12, which itself is positive and smaller than one. In order to decide whether or not the new configuration is accepted, the exponential is compared to a random number between zero and one. The new configuration is accepted if the exponential is larger than the random number and rejected otherwise. Slight uphill moves have a better chance of being accepted than more severe uphill moves. The exponential, and the probability of being accepted, very quickly decreases as the variation of energy with respect to the previously accepted configuration increases. A higher temperature promotes uphill moves by increasing the value of the exponential and the probability of being accepted.

The rate at which trial configurations are accepted depends on the aggressiveness of the random displacements of the nuclear coordinates. Small displacements will produce successive configurations that are very similar and increase the probability of accepting the trial configuration. With this conservative approach, the system will slowly sample the potential energy surface, requiring a larger number of configurations to properly sample the surface. On the other hand, large displacements increase the chance of generating a trial configuration that is higher in energy and less likely to be accepted. A more aggressive approach leads to more efficient sampling of the potential energy surface, but may lead to

long streaks of rejected configurations. Ideally, trial configurations should be accepted at a rate of approximately 50%, reaching a compromise of sort where the sampling efficiency is optimal. The acceptance rate is easily adjusted by modifying the maximal displacement allowed parameter, δ_{max}.

When using a chain of configurations generated with this importance sampling method, the average energy is calculated with a simple arithmetic average.

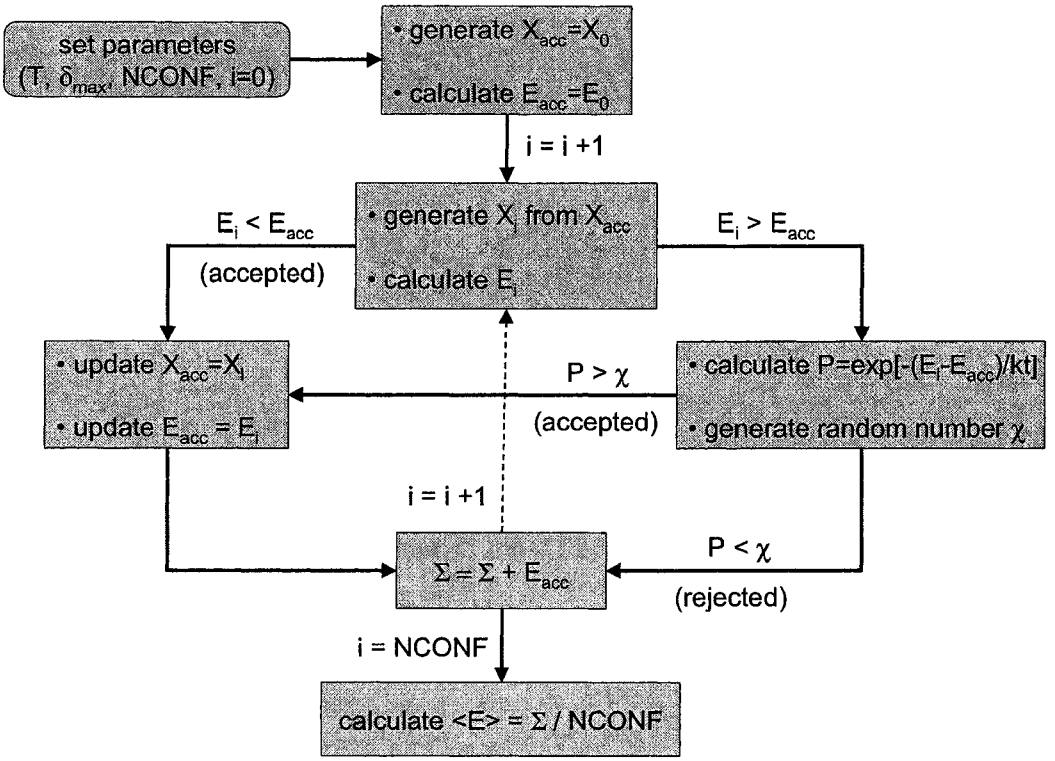
$$\langle E \rangle \approx \frac{\sum_{i=1}^{NCONF} E_i(r_1, r_2, \ldots, r_{3N})}{NCONF} \tag{3.13}$$

As was the case for a traditional MC simulation, the approximation of the average energy during a Metropolis-MC simulation will be exact in the limit of an infinitely long simulation.

$$\langle E \rangle = \lim_{NCONF \to \infty} \frac{\sum_{i=1}^{NCONF} E_i(r_1, r_2, \ldots, r_{3N})}{NCONF} \tag{3.14}$$

When a trial configuration is rejected, the energy of the previously accepted configuration is added instead of the energy of the rejected configuration. Figure 21 summarizes the essential components of a Metropolis-MC simulation aiming to evaluate the average energy of a molecular system (or the average of any property of any system).

Figure 21. Schematisation of a Metropolis-Monte Carlo simulation.



Initially, the parameters of the simulation are set: the temperature, the maximal displacement allowed and the total number of configurations. A first, reasonable, configuration is generated and assigned the tag of last accepted configuration. A trial configuration is generated with random displacements of the nuclear coordinates of the last accepted configuration. The energy of the trial configuration is calculated and compared with the energy of the last accepted configuration. If the variation of energy ($E_i - E_{acc}$) is negative then the trial configuration is automatically accepted. If the variation of energy is positive then the probability function is evaluated and compared to a random number. The trial configuration is accepted if the probability function is larger than the random number and rejected otherwise. When a trial configuration is accepted, it is labelled as the last accepted configuration. After each MC step, the energy of the last accepted configuration is added to the total and the cycle is repeated until the predetermined number of configurations is reached. The average energy is easily calculated afterwards. As is

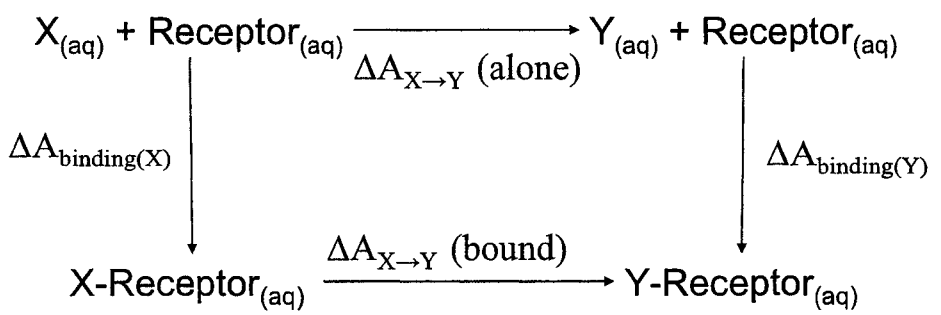
customary in molecular dynamics simulations, an equilibration period may be simulated prior to the actual accumulation of data in order to prevent the final result from being biased by the choice of the initial configuration.

Unfortunately, even with the importance sampling of Metropolis-MC, it is very difficult to accurately calculate the absolute free energy of a system due to the insufficient sampling of finite length simulations. However, a number of methods have been developed to calculate variations of free energy using molecular simulations. Chemists are more often interested in variations of free energy than absolute free energies for most applications, whether it would be to determine the activation free energy of a reaction, the spontaneity of a reaction, which of two molecules is more soluble in a given solvent or which of two drugs binds more strongly to a receptor. Free energy calculations applied to molecular and biochemical systems[57] are crucial and positively complement experimental work.

Free energy perturbation

Free energy perturbation[58] is a method often used to calculate the difference in free energy between two states, X and Y. These two states could be different molecular systems or simply a molecular system in different configurations. For example, X and Y could represent potential candidates for a drug. Suppose the better drug is the one that will bind more strongly with a receptor.

Figure 22. Thermodynamic cycle studied with free energy perturbation.



While it would be very difficult to measure $\Delta A_{binding(X)}$ or $\Delta A_{binding(Y)}$ in a simulation, free energy perturbation can effectively measure $\Delta A_{X \rightarrow Y(alone)}$ and $\Delta A_{X \rightarrow Y(bound)}$.

$$\Delta A_{binding(X)} + \Delta A_{X \rightarrow Y}(bound) - \Delta A_{binding(Y)} - \Delta A_{X \rightarrow Y}(alone) = 0 \qquad (3.15)$$

$$\Delta\Delta A_{binding} = \Delta A_{binding(Y)} - \Delta A_{binding(X)} = \Delta A_{X \rightarrow Y}(bound) - \Delta A_{X \rightarrow Y}(alone) \qquad (3.16)$$

Using the thermodynamic cycle, the difference in binding free energy is calculated from two free energy perturbation simulations. Separate free energy perturbation simulations can approximate the difference in free energy between the two drugs in solution or when bound to the receptor (horizontal paths of figure 22) and ultimately lead to the difference in binding free energy, $\Delta\Delta A_{binding}$. This simple example is just one of the various applications of free energy perturbation.

In the canonical NVT ensemble (closed system under constant volume and temperature), the Helmholtz free energy of a system is calculated from its partition function. If the kinetic energy component is neglected, the partition function is dependent on the nuclear coordinates alone. The kinetic energy contribution can be added independently afterwards.

$$A_X = -k_B T \ln Q_X = -k_B T \ln\left(\int\int ... \int e^{-E_X(r_1, r_2, ..., r_{3N})/k_B T} dr_1 dr_2 ... dr_{3N} \right) \qquad (3.17)$$

$$A_Y = -k_B T \ln Q_Y = -k_B T \ln\left(\int\int ... \int e^{-E_Y(r_1, r_2, ..., r_{3N})/k_B T} dr_1 dr_2 ... dr_{3N} \right) \qquad (3.18)$$

E_X and E_Y represent the potential energies obtained from a Hamiltonian describing states X and Y respectively. A direct evaluation of the absolute free energy of any state from a

partition function obtained via a molecular simulation is unfeasible. Proper sampling of the entire potential energy surface would be required to numerically approximate the integral of the partition function. The difference of free energy between states X and Y is more easily obtained from a simulation. Using simple algebra, the difference of free energy can be expressed with a similar equation.

$$\Delta A_{X\to Y} = -k_B T \ln\left(\int\int\ldots\int e^{-[E_Y(r_1,r_2,\ldots,r_{3N})-E_X(r_1,r_2,\ldots,r_{3N})]/k_B T} dr_1 dr_2 \ldots dr_{3N} \right) \quad (3.19)$$

A simple twist on the standard MC algorithm allows the evaluation of this integral with reasonable accuracy during a simulation of finite length. Configurations for state X are generated just as they would be during a standard MC simulation. The energy of each configuration for state X is calculated. However, prior to moving on to the next configuration, state Y replaces state X and the energy of state Y, in the same configuration as X, is also computed. By calculating the difference in energy between X and Y for each configuration obtained from a Boltzmann distribution, the integral is numerically approximated with the MC simulation. A free energy perturbation simulation is essentially the same as a standard MC simulation with the exception that two energy calculations are required for every configuration. The difference in free energy is calculated from a simple average of the difference in energy for each configuration.

$$\Delta A_{X\to Y} = -k_B T \ln\left\langle e^{-(E_Y - E_X)/k_B T} \right\rangle_X \quad (3.20)$$

The subscript X outside of the bracket signifies that the average is calculated with configurations obtained from the probability distribution of X via the molecular simulation. Unfortunately, the configurations that are generated will tend to favour X and bias the estimation of the difference in free energy by penalizing state Y unfairly. For that reason, the variation of free energy calculated with equation 3.20 is valid only for cases where

states X and Y are quite similar. A simple average of the difference in energy between X and Y can be used to approximate equation 3.20.

$$\Delta A_{X\to Y} \approx \langle E_Y - E_X \rangle_X \tag{3.21}$$

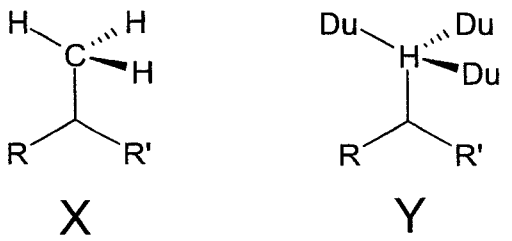
If a better approximation of equation 3.20 is required then a second-order term can be added to equation 3.21.

$$\Delta A_{X\to Y} \approx \langle E_Y - E_X \rangle_X - \frac{\left\langle \left[E_Y - E_X - \langle E_Y - E_X \rangle_X \right]^2 \right\rangle_X}{2k_B T} \tag{3.22}$$

For most applications, states X and Y are not similar enough to calculate the difference of free energy in a single simulation. The free energy perturbation simulation needs to be broken down into multiple windows using intermediate states between X and Y.

Using force fields, non-physical transformations where atoms are progressively morphed into different atoms are possible. For example, suppose the difference between the drug candidates X and Y is simply a methyl group substituted by a hydrogen atom. The carbon atom of the methyl group in X is replaced by a hydrogen atom in Y while the hydrogen atoms of the methyl group in X are replaced by dummy atoms in Y. Dummy atoms are in effect invisible, without a charge or Van der Waals parameters.

Figure 23. Potential candidates for a fictitious drug.



A carbon atom can be gradually morphed into a hydrogen atom while three hydrogen atoms are gradually disappearing. A coupling parameter, λ, is defined to connect states X (λ=0) and Y (λ=1) with any number of intermediate states. The hybrid Hamiltonian is a continuous function of the coupling parameter.

$$H(\lambda) = (1-\lambda)^{\gamma} H_X + \left[1-(1-\lambda)^{\gamma}\right] H_Y \tag{3.23}$$

Linear mixing (γ=1) is usually employed. However, $\gamma > 3$ is sometimes necessary for cases when atoms are completely disappearing during the simulation in order to obtain converging values for the variation of free energy between intermediate states[59]. The actual hybrid parameters involving atoms that are morphing are calculated just as the hybrid Hamiltonian. For example, the bond stretching parameter for the carbon-carbon bond morphing into a carbon-hydrogen bond is calculated with the following equation, where k_Y is the stretching parameter of a carbon-hydrogen bond and k_X is the stretching parameter of a carbon-carbon bond.

$$k(\lambda) = (1-\lambda)^{\gamma} k_X + \left[1-(1-\lambda)^{\gamma} k_Y\right] \tag{3.24}$$

If the pathway connecting states X and Y is divided into NINT intermediate states then the difference in free energy between X and Y will be the sum of NINT-1 differences obtained from individual, independent simulations.

$$\Delta A_{X\to Y} = -k_B T \sum_{k=1}^{NINT-1} \ln\left\langle e^{-[E(\lambda_{k+1})-E(\lambda_k)]/k_B T} \right\rangle_k \tag{3.25}$$

$$\Delta A_{X\to Y} \approx \sum_{k=1}^{NINT-1} \left\langle E(\lambda_{k+1}) - E(\lambda_k) \right\rangle_k \tag{3.26}$$

The width of the windows can remain constant throughout the simulation (for example λ = 0, 0.25, 0.5, 0.75 and 1) or be dynamically modified during the simulation[60]. For an optimal simulation, the windows should be narrower in areas where the potential energy as a function of the coupling parameter varies significantly (large $dE/d\lambda$) and should be wider otherwise.

Thermodynamic Integration

Another type of simulation, similar to free energy perturbation, used to calculate variations of free energy is called thermodynamic integration. Theoretically, the absolute free energy of a state λ is calculated from the partition function of this particular state.

$$A_\lambda = -k_B T \ln Q_\lambda = -k_B T \ln\left(\int\int \ldots \int e^{-E_\lambda(r_1, r_2, \ldots, r_{3N})/k_B T} dr_1 dr_2 \ldots dr_{3N} \right) \quad (3.27)$$

This equation can be differentiated with respect to the coupling parameter.

$$\frac{dA_\lambda}{d\lambda} = \frac{-k_B T \int\int \ldots \int e^{-E_\lambda(r_1, r_2, \ldots, r_{3N})/k_B T} \left(\frac{-1}{k_B T} \right) \left(\frac{\partial E_\lambda(r_1, r_2, \ldots, r_{3N})}{\partial \lambda} \right) dr_1 dr_2 \ldots dr_{3N}}{\int\int \ldots \int e^{-E_\lambda(r_1, r_2, \ldots, r_{3N})/k_B T} dr_1 dr_2 \ldots dr_{3N}} \quad (3.28)$$

After algebraic simplifications, the derivative of the free energy with respect to the coupling parameter yields a new integral, which itself can be numerically approximated by a MC simulation.

$$\frac{dA_\lambda}{d\lambda} = \int\int \ldots \int \frac{\partial E_\lambda(r_1, r_2, \ldots, r_{3N})}{\partial \lambda} dr_1 dr_2 \ldots dr_{3N} \approx \left\langle \frac{\partial E_\lambda(r_1, r_2, \ldots, r_{3N})}{\partial \lambda} \right\rangle_\lambda \quad (3.29)$$

Instead of explicitly calculating the difference of energy between two states for each configuration (as in free energy perturbation simulations), thermodynamic integration simulations evaluate the partial derivative of the energy with respect to the coupling parameter for each configuration. The variation of free energy is equal to the integral of this partial derivative from λ=0 to λ=1.

$$\Delta A_{X\to Y} \approx \int_{\lambda=0}^{\lambda=1} \left\langle \frac{\partial E_\lambda(r_1, r_2, \ldots, r_{3N})}{\partial \lambda} \right\rangle_\lambda d\lambda \qquad (3.30)$$

The integral of equation 3.30 is evaluated in one of two ways. In the slow-growth approach, the Hamiltonian is slightly modified after each configuration[61]. As the name implies, the Hamiltonian slowly changes to "continuously" connect the initial state to the final state. An increment of the coupling parameter is defined as the inverse of the number of configurations (NCONF) the simulation is scheduled to last. The actual value of λ for the i^{th} configuration of the simulation is easily calculated.

$$\lambda_i = i \times \delta\lambda = \frac{i}{NCONF} \qquad (3.31)$$

The partial derivative of the energy with respect to the coupling parameter is evaluated at each configuration and multiplied by $\delta\lambda$ to obtain the variation of free energy for that increment of λ. The global variation of free energy is obtained by adding the variation of free energy of each increment.

$$\Delta A_{X\to Y} \approx \sum_{i=1}^{NCONF} \left(\frac{\partial E_\lambda}{\partial \lambda} \right)_{\lambda_i} \delta\lambda \qquad (3.32)$$

Although the slow-growth approach has the advantage of requiring a single simulation to calculate the global variation of free energy, it often suffers from a lag between the system's configuration and the Hamiltonian[62]. The effect of this lag is the presence of a

fictitious positive contribution to the variation of free energy, regardless of the direction of the simulation. Long slow-growth simulations (large NCONF) are required to minimize the lag effect.

On the other hand, in the multi-configuration approach, selected values of λ are chosen and individual simulations are performed to evaluate the average of the derivative at those specific values of λ only[63]. The integral of equation 3.30 can be approximated numerically with standard integration techniques such as the midpoint rule, the trapezoid rule or Simpson's rule. As in the free energy perturbation technique, the width of the windows in this window-based approach can be fixed or not. Gaussian quadrature formulas are often used to approximate the integral of equation 3.30 with windows of variable width[64]. Quadrature rules compute approximations of definite integral with a weighted sum of the integrand's value at specific points within the domain of integration.

$$\Delta A_{X\to Y} \approx \sum_{k=1}^{n} \left\langle \frac{\partial E_\lambda}{\partial \lambda} \right\rangle_{\lambda_k} w_k \qquad (3.33)$$

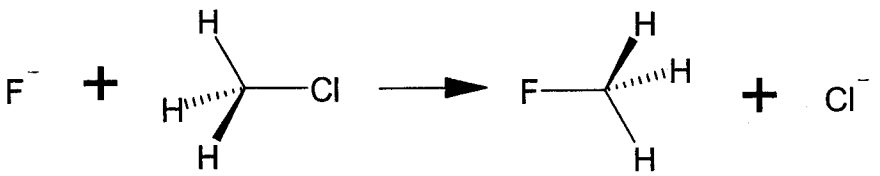
The quality of the approximation can be controlled by the order of the quadrature. A one point quadrature (λ_1=0.5, w_1=1) would obviously not be as accurate as a three point quadrature (λ_1=0.1127, w_1=0.2777, λ_2=0.5, w_2=0.4444, λ_3=0.8873, w_3=0.2777) which itself would not be as accurate as higher order quadratures. For thermodynamic integration, the sum of the weights is always equal to one and the quadrature points are always distributed symmetrically with respect to λ=0.5. Whether the integral is approximated by standard numerical techniques or quadrature formulas, an individual simulation is required for each point, limiting the numerical precision with which the integral can be evaluated.

Potential of mean force

To this point, the present discussion of free energy simulations has focused on the difference of free energy between two different molecular systems. The study of the free energy profile of a reaction along a certain reaction coordinate, whether it would be to determine the variation of free energy of the reaction or the activation free energy, is another common investigation in chemistry. Depending on the reaction, the reaction coordinate can be a bond length, a bond angle, a torsion angle or a combination of many variables. On its own, a molecular simulation will have difficulty overcoming the large energy barriers often encountered in common chemical reactions. Even in cases where a simulation successfully crosses an energy barrier, the system will not stay near the transition state long enough to sample that region of the potential energy surface appropriately.

To remedy this situation, the potential of mean force technique drags the molecular system along a reaction coordinate[65]. The potential of mean force can be seen as an extension of thermodynamic integration, where rather than connecting two states via non-physical mutations (methyl group mutated to a hydrogen atom), the two states are connected by a reaction coordinate. As a simple example, consider the S_N2 reaction of chloromethane with a fluoride anion.

Figure 24. S_N2 reaction of chloromethane and fluoride.



An intuitive reaction coordinate in the case of this S_N2 reaction would involve the F-C and C-Cl bond lengths. As the reaction progresses from reactants to products, the F-C bond gradually shortens while the C-Cl bond gradually lengthens. For the sake of this argument,

let's suppose the F-C bond goes from 3 angstroms to 1.4 angstroms while the C-Cl bond goes from 1.8 angstroms to 3.5 angstroms.

$$\xi_{\text{reactants}} = \{F - C = 3, C - Cl = 1.8\} \tag{3.34}$$

$$\xi_{\text{products}} = \{F - C = 1.4, C - Cl = 3.5\} \tag{3.35}$$

Just as was the case with the free energy perturbation and thermodynamic integration techniques, the initial and final states can be linked by intermediates states, defined with a coupling parameter. However, in the potential of mean force approach, the intermediates states are actual, physical intermediates between the starting point and the end point. A parallel can be made to the equations of thermodynamic integration. The derivative of the energy with respect to the reaction coordinate is calculated.

$$\Delta A_{\xi_i \to \xi_f} \approx \int_{\xi=\xi_i}^{\xi=\xi_f} \left\langle \frac{\partial E_\xi(r_1, r_2, \ldots, r_{3N})}{\partial \xi} \right\rangle_\xi d\xi \tag{3.36}$$

The derivative of the energy with respect to the reaction coordinate is obviously related to the energy gradient (the derivative of the potential energy with respect to position is a force). One intuitive way of thinking about the potential of mean force technique is that the system is dragged along a reaction coordinate while the force required to drag it is measured to ultimately calculate the variation of free energy. When performing a potential of mean force simulation, the reaction coordinate must be chosen carefully[66]. One particular point of emphasis to consider when choosing the reaction coordinate is that it must go through the transition state for a proper evaluation of the activation energy.

Once again, the integral of equation 3.36 can be evaluated with the slow-growth or multi-configuration approach. In the slow-growth approach, the reaction coordinate is gradually modified after each configuration to reach the final state. In the multi-configuration approach, the global change in the reaction coordinate is divided into NINT intermediate

changes. Individual simulations are performed for each intermediate state in which the reaction coordinate is frozen and the average derivative of the energy with respect to the frozen reaction coordinate is computed. Following the individual simulations, the integral can be approximated numerically by adding the contributions of each window.

$$\Delta A_{\xi_i \to \xi_f} \approx \sum_{k=1}^{NINT-1} \left\langle \frac{\partial E}{\partial \xi_k} \right\rangle_{\xi_k} \Delta\xi \qquad (3.37)$$

Equation 3.37 is equivalent to the approximation of the area under a curve (in this case a plot of the derivative of the energy versus the reaction coordinate) with rectangles of fixed width. Other techniques, such as the trapezoid integration or quadratures, can also be used to numerically approximate the integral.

The "hard" constraints keeping the reaction coordinate fixed at a specific value can be applied in various manners[67]. For simple cases, such as the previously discussed S_N2 reaction, the position of particular nuclei (F, C and Cl atoms) can be fixed while the other nuclei move around during the simulation. More complex reactions for which the reaction coordinate involves many internal coordinates are more difficult to constrain to a particular value of the reaction coordinate. Other techniques, such as the popular umbrella sampling method[68], impose "soft" constraints to keep the system in a region of the potential energy surface.

3.2. The introduction of secondary chains into MC simulations

A molecular simulation is an inherently long process, especially when ab initio calculations are implicated. Three parameters will affect the total length of a multi-configuration potential of mean force simulation: the time required for the calculation of the energy and energy gradient of a single configuration, the number of windows used to numerically

evaluate the variation of free energy (equation 3.37) and the number of configurations required in each window to obtain a representative average of the derivative of the energy with respect to the reaction coordinate.

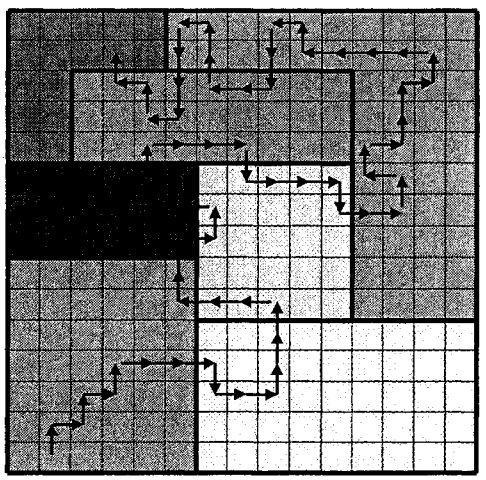
The first parameter, the time required for the calculation of the energy and energy gradient of a single configuration, is entirely dependent on the size of the system and the level of theory. Considering the large number of configurations typically generated during a simulation, one usually cannot afford to spend more than a few seconds per configuration. Larger systems are usually limited to force fields, semi-empirical methods or hybrid QM/MM methods for simulations of reasonable length. If an expensive quantum mechanical method was initially selected for accuracy purposes then there is no point in lowering the quality of the method in order to save time. The second parameter, the number of windows used to evaluate the variation of free energy, is responsible for the accuracy of the numerical integration of equation 3.36. Although it might be tempting to reduce the number of windows because each window requires its own simulation, it would be pointless to accurately simulate each window only to waste the accurate results accumulated when performing the final numerical integration. Additionally, each window can be simulated simultaneously in parallel, providing another deterrent to the reduction of the number of windows.

The last parameter, the number of configurations required in each window, depends on the sampling efficiency of the simulation. Different areas of the potential energy surface will have different contributions to the average property being evaluated during the simulation. In the case of a potential of mean force simulation, an efficient sampling method will require less data (fewer configurations) to accurately calculate the average force in a given window. The sampling efficiency of Metropolis-MC simulations is related to the acceptance rate, which itself is a function of the maximal random displacement allowed during the generation of trial configurations. The maximal random displacement is usually adjusted to reach an acceptance rate of approximately 50% in order to optimize the sampling efficiency. Too low an acceptance rate (large maximal displacement allowed) means the sampling is too aggressive, trying to leap and bound over different areas of the

potential energy surface with little success. Too high an acceptance rate (small maximal displacement allowed) means the sampling is not aggressive enough, covering the potential energy surface in an unnecessary conservative fashion with small steps. One major deficiency of Metropolis-MC simulations is that successive configurations are strongly correlated to each other. In the context of MC simulations, correlation refers to the fact that the configuration X_{i+1} is very similar to the configuration X_i. Even with an acceptance rate of 50%, thousands to millions of configurations are required to properly sample the potential energy surface depending on the system.

Consider a fictitious molecular system whose potential energy surface is represented by the 15x15 grid of figure 25. In order to properly sample this potential energy surface, a simulation must visit each region delimited by a different color (or a thick black line).

Figure 25. Sampling of a potential energy surface with traditional Metropolis-MC.

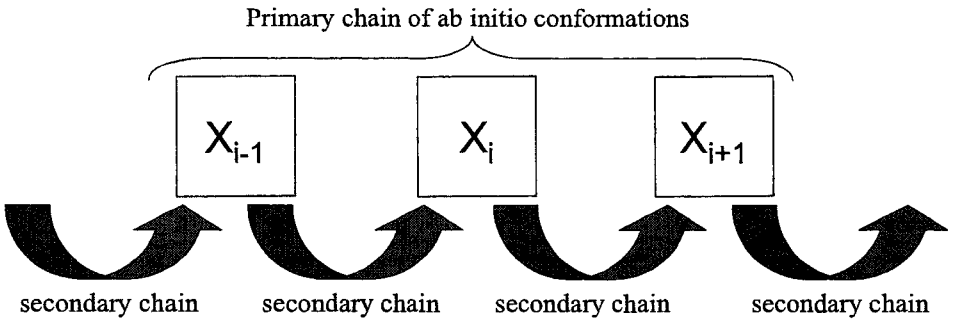


With a traditional MC simulation, configurations are strongly correlated and the movement along the surface is restricted to random steps crossing a single dotted line. Additionally, on average, the steps are accepted only half the time. Starting from a point in the bottom left corner, numerous steps will be required to visit each region. If the objective is to reduce the number of steps required to properly sample the surface (fewer configurations

for a shorter simulation) then it is clear that the correlation between configurations must be weakened. At the other end of this spectrum, configurations generated randomly from scratch, as opposed to the scheme of Metropolis-MC, would be entirely uncorrelated but of such low probability that they would be useless. Sticking with Metropolis-MC, the simplest way to weaken the correlation is to increase the maximal displacement allowed when randomly perturbing the nuclear coordinates to generate a trial configuration. Larger perturbation of the nuclei coordinates lead to more different configurations. Unfortunately, a drastic reduction of the acceptance rate will accompany the increase of the maximal displacement, in effect counteracting the improvement in sampling efficiency gained by the weakened correlation. In order to really improve the sampling efficiency, the correlation between configurations must be weakened without the negative impact of lowering the acceptance rate.

Traditional MC simulations consist of a single Markov chain in which trial configurations are generated by slightly perturbing a previous configuration and are accepted or rejected according to a Boltzmann distribution. The work of Schofield has focused on improving the sampling efficiency of traditional MC simulations at the ab initio level by generating an importance function based on a classical potential. Sampling efficiency has been shown to improve by approximately an order of magnitude with his classical potential importance function for systems in the gas phase[69] or in solution[70]. Secondary Markov chains of fixed length (NSEC steps) simulated with a classical potential are inserted between the configurations of the primary ab initio chain. The secondary chains, usually on the order of 1000 steps in length, are nothing more than short standard MC simulations using a classical potential. They always start with a configuration of the primary ab initio chain. The final configuration obtained after a simulation of NSEC steps using the classical potential serves as the subsequent trial configuration for the primary ab initio chain. The time required to complete a simulation of 1000 steps with a classical potential is negligible compared to a single energy evaluation at the ab initio level.

Figure 26. Concept of secondary Markov chains.



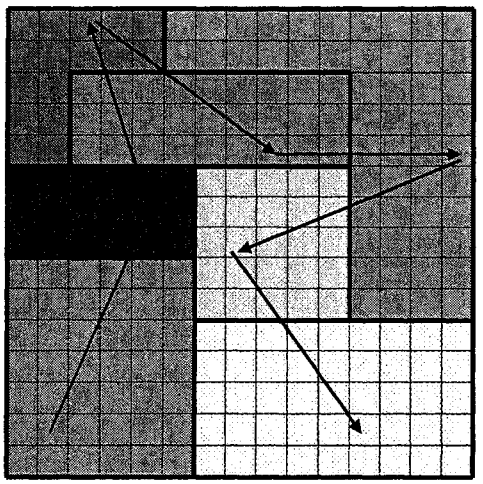
Statistics concerning the secondary chains are not accumulated as their only purpose is to generate trial configurations for the primary chain. The average property of interest is still calculated purely at the ab initio level. The acceptance criterion of Metropolis-MC initially defined by equation 3.12 is slightly modified to ensure that the primary chain's limiting distribution is the Boltzmann distribution.

$$\Delta\Delta E_{i\to i+1} = \Delta E_{i\to i+1}^{\text{ab initio}} - \Delta E_{i\to i+1}^{\text{classical}} \tag{3.38}$$

$$P_{i\to i+1} = \min\left\{1, e^{-\Delta\Delta E_{i\to i+1}/k_B T}\right\} \tag{3.39}$$

The simulation will converge to the same solution regardless of the form of the energy function (classical potentials or other functions) in the secondary chains. In effect, the configurations of the primary chain are now separated by NSEC steps instead of a single step. The correlation between configurations of the primary chain is substantially weakened, explaining the improvement of the sampling efficiency. Returning to the analogy presented in figure 25, the movement along the surface can now proceed with steps that cross many dotted lines.

Figure 27. Improved sampling efficiency using an importance function.



An order of magnitude improvement of the sampling efficiency over traditional MC could translate to steps crossing ten dotted lines instead of a single dotted line. Evidently, fewer steps are required to properly sample the surface in figure 27 when compared with figure 25. If the acceptance rate of this new sampling method remains approximately equal to 50% then it holds a decisive advantage over traditional MC in terms of sampling efficiency.

A closer look at equation 3.39 reveals that the acceptance rate depends on the ability of the classical potential to reproduce variations of energy calculated at the ab initio level. The trial configurations of the primary chain (the endpoint of a secondary chain) are always accepted if the variation of energy calculated at the ab initio level is lower or equal to the one obtained with the classical potential. However, as will be discussed later, the simulation can fall into a "sampling hole" when $\Delta E^{ab\ initio} - \Delta E^{classical}$ is significantly negative (a step that is always accepted). For optimal sampling efficiency, ab initio and classical variations of energy should remain similar.

Although a classical potential was used to generate trial configurations for a more expensive primary chain in the previous discussion, other methods could play the same role. Only two requirements need to be satisfied by the method used in the secondary chains.

Firstly, it must be fast. The larger fraction of the computing efforts should still be focused on the primary chain. An entire secondary chain (typically 500 to 1000 configurations) should be generated in a fraction of the time required for a single energy evaluation in the primary chain. Secondly, it must reproduce the variations of energy calculated with the method of the primary chain with reasonable success. No method can beat a classical potential in terms of speed to calculate the potential energy of a molecular system. However, generating a classical potential which is able to reasonably reproduce results of a quantum mechanical method is not as trivial as it seems, especially for the study of reactions involving the creation and/or destruction of chemical bonds. Different potentials need to be developed on a system to system basis, often requiring prior knowledge of the potential energy surface.

Intuitively, one would expect semi-empirical methods such as AM1[71] and PM3[72] to perform much better than classical potentials when trying to reproduce the results of more sophisticated methods such as DFT. Additionally, semi-empirical calculations are still orders of magnitude faster than the simplest of DFT methods, even for relatively small molecular systems. For MC simulations with DFT methods, semi-empirical methods meet the two requirements cited earlier to generate trial configurations with secondary Markov chains.

3.3. Proof of concept application (ring opening of cyclobutene)

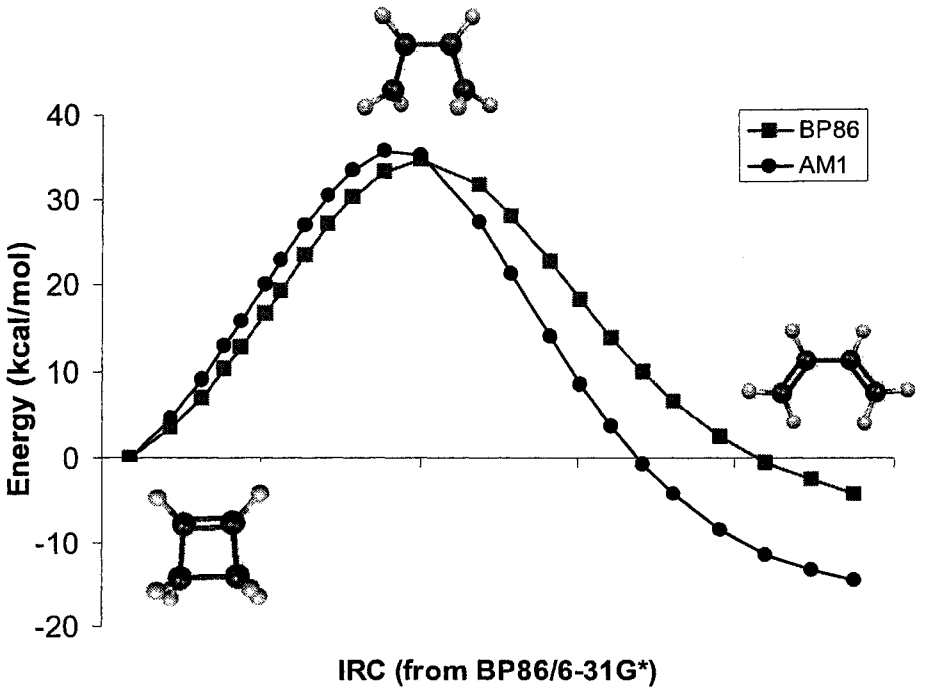
In this proof of concept application, calculations on the primary chain are performed with our own DFT code, DeFT[23], using Becke's B88 exchange functional[14] and Perdew's P86 correlation functional[13], commonly abbreviated as the BP86 functional. Pople's 6-31G* basis set[73] is used for all DFT calculations. The method chosen for the generation of the secondary chains is the semi-empirical AM1[71] Hamiltonian implemented within the Mopac7[74] software package. The objective of this application is not to reproduce experimental results or demonstrate the accuracy of a MC simulation with the BP86

functional on its own. We simply wish to compare the sampling efficiency of MC simulations at the BP86 level with and without the semi-empirical importance function.

The system chosen to test the sampling efficiency of three different MC schemes is the ring opening of cyclobutene to form cis-butadiene. A multi-configuration potential of mean force simulation will be used to obtain the free energy profile of the reaction. Ten intermediate states between cyclobutene and the transition state along with ten intermediate states between the transition state and butadiene were picked from an intrinsic reaction coordinate[75] path (IRC) obtained at the BP86/6-31G* level. The simulation will be broken down into twenty windows of approximately equal width. Figure 28 displays the electronic energy of these twenty intermediate states and the transition state for the BP86 and AM1 methods.

The AM1 activation energy is approximately 2kcal/mol higher than the BP86 activation energy while a difference of approximately 9kcal/mol is observed between the variations of energy of the reaction calculated with the two methods. Additionally, the transition state of the two methods does not occur at exactly the same point along the reaction coordinate. These differences are significant because the acceptance rate of the MC simulations with the AM1 importance function is dependent on the similarity of the AM1 and BP86 potentials. Sampling difficulties are foreseen for windows around the transition state when using the AM1 importance function because of the stronger disagreement between AM1 and BP86 in that area of the potential energy surface.

Figure 28. AM1 and BP86 energy along the BP86 IRC.



Because the objective of the simulation is not to reproduce experimental results, a simple choice of reaction coordinate is sufficient even though it may not lead to the correct activation energy. We chose the C-C bond length as the reaction coordinate, which goes from a value of 1.578 angstroms for cyclobutene to 3.105 angstroms for butadiene.

Figure 29. Reaction coordinate used for the simulation.

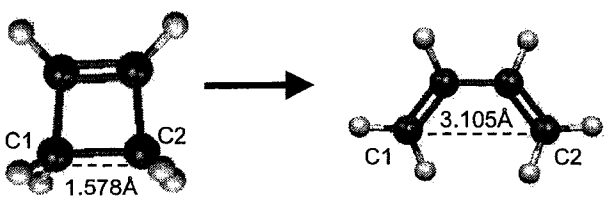


Table 18. Reaction coordinate for 20 windows.

Window	ξ (angstrom)	Window	ξ (angstrom)	Window	ξ (angstrom)
cyclobutene	1.578	8	1.976	15	2.509
1	1.640	9	2.024	16	2.569
2	1.706	10	2.084	17	2.631
3	1.758	TS	2.156	18	2.724
4	1.792	11	2.267	19	2.816
5	1.844	12	2.323	20	2.903
6	1.878	13	2.393	butadiene	3.105
7	1.928	14	2.451		

For the simulation of a given window, the nuclear coordinates of the carbon atoms involved in the reaction coordinate (C_{1x}, C_{1y}, C_{1z}, C_{2x}, C_{2y}, C_{2z}) are frozen while the other atoms are allowed to move during each MC step. For each accepted configuration, the energy gradient is calculated in addition to the electronic energy. Using simple geometry arguments, the components of the energy gradient vector belonging to C_1 and C_2 are aligned on an axis connecting C_1 and C_2 in order to obtain the derivative of the energy with respect to the reaction coordinate.

$$\frac{\partial E}{\partial \xi} = \frac{1}{2} \sum_{i=x,y,z} \frac{\partial E}{\partial C_{1i}} \left(\frac{\xi}{C_{1i} - C_{2i}} \right) + \frac{\partial E}{\partial C_{2i}} \left(\frac{\xi}{C_{2i} - C_{1i}} \right) \qquad (3.40)$$

The average force to pull C_1 and C_2 apart is computed by averaging the results of equation 3.40 during the individual simulations of each window. Variations of free energy can then be calculated by numerically approximating the integral of equation 3.36 with the trapezoid rule from a plot of the average force against the reaction coordinate.

Three different MC schemes will be tested: a pure BP86/6-31G* simulation without a semi-empirical importance function (labelled DFT), a BP86/6-31G* simulation using a rigid

AM1 importance function (labelled DFT-SE/R) and a BP86/6-31G* simulation using a novel adaptive AM1 importance function (labelled DFT-SE/A). The temperature of all the MC simulations is fixed at 300K and the maximal displacement parameter is set to 0.0165 angstroms. The pseudo-random numbers required during the simulation are generated with the *drand* function of Fortran77[41]. The *drand* function generates an infinitely long series of numbers starting from a seed. They are called pseudo-random numbers because the series of numbers is a function of the seed alone (the same seed will give the exact same series of numbers). Different seeds are obtained from a function involving the time of the day at which the simulation starts.

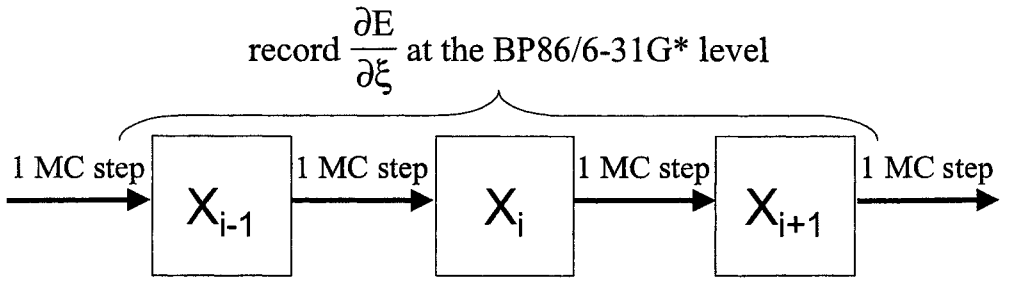
$$\text{seed} = (\text{hour} + 1)(\text{minutes} + 100)(\text{seconds} + 200) \tag{3.41}$$

The arbitrary constants 1, 100 and 200 were added to reduce the probability of having different time resulting in the same seed.

The DFT MC scheme

The DFT scheme is the simplest of the three MC schemes considered in this study. A chain of configurations is generated at the BP86/6-31G* level without any semi-empirical secondary chains inserted between the configurations. Trial configurations are generated with random displacements of all the nuclear coordinates except for the C_1 and C_2 atoms, as described by equation 3.11. A trial configuration is accepted or rejected according to the probability function of equation 3.12.

Figure 30. The DFT Monte Carlo scheme.

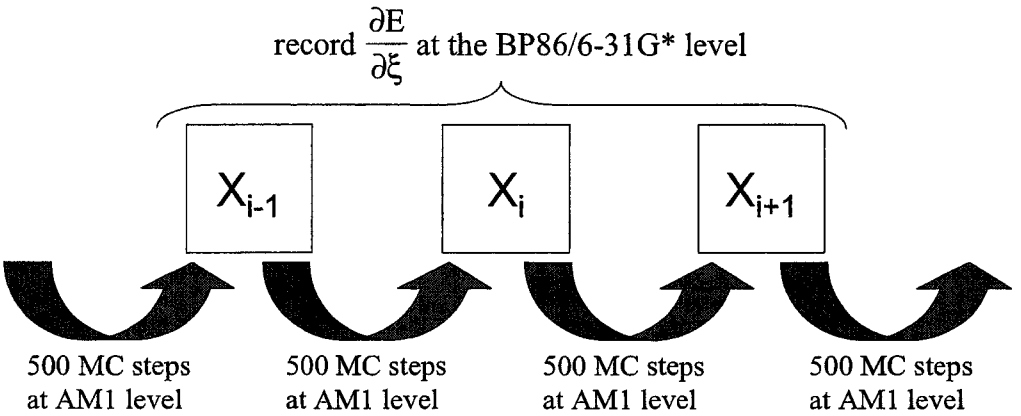


The force acting on the reaction coordinate is recorded for each accepted configuration and the average force is computed after a predetermined number of MC steps. The acceptance rate of this scheme is approximately 50% regardless of which window is simulated (the maximal displacement parameter was adjusted to obtain this acceptance rate). Correlation is at its maximum and poor sampling efficiency is expected with the DFT scheme.

The DFT-SE/R scheme

The DFT-SE/R MC scheme incorporates a rigid semi-empirical importance function to the DFT scheme. A secondary chain of 500 MC steps is generated with the AM1 method between each configuration of the primary chain. The acceptance rate within a secondary chain is approximately 50% since the same maximal displacement parameter is used. Each secondary chain starts with the previously accepted configuration of the primary chain and ends with the subsequent trial configuration of the primary chain. The trial configurations in the secondary chains are generated with the same procedure (random displacements of all atoms with equation 3.11 except for C_1 and C_2). Trial configurations of the primary chain are accepted or rejected according to the probability function of equation 3.39.

Figure 31. The DFT-SE/R Monte Carlo scheme

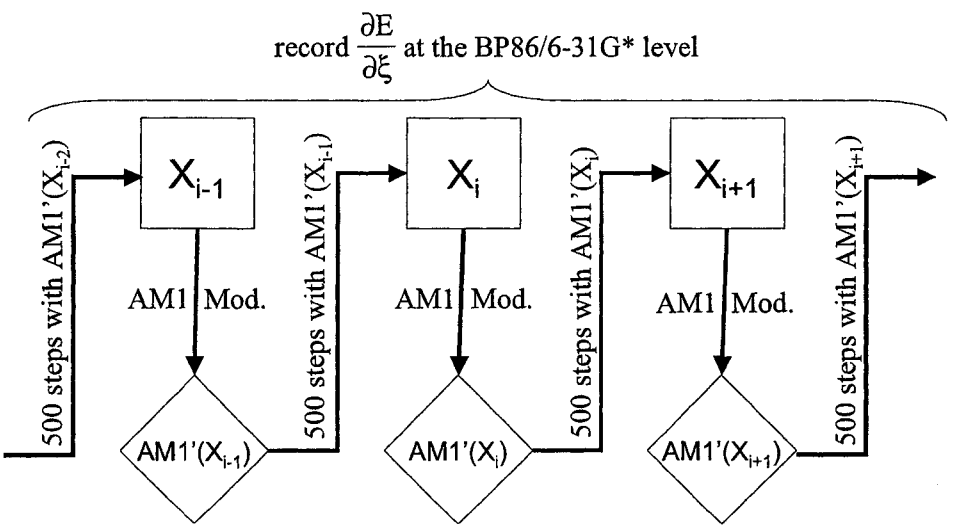


The force acting on the reaction coordinate is calculated strictly at the BP86/6-31G* level as the secondary chains' sole purpose is to generate trial configurations for the primary chain. Correlation between the configurations of the primary chain is substantially weakened compared to the DFT scheme since each configuration of the primary chain is in effect separated by 500 MC steps. The weakened correlation should significantly improve the sampling efficiency if the acceptance rate remains reasonably high. The acceptance rate of the trial configurations in the primary chain will depend on the similarity between the BP86/6-31G* and AM1 potentials. For that reason, different windows will have different acceptance rates. Windows around the transition state are expected to have a lower acceptance rate. The danger of falling into a "sampling hole" increases when significant disagreements exist between the two methods. Sampling holes occur when the transition to a trial configuration is much more favourable for the BP86 method than the AM1 method ($\Delta E_{i\to i+1}^{BP86} <<< \Delta E_{i\to i+1}^{AM1}$). Such configurations are always accepted and are difficult to move away from. Streaks of hundreds to thousands of consecutive rejected configurations are often observed before a trial configuration is finally accepted to escape the sampling hole. Needless to say, the presence of sampling holes negatively impacts the sampling efficiency of the DFT-SE/R scheme.

The DFT-SE/A scheme

The DFT-SE/A MC scheme was developed in order to prevent the simulation from falling into the sampling holes often encountered with the DFT-SE/R scheme. The idea of this new scheme is to parameterize the AM1 Hamiltonian to accentuate the similarity between the AM1 and BP86 potentials. Modifying the parameters of AM1 creates a semi-empirical method that is completely useless on its own. However, the only purpose of the semi-empirical method in this application is to reproduce the results of the BP86 method. It is important to remember that a MC simulation using an importance function will converge to the same result regardless of the secondary potential's identity. In the DFT-SE/A scheme, a slightly different AM1 type potential is used for each secondary chain, as opposed to the rigid true AM1 potential for all secondary chains in the DFT-SE/R scheme. Apart from this crucial difference, DFT-SE/R and DFT-SE/A operate identically. In figure 32, AM1'(X_i) refers to a modified AM1 potential adjusted to better reproduce the BP86 potential around the configuration X_i.

Figure 32. The DFT-SE/A Monte Carlo scheme.



The different AM1' potentials are obtained by modifying selected AM1 parameters to better reproduce the BP86 energy gradient of a given configuration. If the energy gradients of the two methods are similar then variations of energy caused by slight perturbations to the nuclear coordinates should also be similar. For a complete description of the AM1 parameters, their value and their physical meaning, refer to the original AM1 paper[71]. First-row elements are characterized by seven basic parameters (four for hydrogen as *p* orbitals are not involved): U_{ss}, U_{pp}, β_s, β_p, ζ_s, ζ_p and α. Additional parameters are added in the form of Gaussian corrections to the core repulsion function (K_i, L_i, M_i with i=1,4 for carbon and i=1,3 for other elements). Ideally, the entire parameterization process should only take a fraction of the time required for a single BP86 calculation (energy and energy gradient). For that reason, only selected parameters take part in the parameterization process as opposed to the entire set. The α parameter as well as the K and M parameters are ignored and keep their original value throughout the process. To further reduce the number of variables to parameterize, L_1, L_2, L_3 and L_4 are merged into a single L parameter. In all, five parameters for first-row elements and three for hydrogen are treated as flexible variables. In our application, a total of eight parameters need to be considered: U_{ss}(carbon), U_{pp}(carbon), β_s(carbon), β_p(carbon), L(carbon), U_{ss}(hydrogen), β_s(hydrogen) and L(hydrogen). For a given configuration, the AM1' energy is a function of eight variables.

$$P = \left\{U_{ss}^{C}, U_{pp}^{C}, U_{ss}^{H}, \beta_s^{C}, \beta_p^{C}, \beta_s^{H}, L^{C}, L^{H}\right\} \tag{3.42}$$

$$E^{AM1'} = E^{AM1'}(P) \tag{3.43}$$

The following equation represents the function to be minimized during the parameterization process for a given configuration, where N is the number of atoms and C_{ji} is the nuclear coordinate of atom j in direction i.

$$f(P) = \sum_{j=1}^{N} \sum_{i=x,y,z} \left(\frac{\partial}{\partial C_{ji}} E^{BP86} - \frac{\partial}{\partial C_{ji}} E^{AM1'}(P) \right)^2 \tag{3.44}$$

Fortunately, no additional BP86 calculation is required for the parameterization process since the BP86 energy gradient is already computed to evaluate the force acting on the reaction coordinate. Equation 3.44 is partially minimized with eight steps of the steepest descent algorithm[43], always starting from the original AM1 parameters. The partial derivative of equation 3.44 with respect to the eight individual parameters is required and numerically approximated by a finite difference. For example, the partial derivative of equation 3.44 with respect to L^C is approximated with the following equation.

$$\frac{\partial f(P)}{\partial L_C} = \frac{f(P + \Delta L^C) - f(P)}{\Delta L^C} \tag{3.45}$$

The increment added to the parameters to compute the partial derivative is always equal to one ten thousandth of the parameter ($\Delta L^C = 0.0001 L^C$). The parameters are updated using the partial derivatives at each step of the steepest descent algorithm. The partial derivatives are multiplied by an arbitrary factor of 0.001 to ensure that the steepest descent algorithm is not too aggressive in its minimization process.

$$L^C_{n+1} = L^C_n - 0.001 \frac{\partial f(P)}{\partial L^C_n} \tag{3.46}$$

A final set of parameters is obtained after eight steepest descent cycles. The entire optimization process necessitates hundreds of semi-empirical energy calculations but still takes only a fraction of the time required for a single BP86 energy calculation.

Using the DFT-SE/A scheme should improve the acceptance rate and sampling efficiency over DFT-SE/R because the semi-empirical importance function is reparamaterized after each new accepted configuration of the primary chain. The similarity between BP86 and the semi-empirical importance function is enhanced over all regions of the potential energy surface. No prior knowledge of the potential energy surface is required since the parameterization is automatically performed using the already available energy gradient of the primary chain configurations. As will be demonstrated in the following section, the

enhanced similarity is particularly striking for windows around the transition state. This fact is not surprising since semi-empirical methods such as AM1 were originally parameterized against data pertaining to stable molecular configurations.

There are two negative aspects to DFT-SE/A, both of which are not overly detrimental to the MC scheme. The first and more obvious drawback is the time required to parameterize the AM1 method for every accepted configuration of the primary chain. The parameterization process is about twice as long as the generation of an entire secondary chain, adding to the extra cost of using the importance function in the first place. However, the parameterization takes place only when a trial configuration is accepted. The potential improvement in sampling efficiency gained by reducing the occurrences of sampling holes when using DFT-SE/A more than make up for the additional time required for parameterization. The second drawback is related to the condition of detailed balance[76]. When detailed balance is respected, a Markov chain's limiting distribution exists, is unique and is the Boltzmann distribution. A Markov chain respects the detailed balance condition if the chain satisfies microscopic reversibility.

$$T_{i\rightarrow i+1}P_i = T_{i+1\rightarrow i}P_{i+1} \tag{3.47}$$

In equation 3.47, T represents the transition probability of going from one state to the next state in the chain while P represents the probability of being in a given state. It can easily be proven that a standard single chain MC scheme, such as the DFT scheme, satisfies microscopic reversibility. The usage of a rigid semi-empirical function, such as the DFT-SE/R scheme, also satisfies microscopic reversibility, although the mathematical proof is more complex[69]. Detailed balance is respected in the DFT-SE/R scheme because microscopic reversibility is satisfied in the secondary chains and the following equation involving part of the acceptance criterion is true.

$$\Delta\Delta E_{i\rightarrow i+1} = -\Delta\Delta E_{i+1\rightarrow i} \tag{3.48}$$

On the other hand, because of the adaptive importance function, the previous equation does not hold for the DFT-SE/A scheme. AM1'(X_i) is used to calculate $\Delta\Delta E_{i\to i+1}$ but a different AM1'(X_{i+1}) is used to calculate $\Delta\Delta E_{i+1\to i}$. Fortunately, it has been shown that the detailed balance condition is an unnecessarily strict requirement for a correct MC simulation[77]. Any MC scheme that leaves the Boltzmann distribution invariant, as prescribed by the more lenient balance condition, will provide results that agree with the correct Boltzmann distribution. The DFT-SE/A scheme, along with many other MC schemes (for example any scheme using sequential updates of any kind), falls into that category.

3.4. Sampling efficiency of DFT, DFT-SE/R and DFT-SE/A

The technique of block averaging will be used to quantify the sampling efficiency of the three MC schemes. The overall data obtained from a simulation, in this case the force acting on the reaction coordinate for each configuration of the primary chain, is divided into blocks of data. For example, a simulation of 10000 steps (NT for total number of steps) could be divided into 5 blocks (NDIV) of 2000 (NB for number of steps per block) steps each. The average of a particular block is calculated with a simple arithmetic average of the data inside that block (configurations 1 to 2000 in block NDIV=1, 2001 to 4000 in block NDIV=2, etc.).

$$\langle F \rangle_{NDIV=i} = \frac{\sum_{j=NB(i-1)+1}^{NB(i)} F_j}{NB} \tag{3.49}$$

If the overall data is divided into blocks of equal size then the average of the individual block averages is also equal to the overall arithmetic average taken over the entire simulation data.

$$\langle F \rangle = \frac{\sum_{j=1}^{NT} F_j}{NT} = \frac{\sum_{i=1}^{NDIV} \langle F \rangle_i}{NDIV} \tag{3.50}$$

The standard deviation stemming from the average of averages can be calculated with the usual formula for small values of NDIV.

$$s = \sqrt{\frac{\sum_{i=1}^{NDIV} \left(\langle F \rangle_{NDIV=i} - \langle F \rangle \right)^2}{NDIV - 1}} \tag{3.51}$$

If the sampling of a method was perfect then the averages of sufficiently large blocks would be identical and the standard deviation equal to zero. That will obviously not be the case for a real simulation. However, the standard deviation of equation 3.51 is a good indicator of the sampling efficiency of a MC scheme. Efficient sampling will translate to block averages that are closer to the overall average and ultimately a smaller standard deviation.

Long DFT, DFT-SE/R and DFT-SE/A simulations for a particular window

Long simulations of 30000 configurations were performed with all three schemes for one particular window, more precisely window 8 (see table 18). This window is chosen because it resides in a problematic area of the potential energy surface for the DFT-SE/R scheme (discrepancies between BP86 and AM1 potentials) and is well suited to demonstrate the difference between the DFT-SE/R and DFT-SE/A schemes. The 30000 configurations are divided into 60 blocks of 500 configurations. The objective of this simulation is to observe the behaviour of the MC schemes in the limit of a large number of configurations, in particular the acceptance rate, the spread in the block averages and the presence of sampling holes when using an importance function.

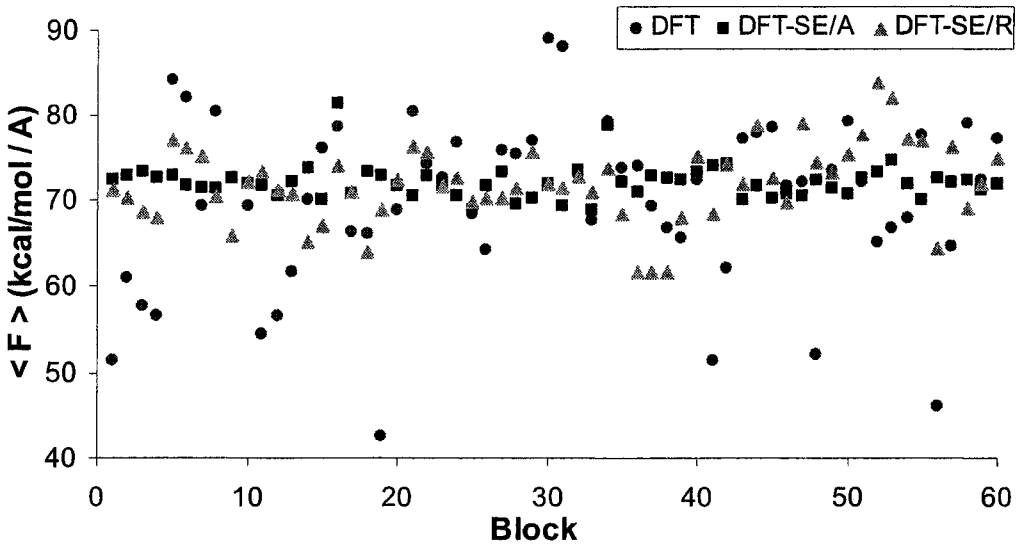
One should not be worried that the average force obtained after 30000 configurations is not exactly the same with the three different MC schemes. It was not our intention to obtain a converged value for the average force. The simulation with the DFT scheme certainly has not converged within 30000 configurations. However, it is encouraging that the average forces obtained with DFT-SE/R and DFT-SE/A are very similar even though our novel DFT-SE/A scheme violates the detailed balance principle.

Table 19. Results of long simulations performed on window 8 with three MC schemes.

	DFT	DFT-SE/R	DFT-SE/A
NT	30 000	30 000	30 000
NDIV	60	60	60
NB	500	500	500
< F > (kcal/mol / angstrom)	69.9	71.9	72.1
s (kcal/mol / angstrom)	9.8	4.6	2.0
Acceptance rate	51.1%	10.7%	36.6%
Longest rejection streak	20	3497	237
50+ rejections streaks	0	126	29
100+ rejections streaks	0	48	5
250+ rejections streaks	0	14	0

The standard deviations obtained with the three schemes differ significantly, from 9.8 kcal/mol/angstrom for DFT to 4.6 kcal/mol/angstrom for DFT-SE/R to 2.0 kcal/mol/angstrom for DFT-SE/A. The difference of standard deviations is clearly seen in figure 33.

Figure 33. Average force of 60 blocks of simulation data.



The average forces of the DFT scheme's blocks are spread out all over the chart, ranging from a minimum of 42 kcal/mol/angstrom to a maximum of 89 kcal/mol/angstrom. This gap is much smaller for DFT-SE/R (61 to 84) and even smaller for DFT-SE/A (68 to 81). Looking at the standard deviations, we can conclude that the DFT-SE/R scheme is successful in reducing the correlation between configurations of the primary chain and improve the sampling efficiency over the DFT scheme. However, the presence of several sampling holes hurts its sampling efficiency. At one particular point, 3497 consecutive trial configurations were rejected. At least 50 consecutive trial configurations were rejected on 126 different occasions, at least 100 consecutive trial configurations were rejected on 48 different occasions and at least 250 consecutive trial configurations were rejected on 14 different occasions. These rejected configurations alone (250x14 + 100x34 + 50x78 = 10800) account for a third of the total simulation and are mostly responsible for the low acceptance rate of 10.7%. In addition to reducing the correlation between configurations of the primary chain, the DFT-SE/A scheme mostly avoids the sampling holes that plague DFT-SE/R. The longest streak of consecutive rejections is relatively short at 237, considering streaks in excess of 250 rejections occurred 14 different times with DFT-SE/R.

Streaks in excess of 50 consecutive rejections happened only 29 times while streaks in excess of 100 consecutive rejections happened only 5 times. By avoiding sampling holes for the most part, the DFT-SE/A scheme is able to raise the acceptance rate to 36.6% and sample the potential energy surface more effectively than DFT-SE/R.

Ultimately, sampling efficiency is measured by the number of configurations required to yield a final average with a given degree of certainty. The more efficient methods will require fewer configurations. From figure 33, it is clear that, at least intuitively, fewer blocks are required for DFT-SE/A to obtain a "good" average when compared with the other two methods. This intuitive observation can be quantified with the help of confidence intervals. If the standard deviations presented in table 19 are good approximations of the standard deviations in the limit of an infinitely large data set then a 95% confidence interval can be defined with the following equation[78].

$$95\%(\text{C.I.}) = \pm\frac{1.96s}{\sqrt{\text{NDIV}}} \tag{3.52}$$

One might ask how many blocks are required to obtain an average force with a 95% confidence interval of 2.5 kcal/mol/angstrom. Using the standard deviation presented in table 19, that question is easily answered by isolating NDIV in equation 3.52 and rounding off the answer to the higher integer. 59 blocks (29500 configurations) would be required for DFT, 14 blocks (7000 configurations) for DFT-SE/R and 3 blocks (1500 configurations) for DFT-SE/A. The drastic reduction in the number of required configurations obviously translates to significant savings in computer time. In the case of these simulations, all the calculations were performed using resources from the High Performance Computing Virtual Laboratory[79]. Calculations on the configurations of the primary chain ran in parallel across eight processors and took approximately 7.5 seconds for a rejected configuration and 12.5 seconds for an accepted configuration (the energy gradient is required for an accepted configuration only). The semi-empirical calculations ran on a single processor and took 1.5 seconds for the generation of an entire secondary chain

(required for DFT-SE/R and DFT-SE/A) and 3.0 seconds for the parameterization (required for DFT-SE/A).

Table 20. Projected time required for simulations with different MC schemes.

	DFT	DFT-SE/R	DFT-SE/A
NT	29 500	7 000	1 500
Acceptance rate	51.1%	10.7%	36.6%
Accepted configurations	15074	749	549
Rejected configurations	14426	6251	951
Primary chain time (hours)	82.4 (100%) [a]	15.6 (84%)	3.9 (78%)
Secondary chain time (hours)	0 (0%)	2.9 (16%)	0.6 (12%)
Parameterization time (hours)	0 (0%)	0 (0%)	0.5 (10%)
Total time (hours)	82.4	18.5	5.0

[a] The percentage represents the fraction of the total time spent on a particular task.

The high correlation between configurations in the DFT scheme is responsible for its poor sampling efficiency and a long projected simulation of 82.4 hours. The improved sampling efficiency in DFT-SE/R results in a simulation that is approximately 4.5 times faster than DFT even though a significant amount of additional time is spent on the generation of secondary chains. Further improvement of the sampling efficiency in DFT-SE/A leads to a simulation that is approximately 16.5 times faster than DFT and 3.7 times faster than DFT-SE/R, even with the additional time required for parameterization. This data conclusively demonstrates the advantages of improving the sampling efficiency in exchange for the additional cost of generating secondary chains and parameterization.

DFT-SE/R versus DFT-SE/A for the entire reaction

The window used in the previous simulations was chosen because DFT-SE/A was expected to significantly outperform DFT-SE/R. To prove that DFT-SE/A is advantageous for any

situation and not only in situations where DFT-SE/R encounters many sampling holes, short simulations of all twenty windows were performed with both schemes. Each window is simulated with 6 blocks of 500 configurations for a total of 3000 configurations while an equilibration period of 100 accepted steps precedes the accumulation of the data. The starting configuration of each window is taken from the BP86 intrinsic reaction coordinate path displayed in figure 28. Refer to table 18 for the actual value of the reaction coordinate for each window.

Table 21. Data for DFT-SE/R and DFT-SE/A simulations of all 20 windows.

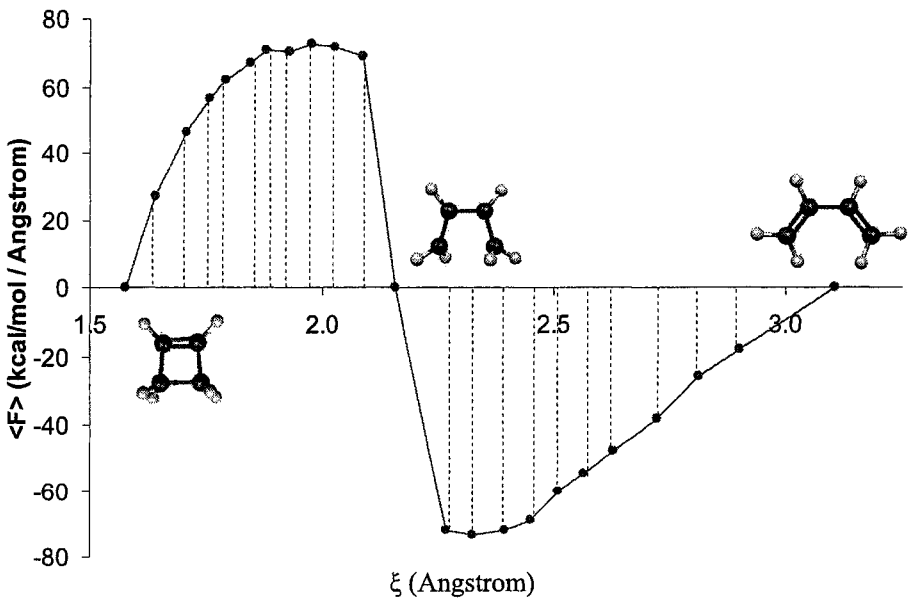
	< F > ± s (accept. rate) [a]			< F > ± s (accept. rate) [a]	
	DFT-SE/R	DFT-SE/A		DFT-SE/R	DFT-SE/A
1	28.6 ± 1.6 (28%)	26.9 ± 1.0 (45%)	11	-64.2 ± 5.4 (13%)	-72.0 ± 3.3 (28%)
2	46.8 ± 2.5 (23%)	46.5 ± 0.9 (44%)	12	-58.9 ± 25 (15%)	-73.8 ± 3.5 (33%)
3	57.0 ± 0.7 (19%)	56.6 ± 1.2 (41%)	13	-72.6 ± 3.3 (20%)	-72.0 ± 1.1 (34%)
4	62.2 ± 1.5 (20%)	61.9 ± 0.9 (43%)	14	-65.1 ± 5.1 (21%)	-68.8 ± 2.9 (38%)
5	66.8 ± 2.6 (15%)	67.3 ± 1.8 (41%)	15	-61.3 ± 2.7 (21%)	-60.1 ± 2.9 (41%)
6	72.3 ± 1.4 (16%)	71.0 ± 0.8 (37%)	16	-56.2 ± 3.5 (24%)	-54.5 ± 2.1 (41%)
7	70.5 ± 2.7 (16%)	70.3 ± 2.5 (39%)	17	-47.7 ± 3.0 (26%)	-48.0 ± 2.2 (47%)
8	71.9 ± 4.7 (11%)	72.5 ± 1.6 (37%)	18	-36.9 ± 3.0 (29%)	-38.0 ± 1.4 (49%)
9	73.1 ± 3.1 (6%)	71.5 ± 1.7 (40%)	19	-25.6 ± 1.7 (25%)	-25.7 ± 1.8 (55%)
10	68.0 ± 5.2 (10%)	69.2 ± 2.0 (40%)	20	-17.8 ± 3.0 (28%)	-17.8 ± 0.2 (56%)

[a] < F > and s are in units of kcal/mol / angstrom

The acceptance rate of DFT-SE/A is consistently two to three times higher than the acceptance rate of DFT-SE/R. The increased mobility in the DFT-SE/A simulations results in lower standard deviations for 17 of the 20 windows. A free energy profile can be created by calculating the variations of free energy when going from one window to the next. The integral of equation 3.36 is approximated by calculating the area under the curve of a plot of the average force against the reaction coordinate, as demonstrated in figure 34. No actual simulation was performed for the cyclobutene, transition state and butadiene windows. For

these three windows, the average force acting on the reaction coordinate is given a value of zero.

Figure 34. Average force versus the reaction coordinate for DFT-SE/A simulation.



The data points in figure 34 are connected with straight lines to emphasize the shape of the trapezoids under the curve. The area under a trapezoid is equal to the variation of free energy between two connected data point.

$$\Delta A_{i\to i+1} = \frac{(\xi_{i+1} - \xi_i)(\langle F\rangle_{i+1} + \langle F\rangle_i)}{2} \tag{3.53}$$

The error associated with a variation of free energy can be calculated from the standard deviations of the average forces with formulas commonly used in error propagation calculations.

$$s_{\Delta Ai\to i+1} = \frac{\sqrt{s^2_{\langle F\rangle i} + s^2_{\langle F\rangle i+1}}}{\langle F\rangle_i + \langle F\rangle_{i+1}} \Delta A_{i\to i+1} \tag{3.54}$$

Creating the free energy profile is simply a matter of adding the different variations of free energy, starting with butadiene as the state of reference and a free energy of zero. The error of each free energy value is easily calculated.

$$A_{i+1} = A_i + \Delta A_{i\to i+1} \tag{3.55}$$

$$s_{Ai+1} = \sqrt{s^2_{Ai} + s^2_{\Delta Ai\to i+1}} \tag{3.56}$$

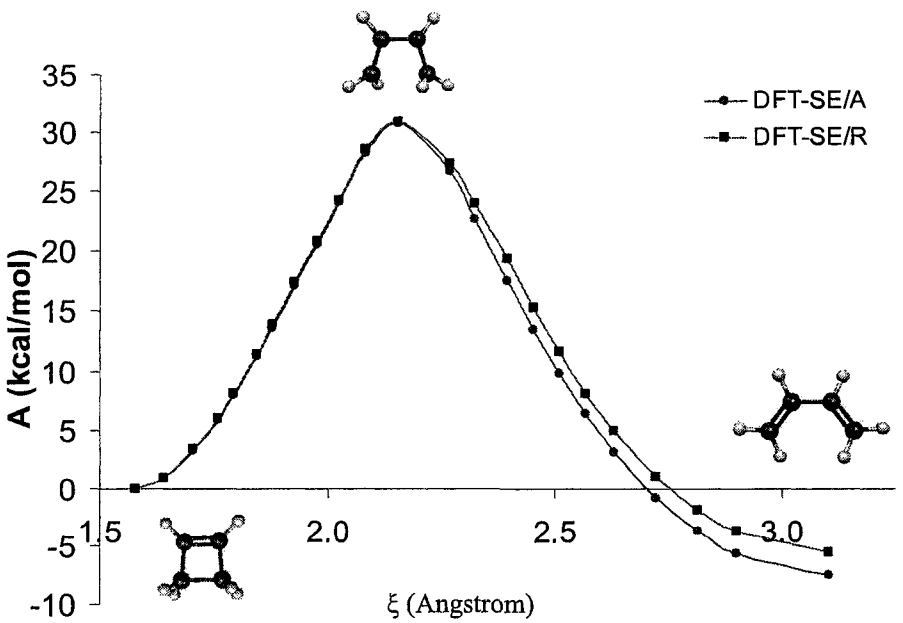
The errors presented in the previous equations are purely statistical in nature and depend entirely on the standard deviations of the average forces. They do not reflect the quality of the BP86 potential and the accuracy of the free energy profile with respect to experimental or other theoretical data. These errors are only related to the sampling efficiency by reflecting the similarity between six different blocks of data for each window.

The free energy profiles created with DFT-SE/A and DFT-SE/R more or less overlap until the first window after the transition state because the averages forces recorded by the two methods are very similar for the early windows. Windows 11 and 12 show significant differences between the average forces from the DFT-SE/A and DFT-SE/R simulations, as is clearly visible in the free energy profile.

Table 22. Free energies obtained with the DFT-SE/R and DFT-SE/A simulations.

	A ± s (kcal/mol)			A ± s (kcal/mol)	
	DFT-SE/R	DFT-SE/A		DFT-SE/R	DFT-SE/A
cyclobutene	0 ± 0	0 ± 0	11	27.40 ± 0.48	26.79 ± 0.26
1	0.89 ± 0.05	0.83 ± 0.03	12	23.95 ± 0.87	22.71 ± 0.29
2	3.37 ± 0.11	3.26 ± 0.05	13	19.35 ± 1.25	17.60 ± 0.32
3	6.07 ± 0.13	5.94 ± 0.07	14	15.35 ± 1.26	13.52 ± 0.33
4	8.10 ± 0.13	7.95 ± 0.07	15	11.69 ± 1.27	9.78 ± 0.35
5	11.45 ± 0.15	11.31 ± 0.09	16	8.17 ± 1.28	6.34 ± 0.37
6	13.81 ± 0.16	13.66 ± 0.10	17	4.94 ± 1.29	3.16 ± 0.38
7	17.38 ± 0.18	17.19 ± 0.12	18	1.01 ± 1.30	-0.84 ± 0.40
8	20.80 ± 0.22	20.62 ± 0.14	19	-1.87 ± 1.31	-3.77 ± 0.41
9	24.28 ± 0.26	24.07 ± 0.15	20	-3.76 ± 1.32	-5.66 ± 0.42
10	28.51 ± 0.32	28.30 ± 0.17	butadiene	-5.56 ± 1.35	-7.46 ± 0.42
TS	30.96 ± 0.37	30.79 ± 0.18			

Figure 35. Free energy profile of the reaction.



However, the results of DFT-SE/A for these two windows are more credible given the much lower standard deviation when compared with DFT-SE/R. The fact that the standard deviations from DFT-SE/A are consistently lower than those of DFT-SE/R over the entire course of the reaction leads to smaller statistical errors in the free energy profile (two times lower for the activation free energy and more than three times lower for ΔA of the reaction).

Table 23. Activation free energy and ΔA of the reaction.

	Activation free energy ± s (kcal/mol)	$\Delta A_{reaction}$ ± s (kcal/mol)
DFT-SE/A	30.79 ± 0.18	-7.46 ± 0.42
DFT-SE/R	30.96 ± 0.37	-5.56 ± 1.35

DFT-SE/A accomplishes exactly what it was designed to do. The simulations of windows DFT-SE/R has trouble handling because of sampling holes and a low acceptance rate are no concern for the DFT-SE/A scheme. Additionally, the improved sampling efficiency of DFT-SE/A is observed over the entire reaction, even for windows which are sampled fairly effectively by DFT-SE/R to begin with. The acceptance rate of DFT-SE/A is consistently in the 40% range and never dropped below 28% over the course of the reaction. The improved sampling efficiency of DFT-SE/A ultimately means fewer configurations are required to obtain an average with a given degree of certainty. The comparison between DFT-SE/A and DFT-SE/R for the reaction focuses on simulations of equal length to emphasize the consistently lower standard deviations of DFT-SE/A. However, as was demonstrated in the long simulation (30000 configurations) of window 8, DFT-SE/A may need less than a quarter of the number of configurations required by DFT-SE/R to obtain an average with the same level of certainty.

3.5. Afterthoughts

This proof of concept application certainly showed that the sampling efficiency of quantum mechanical MC simulations greatly benefits from a semi-empirical importance function. The improved sampling efficiency gained by weakening the correlation between configurations of the primary chain is well worth the additional time required to generate secondary chains at the semi-empirical level. Our novel adaptive importance function based on the AM1 Hamiltonian is successful in further improving the sampling efficiency when compared to a rigid AM1 importance function. The usage of our adaptive importance function removes the sampling holes often observed during simulations with the rigid importance function due to significant discrepancies between the AM1 and BP86 potentials, especially for areas around the transition state.

We can conclude that the sampling of our novel DFT-SE/A scheme is more than an order of magnitude more efficient than a standard single chain scheme. DFT-SE/A simulations are much faster than simulations with the other two schemes (5.0 hours for DFT-SE/A, 18.5 hours for DFT-SE/R and 82.4 hours for DFT), even though a significant amount of time is spent on the generation of secondary chains and the parameterization of the AM1' potentials. In this case, approximately 20% of the total time was spent on the semi-empirical tasks while the remaining 80% focused on the BP86 calculations for a DFT-SE/A simulation. This fraction of the total time spent on semi-empirical tasks is artificially high because semi-empirical calculations were performed in serial while BP86 calculations were performed in parallel across eight processors. The fraction of the time spent on semi-empirical tasks will obviously diminish if the semi-empirical calculations can also be performed in parallel. Additionally, a smaller fraction of the total time would be spent on semi-empirical tasks for larger molecular systems or more sophisticated methods in the primary chain.

The effect of the length of the secondary chains on the acceptance rate was quickly investigated and we concluded that the acceptance rate was more or less constant for secondary chains in excess of 100 configurations. Longer secondary chains further reduce

the correlation between the configurations of the primary chain but result in more time spent on semi-empirical tasks. Chains of 500 configurations were chosen as a compromise between improving the sampling efficiency and minimizing the time spent on semi-empirical tasks. The optimal chain length that will minimize the actual computer time of a simulation would be an important piece of information. Future work will focus on the implementation of a more general reaction coordinate where every MC step is perpendicular to the intrinsic reaction coordinate. It is yet to be determined if the improvements of the sampling efficiency observed for a rigid system such as cyclobutene are truly universal and will translate to larger and more flexible systems.

In conclusion, we have demonstrated that semi-empirical methods are fast enough to serve as importance functions for quantum mechanical MC simulations. DFT-SE/A is extremely easy to use and does not require any prior knowledge of the potential energy surface. The parameterization process simply requires the energy gradient of the configurations in the primary chain, which is already available for a potential of mean force simulation. It is a very general scheme which can be applied to any combination of ab initio and semi-empirical methods. Our novel DFT-SE/A scheme provides a significant improvement to the sampling efficiency over traditional single chain simulations or simulations using a rigid importance function.

Conclusion

Computational chemistry is evolving into an increasingly powerful science. Larger systems are studied theoretically at the quantum mechanical level in an effort to predict or explain real life occurrences, necessitating the development of fast ab initio methods. Additionally, the inclusion of thermal effects requires molecular simulations as opposed to single point energy calculations. Metropolis Monte Carlo simulations applied to molecular systems are often plagued by inefficient sampling of the potential energy surface, resulting in unnecessarily long simulations. These two distinct issues, namely the development of a fast ab initio method and the improvement of the sampling efficiency of Monte Carlo simulations, were addressed in this work.

Our implementation of the analytical Xα functional focuses on efficiency and scaling behaviour in order to perform quantum mechanical calculations on large molecular systems. The analytical implementation improves upon the traditional numerical integration of the Xα exchange energy in terms of speed, especially for the energy gradient. The removal of the grid greatly simplifies the evaluation of the energy gradient, thus making the analytical method even more appealing for applications requiring the energy gradient such as geometry optimizations, molecular dynamics simulations or Monte Carlo simulations. Inherent deficiencies of the Xα functional and other errors introduced by our minimalist approach (minimal orbital basis set, charge fitting basis set and Xα fitting basis set) can be partly recuperated by treating α as a flexible parameter as opposed to a fixed constant. Hybrid methods such as the ONIOM method greatly benefit from the flexible α parameter when the analytical Xα functional is used as a low level of theory coupled with more sophisticated levels of theory.

Several applications involving of a molecular mechanics based importance function for Metropolis Monte Carlo simulations have been published. Our investigation revealed that a semi-empirical importance function is fast enough to be used with a primary Markov chain simulated at the DFT level. Using a semi-empirical importance function eliminates the need to always generate classical potentials that are valid for the specific application at

hand. Significant gains in sampling efficiency are observed when using the semi-empirical importance function when compared with a traditional single chain simulation at the DFT level. Furthermore, we developed a novel adaptive semi-empirical function. The adaptive importance function is obtained by allowing a set of semi-empirical parameters to vary in order to minimize the difference in the energy gradient between the DFT and semi-empirical potentials for each configuration of the primary chain. Using the adaptive importance function raises the acceptance rate of the simulation when compared to a rigid semi-empirical importance function and leads to a more efficient sampling of the potential energy surface. The drastic improvement in sampling efficiency is well-worth the additional cost of generating semi-empirical secondary chains and parameterizing the semi-empirical method for each configuration of the primary chain, ultimately leading to shorter simulations.

In conclusion, our implementation of the analytical Xα functional represents a very fast ab initio method on its own, with the flexibility to adapt itself to better suit more sophisticated methods within hybrid schemes. Our novel adaptive semi-empirical importance function drastically improves the sampling efficiency of Monte Carlo simulations, reducing the number of configurations required and shortening the simulation. Both of these distinct projects can be seen as steps towards the ultimate goal of studying large molecular systems at the quantum mechanical level and accounting for thermal effects, as opposed to small model systems at absolute zero.

References

[1] W.J. Hehre, L. Radom, J. Pople, P.V. Schleyer, Ab Initio Molecular Orbital Theory, Wiley-Interscience, 1986

[2] L.H. Thomas, *Proc. Cam. Phil. Soc.*, **23**, 542 (1927)

[3] E. Fermi, *Rend. Accad. Lincei*, **6**, 602 (1927)

[4] P.A.M. Dirac, *Proc. Cam. Phil. Soc.*, **26**, 376 (1930)

[5] E. Teller, *Rev. Mod. Phys.*, **34**, 627 (1962)

[6] R.G. Parr, W. Yang, Density Functional Theory of Atoms and Molecules, Oxford University Press, 1989

[7] P. Hohenberg, W. Kohn, *Phys. Rev. B*, **136**, 864 (1964)

[8] W. Kohn, L.J. Sham, *Phys. Rev. A*, **140**, 1133 (1965)

[9] U. von Barth, L. Hedin, *J. Phys. C: Solid State Phys.*, **5**, 1629 (1972)

[10] D.M. Ceperley, B.J. Alder, *Phys. Rev. Lett.*, **45**, 566 (1980)

[11] S.H. Vosko, L. Wilk, M. Nusair, *Can. J. Phys.*, **58**, 1200 (1980)

[12] F.J. Sim, A. St-Amant, I. Papai, D.R. Salahub, *J. Am. Chem. Soc.*, **114**, 4391 (1992)

[13] J.P. Perdew, *Phys. Rev. B*, **33**, 8822 (1986)

[14] A.D. Becke, *Phys. Rev. A*, **38**, 3098 (1988)

[15] C. Lee, W. Yang, R.G. Parr, *Phys. Rev. B*, **37**, 785 (1988)

[16] J.P. Perdew, K. Burke, Y. Wang, *Phys. Rev. B*, **54**, 16533 (1996)

[17] J.P. Perdew, K. Burke, M. Ernzerhof, *Phys. Rev. Lett.*, **77**, 3865 (1996)

[18] S. Kurth, J.P. Perdew, P. Blaha, *Int. J. Quant. Chem.*, **75**, 889 (1999)

[19] A.D. Becke, *J. Chem. Phys.*, **98**, 5648 (1993)

[20] T. Van Voorhis, G.E. Scuseria, *J. Chem. Phys.*, **109**, 400 (1998)

[21] J.M. Tao, J.P. Perdew, V.N. Staroverov, G.E. Scuseria, *Phys. Rev. Lett.*, **91**, 146401 (2003)

[22] E.J. Baerends, D.E. Ellis, P.Ros, *Chem. Phys.*, **2**, 41 (1973)

[23] DeFT and related documentation can be obtained by sending inquiries to the author at the University of Ottawa. E-mail address: alain.st-amant@uottawa.ca

[24] B. Delley, *J. Chem. Phys.*, **92**, 508 (1990)

[25] G. te Velde, F.M. Bickelhaupt, S.J.A. van Gisbergen, C. Fonseca Guerra, E.J. Baerends, J.G. Snijders, T. Ziegler, *J. Comput. Chem.*, **22**, 931 (2001)

[26] Gaussian 03, Revision C.02, M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, J. A. Montgomery, Jr., T. Vreven, K. N. Kudin, J. C. Burant, J. M. Millam, S. S. Iyengar, J. Tomasi, V. Barone, B. Mennucci, M. Cossi, G. Scalmani, N. Rega, G. A. Petersson, H. Nakatsuji, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, M. Klene, X. Li, J. E. Knox, H. P. Hratchian, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, P. Y. Ayala, K. Morokuma, G. A. Voth, P. Salvador, J. J. Dannenberg, V. G. Zakrzewski, S. Dapprich, A. D. Daniels, M. C. Strain, O. Farkas, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. V. Ortiz, Q. Cui, A. G. Baboul, S. Clifford, J. Cioslowski, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng, A. Nanayakkara, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, C. Gonzalez, and J. A. Pople, Gaussian, Inc., Wallingford CT, 2004

[27] A. St-Amant, Reviews in Computational Chemistry, Volume 7, Chapter 5, VCH Publishers Inc., 1996

[28] J.C. Slater, *Phys. Rev.*, **81**, 385 (1951)

[29] R. Gáspár, *Acta Phys. Acad. Sci. Hung.*, **3**, 263 (1954)

[30] K. Schwarz, *Phys. Rev. B*, **5**, 2466 (1972)

[31] A.D. Becke, *J. Chem. Phys.*, **88**, 2547 (1988)

[32] K.S. Werpetinski, M. Cook, *J. Chem. Phys.*, **106**, 7124 (1997)

[33] R.E. Stratmann, G.E. Scuseria, M.S. Frisch, *Chem. Phys. Lett.*, **257**, 213 (1996)

[34] B.I. Dunlap, *J. Phys. Chem.*, **90**, 5524 (1986)

[35] B.I. Dunlap, *Int. J. Quant. Chem.*, **69**, 317 (1998)

[36] B.I. Dunlap, *J. Phys. Chem. A*, **107**, 10082 (2003)

[37] R.R. Zope, B.I. Dunlap, *J. Chem. Phys.*, **124**, 44107 (2006)

[38] W.J. Hehre, R.F. Stewart, J.A. Pople, *J. Chem. Phys.*, **51**, 2657 (1969)

[39] B.I. Dunlap, J.W. Connolly, J.R. Sabin, *J. Chem. Phys.*, **71**, 3396 (1979)

[40] J. Andzelm, E. Wimmer, *J. Chem. Phys.*, **96**, 1280 (1992)

[41] More information on the Fortran language is available at www.fortran.com

[42] S. Obara, A. Saika, *J. Chem. Phys.*, **84**, 3963 (1986)

[43] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C: The Art of Scientific Computing, 2nd edition, Cambridge University Press, 1992

[44] C.G. Broyden, *J. Inst. Math. Appl.*, **6**, 76 (1970)

[45] R. Fletcher, *Comput. J.*, **13**, 317 (1970)

[46] D. Goldfarb, *Math. Comput.*, **24**, 23 (1970)

[47] D.F. Shanno, *Math. Comput.*, **24**, 647 (1970)

[48] D.A. Case, D.A. Pearlman, J.W. Caldwell, T.E. Cheatham III, J. Wang, W.S. Ross, C.L. Simmerling, T.A. Darden, K.M. Merz, R.V. Stanton, A.L. Cheng, J.J. Vincent, M. Crowley, V. Tsui, H. Gohlke, R.J. Radmer, Y. Duan, J. Pitera, I. Massova, G.L. Seibel, U.C. Singh, P.K. Weiner, P.A. Kollman, 2002, AMBER 7, University of California, San Francisco

[49] T. Vreven, K.S. Byun, I. Kimaromi, S. Dapprich, J.A. Montgomery Jr., K. Morokuma, M.J. Frisch, *J. Chem. Theory Comput.*, **2**, 815 (2006)

[50] K. Morokuma, Q. Wang, T. Vreven, *J. Chem. Theory Comput.*, **2**, 1317 (2006)

[51] P.B. Karadakov, K. Morokuma, *Chem. Phys. Lett.*, **317**, 589 (2000)

[52] K. Morokuma, D.G. Musaev, T. Vreven, H. Basch, M. Torrent, D.V. Khoroshun, *IBM J. Res. Dev.*, **45**, 367 (2001)

[53] A. Leach, Molecular Modellling: Principles and Applications, 2nd edition, Prentice Hall, 2001

[54] N. Metropolis, S. Ulam, *J. Am. Stat. Assoc.*, **44**, 335 (1949)

[55] D.A. McQuarrie, Statistical Mechanics, 2nd edition, University Science Books, 2000

[56] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, *J. Chem. Phys.*, **21**, 1087 (1953)

[57] P. Kollman, *Chem. Rev.*, **93**, 2395 (1993)

[58] R.W. Zwanzig, *J. Chem. Phys.*, **22**, 1420 (1954)

[59] T. Simonson, Computational Biochemistry and Biophysics, Marcel Dekker, 2001

[60] D.A. Pearlman, P.A. Kollman, *J. Chem. Phys.*, **90**, 2460 (1989)

[61] H. Hu, R.H. Yun, J. Hermans, *Mol. Simul.*, **28**, 67 (2002)

[62] D.A. Pearlman, P.A. Kollman, *J. Chem. Phys.*, **91**, 7831 (1989)

[63] T.P. Straatsma, J.A. McCammon, *J. Chem. Phys.*, **95**, 1175 (1991)

[64] G. Hummer, A. Szabo, *J. Chem. Phys.*, **105**, 2004 (1996)

[65] E. Carter, G. Ciccotti, J. Hynes, R. Kapral, *Chem. Phys. Lett.*, **156**, 472 (1989)

[66] A. Michalak, T. Ziegler, *J. Phys. Chem. A*, **105**, 4333 (2001)

[67] D.R. Matusek, S. Osborne, A. St-Amant, *J. Chem. Phys.*, **128**, 154110 (2008)

[68] G. Torrie, J. Valleau, *Chem. Phys. Lett.*, **28**, 578, 1974

[69] R. Iftimie, D. Salahub, D. Wei, J. Schofield, *J. Chem. Phys.*, **113**, 4852 (2000)

[70] R. Iftimie, D. Salahub, J. Schofield, *J. Chem. Phys.*, **119**, 11285 (2003)

[71] M.J.S. Dewar, E.G. Zoebisch, E.F. Healy, J.J.P. Stewart, *J. Am. Chem. Soc.*, **107**, 3902 (1985)

[72] J.J.P. Stewart, *J. Comp. Chem.*, **10**, 209 (1989)

[73] W.J. Hehre, R. Ditchfield, J.A. Pople, *J. Chem. Phys.*, **56**, 2257 (1972)

[74] MOPAC2007, J.J.P. Stewart, Stewart Computational Chemistry, Colorado Springs, USA, http://openmopac.net (2007)

[75] K. Fukui, *Acc. Chem. Res.*, **14**, 363 (1981)

[76] D. Frenkel, B. Smit, Understanding Molecular Simulation: From Algorithms to Applications, Computational Science Series Vol. 1, 2nd edition, Academic San Diego, 2002

[77] V.I. Manousiouthakis, M.W. Deem, J. Chem. Phys., 110, 2753 (1999)

[78] D.A. Skoog, D.M. West, F.J. Holler, C. Buess-Herman (translator), J. Dauchot-Weymeers (translator), F. Dumont (translator), Chimie analytique, De Boeck Université, 1997

[79] More information is available at http://www.hpcvl.org