Calculations Using The Coupled Coherent States Method:

Quantum Mechanics With Trajectories

John Andrew Kattirtzi

Submitted in accordance with the requirements for the degree of Mastership By Research

University of Leeds

School of Chemistry

September 2009

The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others.

Acknowledgements

This work would not have been possible without my advisor, Dr Dmitrii Shalashilin and I would like to thank him for his supervision, for helping me to improve my understanding of quantum mechanics and for guiding me through the derivation of the CCS equations. I am extremely grateful for how he would stop what he was working on in order to help me. I am sure that the basic concepts of quantum mechanics, which he helped me to understand, are likely to remain useful in the future.

I am also indebted to Dr Stewart Reed for his knowledge and experience of Unix, Fortran and the CCS method. The guidance given when programming helped me to write a well organised code and to overcome technical problems. In addition to this, numerous discussions have helped clarify various difficulties and have inspired new ideas.

Dr David Waller and the IT team provided much needed support with the Linux computers.

I would like to gratefully acknowledge the people who have read through this work for their helpful feedback, in particular: Dr Stewart Reed, Dr Dmitrii Shalashilin, Prof. Ben Whitaker and Michael Kattirtzi. Perfection is driven from criticism.

Lunch/Coffee Breaks and a good social atmosphere have helped me stay motivated throughout this year. I particularly enjoyed lunch with Katie Marriott and Cassey McRae and coffee breaks with Dr Marcelo Miranda, Pablo García Jambrina and others. Regulars who attended the pub also helped provide the good social atmosphere.

As always my family have helped to support me in what I want to do and I am grateful for this. In addition to this, I would like to make a special mention for the support from the Hadjisavva family who invited me for dinner every Sunday, offered me a place to stay when I needed it and were happy to drive me around Leeds.

Finally, funding from the School of Chemistry at the University Of Leeds has made all of this possible and I am very thankful for the studentship.

John Andrew Kattirtzi

Calculations Using The Coupled Coherent States Method: Quantum Mechanics With Trajectories

Masters By Research

September 2009

Abstract

Quantum dynamical methods are known to suffer from the ‘curse of exponential scaling’, which is associated with the computational cost of the simulation. Over the years a wide range of methods have been developed to overcome this high computational cost. One such promising method is known as the ‘Coupled Coherent States’ method. In this method the wave function is simulated using a basis set of time dependent basis functions that are multiplied by coupled time dependent coefficients. The method has been used previously for a variety of applications and has been shown to achieve good accuracy for a variety of problems but with a lower computational cost. Further work on applying the CCS method to different systems is therefore required.

In this work, original software has been written in the Fortran programming language that implements the CCS method, which was then tested. A Gaussian wave function was positioned at an excited state and then propagated. The results were then used to calculate a Franck-Condon absorption spectra for the Harmonic and Morse Oscillator. The main peaks of the spectra were found to be in excellent agreement with the exact eigenvalues for the two systems. Further work using this wave function for the Morse Oscillator with symmetry arguments showed how similar results could be obtained with a lower computational cost.

A Gaussian wave function was also used for simulations using the Bose-Hubbard model. This model describes a wave function of a Bose-Einstein Condensate, which is able to tunnel between two potential wells or self-trap in each potential well. The results are analysed using different basis sets and a comparison is made between the quantum CCS calculations and semi-classical calculations.

Although in this work only relatively simple one and two dimensional problems have been considered, the code could be used in the future to study more complex problems.

Contents

List of Figures

List of Tables

List of Abbreviations

$\Im$ or Im	Imaginary component
$\Re$ or Re	Real component
ACF	Autocorrelation Function
au	Atomic Unit
BEC	Bose Einstein Condensate
CCS	Coupled Coherent States
DFT	Density Functional Theory
FGA	Frozen Gaussian Approximation
G-MCTDH	Gaussian Multi-Configurational Time Dependent Hartree
HK	Herman-Kluk
IVR	Initial Value Representation
MAE	Mean Absolute Error
MCTDH	Multi-Configurational Time Dependent Hartree
TDH	Time Dependent Hartree
TDSE	Time Dependent Schrödinger Equation
TISE	Time Independent Schrödinger Equation
vMCG	variational Multi-Configurational Gaussian

Chapter 1

Introduction

1.1 Theoretical Background

Chemistry, the study of reactions of atoms and molecules, has traditionally been investigated experimentally. However, theoretical methods, which are rooted in physics, allow a theoretical insight into chemical processes. The choice of which theoretical method to use depends on the system and what is being investigated. These methods can be grouped into several classes. The distinction between the classes is often ambiguous in the literature. Therefore, the theoretical methods discussed here are divided into groups, which are defined in this work and may not necessarily correspond to the group names used in the literature. One way to divide these methods is to separate them into 'dynamical' and 'static' methods. The distinction between these methods, defined here, depends on whether the equations of motion are used. If the theoretical method solves equations of motion then the method is categorised as a 'dynamical' method, otherwise the method is categorised as a 'static' method.

Within the static methods class is a large group called 'electronic structure' methods. These techniques allow the electronic structure of atoms and molecules to be calculated and from this chemical properties, which may be useful in experimental studies, can be determined.[1] Electronic structure methods are used to solve the Time Independent Schrödinger Equation (TISE) and different electronic structure methods suggest different approximations for doing this. Often the calculation that gives the most accurate electronic structure also gives the most accurate properties. The primary difficultly is calculating the correlation of the electrons in atoms and molecules. The techniques range from Hartree-Fock theory, which has a lower computational cost but neglects electron correlation, to Couple Cluster theory, which has a relatively high computational cost but gives a good description of electron correlation. Density Functional Theory (DFT) is a popular choice for electronic structure calculations as the results obtained can be very accurate, whilst the computational cost is comparatively low.

On the other hand, dynamical methods are concerned with the motion of a system and allow dynamical simulations of a chemical system. Two types of dynamical methods discussed here are Molecular Dynamics and Quantum Molecular Dynamics. Both of these methods are used to solve the equations of motions for the system. However, Molecular Dynamics uses a classical description, whilst Quantum Molecular Dynamics uses a quantum wave function, to describe the system.

Molecular Dynamics calculates the motion of atoms and molecules using classical equations of motion. In order to do this, a potential energy function is required, which is also known as a force field. This can be calculated using chemical properties, such as bond lengths taken from experimental data, or by using electronic structure methods to calculate these properties. Often a combination of both is used.

Quantum Molecular Dynamics calculates the motion of a wave function (or wave packet) in accordance with the Time Dependent Schrödinger Equation (TDSE). The traditional method to do this is by using the TISE to solve the TDSE.[2] It can be shown that the TISE is a particular solution to the TDSE. This is done by separating the variables of the wave function into a product of two functions, one that is dependent on position and one that is dependent on time. The linear combination of solutions of the TISE is a solution to the TDSE.[3] Although this method is in principle exact, it is restricted to small systems by the computational cost associated with calculating each TISE. This computational cost is proportional to the number of equations that need to be solved (N), where $N = l^m$ and l is the number of basis functions in 1 degree of freedom. The number of degrees of freedom (or dimensions) of the wave function is given by m. For a wave function to give a good description of a large chemical system, many degrees of freedom are required and so the computational cost for solving the TDSE, using this method, is very high. Therefore, more efficient methods to solve the TDSE are required.

One efficient semi-classical approach to solving the TDSE, suggested by Heller, is called the ‘Frozen Gaussian Approximation’ (FGA).[4] In this approach the wave function is written as a sum of Gaussian basis functions. These basis function can then be propagated over time using the classical equations of motion. The Gaussian functions are termed ‘Frozen’ as the width of the Gaussian functions remain fixed. Solving the TDSE in this way requires initial values of the wave function and such methods are known as ‘initial value representation’ (IVR) methods. Heller showed that the FGA can be used to solve simple systems, up to quadratic potentials, and that interesting properties such as Franck-Condon absorption spectra can be calculated.[5,6] The Herman-Kluk method is a more developed version of this, which has superseded the FGA method.[7–10] Both of these methods reduce the computational cost but give a semi-classical description.

Further developments have been made with other initial value representation methods that are fully quantum mechanical but decrease the computational cost and so can be used for large chemical systems. The Time Dependent Hartree (TDH) method uses a product of one dimensional Hartree products for a multidimensional wave function. This decreases the computational cost of the calculation significantly by decreasing the size of the base (l). Unfortunately the correlation between the different dimensions is neglected. One approach to overcome this is the 'Multi-Configurational Time Dependent Hartree' (MCTDH) method, where a multi-configurational wave function is used.[11,12] The accuracy of a simulation can be further enhanced by using more configurations of the wave function. However the larger number of configurations results in a larger size of the base.

The MCTDH method has gained much attention since 1990 and has been used for a variety of multidimensional problems.[11] Many variations of this have also been developed. One variation named 'G-MCTDH' uses Gaussian functions with variable widths, instead of the flexible basis functions used in MCTDH.[13,14] If the basis functions are all Frozen Gaussians the method is called the 'variational Multi-Configurational Gaussian' (vMCG) method. In this approach, the variational principle is applied to the whole wave function and allows the trajectories of the basis functions to reach classically forbidden areas.

A computationally less expensive approach is to use a similar method called the 'multiple spawning technique'.[15–17] In this method the wave function is written as a linear combination of multidimensional frozen Gaussian basis functions that follow classical trajectories. The basis functions are multiplied by time dependent coefficients and these are referred to as the amplitudes of the basis functions. The equations of motions of the amplitudes are derived using the TDSE. In order to be more computationally efficient the wave function uses the centre of the Gaussian basis functions and treats this as a particle like object. The spawning algorithm is also more computationally efficient as more basis functions can be generated in the simulation, if required, and so a large basis set is not required initially. The computational efficiency of the method allows the quantum dynamic technique to be to be combined with an electronic structure method.[18] For a system where the Hamiltonian is not known exactly the electronic structure method is able to approximate the potential energy, which is then used for the dynamical simulation. Combining these methods results in a powerful tool that can be used to describe the quantum dynamics of a system approximately.

A similar method called 'Coupled Coherent States' (CCS), which in principle describes the quantum dynamics exactly, is chosen for this work. The computational cost lies in between that of the MCTDH method and the multiple spawning technique.[19] The wave function is written as a linear combination of multidimensional frozen Gaussian basis functions as used in the vMCG technique. This is similar to the multiple spawning technique, but where a quantum averaged Hamiltonian is used to calculate the trajectories of the basis functions, which is different to the classical Hamiltonian used in the multiple spawning technique. The variational principle is used to calculate the trajectories of the individual basis functions, unlike the MCTDH technique, where the variational principle is applied to the whole wave function. The time dependent coefficients are coupled together, unlike in the semi-classical methods. The details of this theory are discussed in the following chapter. Although the cost of the calculation is greater using the CCS method compared to the multiple spawning method it is significantly lower than the MCTDH method. This allows the method to use a multidimensional wave function that describes a large system.

1.2 Applications

The MCTDH method is the most widely used quantum dynamical method that has been discussed and there are numerous recent examples of this being used in the literature.[20–22] Studies include calculations of pyrazine, malonaldehyde, allene and the butatriene cation.[23–28] Properties that were calculated include a Franck-Condon absorption spectra, tunnelling splittings and rate constants. These calculations are often used as benchmark for other calculations as the MCTDH method yields accurate results that are comparable with experimental data. A review of the applications of the MCTDH method would go beyond the scope of this work but the aim of the CCS method is to reach the accuracy of the MCTDH method, without the high computational cost.

Initially the CCS method was tested for a one dimensional problem in which the evolution of a Gaussian wave packet in a Morse potential was investigated.[29] The wave packet was expanded using a basis set of Gaussian functions in which the centres of the basis functions were distributed uniformly on a rectangular grid. It was found that if the grid of basis functions was stationary then the calculations were fast and gave accurate results. When the grid of basis functions was able to move the calculations took longer to run. The results were disappointing when the basis functions were chosen randomly.

An interesting area of Quantum Dynamics is the ability to simulate quantum tunnelling. This application is particularly interesting for the CCS method as it uses classical trajectories, whilst tunnelling is considered to be a quantum phenomenon. Thus the ability of CCS to simulate tunnelling shows the true quantum nature of the method. Sherratt *et al.* used two different multidimensional Hamiltonians, to investigate tunnelling in symmetric and asymmetric double well potentials.[30] The Hamiltonians consisted of a double well potential that was coupled to harmonic ‘bath’ modes and were used previously by Wu and Batista.[31] Unlike the previous study described, the basis functions were chosen randomly from a coherent state function, which matched the distribution of the initial wave function. Overall, the CCS calculations were shown to agree with work from other studies.[31] Therefore, this study is one example that shows that the CCS method can be used for multidimensional quantum problems where the basis functions are chosen randomly in a coherent state distribution.

More recently, Reed *et al.* have applied the CCS method to the $HOSO_2$ complex.[19]

This molecule is atmospherically interesting as it is a precursor for acid rain, which is formed from a chemical reaction of OH and SO_2. In general, it is assumed that the redistribution of internal energy of an excited complex occurs significantly faster than the rate of chemical reactions and dissociation. However, there are a number of cases where this does not hold true and understanding this process is vital to understanding the chemical dynamics of the dissociation process.[32] The CCS method was used to investigate the dissociation of $HOSO_2$ formed by the reaction of vibrationally excited OH with SO_2 and study the decay of the excited vibrational energy of OH. The system consists of 9 vibrational energy modes (or degrees of freedom), which are coupled together, and the potential energy surface was derived from an internal coordinate potential energy surface. It was found that most of the normal modes of the complex could be accurately described by Harmonic or Morse potentials or a modified version of these potentials. The basis functions that describe the wave function for a specified degree of freedom were chosen from a random distribution that represent the shape of the wave function for the degree of freedom. In most cases this was a Gaussian distribution, however, for the excited $v = 3$ OH energy mode an excited coherent state distribution was used. These CCS calculations showed that the dissociation of the complex is faster than the internal energy redistribution and this was consistent with classical molecular dynamic calculations.

The CCS method can be compared to the MCTDH and FGA/HK methods. The pyrazine molecule was a good benchmark for one study as it has 26 degrees of freedom and for a suitable benchmark to be used many degrees of freedom are required. This is because the computational calculations are more simple when fewer degrees of freedom are used and so similar results, with both methods, are expected. Furthermore, the absorption spectrum of pyrazine had been studied with the MCTDH, semi-classical FGA and HK methods. Shalashilin and Child calculated this with the CCS method and showed how the spectra calculated using the four different methods compared to the experimental spectrum.[33] The experimental spectrum shows three peaks where the large peak is at approximately 4.75 eV, a smaller shoulder peak is at approximately 4.6 eV and a broader shoulder peak is in the range 4.8 eV to 5 eV. All of the methods used show the larger peak at 4.75 eV. The FGA method, which is equivalent to the CCS method but in which uncoupled equations are used for the amplitudes, failed to give a good description of the shoulder peaks. The HK method, which supersedes the FGA method, gave improved results and the broad shoulder peak was described well, better than with the CCS method. However, the spectrum calculated with the CCS method gave smooth lines for the peaks whilst the HK spectrum consisted of many side peaks, which were most probably due to numerical noise and these can be confused to be shoulder peaks. The MCTDH spectrum shows a good description of the shoulder peaks but with smaller peaks found around these. Overall the accuracy of the spectrum calculated is in the order MCTDH $>$ CCS $>$ HK $>$ FGA. However, taking into account the significantly lower computational cost of the CCS method compared to the MCTDH, it is clear that CCS method is an efficient method, which is almost as accurate as the MCTDH method. Currently further calculations are being done on pyrazine with the CCS method to investigate if more accurate results can be achieved with better sampling.

Another study by Shalashilin and Child used different basis set sampling methods with the CCS method and compared spectra using the Henon-Heiles potential with the MCTDH method.[34] This work followed from a previous study that showed large errors in the results obtained with the CCS method, which was due to incorrect sampling of the basis set.[35] The corrected study used three sampling methods ('swarms', 'pancakes' and 'trains'), to calculate an autocorrelation function and a spectrum from this. The swarm and pancake sampling methods chose the basis functions randomly from different distributions whilst the train sampling methods used batches of basis functions positioned at specified points. The results of the calculations with 32 dimensions showed an almost exact agreement with the MCTDH method, whilst the computational cost of the CCS calculations were significantly lower than the MCTDH calculations. This study not only shows the significance of the sampling methods used for the CCS method but also shows that the CCS method is an efficient one that can be used to achieve the same accuracy as is possible with the MCTDH calculations.

The examples that use the CCS method discussed are a small selection of examples found in the literature.[36] These examples were chosen to show that the CCS method can be applied to a variety of systems, from small systems consisting of one or two degrees of freedom to larger systems that describe the wave function of molecules. The CCS method is therefore a promising, albeit relatively new method and it will be interesting to apply this to different chemical problems.

1.3 This Work

The following work describes the CCS method, its implementation and application. The key aim of the project was to write an original computer program, using the Fortran programming language, that implements the CCS method. The implemented software is provided on a CD and the method is discussed in Chapter 2. Tests using the simple one dimensional Free Particle and Harmonic Oscillator systems were shown to give low numerical errors. The software was then used to simulate the one dimensional Morse Oscillator system, which has a variety of applications, using a different sampling method for the basis functions compared with previous studies. Further work on the Harmonic and Morse Oscillator systems was carried out to obtain Franck-Condon spectra and the eigenvalues from these spectra were compared with theoretical values. A new method to decrease the computational cost of the simulation, based on symmetry, was also investigated.

One interesting area of application that had not been studied before using the CCS method is the Bose-Hubbard model. The reason for these calculations is that they can be used to model the tunnelling that exists when atoms or molecules are in a state of matter called the ‘Bose-Einstein Condensate’ (BEC). Other methods, such as the MCTDH method were not used as this is a two dimensional problem, which is computationally easier to calculate when compared with benchmark problems, and so similar results are expected with both methods. In addition to this, traditional Quantum Dynamic methods that do not use trajectory guided basis sets are not able to easily simulate the Bose-Hubbard model as a large number of static basis functions are required.

Chapter 2

Coupled Coherent States

The Coupled Coherent States theory, which is used in this work, is explained in this chapter. The motion of the wave function is described using the equations of motion shown here for the basis functions and the time dependent coefficients. The calculation of properties of the wave function are also discussed and the appropriate equations for this are shown.

2.1 The Wave Function

The initial premise of this method is that a wave function can be written that describes a system. This may be a complicated system consisting of many atoms or molecules or a more simple theoretical system. The CCS method assumes that the wave function can be written as a linear combination of time dependent basis functions multiplied by time dependent coefficients. This is shown below:

$$\Psi(t) = \sum_j B_j(t)\ \Phi_j(t)\ . \tag{2.1.1}$$

Although equation 2.1.1 is at first more complicated than the equations used in other methods, which use either time dependent coefficients or time dependent basis functions, allowing both the coefficients and the basis functions to vary with time decreases the overall computational cost within the CCS method, whilst remaining a fully quantum method.

For a multidimensional wave function, which is required to describe the wave function of a molecule, the basis functions, Φ are multidimensional. These are calculated from the one dimensional basis functions ϕ:

$$\Phi = \prod_\alpha \phi^{(\alpha)}\ , \tag{2.1.2}$$

where α is the degree of freedom. The general rule is that the lower case letter (b or ϕ) refers to the symbols for one dimensional wave functions and the upper case letters (B or Φ) refer to the symbols for multidimensional wave functions. The notation $^{(\alpha)}$ is also used to refer to the α degree of freedom. For simplicity a one dimensional wave function is used to derive the following equations in this chapter. However, most of the equations are essentially the same for the multidimensional case but with the upper case symbols replacing the lower case symbols.

2.2 Basis Functions

The Coupled Coherent States method uses coherent states $|z\rangle$ for the basis functions ϕ, which are non-orthogonal. These are defined as eigenstates of the non-Hermitian annihilation operator $\hat{a}$. The complex conjugate of this is the creation operator $\hat{a}^{\dagger}$. If the annihilation or creation operators act on an eigenstate $|n\rangle$ of the Hamiltonian operator, a new eigenstate of the Hamiltonian will be formed, $|n-1\rangle$ or $|n+1\rangle$, which will have an energy below or above the original eigenstate $|n\rangle$. However, defining the coherent state $|z\rangle$ as an eigenstate of the annihilation operator gives:

$$\hat{a}|z\rangle \quad = \quad z|z\rangle \tag{2.2.1}$$

and

$$\langle z|\hat{a}^{\dagger} \quad = \quad \langle z|z^{*} \; . \tag{2.2.2}$$

The eigenvalue z is a complex dimensionless number:

$$z = \left(\frac{\gamma}{2}\right)^{\frac{1}{2}} q + \frac{i}{\hbar}\left(\frac{1}{2\gamma}\right)^{\frac{1}{2}} p \; , \tag{2.2.3}$$

where q gives the position and p gives the momentum of the centre (or expectation value) of the coherent state and γ is a constant that describes the shape of the coherent state. The constant γ is also known as a squeezing parameter. The complex conjugate is:

$$z^{*} = \left(\frac{\gamma}{2}\right)^{\frac{1}{2}} q - \frac{i}{\hbar}\left(\frac{1}{2\gamma}\right)^{\frac{1}{2}} p \; . \tag{2.2.4}$$

The annihilation operator is:

$$\hat{a} \quad = \quad \left(\frac{\gamma}{2}\right)^{\frac{1}{2}} \hat{q} + \frac{i}{\hbar}\left(\frac{1}{2\gamma}\right)^{\frac{1}{2}} \hat{p} \; , \tag{2.2.5}$$

and the creation operator is:

$$\hat{a}^\dagger \;=\; \left(\frac{\gamma}{2}\right)^{\frac{1}{2}} \hat{q} - \frac{i}{\hbar}\left(\frac{1}{2\gamma}\right)^{\frac{1}{2}} \hat{p}\,. \tag{2.2.6}$$

The operators $\hat{q}$ and $\hat{p}$ are the position and momentum operators respectively.

The projection of the coherent state $|z\rangle$ onto the position coordinate x yields:

$$\langle x|z\rangle = \left(\frac{\gamma}{\pi}\right)^{\frac{1}{4}} \exp\left(-\frac{\gamma}{2}(x-q)^2 + \frac{i}{\hbar}p(x-q) + \frac{ipq}{2\hbar}\right)\,. \tag{2.2.7}$$

The derivation of this equation, which goes beyond the scope of this work, is shown by Louisell.[37] This equation shows that the envelope of the coherent state is a complex Gaussian function in position representation and that the parameter γ affects the shape (or degree of squeezing) of the Gaussian function.

The position and momentum of the centre of the basis function can be written in terms of z and its complex conjugate z^*, as:

$$q \;= \left(\tfrac{1}{2\gamma}\right)^{\frac{1}{2}} (z^* + z) \tag{2.2.8}$$

and

$$p \;= i\hbar \left(\tfrac{\gamma}{2}\right)^{\frac{1}{2}} (z^* - z) \quad . \tag{2.2.9}$$

The complex vector comprising each eigenvalue z for all the basis functions is written as $\mathbf{z}$ and in accordance with the rule for multidimensional wave functions $\mathbf{Z}$ refers to an array that consists of each z for each degree of freedom.

2.3 Coefficients

The one dimensional coefficients, b, can be split into a smooth function and an oscillatory phase part to give:

$$b(t) = d(t)\exp\left(\frac{i}{\hbar}s(t)\right)\,, \tag{2.3.1}$$

where s is the time dependent classical action and d is a smooth function of time. Shalashilin *et al.*[14] have shown that this form can be derived by applying Hamilton's principle of least action onto the trajectory of the coefficient of the basis function b. It is convenient to use the coefficients in this form when using numerical techniques implemented in a computer program. Another convenient form is using the coefficients c. These can be calculated from the coefficients d using

$$c_j = \sum_k \langle z_j|z_k\rangle d_k \exp\left(\frac{i}{\hbar}(s_k - s_j)\right) , \tag{2.3.2}$$

where $\langle z_j|z_k\rangle$ is the overlap between the basis functions k and j. This is:

$$\langle z_j|z_k\rangle = \exp\left(z_j^* z_k - \frac{1}{2}z_k^* z_k - \frac{1}{2}z_j^* z_j\right) . \tag{2.3.3}$$

The difference between c and d occurs due to the basis functions being diffuse functions which overlap. There is also a difference in the phase due to the different indexes that are used in each coefficient, which results in a difference of the action. The coefficients c are a convenient form to use for a physical description. This is because it can be shown that the projection of the wave function Ψ onto the basis function labelled z yields:

$$\langle z_j|\Psi\rangle = c_j \exp\left(\frac{i}{\hbar}s_j\right) . \tag{2.3.4}$$

This gives the amplitude of the basis function. However, the term 'amplitude' is used in the literature for either of the coefficients b, c or d as it is possible to use the equations shown to convert from one coefficient to another.[14,19]

2.4 Equations of Motion

The coherent state basis functions are expected to follow classical like trajectories in phase space as shown by Heller in the Frozen Gaussian Approximation.[4] Therefore Hamilton's classical equations of motions are used. The fact that coherent states follow classical like trajectories is the main advantages of using a basis set of coherent states. In the z notation, the equations of motion are:

$$\dot{z}_n = -\frac{i}{\hbar}\frac{\partial H_{ord}(z_n^*, z_n)}{\partial z_n^*} \tag{2.4.1}$$

and

$$\dot{z}_n^* = \frac{i}{\hbar}\frac{\partial H_{ord}(z_n^*, z_n)}{\partial z_n} . \tag{2.4.2}$$

The dot indicates the derivative with respect to time.

The function $H_{ord}(z_n^*, z_n)$, in the above equations, is an averaged or reordered Hamiltonian function. In general, a Hamiltonian operator, in terms of position and momentum operators, $\hat{q}$ and $\hat{p}$ respectively, can be rewritten in terms of the creation and annihilation operators.

The H_{ord} function is derived from the quantum Hamiltonian, $\hat{H}$, by multiplying the Hamiltonian by $\langle z_2|$ and $|z_1\rangle$ and using equations 2.2.1 and 2.2.2. This gives:

$$\langle z_2|\hat{H}|z_1\rangle = \langle z_2|z_1\rangle H_{ord}(z_2^*, z_1) \ , \tag{2.4.3}$$

where $\langle z_2|z_1\rangle$ is the overlap between the basis functions labelled z_2 and z_1. This is given in equation 2.3.3. To use equations 2.2.1 and 2.2.2, the Hamiltonian operator, written in terms of the creation and annihilation operators, must be reordered such that the creation operator acts on the bra $\langle z_2|$ and the annihilation operator acts on the ket $|z_1\rangle$.

A simple example to show Hamiltonian reordering can be shown for the Hamiltonian:

$$\hat{H} = \hat{a}\hat{a}^\dagger. \tag{2.4.4}$$

Multiplying by $\langle z_2|$ and $|z_1\rangle$ gives:

$$\langle z_2|\hat{H}|z_1\rangle = \langle z_2|\hat{a}\hat{a}^\dagger|z_1\rangle \ . \tag{2.4.5}$$

Equation 2.4.3 cannot be used when the Hamiltonian is ordered in this way, as neither $\hat{a}^\dagger|z\rangle$ nor $\langle z|\hat{a}$ have been defined, and so it must be reordered using the commutator relationship $\hat{a}\hat{a}^\dagger - \hat{a}^\dagger\hat{a} = 1$. This yields:

$$\langle z_2|\hat{H}|z_1\rangle = \langle z_2|1 - \hat{a}^\dagger\hat{a}|z_1\rangle \tag{2.4.6}$$

$$= \langle z_2|z_1\rangle H_{ord}(z_2^*, z_1) \ . \tag{2.4.7}$$

The H_{ord} function is therefore given by:

$$H_{ord} = 1 - z_2^* z_1 \ , \tag{2.4.8}$$

where $z_2^*\langle z_2|$ and $|z_1\rangle z_1$ replace $\langle z_2|\hat{a}^\dagger$ and $\hat{a}|z_1\rangle$, respectively. The H_{ord} function can be regarded as an averaged Hamiltonian function as it uses z, which consists of the expectation values for the position and momentum of the coherent state.

To derive the equations of motions for the coefficients b, the wave function

$$|\Psi\rangle = \sum_j b_j|z_j\rangle \tag{2.4.9}$$

is substituted into the TDSE

$$\frac{\partial|\Psi\rangle}{\partial t} = -\frac{i}{\hbar}\hat{H}|\Psi\rangle \ . \tag{2.4.10}$$

This is differentiated, multiplied by $\langle z_k|$ and rearranged to give:

$$\sum_j \langle z_k|z_j\rangle \dot{b}_j = -\frac{i}{\hbar}\sum_j \langle z_k|\hat{H}|z_j\rangle b_j - \sum_j \langle z_k|\dot{z}_j\rangle b_j \ . \tag{2.4.11}$$

Using equation 2.4.3 yields:

$$\sum_j \langle z_k|z_j\rangle \dot{b}_j = -\frac{i}{\hbar}\sum_j \langle z_k|z_j\rangle H_{ord}(z_k^*, z_j) b_j - \sum_j \langle z_k|\dot{z}_j\rangle b_j \ . \tag{2.4.12}$$

This equation can be used to find the equation of motion in terms of the coefficient d_j. In order to do this, equation 2.3.1 must be used with the time derivative of the action i.e. the Lagrangian:

$$\dot{s_j} = \frac{i\hbar}{2}\left(z_j^*\dot{z}_j - \dot{z}_j^* z_j\right) - H_{ord}(z_j^*, z_j) \ . \tag{2.4.13}$$

In addition to this

$$\langle z_k|\dot{z}_j\rangle = (z_k^* - z_j^*)\dot{z}_j + \frac{z_k^*\dot{z}_k}{2} + \frac{z_k\dot{z}_j^*}{2} \tag{2.4.14}$$

is required. This yields:

$$\begin{aligned}\sum_j \dot{d}_j \exp\left(\frac{i}{\hbar}s_j\right)\langle z_k|z_j\rangle \ &= \ \frac{i}{\hbar}\sum_j d_j \exp\left(\frac{i}{\hbar}s_j\right)\langle z_k|z_j\rangle \\ &\times \left(H_{ord}(z_j^*, z_j) - H_{ord}(z_k^*, z_j) + \frac{\partial H_{ord}(z_j^*, z_j)}{\partial z_j^*}(z_k^* - z_j^*)\right).\end{aligned} \tag{2.4.15}$$

Defining

$$\delta^2 H_{ord}(z_j^*, z_k) = \left(H_{ord}(z_j^*, z_j) - H_{ord}(z_k^*, z_j) + \frac{\partial H_{ord}(z_j^*, z_j)}{\partial z_j^*}(z_k^* - z_j^*)\right) \tag{2.4.16}$$

gives:

$$\sum_j \dot{d}_j \exp\left(\frac{i}{\hbar}s_j\right)\langle z_k|z_j\rangle \ = \ \frac{i}{\hbar}\sum_j d_j \exp\left(\frac{i}{\hbar}s_j\right)\langle z_k|z_j\rangle \delta^2 H_{ord}(z_j^*, z_k) \ . \tag{2.4.17}$$

The index j in the summation on the right hand side of the equation can be changed to ℓ :

$$\sum_j \dot{d}_j \exp\left(\frac{i}{\hbar}s_j\right)\langle z_k|z_j\rangle = \frac{i}{\hbar}\sum_\ell d_\ell \exp\left(\frac{i}{\hbar}s_\ell\right)\langle z_k|z_\ell\rangle\delta^2 H_{ord}(z_\ell^*, z_k)\ . \tag{2.4.18}$$

This clearly shows that terms in the summations on each side of the equation do not cancel.

Using the equations of motion for the trajectories of the basis functions and their amplitudes, the wave function can be propagated. This is done by solving the differential equations 2.4.1, 2.4.13 and 2.4.18 using a numerical integration technique, such as the fourth order Runge-Kutta technique, which is discussed in the next chapter.

2.5 Properties

Throughout the simulation of a wave function, a number of properties can be calculated. These include the total classical energy of the basis functions, the total quantum expectation energy of the wave function, the norm of the wave function and an autocorrelation function.

The classical energy of the basis function j is given by $H_{ord}(z_j^*, z_j)$. The total classical energy for the basis set is calculated as the sum of the classical energy of each basis function in the basis set, i.e. $\sum\limits_j H_{ord}(z_j^*, z_j)$. This classical energy will be conserved well throughout the simulation as classical trajectories are used for the basis functions and so this is in accordance with the classical laws of physics.

The total quantum expectation energy of the wave function is calculated by using the Hamiltonian operator on the wave function as:

$$\langle\Psi|\hat{H}|\Psi\rangle = \sum_k\sum_j d_k^* d_j \exp\left[\frac{i}{\hbar}(s_j - s_k)\right] H_{ord}(z_k^*, z_j)\langle z_k|z_j\rangle\ . \tag{2.5.1}$$

This property is different from the total classical energy as it uses the quantum Hamiltonian operator on the whole wave function, whilst the total classical energy uses the ordered Hamiltonian function (H_{ord}) with each z. Unlike the total classical energy, the total quantum expectation energy is not conserved here. This is because the Hamiltonian operator is not used to derive the equations of motion for the whole wave function, as the basis functions follow Hamilton's classical equations of motion.

The term 'norm' is used for a property that depends on the coefficients in addition to the positions and momenta of the basis functions. It is calculated by multiplying the wave function

$$|\Psi\rangle = \sum_j d_j \ \exp\left(\frac{i}{\hbar}s_j\right)|z_j\rangle \tag{2.5.2}$$

with its complex conjugate

$$\langle\Psi| = \sum_k d_k^* \ \exp\left(-\frac{i}{\hbar}s_k\right)\langle z_k| \tag{2.5.3}$$

to give:

$$\langle\Psi|\Psi\rangle \ = \ \sum_k\sum_j d_k^*\langle z_k|z_j\rangle \exp\left[\frac{i}{\hbar}(s_j - s_k)\right] d_j \ . \tag{2.5.4}$$

For a wave function that is normalised appropriately, the norm should be equal to 1. This property indicates how well the wave function is described by the basis functions and the coefficients, and gives the overlap between the wave function (Ψ) with that calculated from the basis set. If it is not equal to 1 initially then the initial basis functions and the amplitudes do not give an appropriate description of the wave function and these must be reinitialised. In addition to this, if the norm changes from 1 during the simulation then the simulation cannot be regarded as reliable. This conservation of the norm arises from the equations of motions for the coefficients.

In general, an autocorrelation function (ACF) is a function that correlates a function at time t with itself at time 0. This can be used to determine the overlap of the initial wave function with that at time t. The equation that describes this for a coherent state (normalised) wave function is similar to the norm as can be seen by comparing equation 2.5.4 with

$$\langle\Psi(0)|\Psi(t)\rangle = \sum_k\sum_j d_k^*(0)\exp\left[\frac{i}{\hbar}(s_j(t))\right] d_j(t)\langle z_k(0)|z_j(t)\rangle \ . \tag{2.5.5}$$

The difference between equations 2.5.4 and 2.5.5 is the fact that the wave function in the bra is at the initial time 0 and so all the terms associated with this are also at the initial time. For any system, the ACF will be at its peak initially as the wave functions are equal. As the wave function moves along the trajectory the ACF typically decreases as the overlap between the two wave functions decrease. However, for an oscillator (either Harmonic or Morse) the ACF increases again and peaks as the wave function returns to the initial position to start another period of oscillation.

In addition to calculating the ACF to investigate how the wave function moves over time, the ACF is also required in order to calculate a Franck-Condon absorption spectrum. This can be used to find the eigenvalues for the vibrational energy levels of the Harmonic or Morse Oscillators. The spectrum is calculated by taking the Fourier transform of the ACF.[6] The Fourier transform function $F(E)$ is therefore

$$F(E) = \frac{1}{2\pi\hbar}\int_{-\infty}^{\infty} \exp\left(\frac{iE(t)}{\hbar}\right)\langle\Psi(0)|\Psi(t)\rangle dt \qquad (2.5.6)$$

The graph of the function is a Franck-Condon spectrum, where the peaks of this correspond to the eigenvalues of the energy levels of the system that is being investigated.

2.6 Summary

The CCS theory and the properties of the wave function discussed in this chapter explain the theoretical methods used for the simulations investigated. The equations shown here must must be implemented into a Fortran computer code so as to be used to run the simulations and investigate chemical problems.

Chapter 3

Implementation

This chapter first discusses some numerical techniques used in programming and then describes the original Fortran computer software that implements the CCS method. The fourth order Runge-Kutta technique, used to solve the CCS equations of motion, and a simple Fourier transform technique, required to calculate a Franck-Condon spectrum, are also described. An overview of the software is given, followed by a description of the individual modules used in the main program.

3.1 Numerical Integration

To simulate the wave function using the CCS method the equations of motions shown in the previous chapter must be solved. For certain potential functions such as the Harmonic Oscillator or the Morse Oscillator an analytical solution is possible. However, this is not always the case and so it is preferable to write a general code that uses an accurate approximate numerical integration technique that can be used for a variety of systems. The numerical integration technique discussed here is the fourth order Runge-Kutta technique. Previous studies have shown that this technique can provide accurate and efficient results for the CCS equations of motion.[29]

The Runge-Kutta technique provides a numerical solution to the integral of a function $y(x)$, where it is assumed that the differential function

$$y'_n = f_n(x, y) \ . \tag{3.1.1}$$

The Runge-Kutta method provides a numerical solution for the y_{n+1} value using the previous value y_n. The initial value of y_0 is required. The solution is given by

$$y_{n+1} = y_n + R \ . \tag{3.1.2}$$

R is used to give a good description of the gradient to the function. For the fourth order Runge-Kutta method R is calculated from:

$$R = \frac{1}{6}\left(r_1 + 2r_2 + 2r_3 + r_4\right) \ , \tag{3.1.3}$$

with

$$\begin{aligned}
r_1 &= \delta f(x_n, y_n); && (3.1.4)\\
r_2 &= \delta f\left(x_n + \frac{1}{2}\delta, y_n + \frac{1}{2}r_1\right); && (3.1.5)\\
r_3 &= \delta f\left(x_n + \frac{1}{2}\delta, y_n + \frac{1}{2}r_2\right); && (3.1.6)\\
r_4 &= \delta f\left(x_n + \delta, y_n + r_3\right) \ , && (3.1.7)
\end{aligned}$$

where δ is the time step that determines how far forward from the starting point the value is advanced.[38] A smaller value of δ indicates that the solution is not advanced far and so is less likely to deviate from the exact solution, whilst a larger value of δ will advance further along the function and thus more likely to deviate from the solution. An error is introduced and this is compounded with the error from the previous value calculated as the previous point is used to calculate the next. However if δ is too small, a large number of steps must be taken in order to calculate the same range of the function and thus the time of the calculation can significantly increase. Numerical difficulties also arise from using steps that are too small.

The method works by calculating a weighted mean gradient over the time step using the gradients r_1 to r_4. The gradient at the beginning of the time step is given by r_1 and r_2 is the gradient at the midpoint of the time step, calculated using r_1. The r_3 value is also the gradient at the midpoint but calculated using r_2 and r_4 is the gradient at the end point using r_3. Finally, R is calculated by using a weighted average of r_1 to r_4. This is found to give a good description of the slope and thus an accurate solution. It is possible to change the order of the Runge-Kutta method, but fourth order is found to give accurate solutions and is the most popular.

3.2 Fourier Transform

In the previous chapter, the use of a Fourier transform to calculate a Franck-Condon spectrum was discussed. The integral that gives the Fourier transform of the autocorrelation function, for a normalised wave function, is:

$$F(E) = \frac{1}{2\pi\hbar}\int_{-\infty}^{\infty} \exp\left(\frac{iE(t)}{\hbar}\right)\langle\Psi(0)|\Psi(t)\rangle dt \ . \tag{3.2.1}$$

This can be approximated using the discrete sum:

$$F(E) = \frac{1}{2\pi\hbar} \sum_j \exp\left(\frac{iE(t_j)}{\hbar}\right) \langle\Psi(0)|\Psi(t_j)\rangle\Delta t \ , \tag{3.2.2}$$

where Δt is the time step. This method is not the most computationally efficient method but it is accurate and also simple to implement. Furthermore, this is used in the analysis of the results, rather than during the simulation where efficiency in the code is more important.

One property of spectra calculated from a Fourier transform is that many smaller background peaks are observed. These occur due to the numerical calculations and the spectrum with these peaks is not clear and thus difficult to analyse. In order to produce a clear Fourier transform the function, which is Fourier transformed, must be converged. In order to do this a window function may be used. In general a window function is one that will give a value of zero for the function outside a specified range. Therefore, a clear Franck-Condon spectrum can be obtained by forcing convergence of the ACF to zero and multiplying the function beyond this point by zero. This is obtained by multiplying the ACF by:

$$W(t) = \begin{cases} \cos^2\left(\frac{2\pi t}{4t_{cut}}\right) & : t < t_{cut} \\ 0 & : t \geq t_{cut} \end{cases} \tag{3.2.3}$$

where t is the time and t_{cut} is the maximum time allowed. Previously this function has produced clear Franck-Condon absorption spectra.

3.3 Fortran

The Fortran programming language was originally written as a numerical programming language. It is used to implement the CCS method and many other methods in theoretical chemistry due to its efficiency in numerical calculations. However, it is not an object orientated language, which is the case for many modern computer programs. Object orientated programming is when the program is written using objects that have been defined. For example, the C++ programming language defines an object in a ‘class’ and uses the class in the program. This is in contrast to the modular structure of the Fortran programming language where different modules often have specified tasks rather than defining a data structure as an object. As more modern programming languages are object orientated the Fortran language has developed considerably in this area.

Key features of the Fortran programming language include the predefined complex number type, which is used extensively in the code, and the ability to handle arrays as a whole data structure, e.g. $A(:) = 0$ sets the entire array to zero. Fortran 2003 is used to take advantage of some of the newer developments of Fortran.[39] One of these developments is the ability to define a new allocatable type. This is used in the program by defining a new data type that consists of an allocatable array of complex numbers for the z values, where there is one complex number per degree of freedom. In addition to this array, there is a complex number for the coefficient D of the basis function and a real number for the action S of the basis function. The type therefore consists of the information that labels a single basis function and an array of this type gives the basis set. The advantage of this is that the basis set can be called into different subroutines as an object. This is particularly useful for the Runge-Kutta integration technique, where the differential equations require the whole basis set. The derived type and the Runge-Kutta technique can be found in Appendix C. The disadvantage of using the defined type is that the latest compilers are required to compile the program.

3.4 Solving Linear Equations

To solve the equations of motion for the amplitudes (equation 2.4.18) a set of linear equations must be solved. Analytically, a set of linear equations can be solved using matrix algebra, where one matrix must be inverted. For these equations, the matrix that must be inverted is largely due to the overlap of the non-orthogonal basis functions shown on the right hand side of equation 2.4.18. These equations are numerically difficult to solve when some values are very small and approach zero or when degenerate values in the matrix occur. When this occurs it is numerically difficult to calculate the inverse of the matrix and thus to solve the set of linear equations. The computational difficulty arises due to round off errors in the storage of the matrix, caused by insufficient precision to accurately represent the values. In order to reduce the round off errors double precision has been used to store the data. This allocates double the memory for storing the data compared with the standard single precision available in Fortran. In addition to this the LAPACK routine ‘ZGESV’ has been used, which has been specifically written to solve a set of linear equations for complex, double precision data types. This routine decreases the computational cost and increases the accuracy of the calculations compared to explicitly inversing the matrix. Note that this routine is also required to solve equation 2.3.2.

3.5 Software Overview

The code consists of one main program and two smaller subsidiary programs. An overview of the code is given here, followed by a description of the individual modules in the main program. More details on compiling and running the code can be found in Appendix B.

The overall procedure to run a simulation using the software is:

1. Using one of the subsidiary programs to read in parameters and configure a basis set. This gives a basisset.dat output file.
2. Using the main program:
    (a) Read in the inputs from the basisset.dat and a second input file, which contains parameters for the system investigated.
    (b) Calculate initial parameters, such as the classical energy, from the basis set.
    (c) Propagate the basis functions and calculate properties.
3. Use a subsidiary program to calculate the Fourier transform of the autocorrelation function results, if required.

One advantage of using subsidiary programs to calculate the basis set and the Fourier transform is that the main program is smaller and so easier to follow. Additionally the same basis sets can be used for different calculations. See Appendix A for a flow chart of the procedure described above.

3.5.1 Basis Set

The basis set is configured in a smaller subsidiary program, which allows the same basis sets to be used for a variety of calculations. This program has been written to choose a random number from a Gaussian distribution. The basis functions are chosen randomly from the initial wave function. The initial wave function is defined in the program to be a coherent state centred at z_0 in phase space. The random numbers from the Gaussian distribution are chosen using a subroutine written by Chandler *et al.*[40] The region of phase space that the basis functions are chosen from can be altered using a standard deviation parameter. Additional optional restrictions such as the energy of the basis functions can also be set.

The main advantage of initialising the basis set in this way is that it is efficient and it provides a good approximate solution for many systems. However, such a basis set is not always ideal as it does not always describe the wave function exactly. In order to provide a better approximation for other system the basis functions must be distributed such that they correspond to the shape of the wave function. A different program has already been written by Reed that can be used to do this for excited states of the Harmonic Oscillator.[41]

3.5.2 Calculating Properties

As discussed in the previous chapter, a wide range of properties can be calculated during the simulation. The properties that have been implemented are: the classical energy of the basis set, the norm of the wave function, the autocorrelation function and the Franck-Condon absorption spectrum. These are calculated using the equations already shown. The norm and the classical energy are useful properties as these should be conserved well throughout the simulation. Therefore, calculating these properties tests the accuracy and reliability of the simulations. The Fourier transform of the autocorrelation function, that gives the Franck-Condon spectrum, is calculated after with a subsidiary program. The advantage of this is that other routines for the window function or the Fourier transform can be easily implemented without affecting the main program and running the simulation again.

3.5.3 Propagating The Basis Set

The wave function is propagated in the main program by propagating the basis functions. This is done by using the fourth order Runge-Kutta numerical integration technique, described previously. To propagate the basis functions the time derivatives of $\mathbf{Z}$, S and D in the CCS equations of motion (equations 2.4.1, 2.4.2, 2.4.13 and 2.4.18), given in the previous chapter are calculated in the computer program and used in the integration routine to calculate new values of $\mathbf{Z}$, S and D. A loop is required in the main program that increments time and calls the integration technique, which continues till the maximum time of the simulation is reached.

3.6 Modules Description

The main program is split into six modules. These are called: ‘BSET’, ‘Integration’, ‘HamiltonianSpec’, ‘GenHam’, ‘OutMod’ and ‘ACFMOD’. Each module is written so that a change in one module will not result in large changes in the main program or in other modules. The purpose of each module is discussed in the follow subsections. Figure A.2 in Appendix A shows how the main program calls these modules.

3.6.1 Module: BSET

This module configures the basis set. This involves defining the new basis function type, which can be seen in Appendix C. Defining the type here means that the module is required by the other modules and the main program. The module consists of subroutines that read in the parameters required for the basis set, allocate the memory for storing the basis set and read in the values of $\mathbf{Z}$ and C. The module also includes of a subroutine that will calculate the initial D value from the $\mathbf{Z}$ and C values provided. This is done using

LAPACK routines to solve:

$$C_j = \sum_k \Omega_{jk} D_k \ . \tag{3.6.1}$$

Note that C is calculated in the subsidiary program that creates the basis set as LAPACK routines are not required but then converted to D in this module as D is easier to use. When the initial time of the simulation is at $t = 0$ au the S value is initialised to 0 au also.

3.6.2 Module: Integration

The Runge-Kutta routine and the subroutine that returns the time derivatives of **Z**, S and D are written in this module. If a different numerical integration technique is required it can be written here. The derivs routine and the relatively simple Runge-Kutta routine are found in Appendix C.

3.6.3 Module: GenHam

This module provides a generic interface between the Integration module, which uses the H_{ord} function and its derivatives, and the HamiltonianSpec module. The name of the system is read from the input file and the appropriate function is linked to the functions that are used in the Integration module.

3.6.4 Module: HamiltonianSpec

The H_{ord} functions and its derivative, with respect to z or z^*, for specific systems are given in this module. The systems which have been included are the Free Particle, Harmonic Oscillator, Morse Oscillator and the Hubbard Model.

3.6.5 Module: OutMod

The subroutines which produce the output files for the classical trajectories of the basis functions, the conservation of the classical energy and the norm are written here. To do this the names of the output files are read in from the input files.

3.6.6 Module: ACFMOD

This module has subroutines to calculate the autocorrelation function and to write the output of the autocorrelation function results. A subroutine that reads in if the autocorrelation results are required is also written here. This decreases the computational cost of the program if the autocorrelation function does not need to be calculated.

3.7 Summary

A well structured code has been written that implements the CCS method. New features of the Fortran programming language are used to keep the structure of the main program simple. In addition to this, two smaller subsidiary programs have also been written, which initialise the basis set and analyse the output. Testing the software is discussed in the next chapter.

Chapter 4

Testing The Code

The Fortran CCS software was tested thoroughly for the one dimensional Free Particle, Harmonic Oscillator and Morse Oscillator systems. These systems are described in this chapter and the results showing the conservation of the norm and the classical energy of the basis set are discussed. The trajectories of the basis functions for all three systems are shown in phase space.

4.1 Free Particle

The free particle system describes the motion of a system (particle or wave function) that is not affected by a force. There is no potential energy term in the Hamiltonian and so it is given by:

$$\hat{H} = \frac{\hat{p}^2}{2m} , \tag{4.1.1}$$

where $\hat{p}$ is the momentum operator and m is the mass of the system. The operator $\hat{p}$ can be replaced by the creation and annihilation operators by using equations 2.2.5 and 2.2.6. The resulting expression must then be reordered appropriately. Equation 2.4.3 is then used, where the H_{ord} function is:

$$H_{ord} = -\frac{1}{4m\gamma\hbar}(z - z^*)^2 . \tag{4.1.2}$$

As there is no potential energy in the Hamiltonian, the particle in this system will maintain the same velocity throughout its trajectory.

4.2 Harmonic Oscillator

The Harmonic Oscillator system is well-known in both classical and quantum mechanics. In this system, a restoring force is applied that pulls the system back towards its equilibrium position. The Hamiltonian is:

$$\hat{H} = \frac{\hat{p}^2}{2m} + \frac{k\hat{q}^2}{2} \tag{4.2.1}$$

where $k = m\omega^2$ is the force constant. The corresponding H_{ord} function is:

$$H_{ord} = \hbar\omega \left(zz^* + \frac{1}{2} \right) \ . \tag{4.2.2}$$

This is derived using the creation and annihilation operators to replace the $\hat{p}$ and $\hat{q}$ operators, as shown previously. The energy levels of this system are equally spaced and the eigenvalues of these levels are:

$$\epsilon = \hbar\omega \left(n + \frac{1}{2} \right) \ , \tag{4.2.3}$$

with the integer n starting from 0. Figure 4.1 shows the first twelve energy levels of the Harmonic Oscillator and how the potential energy increases as the position increases. This system can be used to describe the vibrational energy levels of a diatomic molecule.

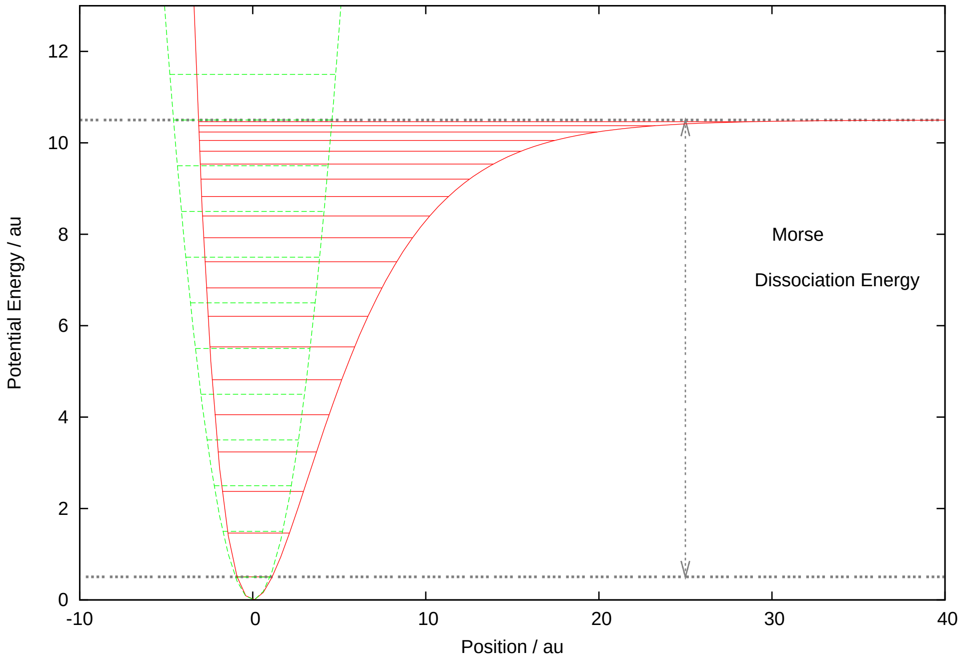

Figure 4.1: The potential energy of the Harmonic Oscillator, equation 4.2.1, (green) using $|\hbar| = |k| = |m| = 1$ and the Morse Oscillator, equation 4.3.1, (red) using $D_0 = 10.5$ au and $a_0 = 0.22$ au. The horizontal lines indicate the vibrational energy levels on each of the potential functions.

4.3 Morse Oscillator

A better approximation to the describe the vibrational energy levels of a diatomic molecule is the Morse Oscillator. The Hamiltonian is:

$$\hat{H} = \frac{\hat{p}^2}{2m} + D_0(1 - \exp(-a_0\hat{q}))^2 \ , \tag{4.3.1}$$

where D_0 is the dissociation energy and a_0 is a parameter that determines the shape of the potential. The H_{ord} function for this system is:

$$\begin{aligned} H_{ord} \quad = \quad & -\frac{1}{4m^2\omega}(z - z^*)^2 + D_0\left(1 + \exp\left(-\sqrt{2}a_0(z^* + z)\right)\exp({a_0}^2)\right) \\ & -2D_0 \exp\left(-\frac{a_0}{\sqrt{2}}(z^* + z)\right)\exp\left(\frac{{a_0}^2}{4}\right) \ . \end{aligned} \tag{4.3.2}$$

The first term is simply the kinetic energy, given by the free particle system. The second term is derived by replacing the operator $\hat{q}$ by the creation and annihilation operators and using $\exp(\hat{A} + \hat{B}) = \exp(\hat{A})\exp(\hat{B})\exp -[A, B]$. The eigenvalues of the energy levels are:

$$\epsilon = \hbar a_0\sqrt{2D_0/m}\left(n + \frac{1}{2}\right) - \frac{(\hbar a_0)^2}{2m}\left(n + \frac{1}{2}\right)^2 \ . \tag{4.3.3}$$

The first term in this equation gives the Harmonic Oscillator levels and these are corrected by the additional quadratic term. This extra term decreases the spacing of the energy levels as the potential energy increases towards the dissociation energy. This is shown in figure 4.1 where the dissociation energy is at 10.5 au and is indicated by the grey lines.

4.4 Computational Set Up

The units used in the code and the ones which are reported in this work are atomic units (au), defined as $|\gamma| = |\hbar| = |m| = |\omega| = 1$. The parameters for the one dimensional Hamiltonians were $k = 1$ au for the Harmonic Oscillator, and for the Morse Oscillator $D_0 = 10.25$ au and $a_0 = 0.2209$ au. The initial conditions of the wave function were chosen to match an earlier study in which $q_0 = 5$ au and $p_0 = 0$ au. The basis functions used to represent this were chosen randomly from Gaussian functions with the mean of the Gaussians equal to q_0 and p_0 of the initial wave function. The number of basis functions and the degree of overlap between them differs between the calculations and the tests required. For testing the norm and the classical energy, the initial time was set at 0 au and the maximum time was 20 au. The trajectory calculations were given a longer maximum time of 50 au for the simulation. Note that the norm is a dimensionless quantity.

4.5 Initial Norm

It is clear, from the discussion of the norm given previously, that the norm of the wave function should be equal to 1. When the basis functions are chosen randomly, increasing the number of basis functions in a given area, will increase the norm so that it is closer to 1. This was checked by using a fixed standard deviation parameter of 1 for the Gaussian function, from which the basis functions are chosen, and using $1, 2, 4, 6, 8, 10$ and 20 basis functions. The calculations were repeated 10 times, with different basis sets, and the mean was taken. The mean initial norm values, calculated with $1, 2, 4, 6, 8$ and 10 basis functions, are shown in figure 4.2.

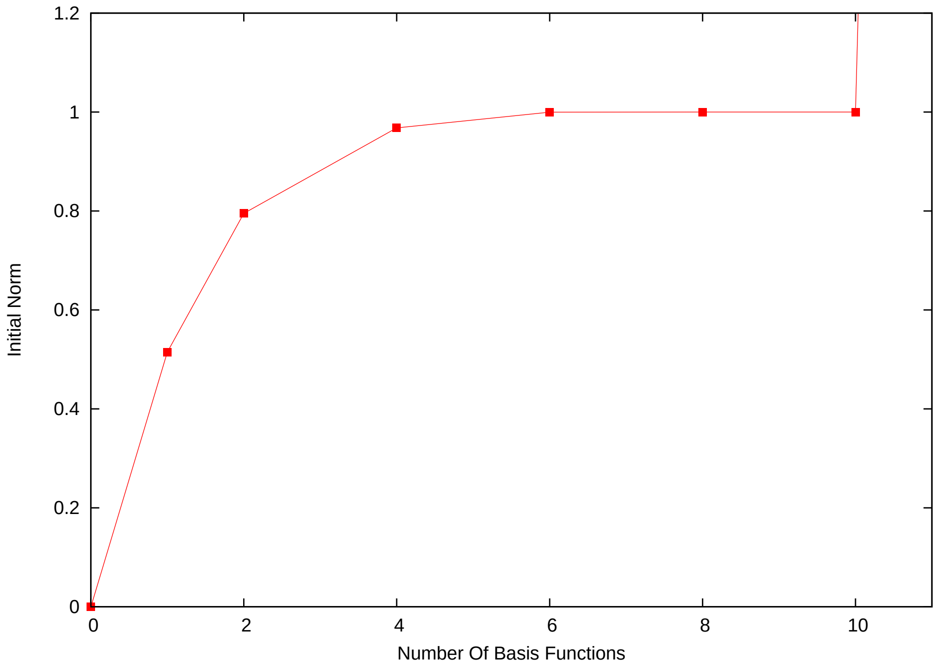

Figure 4.2: Convergence of the norm towards 1 as the number of basis functions increase. Each point is a mean average over 10 different basis sets, chosen randomly from a Gaussian distribution with a standard deviation of 1. Numerical difficulties arise with more than 10 basis functions.

It can be seen that the norm increases and is approximately 1 with 6 to 10 basis functions. There is a sharp increase in the norm after 10 basis functions as the mean norm calculated with 20 basis functions is very large (approximately 59). This occurs due to numerical errors in the calculations, which arise as the overlap matrix becomes more difficult to calculate due to the large overlap between the basis functions. This shows that the software initialises the basis set correctly, providing that there is not too high a degree of overlap between the basis functions. To decrease the overlap between the basis functions a larger spread, which is determined by the standard deviation parameter, of the Gaussian function is required.

4.6 Conservation

The conservation of the classical energy and the norm is very important and the code was tested thoroughly to check that these properties were conserved throughout the simulation. The same basis sets and initial conditions were used as described earlier for all systems. The average initial classical energy and the average norm were calculated for the 10 repetitions using $1, 2, 4, 6, 8, 10$ and 20 basis functions. These properties were also calculated at a time interval of 1 au and the average was taken. The absolute error was then calculated at each time step by calculating the deviation between the initial values and the value at the final time step. The mean of the absolute errors are calculated and these are shown in table 4.6. A mean absolute error (MAE) of 1E-15 au or lower is regarded as very small and is reported as 0E-15.

System:Free Particle		
N Basis Functions	MAE Classical Energy (au)	MAE Norm
1	9.65E-013	0.00E-015
2	2.37E-013	7.44E-012
4	2.56E-013	1.86E-011
6	1.98E-013	3.51E-011
8	6.58E-014	2.88E-011
10	1.15E-013	1.47E-008
20	7.17E-014	2.22E+003

System:Harmonic Oscillator		
N Basis Functions	MAE Classical Energy (au)	MAE Norm
1	0.00E-015	0.00E-015
2	0.00E-015	5.78E-013
4	0.00E-015	2.41E-012
6	0.00E-015	3.02E-012
8	0.00E-015	1.29E-012
10	0.00E-015	1.01E-011
20	0.00E-015	1.15E+003

System:Morse Oscillator		
N Basis Functions	MAE Classical Energy (au)	MAE Norm
1	7.27E-013	0.00E-015
2	6.94E-013	2.52E-012
4	7.61E-013	5.38E-012
6	6.58E-013	5.28E-011
8	7.85E-013	9.09E-011
10	6.72E-013	4.31E-005
20	6.94E-013	6.52E+003

Table 4.1: A table to show the mean absolute error (MAE) in the calculation of the classical energy and the norm. The average property was calculated, for the different basis sets, and the absolute error was then calculated as the deviation between the initial values and the value at the final time step. A MAE of 1E-15 au or lower is reported as 0E-15.

These results show that the code conserves the classical energy and the norm for all systems very well. The classical energy is conserved to at least 10 significant figures for all systems. The norm is also conserved to 10 or more significant figures for all systems with fewer than 10 basis functions. However, for 10 basis functions the norm is only conserved to 10 significant figures for the Harmonic Oscillator. The Free Particle conserves the norm to 8 significant figures and the Morse Oscillator conserves the norm to 5 significant figures for 10 basis functions. For 20 basis functions the conservation of the norm breaks down, which is not surprising given that the initial norm was very high due to numerical errors. The classical energy is conserved very well in every case, even when 20 basis functions are used, which is expected as it is a sum of the independent energies of the individual basis functions. This shows that the difficulty is in calculating the trajectories of the coefficients, which are coupled.

Overall, the three systems showed a good conservation of the norm and the classical energy of the basis set. However, it is clear that the Harmonic Oscillator had the lowest mean percentage error and the Morse Oscillator shows the largest mean percentage error. One reason for this is that the ordered Hamiltonian function H_{ord} shown in equation 4.2.2 is much simpler than the H_{ord} function for the Free Particle and the Harmonic Oscillator, which decreases the numerical errors in the calculations. Furthermore the H_{ord} function for the Morse Oscillator is computationally more difficult to calculate than the H_{ord} function of the Free Particle due to the additional exponential term. It can be seen that the numerical error between the different systems is negligible compared to the numerical error obtained when a large number of basis functions are used with the same standard deviation parameter. Therefore all three systems can be used with this software providing that a suitable basis set is chosen.

4.7 Trajectories Of Basis Functions

The trajectories of the basis functions for the three systems were investigated with 10 basis functions and a larger standard deviation parameter of 1.5. The maximum time of the simulation was also increased to 50 au. The trajectories of $\Re(z)$ are shown for one basis function in each of the three systems in figures 4.3-4.5.

The trajectory graphs show that the basis function moves in a straight line in the Free Particle system whilst the basis function oscillates in the Harmonic and Morse Oscillator systems. The period of oscillation is approximately 2π au for the Harmonic Oscillator but it is significantly larger than this for the Morse Oscillator (approximately double, 4π au). A larger period of oscillation is expected with the Morse Oscillator due to the anharmonicity of the Potential. This occurs because the basis function moves more slowly on the right hand side of the potential well, shown in figure 4.1, compared to the steeper left hand side of the potential.

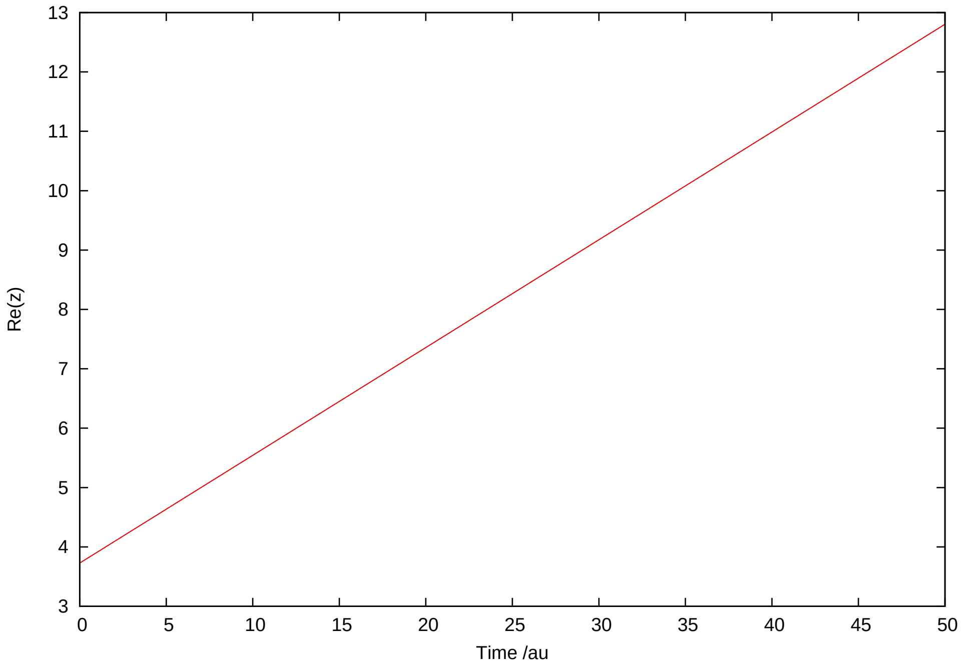

Figure 4.3: Trajectory of one basis function from a basis set of 10 basis functions, chosen randomly from a Gaussian distribution with a standard deviation of 1.5 for the Free Particle system with $\hbar = 1$ au.

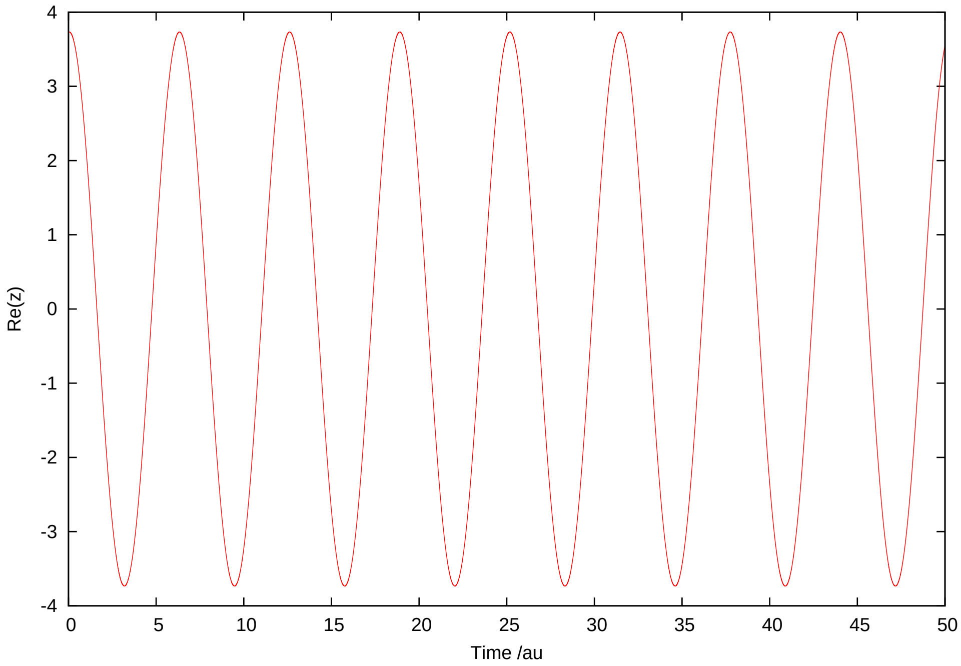

Figure 4.4: Trajectory of one basis function, from a basis set of 10 basis functions, chosen randomly from a Gaussian distribution with a standard deviation of 1.5 for the Harmonic Oscillator system with $|\hbar| = |k| = |m| = 1$.

Another possible representation, often used in physics to represent trajectories, is the phase space representation. This is a graph of position and momentum and is particularly useful as the full trajectory with both variables can be examined. The phase space

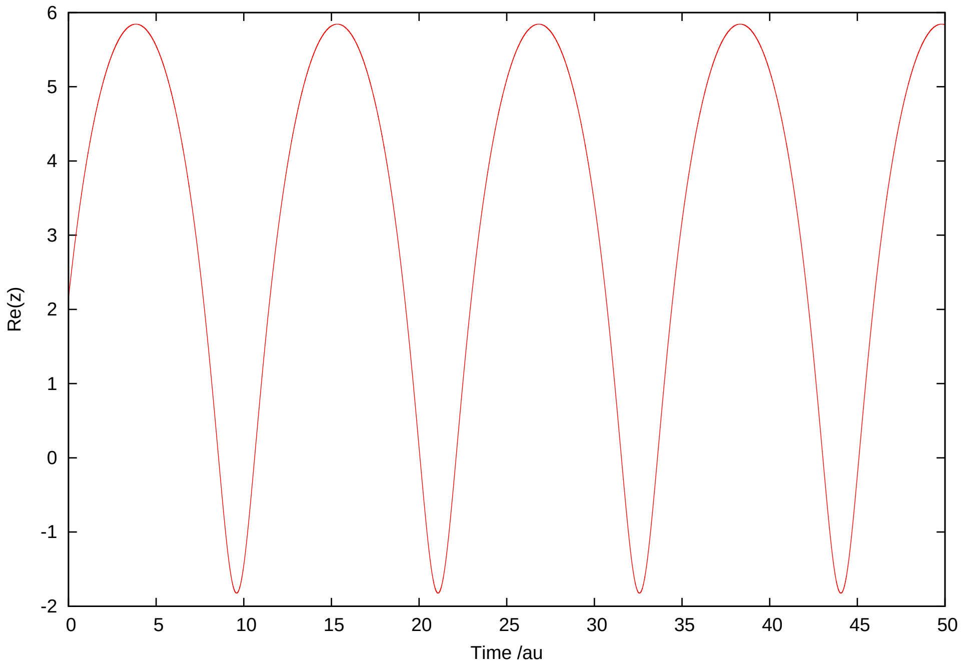

Figure 4.5: Trajectory of one basis function, from a basis set of 10 basis functions, chosen randomly from a Gaussian distribution with a standard deviation of 1.5 for the Morse Oscillator system with $|\hbar| = |m| = 1$, $D_0 = 10.25$ au and $a_0 = 0.2209$ au.

trajectory graphs can be seen in figures 4.6-4.8.

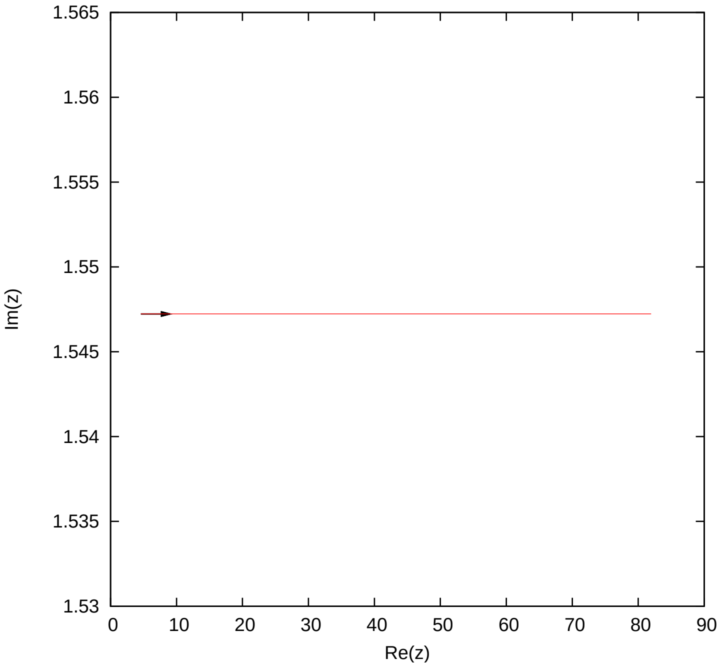

Figure 4.6: A phase space representation of the trajectory of the basis function in the Free Particle system shown in figure 4.3. The arrow shows the direction of motion in phase space.

Figure 4.6 shows how the basis function moves along a straight line trajectory. This

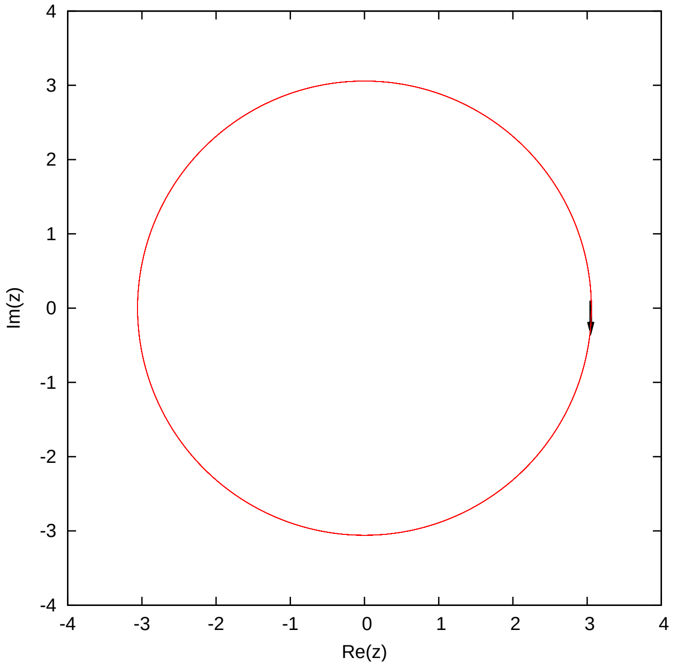

Figure 4.7: A phase space representation of the trajectory of the basis function in the Harmonic Oscillator system shown in figure 4.4. The arrow shows the direction of motion in phase space.

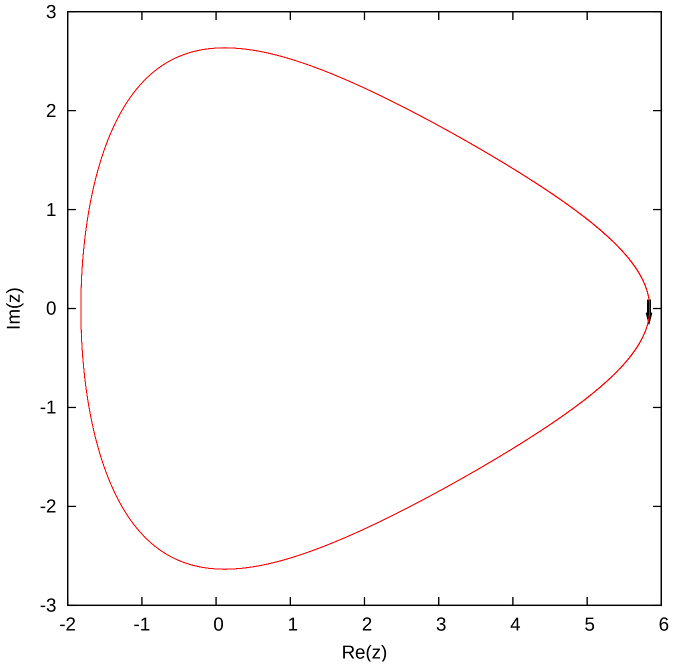

Figure 4.8: A phase space representation of the trajectory of the basis function in the Morse Oscillator system shown in figure 4.5. The arrow shows the direction of motion in phase space.

occurs as there is no force to change the direction of the basis function whilst the restoring force in the Harmonic and Morse Oscillator systems forces the basis function to move around phase space and return back to its original position. The arrow indicates the direction of the basis function in phase space. Comparing figures 4.7 and 4.8, it can be seen that figure 4.7 is symmetric about the line $\Re(z) = 0$, whilst figure 4.8 is asymmetric about this line. This resembles the symmetry in the potential as the Harmonic Oscillator potential is symmetric about the $q = 0$ au line in figure 4.1, which is the position of the potential well, whilst the Morse Oscillator is asymmetric about this line. In figure 4.1 the right hand side of the Morse Oscillator potential increases towards the dissociation energy whilst the left hand side increases asymptotically towards infinity. The slow increase of the potential energy corresponds to the elongated peak at approximately $\Re(z) = 5$ in the phase space trajectory (figure 4.8), whilst the sharp increase in the potential on the left of the potential well corresponds to the sharp vertical wall in the phase space trajectory at approximately $\Re(z) = -1.75$. The trajectory graphs for the three systems show that the code accurately calculates the trajectories of the basis functions.

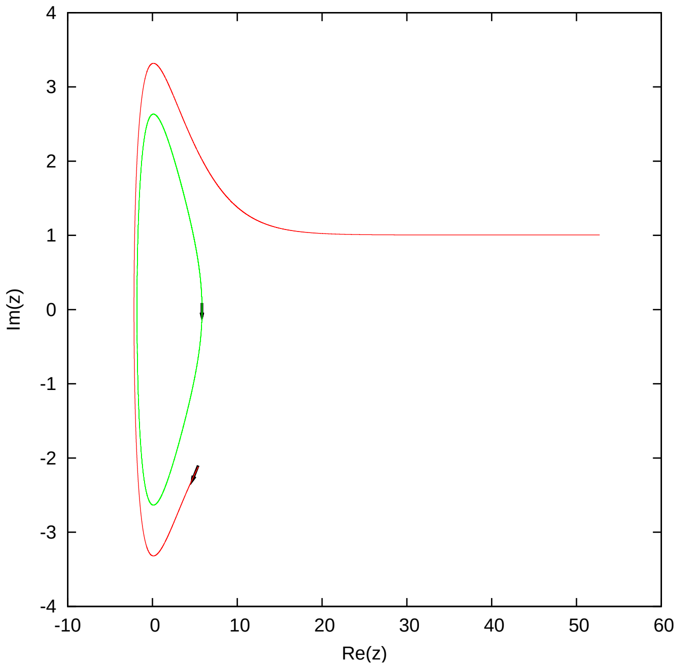

Figure 4.9: Phase space trajectory of two basis function in the Morse Oscillator system, with the same parameters that were used for figure 4.5. The basis function that follows the green trajectory has an energy comparable with the basis function used in figure 4.5 (approximately 5 au), whilst the basis function that follows the red trajectory has an energy higher than the dissociation energy (10.25 au), which causes it to dissociate. The dissociated basis function is uncoupled from the ensemble of basis functions that describe the wave function.

The trajectories discussed so far have been for basis functions that are coupled throughout the simulation. as indicated by the complete oscillation in phase space. However, if the energy of the basis function is high it is possible that the basis function becomes uncoupled. This often occurs for the Morse Oscillator when the energy of one basis function is above the Dissociation energy of the potential. In this case the basis function may not complete an oscillation but move away from the ensemble of basis functions independently, whilst the remaining basis functions continue to oscillate. This is shown in figure 4.9 where the basis functions moves in a straight line approximately along the $\Im(z) = 1$ line. When this occurs the basis function is no longer described quantum mechanically but it is described semi-classically as the amplitudes are no longer coupled. Therefore for a quantum description of the wave function the trajectories of the basis functions must also be checked to ensure that the basis functions remained coupled. This does not show a weakness of the CCS theory or the original code implemented but rather a general problem that must be checked when simulating a wave function in this way.

4.8 Summary

Testing the code has shown that it is able to calculate properties such as the norm and the classical energy correctly. These properties were conserved well for all the simulations, with the exception of the norm in one set of simulations where too many basis functions were used. The software can now be used to calculate more interesting properties.

Chapter 5

Absorption Spectra

A Gaussian wave function, simulated with the Harmonic and Morse Oscillator systems, is used to obtain Franck-Condon absorption spectra. The autocorrelation function calculated during the simulations and the corresponding Franck-Condon absorption spectra are shown in this chapter. A comparison is made between these calculated results and expected theoretical results obtained from the well-known equations for the eigenvalues of the systems. Similar calculations have been investigated previously but an original set up for the initial conditions of the wave function for the Morse Oscillator system is used here and so it will be interesting to see if the results obtained will be accurate.

5.1 Franck-Condon Spectra

In this work the Franck-Condon spectra discussed are absorption spectra that are in accordance with the Franck-Condon principle. This principle uses the Born-Oppenheimer approximation, which separates the nuclear wave function from the electronic wave function. This is because the motion of electrons is much faster than the motion of the nuclei due to the differences in the masses. An electronic excitation can occur from one electronic energy level to a higher electronic energy level but in which the nuclei remain fixed. However, as the system is in a higher electronic state the nuclei experience forces which causes them to vibrate. The Franck-Condon spectrum therefore shows the absorption spectrum of the transitions from the ground state electronic energy level to the vibrational energy levels in the higher electronic energy level. The Franck-Condon principle states that such a transition is more likely to occur when the overlap between the nuclear wave functions in the excited and ground state is greatest.

Experimentally the Franck-Condon spectrum is given by:

$$\sigma(\omega) = \sum_n |c_n|^2 \delta(\omega - \omega_n), \tag{5.1.1}$$

where $\sigma(\omega)$ gives the peak observed, c_n is a coefficient that depends on the overlap between the two nuclear wave functions and $\delta(\omega - \omega_n)$ is the Dirac Delta for the frequencies. Tannor has shown that this is equivalent to the Fourier transform equation shown in equation 2.5.6.[3] This is rationalised by the fact that the autocorrelation function describes how the system vibrates with respect to the initial coordinates. The Fourier transform of this then gives the spectrum.

It should be noted that although equation 2.5.6 is the equation that is widely used for calculating Franck-Condon absorption spectra it is not complete and the complete equation (for a normalised wave function) is:

$$\sigma(\omega_n) = \frac{2\pi\omega_n}{3\hbar c} \int_{-\infty}^{\infty} \exp\left(i(\omega + \omega_n(t))\right) \langle \Psi(0)|\Psi(t)\rangle dt \tag{5.1.2}$$

where ω_n is the incident frequency, ω is the initial frequency and c is the speed of light. The differences between equations 2.5.6 and 5.1.2, apart from the conversion from frequency to energy, is that there is an additional (frequency) term in the exponential and the constant $\frac{1}{2\pi\hbar}$ is now $\frac{2\pi\omega_n}{3\hbar c}$. However, these differences will not affect the positions of the peaks in the spectra.

5.2 Computational Set Up

The initial wave function was assumed to be a Gaussian function and was positioned at $q_0 = 5$ au and $p_0 = 0$ au for both systems. The standard deviation of this Gaussian was defined to be 1. The same atomic units and parameters for the Hamiltonians, i.e $k = 1$ au for the Harmonic Oscillator and for the Morse Oscillator $D = 10.25$ au and $\alpha = 0.2209$ au. Twelve basis functions were used and these were sampled randomly from a Gaussian distribution, which corresponded to the shape of the wave function. The number of basis functions was decided from the test calculations that were discussed earlier, as it was the most number of basis functions that could be used in the small region of phase space without causing high numerical errors. Using a Gaussian wave function positioned at a positive value of q_0 and sampling the basis functions from this Gaussian distribution for the Morse Oscillator has never been investigated before.

5.3 Autocorrelation Function

The autocorrelation function results, calculated during the simulations for the Harmonic and Morse Oscillators, can be seen in figures 5.1 and 5.2. Note that like the norm this is a dimensionless quantity. These figures show how the real part of the autocorrelation function changes over time. In both cases the autocorrelation function increases periodically. For the Harmonic Oscillator there is a period of 2π au. However, the sign of the peaks

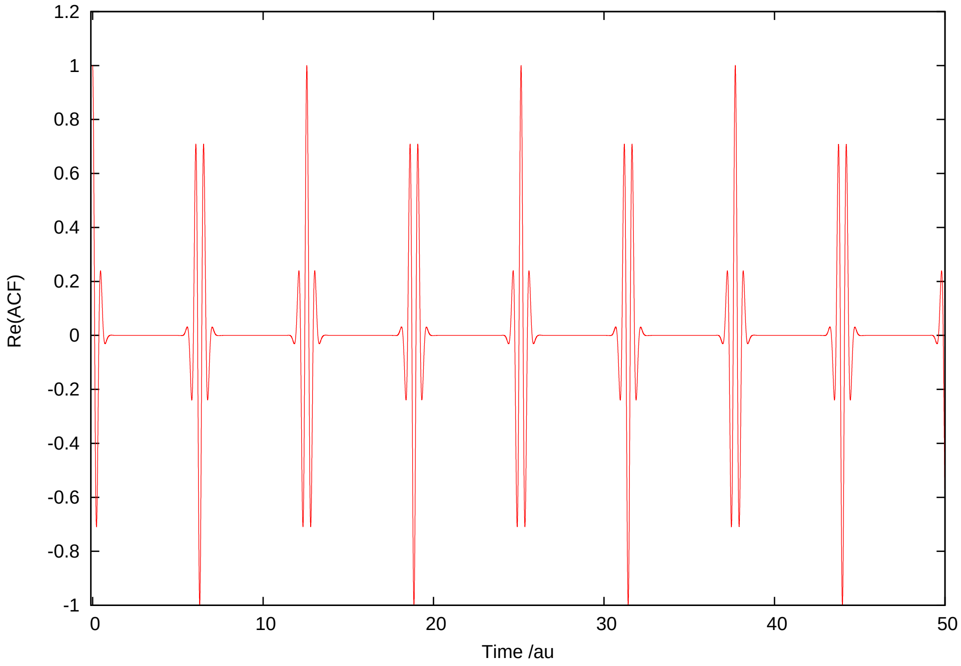

Figure 5.1: Real part of the autocorrelation function for the Harmonic Oscillator, with $|\hbar| = |k| = |m| = 1$. The basis set used was randomly generated from a Gaussian distribution with a seed parameter of 20 and a standard deviation of 1.

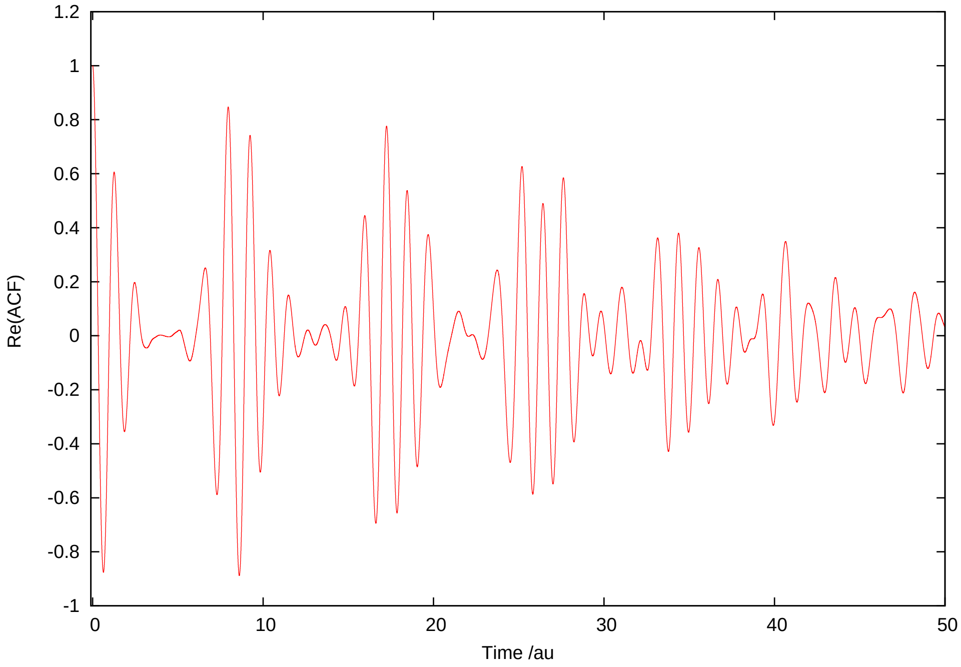

Figure 5.2: Real part of the autocorrelation function for the Morse Oscillator, with $|\hbar| = |m| = 1$, $D_0 = 10.25$ au and $a_0 = 0.2209$ au. The same basis set was used as in figure 5.1.

changes with every oscillation. This is due to the phase of the wave function that changes during the simulation of the wave function.

The period of oscillation of autocorrelation peaks for the Morse Oscillator is longer than 2π au. These periods are consistent with the average periods of the trajectories of the basis functions shown in the previous chapter.

The autocorrelation function was then multiplied by the window function shown in equation 3.2.3, where $t_{cut} = 15$ au was used. These results are shown in figures 5.3 and 5.4. Three peaks can be seen for the Harmonic Oscillator whilst two peaks can be seen for the Morse Oscillator as this has a longer period of oscillation. The graphs show how the magnitude of the amplitude of the peaks decrease up to $t = 15$ au and then are forced to 0 after this point.

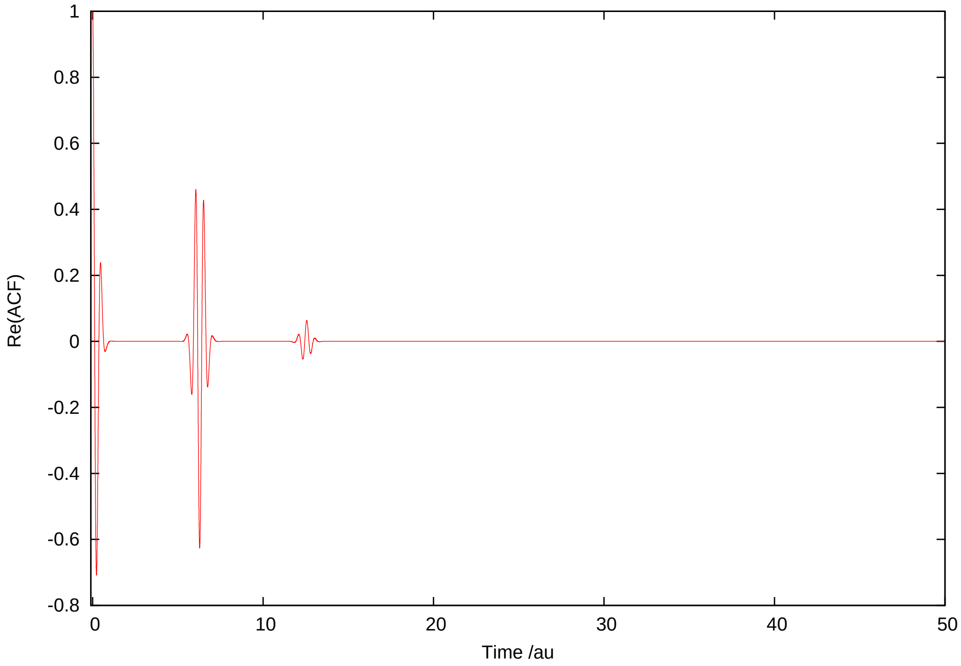

Figure 5.3: Real part of the autocorrelation function for the Harmonic Oscillator, shown in figure 5.1, after being multiplied by the window function, with $t_{cut} = 15$ au.

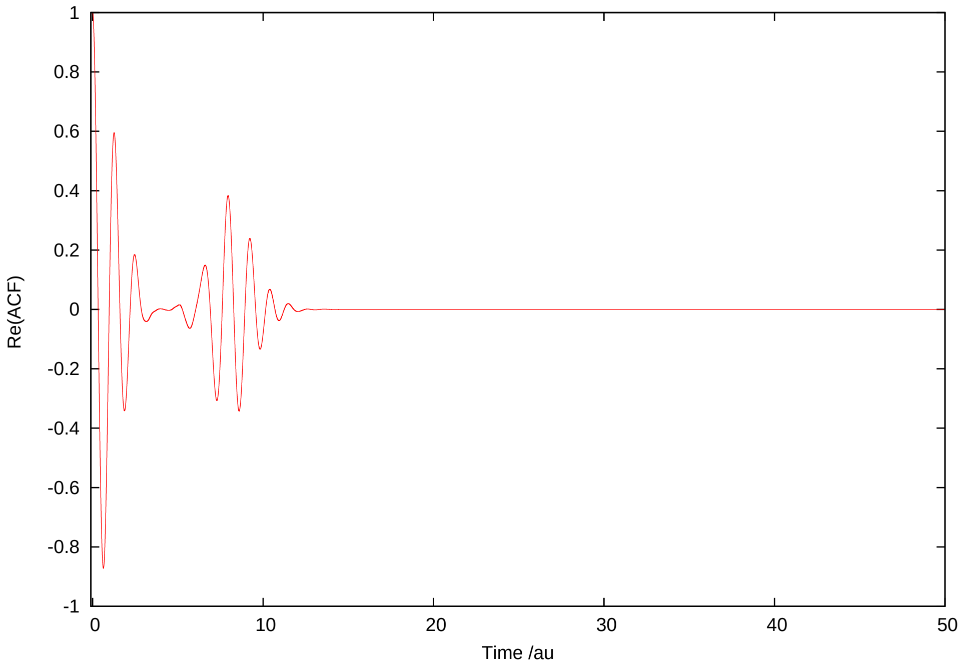

Figure 5.4: Real part of the autocorrelation function of the Morse Oscillator, shown in figure 5.2, after being multiplied by the window function, with $t_{cut} = 15$ au.

5.4 Fourier Transform

The Fourier transform of the complete autocorrelation functions shown in figures 5.1 and 5.2 were calculated. These are shown in figures 5.5 and 5.6 as Franck-Condon absorption spectra. The dashed vertical lines are the positions of the eigenvalues calculated using the well-known equations for the eigenvalues of the Harmonic and Morse Oscillator (equations 4.2.3 and 4.3.3).

The peaks of the calculated spectra, shown in figures 5.5 and 5.6, match the vertical dashed lines, which give the positions of the eigenvalues. This shows the accuracy of the calculations. However, in both sets of results there are many smaller background peaks and it is sometimes hard to distinguish between the peaks associated with the eigenvalues from the background peaks. In order to reduce these peaks and obtain clear spectra the Fourier transform of the product of the window function and the autocorrelation function is taken, as described previously. These are shown in figures 5.7 and 5.8. The value of t_{cut} was set at 15 au.

The Franck-Condon absorption spectra shown in figures 5.7 and 5.8 are significantly clearer and so easier to analyse, than the spectra shown in figures 5.5 and 5.6. This shows that using a window function on the autocorrelation function helps reduce the smaller background peaks. The peaks of the spectra have not moved and so the very good agreement between the peaks of the spectra and the eigenvalues of the system is still retained. Note that the largest peaks in figure 5.8 are those near $\nu = 5$ and this corresponds to the initial mean position of the Gaussian wave function. Comparing figures 5.7 and 5.8 shows

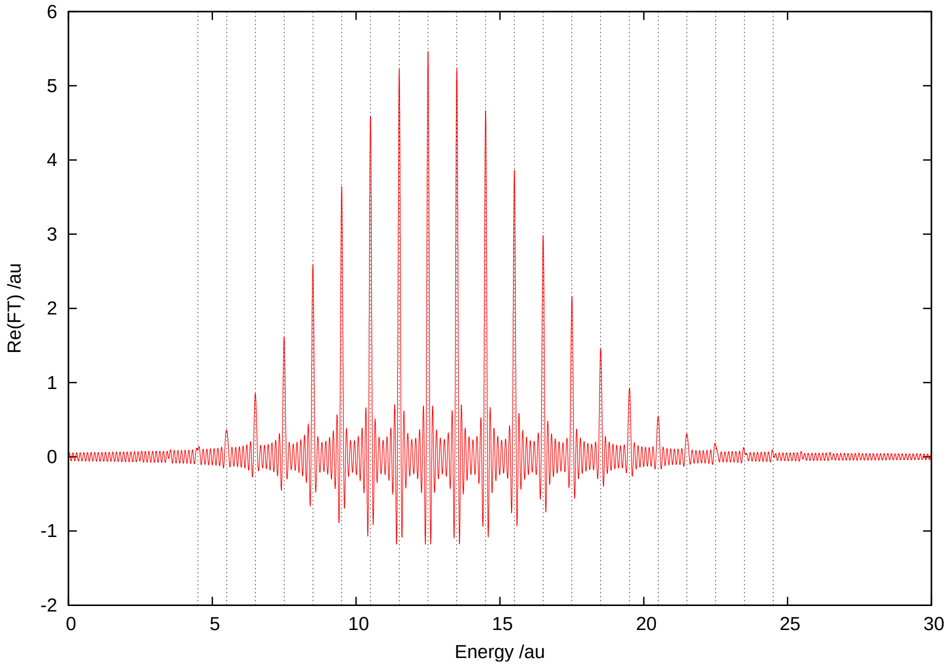

Figure 5.5: The Franck-Condon absorption spectrum for the Harmonic Oscillator (red line) obtained by Fourier transforming the autocorrelation function shown in figure 5.1. The vertical dashed lines show the expected positions of the eigenvalues of the vibrational energy levels for the Harmonic Oscillator system calculated using equation 4.2.3.

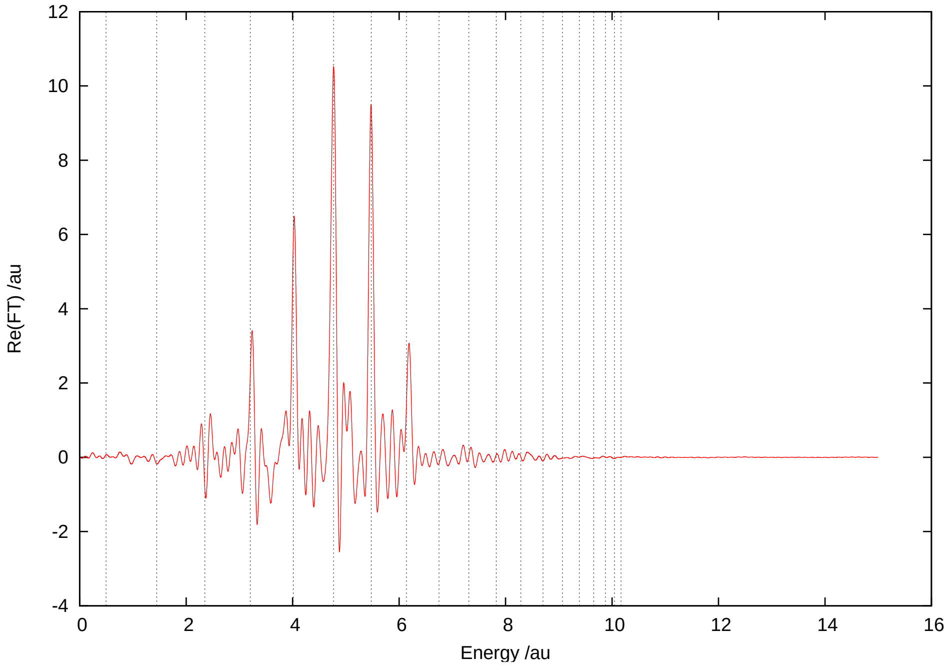

Figure 5.6: The Franck-Condon absorption spectrum for the Morse Oscillator (red line) obtained by Fourier transforming the autocorrelation function shown in figure 5.2. The vertical dashed lines show the expected positions of the eigenvalues of the vibrational energy levels for the Morse Oscillator system calculated using equation 4.3.3.

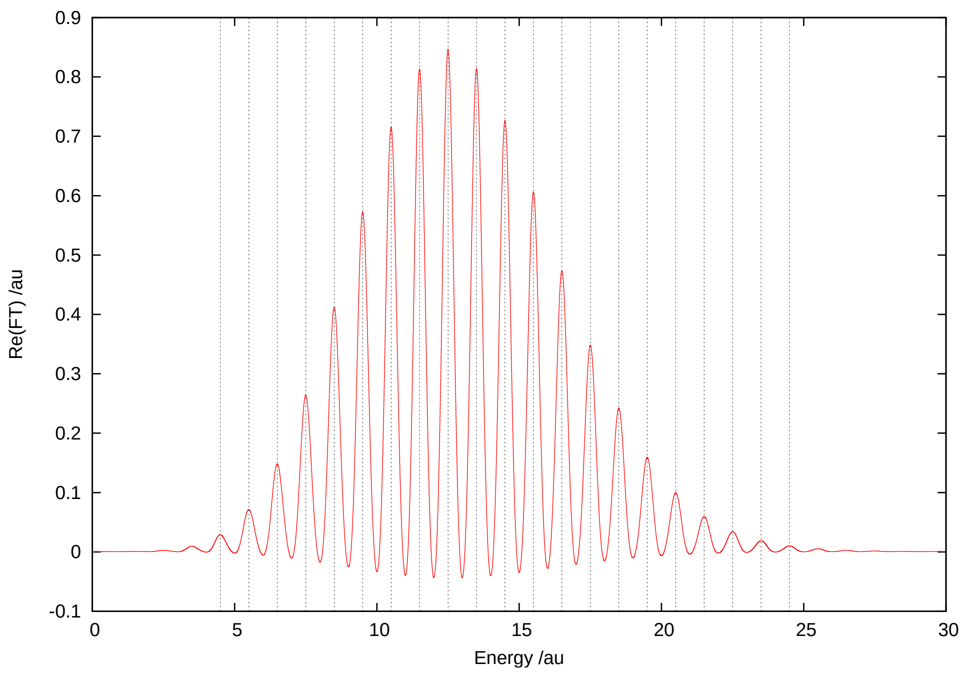

Figure 5.7: The Franck-Condon absorption spectrum for the Harmonic Oscillator (red line) obtained by Fourier transforming the product of the autocorrelation function and the window function, as shown in figure 5.3. The vertical dashed lines show the expected positions of the eigenvalues of the vibrational energy levels for the Harmonic Oscillator system calculated using equation 4.2.3.

that more eigenvalues are calculated using the Harmonic Oscillator system compared with the Morse Oscillator.

The clear absorption spectra were obtained after using a range of different values for t_{cut} and different parameters in the simulations. One spectrum that was also obtained for the Morse Oscillator, with a different basis set, can be seen in figure 5.9. This spectrum was taken after the autocorrelation function was multiplied by the window function with $t_{cut} \approx 2\pi$ au. The spectrum has larger background peaks and the main peaks of the spectrum do not match the positions of the eigenvalues closely, when compared to figure 5.8. Most of the differences between these spectra arise from the fact that a different basis set was used. In order to reduce the differences between the spectra for different basis sets the calculations must be repeated many times and an average of the autocorrelation function must be taken. Convergence in the results is reached when it is found that increasing the number of repetitions in the average taken does not affect the results obtained. This can sometimes be difficult to obtain and given the high accuracy shown in figure 5.8 it is unlikely that the results obtained will be significantly more accurate than this. For this reason averaged results are not shown here for the Harmonic and Morse Oscillators.

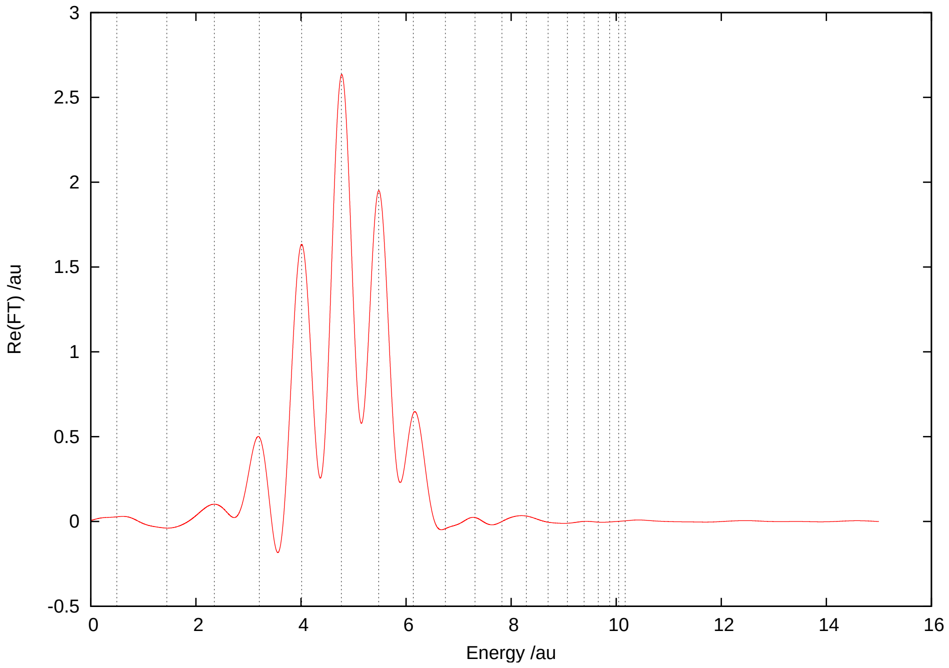

Figure 5.8: The Franck-Condon absorption spectrum for the Morse Oscillator (red line) obtained by Fourier transforming the product of the autocorrelation function and the window function, as shown in figure 5.4. The vertical dashed lines show the expected positions of the eigenvalues of the vibrational energy levels as before.

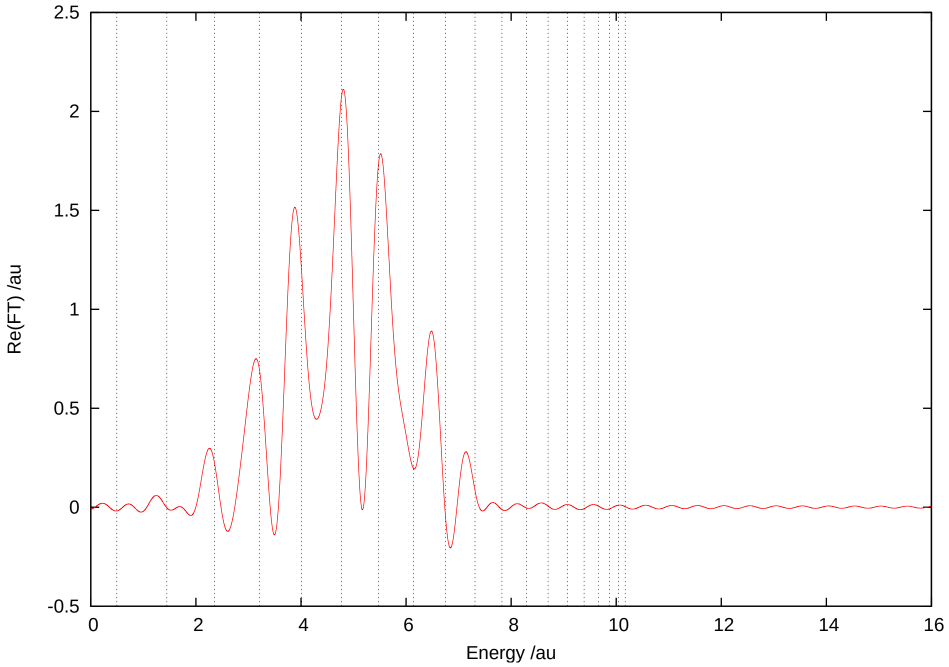

Figure 5.9: The Franck-Condon absorption spectrum for the Morse Oscillator (red line) obtained by Fourier transforming the product of the autocorrelation function and the window function. A random seed parameter of 12232321796 was used for the basis set whilst the values of the other parameters were the same as those in the previous simulations used for figure 5.8.

5.5 Summary

The results in this chapter show that an accurate Franck-Condon spectrum for the Harmonic and Morse Oscillator systems can be obtained by simulating a Gaussian wave function positioned at an excited state ($q_0 = 5$ au in this case). A clearer spectrum can be calculated by using the window function and by using a suitable basis set or by repeating the calculations many times and taking an average.

Chapter 6

Efficient Calculations

The Morse Oscillator has been used in the previous chapters for calculating trajectories, an autocorrelation function and Franck-Condon absorption spectra. Although it is described as an asymmetric potential due to the fact that there is no symmetry on either side of the potential well, there may be a symmetry in the trajectories of the basis functions. If a symmetry does exist in the trajectories of the basis functions it may be possible to use this to decrease the computational cost of the simulations. One possible method of decreasing the computational cost for calculating the autocorrelation function, which uses the symmetry arguments, is explored.

6.1 Symmetry Ideas

A typical basis function follows the trajectory in phase space as shown in figure 6.1. There is no symmetry about the $\Re(z) = 0$ line but there is symmetry about the $\Im(z) = 0$ line, which may be used. Given that the basis functions follow similar trajectories throughout the simulation any symmetry between them may be difficult to see. To investigate this for the Morse Oscillator, the wave function was defined to be a Gaussian function with zero momentum but with a positive value for the position. Note that this wave function was also used when investigating the trajectories, autocorrelation function results and the Franck-Condon spectra, calculated previously. The accuracy from these previous calculations show that this is a suitable wave function for these calculations.

6.1.1 Symmetry Arguments

A wave function with two basis functions is given by

$$\begin{aligned} \Psi &= \sum_i B_i \,|Z_i\rangle && (6.1.1) \\ &= B_1|Z_1\rangle + B_2|Z_2\rangle \ . && (6.1.2) \end{aligned}$$

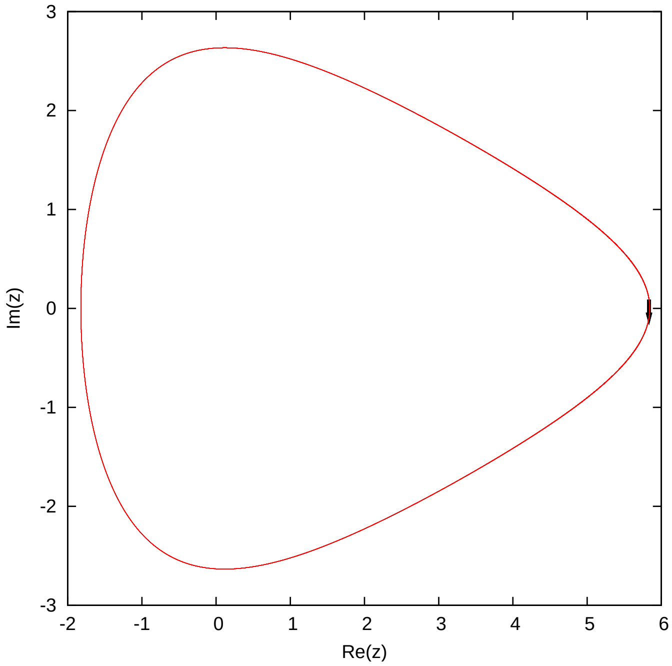

Figure 6.1: A phase space trajectory of one basis function for the Morse Oscillator system, with $D_0 = 10.25$ au and $a_0 = 0.2209$ au. A Gaussian wave function was used with the centre positioned at $q_0 = 5$ au and $p_0 = 0$ au in phase space.

The initial conditions of these basis functions can be chosen such that:

$$Z_1(0) = Z_2^*(0) \tag{6.1.3}$$

and

$$B_1(0) = B_2^*(0)\ . \tag{6.1.4}$$

Propagating the wave function forward in time gives:

$$\Psi \quad = \quad B_1(t)|Z_1(t)\rangle + B_2(t)|Z_2(t)\rangle \tag{6.1.5}$$

and propagating the wave function backwards in time gives:

$$\Psi \quad = \quad B_1(-t)|Z_1(-t)\rangle + B_2(-t)|Z_2(-t)\rangle\ . \tag{6.1.6}$$

If the symmetry in the initial conditions is retained then:

$$Z_1(-t) = Z_2^*(t) \tag{6.1.7}$$

and

$$B_1(-t) = B_2^*(t) \ . \tag{6.1.8}$$

6.1.2 Symmetric Trajectories

To investigate the symmetry, calculations were run for the Morse Oscillator from $t = 0$ au to $t_{max} = 20$ au, with $dt = 0.001$ au and from $t = 0$ au to $t_{max} = -20$ au with $dt = -0.001$ au. Twelve basis functions were used of which six basis functions were distributed from a Gaussian distribution randomly while the other half were their ‘reflections’ using equations 6.1.3 and 6.1.4. A standard deviation parameter of 1 was used. The trajectories of the symmetric basis functions were studied.

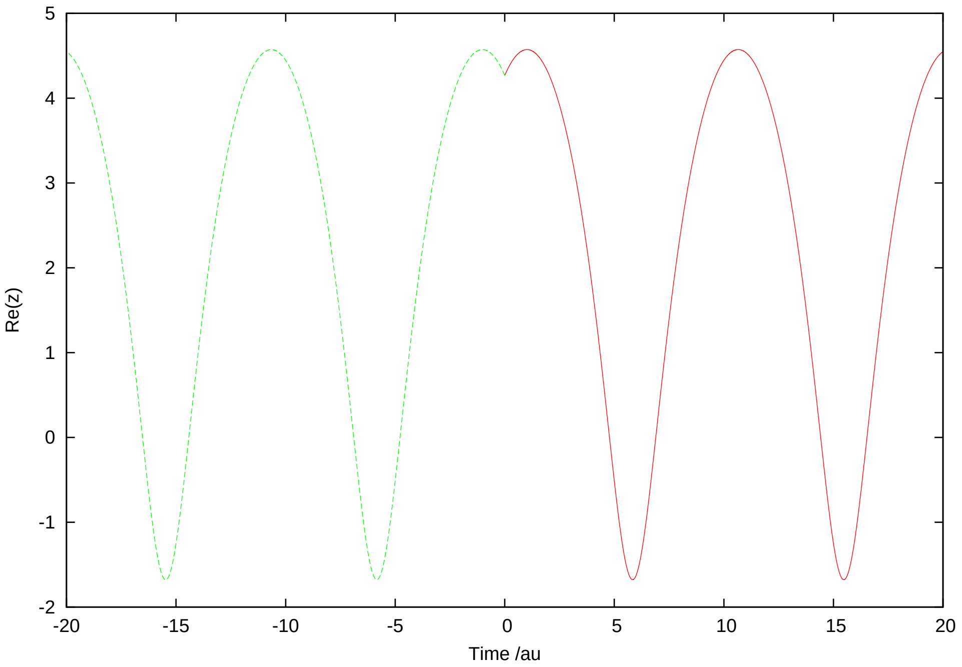

Figure 6.2: The time dependence of the real part of Z for two basis functions, one propagated forward in time (Z_2^* in red) and one propagated backwards in time (Z_1 in green). This used the Morse Oscillator system with the same parameters as used for figure 6.1. Initially $Z_1 = Z_2^*$ and $B_1 = B_2^*$.

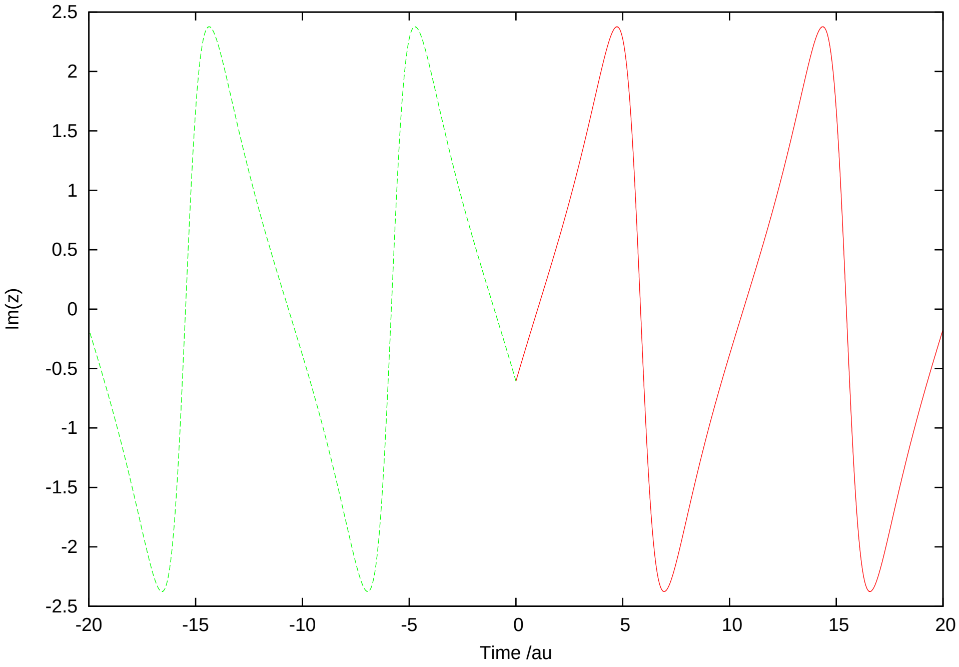

Figure 6.3: The time dependence of the imaginary part of Z for two basis functions, one propagated forward in time (Z_2^* in red) and one propagated backwards in time (Z_1 in green). The same simulation that was used for figure 6.2 is used here but where $\Im(Z)$ of the two basis functions are analysed instead.

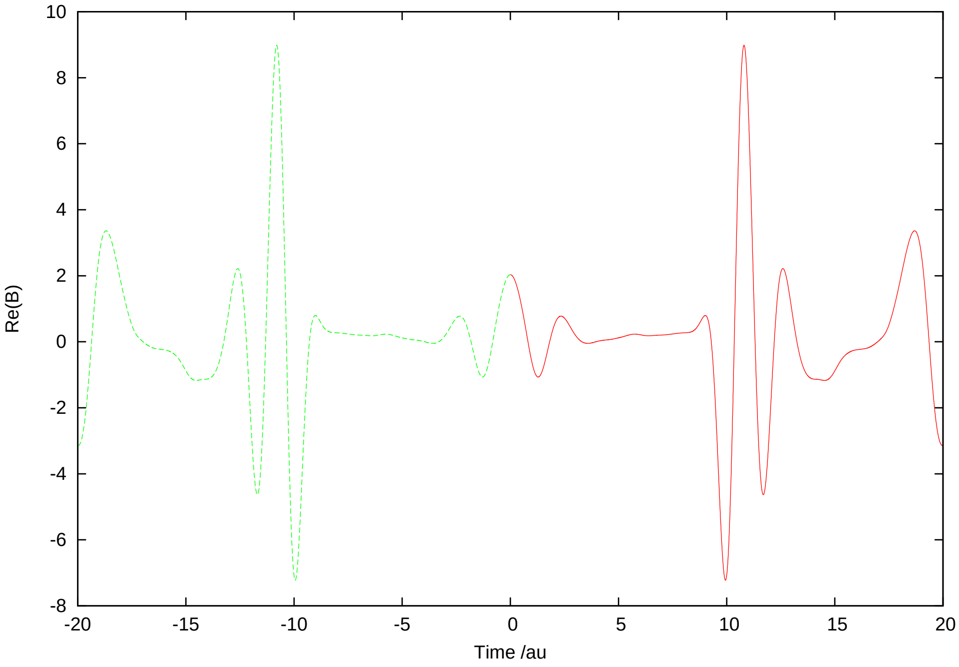

Figure 6.4: The time dependence of the real part of B for two basis functions, one propagated forward in time (B_2^* in red) and one propagated backwards in time (B_1 in green). The same simulation that was used for figure 6.2 is used here but where $\Re(B)$ of the two basis functions are analysed instead.

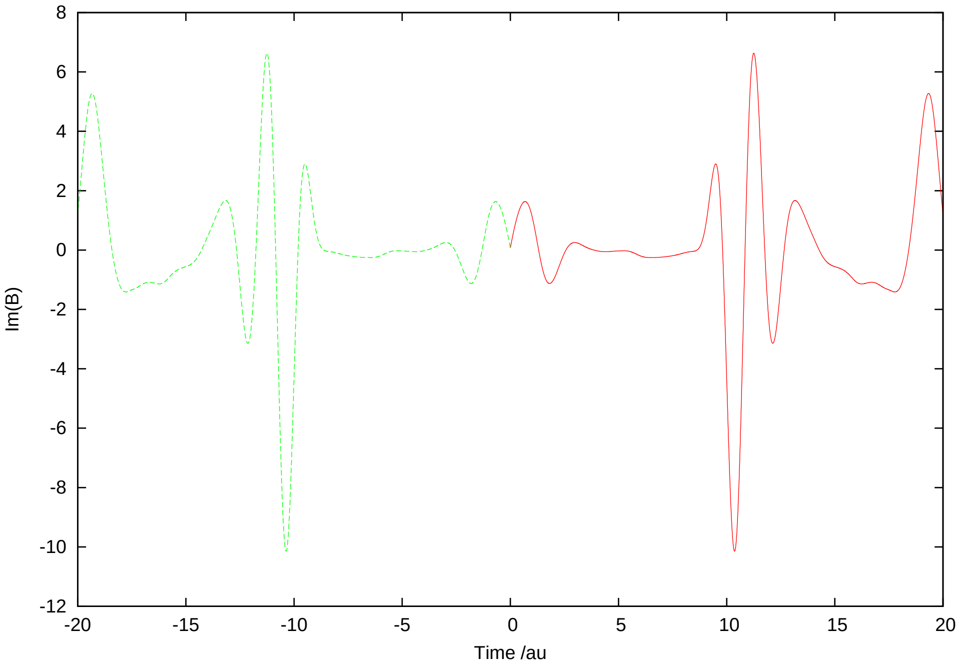

Figure 6.5: The time dependence of the imaginary part of B for two basis functions, one propagated forward in time (B_2^* in red) and one propagated backwards in time (B_1 in green). The same simulation that was used for figure 6.2 is used here but where $\Im(B)$ of the two basis functions are analysed instead.

Figures 6.2 to 6.4 show the time dependence of Z and B, for two symmetric basis functions, when propagated forwards and backwards from $t = 0$ au. The green lines show the time dependence of $Z_1(-t)$ or $B_1(-t)$ and the red lines show the time dependence of Z_2^* or B_2^*. In each case there is a symmetry about the $t = 0$ au line. The real and imaginary components of Z and B are shown separately for clarity. These figures confirm that the symmetry relationships given in equations 6.1.7 and 6.1.8 hold true throughout the simulation. Furthermore, the arguments based on the symmetry show that the random numerical errors have a small effect on the symmetry of the simulation.

6.2 Spectra Reviewed

The autocorrelation function can be rewritten, using the symmetry arguments in equations 6.1.7 and 6.1.8, in a computationally more efficient form. The wave function propagated over time can be written as:

$$|\Psi(t)\rangle = \exp\left(-i\hat{H}t\right)|\Psi(0)\rangle \ , \tag{6.2.1}$$

where $\exp(-i\hat{H}t)$ propagates the wave function and is called the 'propagator'. Multiplying equation 6.2.1 by the bra $\langle\Psi(0)|$ gives the autocorrelation function:

$$\langle\Psi(0)|\Psi(t)\rangle = \langle\Psi(0)|\exp(-i\hat{H}t)|\Psi(0)\rangle \ . \tag{6.2.2}$$

This can be rewritten as

$$\langle\Psi(0)|\exp(-i\hat{H}t)\,|\Psi(0)\rangle = \langle\Psi(0)|\exp\left(-i\hat{H}\frac{t}{2}\right)\,\exp\left(-i\hat{H}\frac{t}{2}\right)|\Psi(0)\rangle \ . \tag{6.2.3}$$

The ket is then analysed as

$$\exp\left(-i\hat{H}\frac{t}{2}\right)|\Psi(0)\rangle \ = \left|\Psi\left(\frac{t}{2}\right)\right\rangle \ , \tag{6.2.4}$$

with the complex conjugate

$$\langle\Psi(0)|\exp\left(i\hat{H}\frac{t}{2}\right) \ = \left\langle\Psi\left(\frac{t}{2}\right)\right| \ . \tag{6.2.5}$$

Multiplying the exponent in the propagator by -1 gives:

$$\langle\Psi(0)|\exp\left(-i\hat{H}\frac{t}{2}\right) \ = \left\langle\Psi\left(\frac{-t}{2}\right)\right| \ . \tag{6.2.6}$$

Assuming the wave function to be real ($p_0 = 0$ au), combined with the symmetry arguments in equations 6.1.7 and 6.1.8 gives

$$\langle\Psi(0)|\exp\left(-i\hat{H}\frac{t}{2}\right) \ = \left\langle\Psi^*\left(\frac{t}{2}\right)\right| \ , \tag{6.2.7}$$

which can be substituted into equation 6.2.3 to give

$$\langle\Psi(0)|\Psi(t)\rangle = \left\langle\Psi^*\left(\frac{t}{2}\right)\right|\Psi\left(\frac{t}{2}\right)\right\rangle \tag{6.2.8}$$

Expanding the wave function in the bra $\langle\Psi^*\left(\frac{t}{2}\right)|$ and the ket $|\Psi\left(\frac{t}{2}\right)\rangle$ in terms of a basis set gives:

$$\left\langle\Psi^*\left(\frac{t}{2}\right)\right| = \sum_k D_k \exp\frac{i}{\hbar}S_k\left(\frac{t}{2}\right)\left\langle z_k^*\left(\frac{t}{2}\right)\right| \tag{6.2.9}$$

and

$$\left|\Psi\left(\frac{t}{2}\right)\right\rangle = \sum_j D_j \exp\frac{i}{\hbar}S_j\left(\frac{t}{2}\right)\left|z_j\left(\frac{t}{2}\right)\right\rangle \tag{6.2.10}$$

Multiplying these together gives the autocorrelation function:

$$\begin{aligned} \langle \Psi(0) | \, \Psi(t) \rangle &= \sum_{k} \sum_{j} \exp \frac{i}{\hbar} \left(S_j \left(\frac{t}{2} \right) + S_k \left(\frac{t}{2} \right) \right) D_j \left(\frac{t}{2} \right) \\ &\quad \times D_k \left(\frac{t}{2} \right) \left\langle z_k^* \left(\frac{t}{2} \right) \middle| \, z_j \left(\frac{t}{2} \right) \right\rangle \end{aligned} \tag{6.2.11}$$

The equation shown for the autocorrelation function, using the symmetry arguments, has never been investigated for the CCS method and if this equation can provide accurate results then it may be used for a larger system.

6.2.1 ACF Results

The autocorrelation function was calculated using equation 6.2.11 for a Gaussian wave function positioned at $q_0 = 5$ au and $p_0 = 0$ au in phase space. These results are shown in red in figure 6.6 and are compared with the previous results, which are shown in green.

The exact same basis sets were used in both cases to allow a direct comparison of the results. The wave function was simulated from time $t = 0$ au to $t = 25$ au, using $dt = 0.001$ au with equation 6.2.11 that calculated the autocorrelation function to $t = 50$ au. The previous autocorrelation function results were calculated with equation 2.5.5, using $dt = 0.001$ au but where the simulation ran from $t = 0$ au to $t = 50$ au and the autocorrelation function was calculated at each time step.

Comparing the results of the two different autocorrelation functions in figure 6.6 shows some similarities and differences. At the beginning of the simulation the two results are identical as the overlap between them is exact. However, differences are observed at approximately $t = 5$ au in which the red line shows a slight oscillation. Interestingly the main peak at approximately 2π au is similar in both cases but the peaks after this point differ significantly. The main difference is that the red lines shows larger peaks later on in the simulation. There are also differences in the positions of some of the peaks, for example the peaks at approximately $t = 30$ au differ in their positions. The greatest differences are observed at $t = 40$ au where the positions and the magnitude of the peaks differ significantly.

In general the autocorrelation function calculated for the Morse Oscillator is expected to peak at approximately every 2π au with the same magnitude of the peak. The results shown here are interesting as the red line, calculated with equation 6.2.11 seems to match this closer than the green line calculated with 2.5.5. This indicates that the autocorrelation function shown in red is more reliable than the autocorrelation function shown in green.

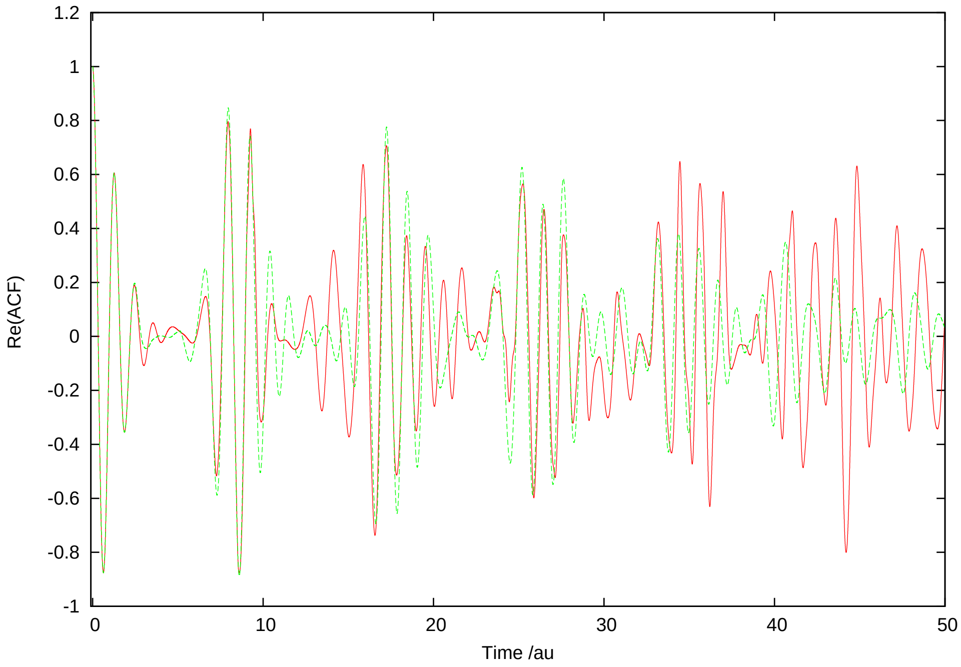

Figure 6.6: The real part of the autocorrelation functions calculated using equation 2.5.5 (green) and equation 6.2.11 (red). This used the Morse Oscillator system with the same parameters as were used for figure 6.1. The basis functions were distributed randomly from a Gaussian distribution with a standard deviation of 1, which was positioned at $q_0 = 5$ au and $p = 0$ au in phase space.

6.2.2 ACF Symmetric Basis Set

The calculations were repeated with the same initial position of the wave function, in phase space, but with symmetric basis sets, where the first half of the basis functions were chosen randomly from a Gaussian distribution and the remaining basis functions were the reflections of these. These are shown in figure 6.7.

The autocorrelation functions in purple and blue in figure 6.7 shows similar trends to the autocorrelation functions in figure 6.6. In both cases the autocorrelation function calculated with equation 6.2.11 show large peaks after $t = 30$ au, which are not observed in the autocorrelation function with 2.5.5. In addition to this, the initial close overlap of the two autocorrelation functions in figure 6.6 is also found in figure 6.7. Therefore, these features are likely to be due to the equations that are used as they are not affected by the basis sets that are used.

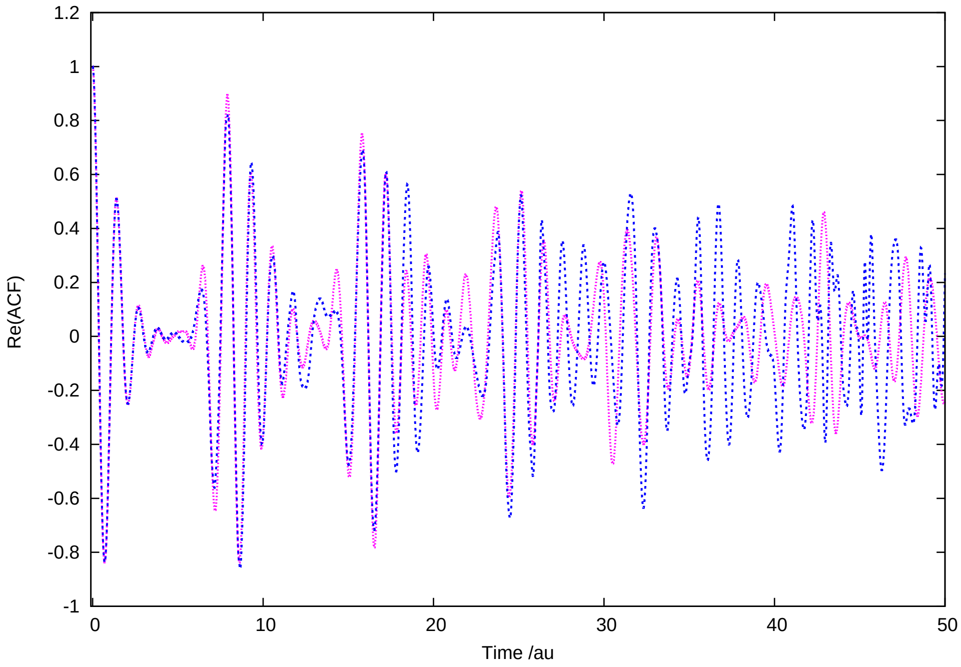

Figure 6.7: The real part of the autocorrelation functions calculated using equation 2.5.5 (purple) and equation 6.2.11 (blue). This used the Morse Oscillator system with the same parameters as were used for figure 6.6. However, a symmetric basis set was used where half of the basis functions were distributed randomly from a Gaussian distribution, with a standard deviation of 1, whilst the other half were distributed symmetrically using equations 6.1.3 and 6.1.4.

6.2.3 Franck-Condon Spectra

The autocorrelation function results in figure 6.6 were used to calculate the Fourier transform. A window function was not used for the results shown in figure 6.8. The exact eigenvalues of the Morse Oscillator are shown by dashed lines and the peaks of the Fourier transform are expected to go through these points as shown previously.

The first noticeable difference between the two absorption spectra shown in red and green lines in figure 6.8 is that the magnitude of the peaks in the red lines are significantly lower than the magnitude of the red lines. This difference arises because the Fourier transform uses the autocorrelation function calculated with equation 6.2.11 and this has half the data points in a specified time interval compared to the autocorrelation function calculated with 2.5.5. Therefore this is one consequence of using equation 6.2.11 to calculate the autocorrelation function at time $2t$.

Another interesting difference is that the red line shows more smaller peaks in the spectra, which do not correspond to energy levels. These are observed in between the larger peaks and on the base line of the spectrum. This seems to suggest that the spectra in red consists of more numerical errors than the spectra in green. One possible explanation of this could be that the Fourier transform is very sensitive to the number of points in the autocorrelation function.

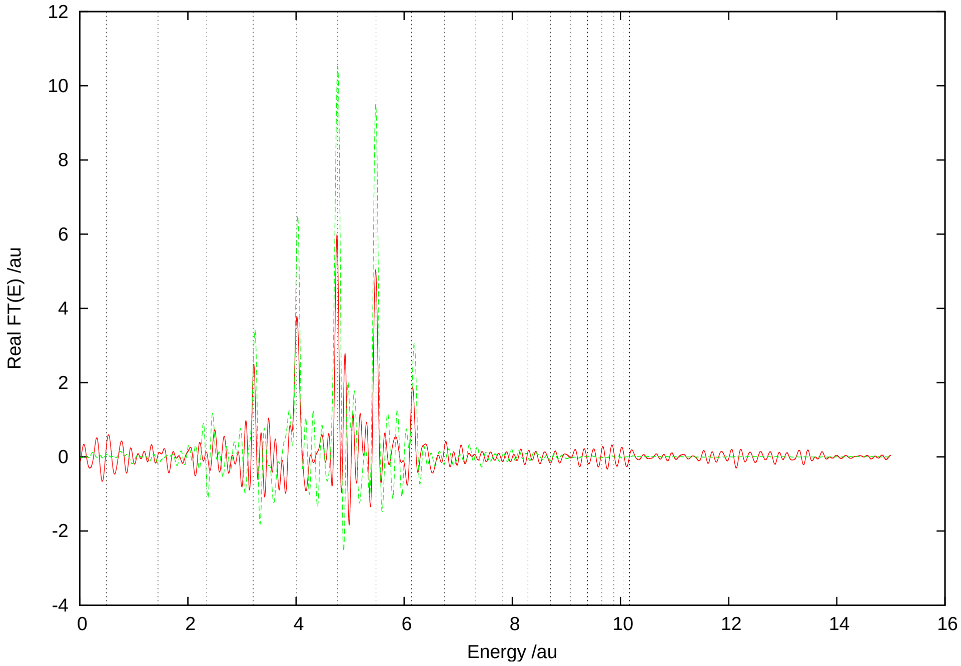

Figure 6.8: The Franck-Condon absorption spectra for the Morse Oscillator, shown in red and green, obtained by Fourier transforming the autocorrelation functions, shown in red and green respectively, in figure 6.6. The vertical dashed lines show the expected positions of the eigenvalues of the vibrational energy levels.

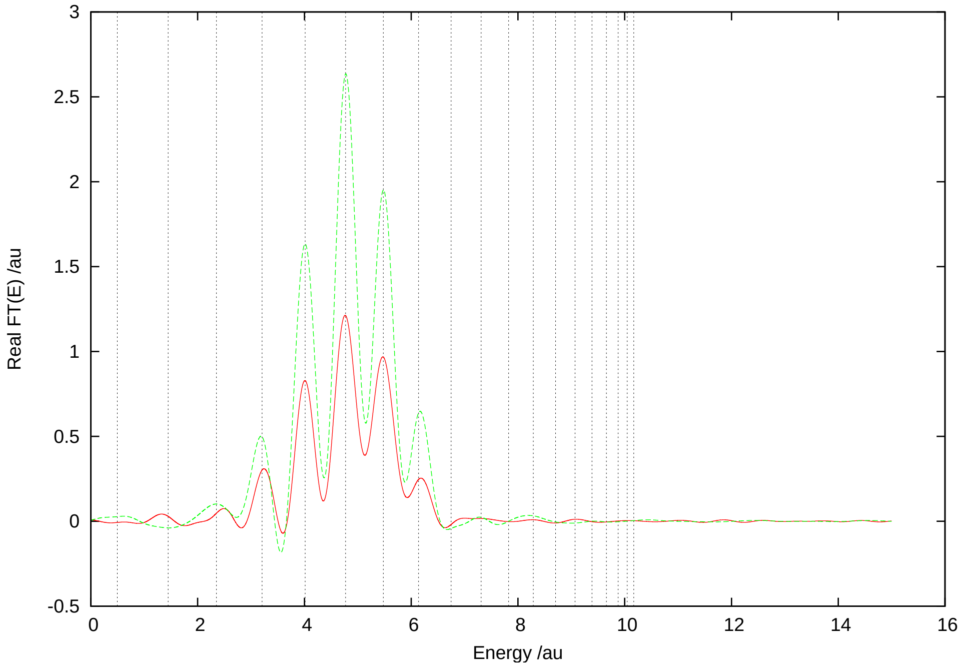

Figure 6.9: The Franck-Condon absorption spectra for the Morse Oscillator, shown in red and green, obtained by Fourier transforming the product of the autocorrelation functions, shown in red and green respectively, in figure 6.6 with the window function, where $t_{cut} = 15$ au. The vertical dashed lines show the expected positions of the eigenvalues of the vibrational energy levels.

The spectra shown in figure 6.9 were calculated using the window function with $t_{cut} = 15$ au. As before the magnitude of the peaks in red are smaller compared with the peaks in green. However, in both the red and green coloured spectra clear results are found. This is unsurprising given that autocorrelation functions are similar in both cases up to $t = 15$ au, which is the value of t_{cut}. The peaks of these spectra match the positions of the eigenvalues almost exactly. Therefore the autocorrelation function in equation 6.2.11 can be used to produce clear spectra but with smaller peaks compared to the spectra obtained using equation 2.5.5.

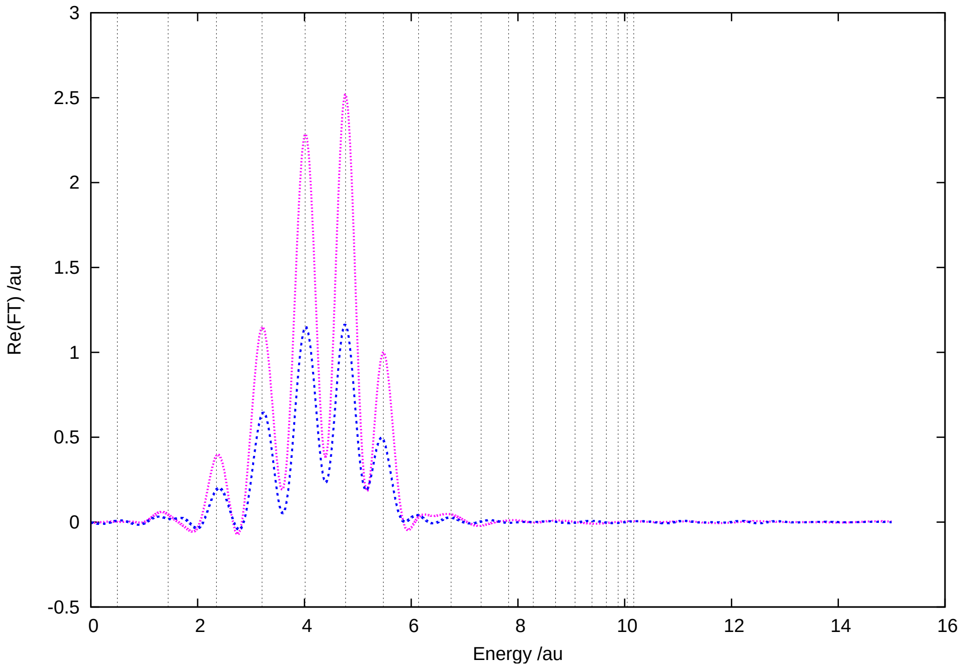

Figure 6.10: The Franck-Condon absorption spectra for the Morse Oscillator, shown in purple and blue, obtained by Fourier transforming the product of the autocorrelation functions, shown in purple and blue respectively, in figure 6.7 with the window function, where $t_{cut} = 15$ au. The vertical dashed lines show the expected positions of the eigenvalues of the vibrational energy levels.

Franck-Condon absorption spectra were also calculated using the autocorrelation functions shown in figure 6.7 with symmetric basis sets and the window function with $t_{cut} = 15$ au. These results are shown in figure 6.10. The spectra are similar to those calculated in figure 6.9 as the peaks in both sets of results match the eigenvalues of the Morse Oscillator and the intensity of the peak is greater when equation 2.5.5 is used.

6.3 Summary

The relatively simple arguments of symmetry have been shown to hold true throughout the simulation of the wave function in the Morse Oscillator for the trajectories of the basis functions and the time dependence of their amplitudes. Further work showed that the equation for the autocorrelation function can be rewritten using the symmetry arguments. The autocorrelation function results obtained with this equation are possibly more reliable and with half the computational cost. The Fourier transform calculated with this equation was shown to give clear results when a window function was used but with smaller magnitudes for the intensity of the main peaks. Using a symmetric basis set with the rewritten autocorrelation function confirms the conclusions found with the randomly generated basis functions.

Chapter 7

Bose-Hubbard Simulations

The simulations discussed so far have been for simple systems that consist of one degree of freedom. These systems have been studied previously using the CCS method with different approaches. The Fortran software is now used with a new system consisting of two degrees of freedom, the 'Bose-Hubbard' model. However, unlike the other systems investigated, this model is not easy to calculate with other methods and quantum dynamical methods that use trajectory guided basis sets, such as CCS, may prove to be useful.

The Hubbard Model is used to describe the dynamics of many systems and the Bose-Hubbard model is used to describe a system that consists of Bosons. The purpose of these calculations is to use this model to simulate the tunnelling and self-trapping of a wave function, which is similar to the recent exciting experimental investigations of tunnelling of many atoms in a specific state called the 'Bose-Einstein Condensate' (BEC). A 'Bose Einstein Condensate' is a state of matter in which the particles act like bosons and are in the lowest quantum state. In doing so they exhibit quantum properties and in contrast to fermions many bosons can occupy the same quantum state.

7.1 Model

The Hamiltonian for the Bose-Hubbard Model comprises of two degrees of freedom and is given by:

$$\hat{H} = \epsilon(\hat{n}^{(1)} - \hat{n}^{(2)}) + \nu(\hat{a}^{\dagger(1)}\hat{a}^{(2)} + \hat{a}^{\dagger(2)}\hat{a}^{(1)}) + g(\hat{n}^{(1)^2} + \hat{n}^{(2)^2}) \, , \tag{7.1.1}$$

where $\hat{a}^\dagger$ and $\hat{a}$ are the creation and annihilation operators and the number operator $\hat{n} = \hat{a}^\dagger\hat{a}$ counts the number of particles.[42] The superscripts $^{(1)}$ and $^{(2)}$ refer to the degrees of freedom. This model is identical to two Harmonic Oscillators, one for each degree of freedom, that are coupled together. The ϵ term gives the energy difference between the two Harmonic Oscillators, the ν term describes the coupling between them and g is a negative number that describes the self on-site attraction in each potential well (of the

Harmonic Oscillator) in each degree of freedom. The potential wells are asymmetric if $\epsilon \neq 0$.

The Bose-Hubbard model is able to model the tunnelling and self-trapping of a wave function. In the model, tunnelling occurs when the particles move from the Harmonic Oscillator potential describing one degree of freedom to the Harmonic Oscillator potential describing the second degree of freedom. The degree of tunnelling is dependent on the self on-site interaction parameter g. If g has a small magnitude ($\frac{1}{10}$, for 10 particles in one potential well) then the self on-site interaction of the wave function will be low and the wave function will tunnel between the two potential wells. However, if the magnitude of g is large ($\frac{3}{10}$) the self on-site interaction within the potential well is high and so little or no tunnelling between the potential wells occurs. This phenomenon is called self-trapping as the wave function is trapped within one of the potential wells.

The ordered Hamiltonian function that is used is:

$$\begin{aligned} H_{ord} &= \epsilon\big(z^{*(1)}z^{(1)} - z^{*(2)}z^{(2)}\big) + \nu\big(z^{*(1)}z^{(2)} + z^{*(2)}z^{(1)}\big) + \\ &\quad g\left((z^{*(1)}z^{(1)})^2 - (z^{*(2)}z^{(2)})^2\right) . \end{aligned} \tag{7.1.2}$$

This is derived by using the commutator relationship $\hat{a}\hat{a}^\dagger - \hat{a}^\dagger\hat{a} = 1$ with the g terms in equation 7.1.1.

A wave function consisting of two degrees of freedom is required in the Hubbard model, where each degree of freedom corresponds to a different potential well. The population of the wave function in the α degree of freedom is given by:

$$\langle\Psi|\hat{n}^{(\alpha)}|\Psi\rangle = \sum_{j,k} d_k^* e^{-\frac{i}{\hbar}S_k} d_j e^{\frac{i}{\hbar}S_j} \langle z_k|z_j\rangle \left(z^{*(\alpha)}_{k} z_j^{(\alpha))}\right) . \tag{7.1.3}$$

This is the expectation value of the number operator $\hat{n}$ and is derived by using $\hat{n} = \hat{a}^\dagger\hat{a}$, where $\hat{a}^\dagger$ acts on the bra $\langle\psi|$ and $\hat{a}$ acts on the ket $|\Psi\rangle$. The degree of tunnelling is given by using equation 7.1.3 to calculate the population in each degree of freedom throughout the simulation and seeing how it changes with time.

This model is not easy to simulate with standard quantum dynamical techniques as the range of the moving wave function in phase space is large and so a large static basis set is required. On the contrary, methods that use trajectory based basis sets, such as the CCS method, provide a natural way to minimise the range that the wave function occupies in phase space.

7.2 Experimental Study

Recently experimental work was carried out in which tunnelling of a Bose-Einstein condensate (BEC) was investigated.[43] Albiez *et al.* were able to observe *in situ* two weakly coupled BECs in a macroscopic double-well potential and carry out experiments to investigate tunnelling. In order to do this the ^{87}Rb BEC was created and split using low temperatures and two focused laser beams. The separation of the split BEC was given as $4.4 \pm 2\,\mu$m. The initial population of the split BEC in each well could be controlled using the focused laser beams. This was important so as to investigate the degree of tunnelling with respect to the population in each well. Absorption images of the population in each well could be taken over time and these were reported.

Overall, it was found that when the population was below a critical value tunnelling between the two potential wells could be observed and a period of 40 ± 2 ms was reported. Conversely, when the population was above a critical value, self-trapping occurred. The significance of the population in each well with respect to the critical value can be rationalised in terms of the self on-site interaction of the BEC. When the population in a potential well is below the critical value the self on-site interaction is low, which allows tunnelling between the two potential wells. However, when the population in a potential well is above the critical population the self on-site interaction of the BEC is higher. This results in stronger interactions between the BEC in the potential well, which prevents tunnelling from occurring into the other potential well.

7.3 Computational Set Up

The ordered Hamiltonian for the Hubbard model, equation 7.1.2, was used with the CCS equations of motion in the simulation of a wave function. The initial wave function is a coherent state, similar to the previous simulations. Many simulations were run and the number of basis functions and the region of phase space in which the basis functions were sampled from was investigated.

The central aim of the investigation of the Hubbard model was to determine whether the CCS method is able to model tunnelling of the wave function when the self on-site interaction is low and to model the self-trapping effect when the self on-site interaction is high. This was studied by calculating the population throughout the simulation in each potential well. The tunnelling of the wave function was controlled by changing the parameter g in the Hamiltonian from -0.1 au to -0.3 au. The simulations that used one basis function are semi-classical calculations as quantum calculations require an ensemble of basis functions with amplitudes that are coupled. The simulations ran from $t = 0$ au to $t = 20$ au with a time step of $dt = 0.001$ au. The atomic units, defined previously, were used with the parameters of the Hamiltonian set to $\nu = 1$ au, $\epsilon = -0.5$ au and $g = -0.1$ au or $g = -0.3$ au. However the populations calculated are dimensionless. The initial phase space position of the wave function in the first degree of freedom is given by $z_0^{(1)} = \sqrt{10} + 0i$ and in the second degree of freedom by $z_0^{(2)} = 0 + 0i$. This gives an initial population of 10 and 0 in each potential well respectively. The first potential well was defined to be at $z_0^{(1)}$ in phase space and the second potential well was defined to be at $z_0^{(1)}$ in phase space.

7.4 Semi-Classical Results

The previous simulations discussed involved using basis functions chosen randomly from a Gaussian function. A different procedure is used for the semi-classical results shown here, where the position of the basis function in phase space is specified. The most accurate description of the wave function that is possible with one basis function is where the mean position of the basis function in phase space is set to be equal to the mean position of the wave function in phase space.

The norm and the classical energy were conserved very well throughout the simulations (to at least ten significant figures), for simulations run with both $g = -0.1$ au and $g = -0.3$ au. The norm was equal to 1, as expected for one basis function that is set to the position of the wave function in phase space, and this was conserved to machine precision.

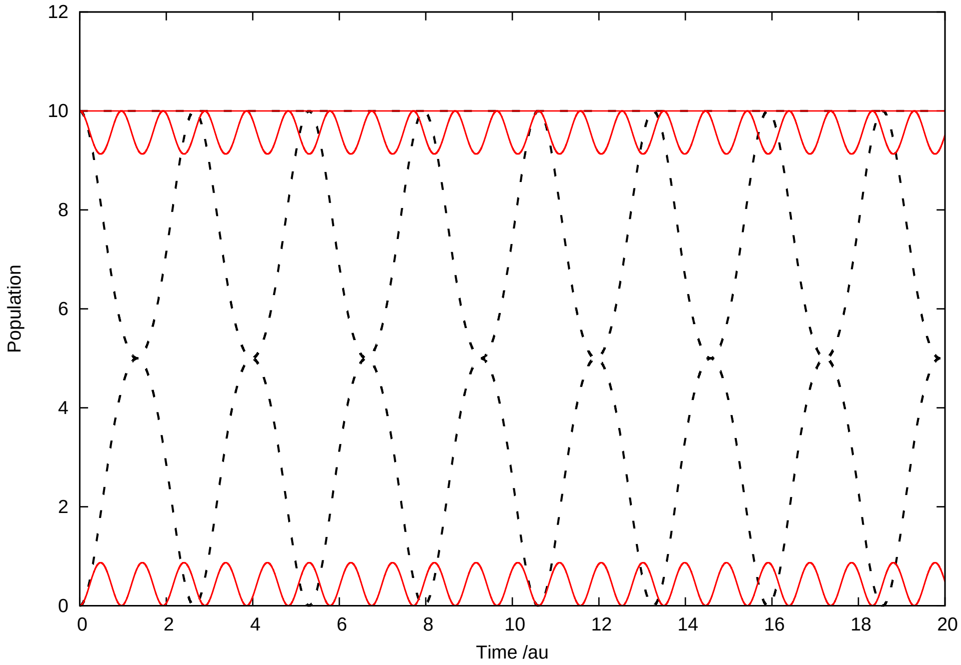

Figure 7.1: The time dependence of the populations of the two potential wells over time for two simulations, one with $g = -0.3$ au (red line) and the other with $g = -0.1$ au (black dashed line). One basis function was used and this was initially chosen to be at the position of the centre of the wave function ($z_0^{(1)} = \sqrt{10} + 0i$ and $z_0^{(2)} = \sqrt{0} + 0i$ for the two degrees of freedom) in phase space. The top line (at Population $= 10$) shows the total population whilst the middle and bottom lines show the population in each potential well.

The populations calculated during both simulations are shown in figure 7.1. In both cases, the initial population is 10 in the first potential well and 0 in the second and this changes over time. The population starts to decreases to 5 in the first well as the population tunnels into the second well, which then returns back to the first well, for the case where $g = -0.1$ au, indicated by the black dashed line. This occurs with a period of approximately 3 au. The total population in both potential wells is conserved, as shown by the horizontal line, with the population equal to approximately 10. The decrease in the population in one potential well as the population in the other potential increases represents the tunnelling that occurs between the two potential wells. When $g = -0.3$ au this tunnelling process occurs significantly less, indicated by the small oscillations shown in red but the total population remains well conserved as shown by the red horizontal line that always has a population of 10. The period of these smaller oscillations is also smaller, with a period of less than 1 au.

The semi-classical simulations show that when the magnitude of g is low the degree of tunnelling between the two potential wells is large but when the magnitude of g is high the degree of tunnelling significantly decreases and so self-trapping occurs. This is understood by the fact that the parameter g determines the self on-site interaction in the potential well and so when this is high the wave function self-traps. Therefore these results are a good description of what was found experimentally by Albiez *et al.* and also agree with the accurate semi-classical calculations by Graefe and Korsch.[42,43]

7.5 Quantum Results

As stated previously, the basis functions can be chosen from a variety of different ways and choosing them randomly has been shown to be efficient. The previous CCS simulations that were investigated involved choosing them randomly from a Gaussian distribution, where the initial wave function was a Gaussian function in phase space. Test calculations indicated that twenty basis functions would be a suitable balanced between the computational cost of the calculations and the fact that more basis functions give a better a description of the wave function. The size of the area that was sampled was adjusted such that the norm would be in the range 0.9900 to 0.9999. This range was believed to be close enough to 1 without introducing a large degree of numerical errors.

One set of results carried out with 20 basis functions gave a norm value of 0.996 (correct to 3 significant figures). These results conserved the classical energy and the norm value very well throughout the simulation, to at least 10 significant figures. However, the total population calculated was not conserved well in either simulations. This is shown in figure 7.2 for both $g = -0.1$ au and $g = -0.3$ au simulations, where the top black dashed line increases from 10 to over 12 and the red line increases from 12 to over 14. By comparing these two total population lines it can be seen that the total population is better conserved when $g = -0.1$ au than when $g = -0.3$ au. The increase in the total populations is not expected and may be due to numerical errors in the calculations or possibly due to the approximations in the model that is being used.

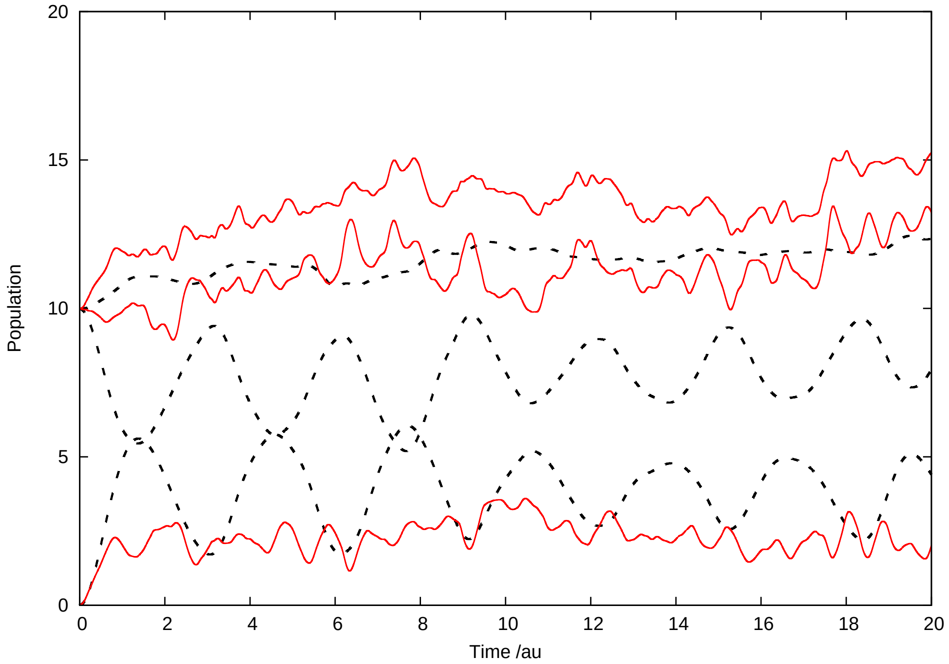

Figure 7.2: The time dependence of the populations of the two potential wells over time for two simulations, one with $g = -0.3$ au (red line) and the other with $g = -0.1$ au (black dashed line). Twenty basis functions were used and these were randomly chosen from a Gaussian function, which was positioned at the centre of the wave function ($z_0^{(1)} = \sqrt{10} + 0i$ and $z_0^{(2)} = \sqrt{0} + 0i$ for the two degrees of freedom) in phase space. The top lines show the total population whilst the middle and bottom lines show the population in each potential well.

For the $g = -0.3$ au results, the population in potential well 1 increases from a population of 10 to approximately 12 and the population in well 2 increases from 0 to 2. It is difficult to determine if any periodic tunnelling is occurring due to high fluctuations in the total population. However, if any tunnelling is occurring it is likely to be relatively small, which implies that self-trapping is occurring.

The dashed black line shows that when $g = -0.1$ au tunnelling is occurring between the potential wells. However, unlike the results with 1 basis function, the amplitudes of the oscillations of the tunnelling are not as consistent. The population in potential well 2 was initially 0 and increased to over 5 but then decreased to 2 and did not return to 0. The population increase in this potential well also changed throughout the simulation as did the period of the oscillation. This is clearly very different to the semi-classical results where the population increases and decreases by the same amount periodically.

Renormalising the population in each potential well by calculating $P_1/(P_1 + P_2)$ for the first degree of freedom and $P_2/(P_1 + P_2)$ for the second degree of freedom, where P_n is the calculated population in the n^{th} degree of freedom, gives clearer results as shown in figure 7.3. In these results it is clear that for the $g = -0.3$ au calculations there are

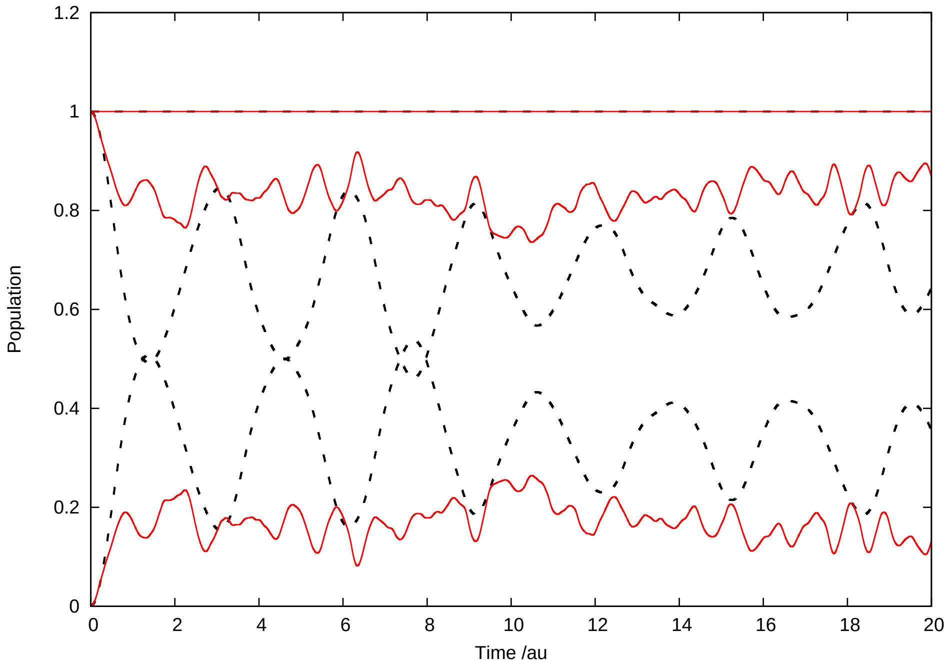

Figure 7.3: The time dependence of the populations shown in figure 7.2 but renormalised to give total populations of 10.

some small oscillations between the potential wells but overall self-trapping occurs. The renormalised results with $g = -0.1$ au do not show any significant differences.

Although these results show that when $g = -0.1$ au tunnelling is occurring and that when $g = -0.3$ au self trapping occurs, it would be more desirable to have results where the total population remains conserved without the need to renormalise. One method of decreasing the numerical errors in the calculations, discussed previously, is to repeat the calculations with different randomly chosen basis functions and take a mean average. The calculations were repeated 100 times and the averaged results can be seen in figure 7.4 (for the $g = -0.3$ au calculation) and figure 7.5 (for the $g = -0.1$ au) calculation. The error bars at the points shown on the graphs are calculated from the standard deviation of the mean of the averaged results that were repeated 100 times.

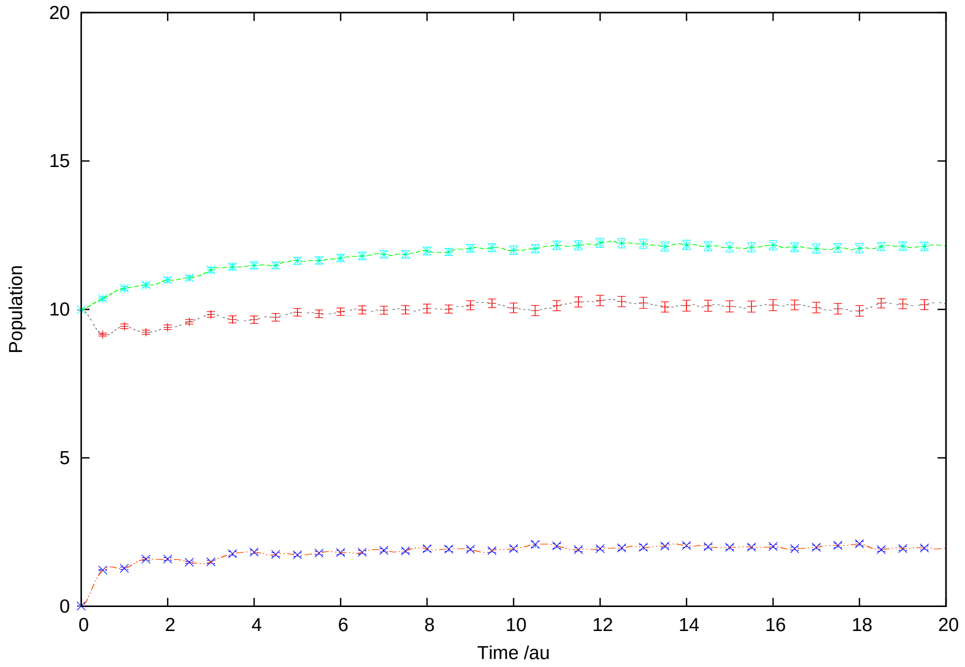

Figure 7.4: The time dependence of the populations of the two potential wells over time. The mean average was calculated using 100 simulations with $g = -0.3$ au. The basis sets consisted of twenty basis functions that were randomly chosen from a Gaussian function, which was positioned at the centre of the wave function as before. The first potential well is shown by the red line (with blue error bars), the second potential well is represented by the grey line (with red error bars) and the total population is shown by the green line (with turquoise error bars).

The results in figures 7.4 and 7.5 show smoother trends in the population compared to the previous results shown in figure 7.2. Therefore the oscillations shown for the simulation using $g = -0.3$ au in figure 7.2 are most likely due to numerical errors, which have now been reduced. The amplitudes of the oscillations that represent tunnelling for the simulation using $g = -0.1$ au are also more consistent, but overall the population appears to be tending towards a steady state, which is not consistent with the semi-classical results.

The overall total populations are better conserved when the average is taken for 100 results. For the simulation with $g = -0.3$ au the total population increases by approximately 2, which is better than the increase of approximately 5 in figure 7.2. The populations in each potential well are relatively constant after the initial decrease in the population of potential well 1 and increase in the population at potential well 2. However, the population in potential well 2 does not return back to 0. Comparing the total populations when $g = -0.1$ au in the same way does not show the same improvement and so this suggests that the numerical errors are larger when $g = -0.3$ au.

A different method for sampling a basis set, which can yield more accurate results, is by choosing random basis functions from a Gaussian function that satisfy additional constraints. One possible constraint is that the energy of the basis function is within a

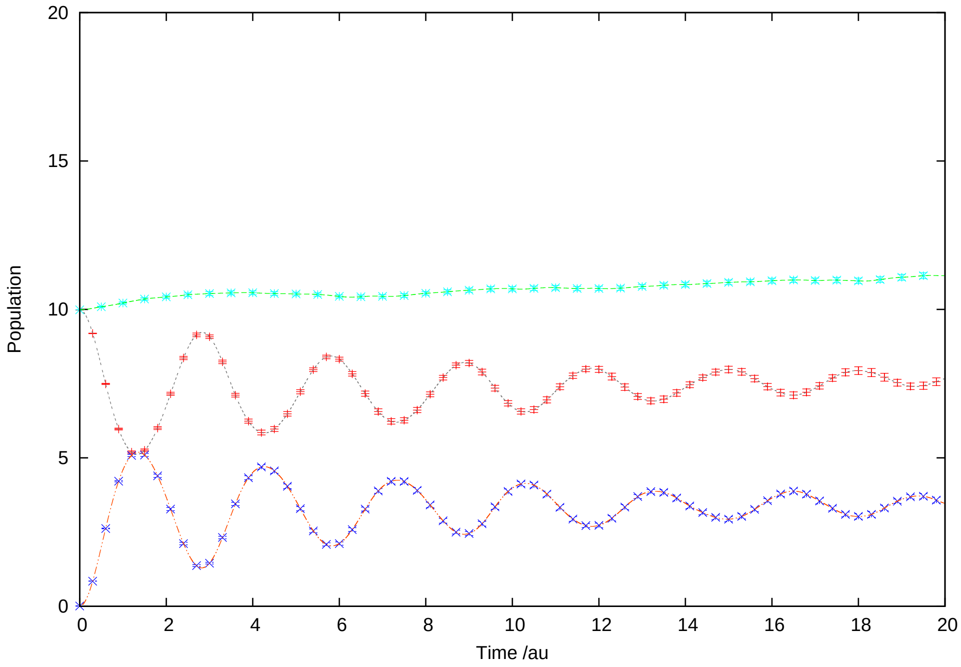

Figure 7.5: The time dependence of the populations of the two potential wells over time as shown in figure 7.4 but with $g = -0.1$ au.

specified range. In addition to this constraint, the population of the basis function in each degree of freedom will also be restricted to be within a specified range. The energy range allowed was calculated by using $E_1 \pm \Delta E$, where E_1 is the energy calculated with one basis function that is positioned in phase space at the initial position of the wave function. The range for the population was calculated in the same way using $P_1 \pm \Delta P$ and a value of $\Delta P = 2$ was found to give the best conservation of the total population.

In investigating the ΔE values, one rule of thumb that has been used previously to obtain clear results was $\Delta E = \pm E/2$. The results shown in figures 7.6 and 7.7 were obtained by using this rule. E was calculated for one basis function and ΔE was selected using this rule. One hundred basis sets were configured in this way and the calculations were run for $g = -0.1$ au and $g = -0.3$ au. Figures 7.6 and 7.7 show the averaged results obtained from these simulations.

A comparison between the set of results that had no energy and population restrictions, figures 7.4 and 7.5, and ones that had energy and population restrictions, figures 7.6 and 7.7, shows that the restrictions give a much better conservation for the total population. For the $g = -0.3$ au results the total population decreases from approximately 12, without restrictions, to approximately 10.5 with restrictions. The magnitude of the error bars in potential well 1 also decrease. A further difference is that in figure 7.6 the population in potential well 1 tends to decrease from 10 to 8 whilst in figure 7.4 the population in potential well 1 remains at approximately 10, after fluctuating initially. The good conservation of the total population suggests that these results may be the most accurate

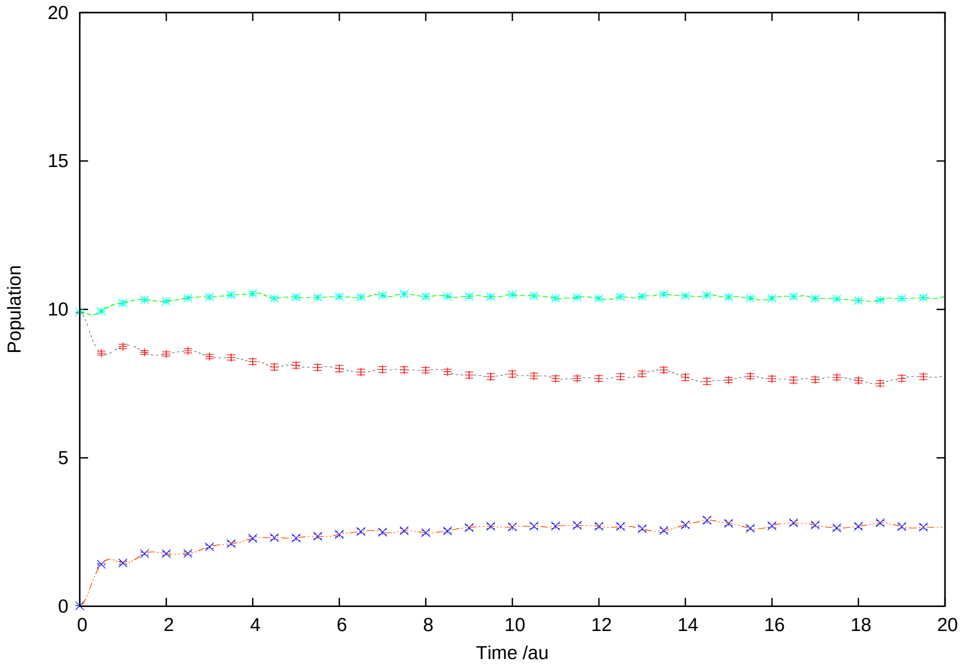

Figure 7.6: The time dependence of the populations of the two potential wells over time. The mean average was calculated using 100 simulations with $g = -0.3$ au. The basis sets consisted of twenty basis functions that were randomly chosen from a Gaussian function with the initial total population restricted to be in the range $8 - 12$ and the initial classical energy of each basis function to be $\pm E/2$, where E is the classical energy of one basis function positioned at the centre of the wave function in phase space. The centre of the Gaussian function was positioned at the centre of the wave function as before. The first potential well is shown by the red line (with blue error bars), the second potential well is represented by the grey line (with red error bars) and the total population is shown by the green line (with turquoise error bars).

ones calculated with 20 basis functions when $g = -0.3$ au. However, comparing these results to the renormalised results for one simulation in figure 7.3 suggests otherwise as the small oscillations that agree with the semi-classical results are removed.

The $g = -0.1$ au set of results are also interesting. There is a clear increase in the total populations at approximately $t = 3$ au in both figures 7.5 and 7.7. However, in figure 7.7 the total population is better conserved before and after $t = 3$ au at approximately 10.5. The population in well 1 decreases more in figure 7.7 compared with figure 7.5. It is interesting to note that the oscillations due to tunnelling in figure 7.7 do not differ significantly from those in figure 7.5 but that these differ significantly from the semi-classical results. The oscillations in these results decrease in magnitude over time whilst in the semi-classical results the magnitudes are unaffected. However, unlike the $g = -0.3$ au results, this agrees with the renormalised results for one simulation in figure 7.3.

The implications of the quantum calculations are that over time the populations between the two potential wells start to balance out to a certain extent. In the $g = -0.3$ au this occurs slowly over time whilst in the $g = -0.1$ au results this occurs from periodic

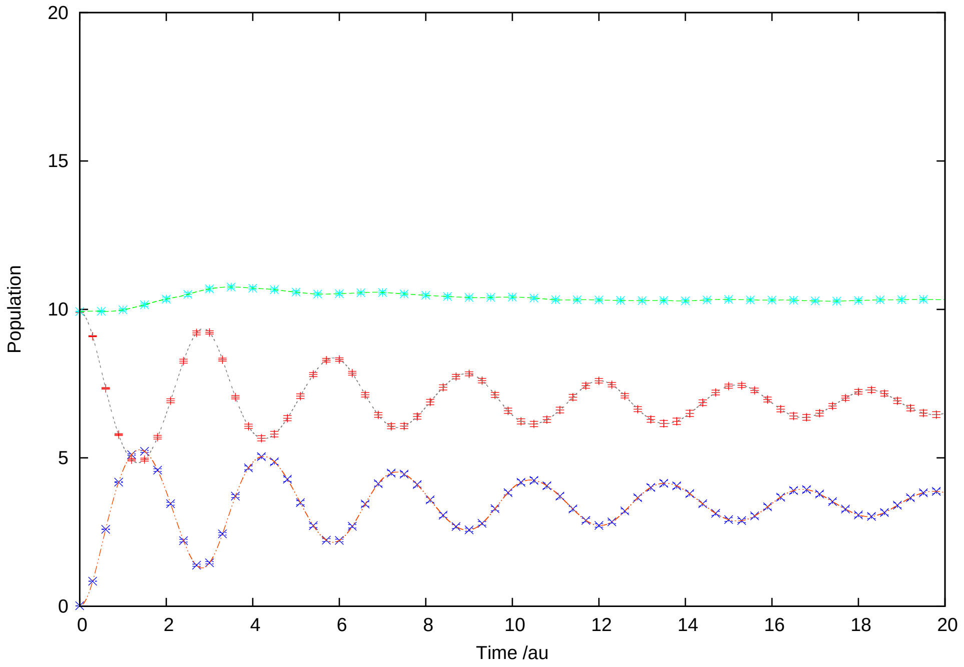

Figure 7.7: The time dependence of the populations of the two potential wells over time, with initial population and energy restrictions as shown in figure 7.6 but with $g = -0.1$ au.

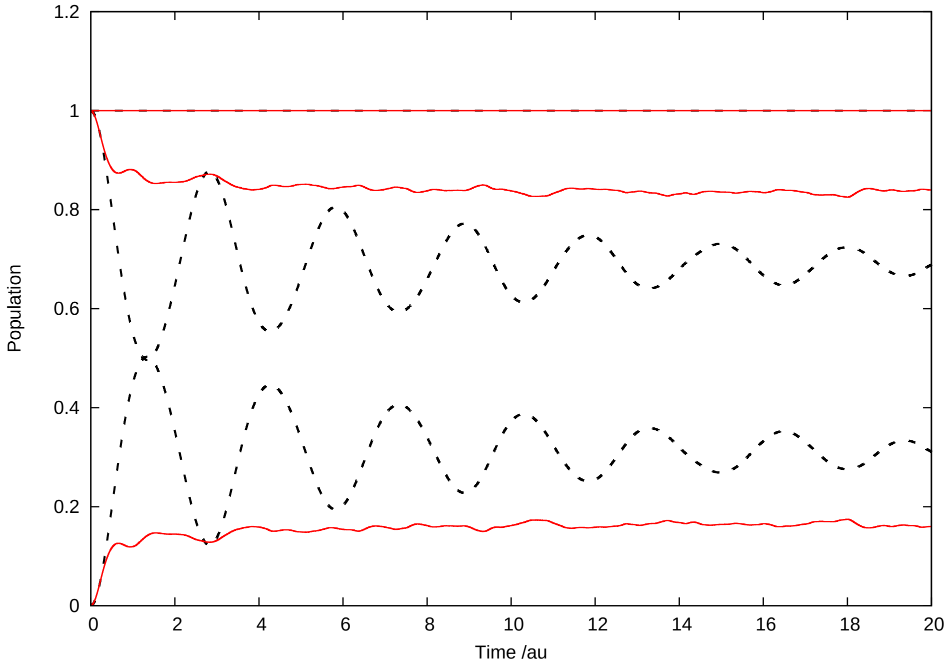

Figure 7.8: The time dependence of the populations shown in figures 7.4 ($g = -0.3$ au, shown in red) and 7.5 ($g = -0.1$ au, shown in black dashed lines) but renormalised to give total populations of 10.

oscillations where a large degree of tunnelling is occurring between the potential wells. Overall periodic tunnelling is occurring when $g = -0.1$ au and this decreases significantly when $g = -0.3$ au, which indicates that self-trapping is occurring. In both cases

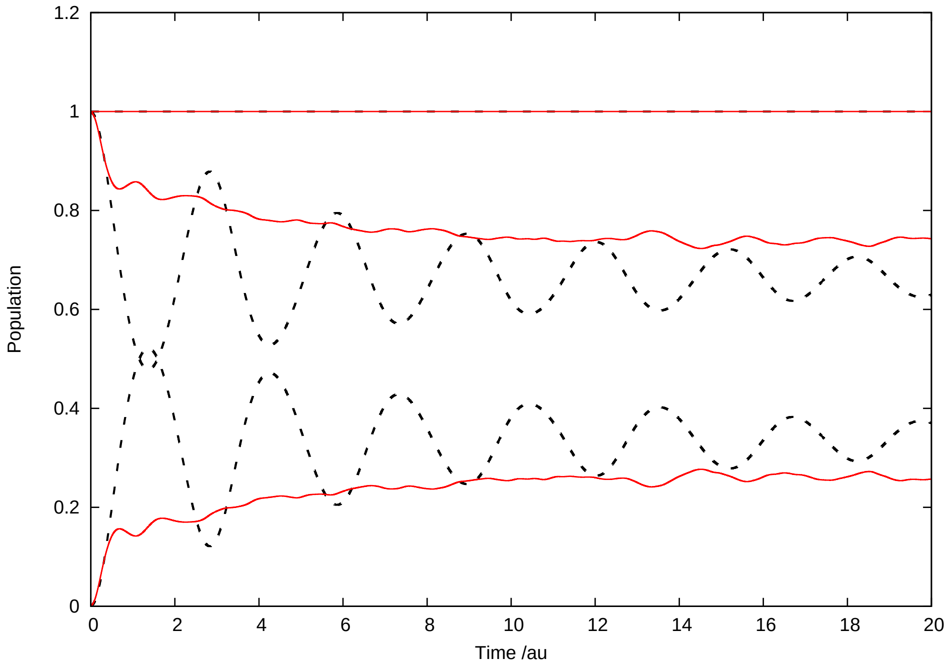

Figure 7.9: The time dependence of the populations shown in figure 7.6 ($g = -0.3$ au, shown in red) and in figure 7.7 ($g = -0.1$ au, shown in black dashed lines), which used initial population and energy restrictions used for choosing the basis functions. These results have been renormalised to give total populations of 10.

the conservation of the population is significantly improved when energy and population restrictions are used.

The averaged results were renormalised, as before, and these are shown in figure 7.8 when no restrictions were used on choosing the basis functions and in figure 7.9 when the basis functions were chosen with energy and population restrictions. The most noticeable difference between these two sets of results is that when $g = -0.3$ au in figure 7.8, which did not use energy and population restrictions for choosing the basis functions, the populations in each well are closer to their initial populations when compared with the results in figure 7.9 when $g = -0.3$ au, which used energy and population restrictions when choosing the basis functions. However, overall the two sets of results are very similar.

7.6 Large Basis Sets

In general a larger basis set will give a better description of a wave function. The difficulty occurs in the practical implementation as larger basis sets result in a greater computational cost and if the area that the basis functions are chosen from is too small then large numerical errors are found. The calculations were repeated with two hundred basis functions to see if any noticeable differences would be found, using a large area of phase space. The same parameters used for the calculations with twenty basis functions were used. The distributions of the basis functions were very broad, with a norm in the range 0.9900 to 0.9999, so as to decrease the numerical errors in the calculations.

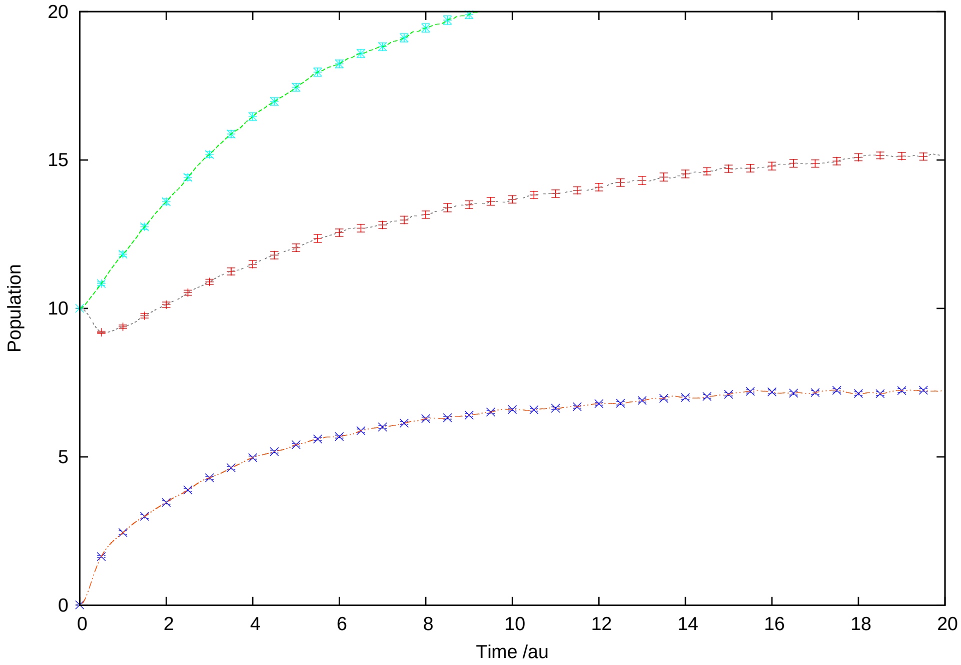

Figure 7.10: The time dependence of the populations of the two potential wells over time. The mean average was calculated using 100 simulations with $g = -0.3$ au. The basis sets consisted of two hundred basis functions that were randomly chosen from a Gaussian function, which was positioned at the centre of the wave function in phase space, as before. The first potential well is shown by the red line (with blue error bars), the second potential well is represented by the grey line (with red error bars) and the total population is shown by the green line (with turquoise error bars).

The results shown in figures 7.10 and 7.11 were obtained when no restrictions were used on selecting the basis functions to see if increasing the size of the basis set would conserve the total population. Unfortunately, in both cases the total population was not conserved well. For the $g = -0.3$ au set of results this increased to approximately 22, which is the largest increase in the total population obtained. The total population for the $g = -0.1$ au set of results were also the least conserved set of $g = -0.1$ au results. Due to the poor conservation of the total population in both cases, these results can not be used to give a good description of the system.

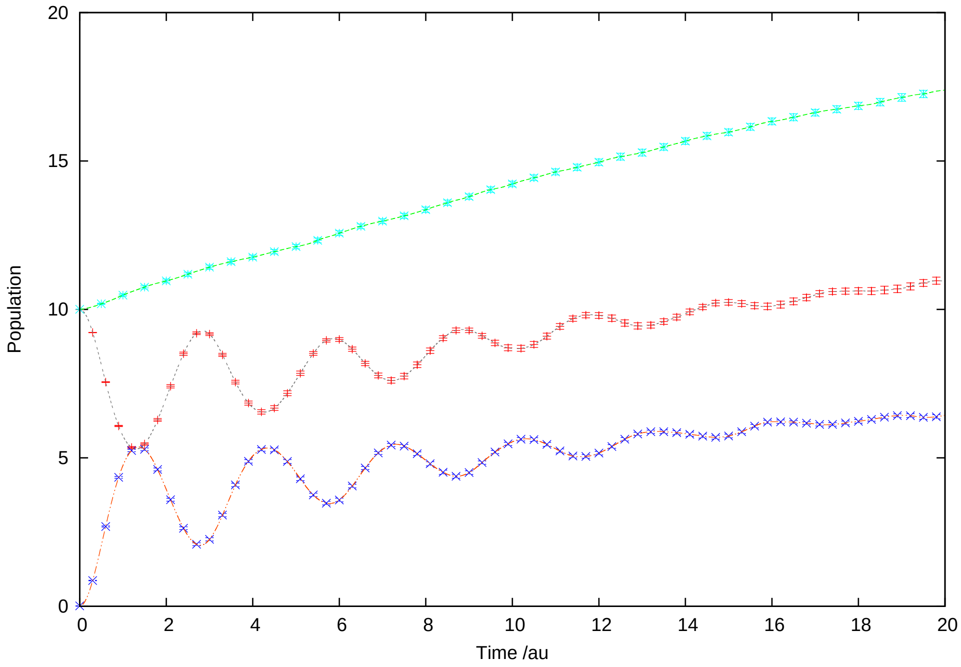

Figure 7.11: The time dependence of the populations of the two potential wells over time as shown in figure 7.10 but with $g = -0.1$ au.

The renormalised results can be seen in figure 7.12. These results do show that tunnelling occurs when $g = -0.1$ au and that self-trapping occurs when $g = -0.3$ au. However, in both cases the populations in each well tend towards the same population, which is unexpected. When $g = -0.1$ au this occurs as the degree of tunnelling decreases but when $g = -0.3$ au this occurs as the population in well 2 increases rapidly initially as the population in well 1 increases rapidly.

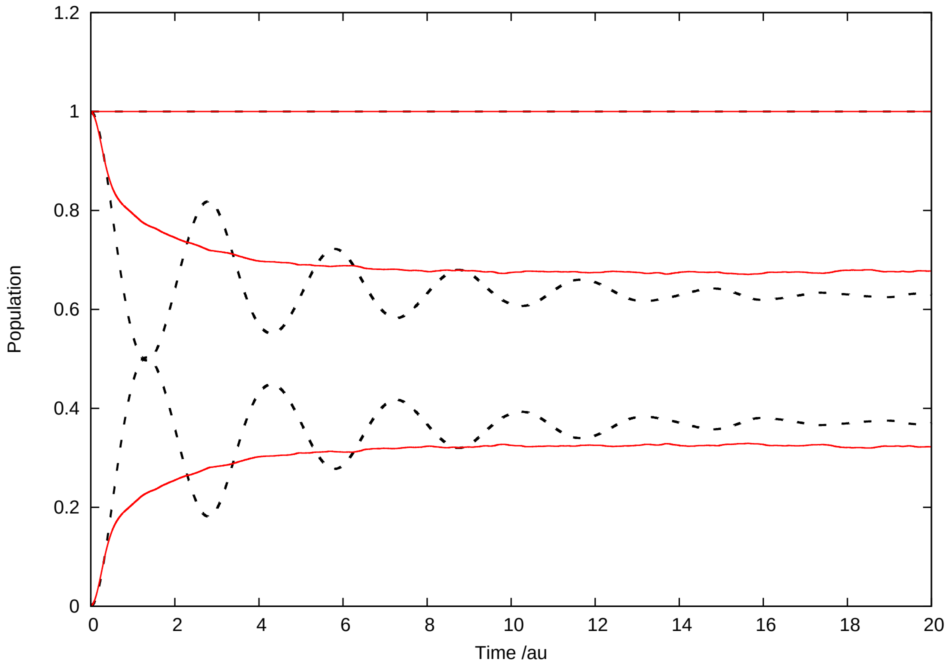

Figure 7.12: The time dependence of the populations shown in figures 7.10 ($g = -0.3$ au, shown in red) and 7.11 ($g = -0.1$ au, shown in black dashed lines) but renormalised to give total populations of 10.

The calculations were repeated with basis sets that had population and energy restrictions, similar to the previous calculations, where $\Delta P = \pm 2$ and $\Delta E = \pm E/2$. The basis functions were broadly distributed to keep the average norm within the same range used previously. These results are shown in figures 7.13 and 7.14.

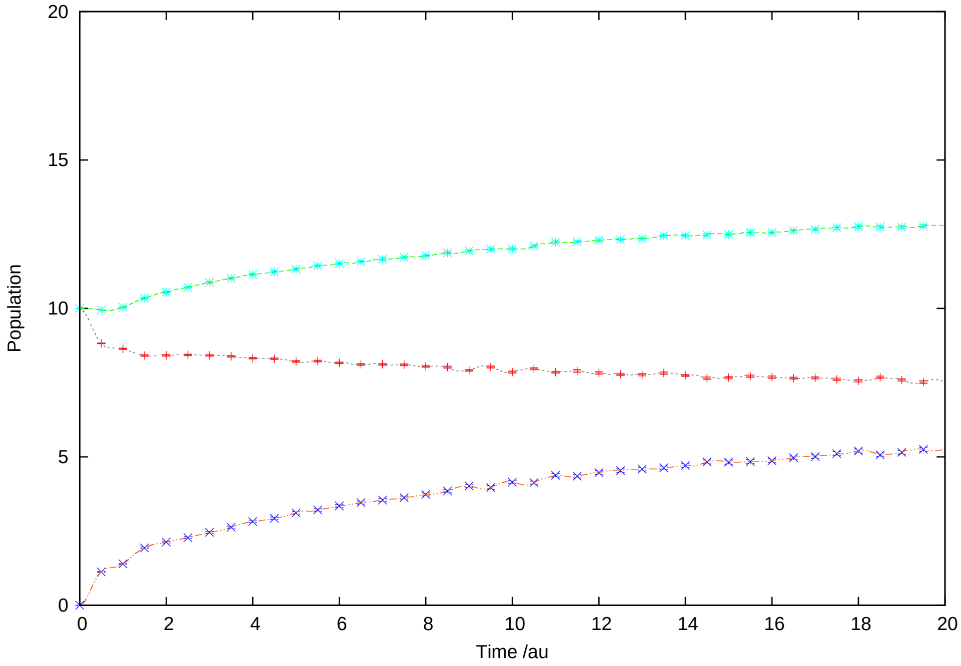

Figure 7.13: The time dependence of the populations of the two potential wells over time. The mean average was calculated using 100 simulations with $g = -0.3$ au. The basis sets consisted of two hundred basis functions that were randomly chosen from a Gaussian function, which was positioned at the centre of the wave function as before. Initial population and energy restrictions were used as described for figure 7.6. The first potential well is shown by the red line (with blue error bars), the second potential well is represented by the grey line (with red error bars) and the total population is shown by the green line (with turquoise error bars).

It is clear that in both cases, using the restrictions of energy and populations on the basis sets, improves the conservation of the total population in the simulation. For the $g = -0.3$ au results the population increased to over 12, which is significantly less than the increase to 22 when restrictions were not used. The total population was increased to approximately 11.5, when the restrictions were used, for the $g = -0.1$ au results, which is also significantly less than the increase to approximately 17 without the restrictions. It is also interesting to note that in the $g = -0.3$ au results the population in potential well 1 decreases linearly whilst the population of potential well 2 increases linearly. Renormalising the results produced the same trends and so the renormalised results have not been shown.

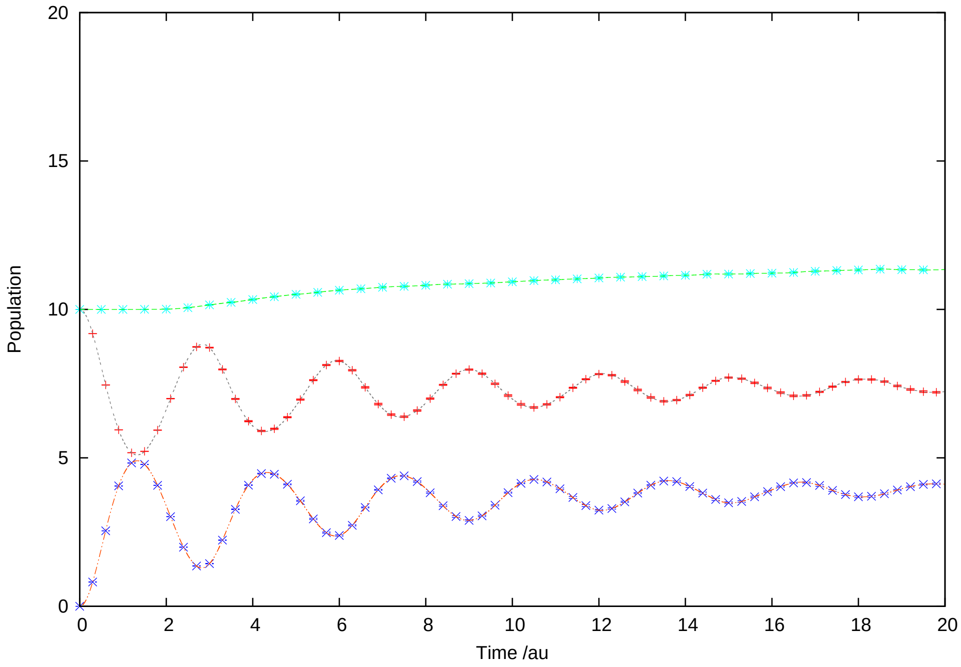

Figure 7.14: The time dependence of the populations of the two potential wells over time as shown in figure 7.13 but with $g = -0.1$ au.

7.7 Excited State Model

A different approach to simulate the Hubbard model, using the CCS method, is to use a wave function that corresponds to the tenth excited state of the Harmonic Oscillator for the first degree of freedom to describe potential well 1. The excitation level of the wave function in each degree of freedom corresponds to the population in each degree of freedom via the number operator. The wave function of the ground state and the tenth excited state of the Harmonic Oscillator can be seen in figures 7.15 and 7.16. There are clear differences between the shapes of these wave functions, such as the hole in the centre of the tenth excited state. The Harmonic Oscillator wave functions are used for the Hubbard model as the Hamiltonian is simply two coupled harmonic oscillators. A computer program that is able to choose the basis functions from an excited state Harmonic Oscillator wave function was written by Reed with possible population restrictions.[41] However, using population restrictions changes the shape of the distribution of the basis functions for the excited state. A Gaussian wave function was still used for the second degree of freedom. The basis sets were configured with this subsidiary program, using 200 basis functions and a compression factor of 2 in the first degree of freedom and 1 for the second degree of freedom, where the norm was in the range 0.9900 to 0.999. The results were repeated 100 times and the averaged results are shown in figures 7.17 and 7.18.

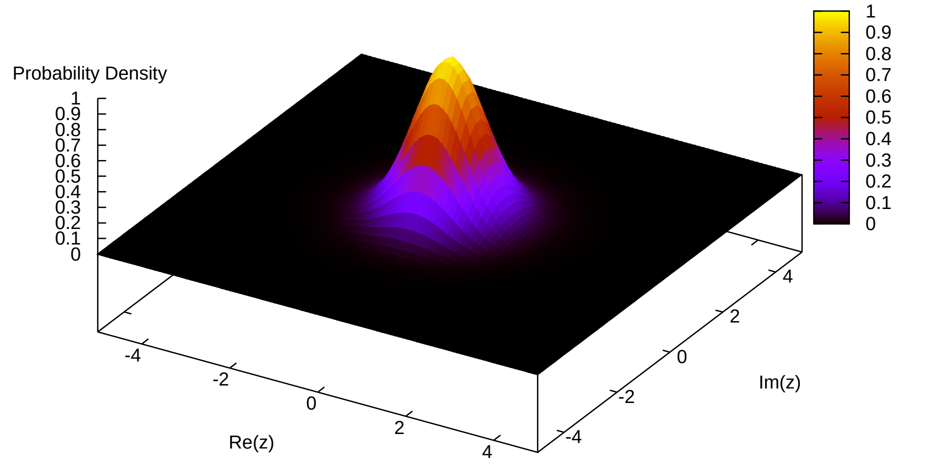

Figure 7.15: The probability density (dimensionless) of the ground state wave function of the Harmonic Oscillator in phase space.

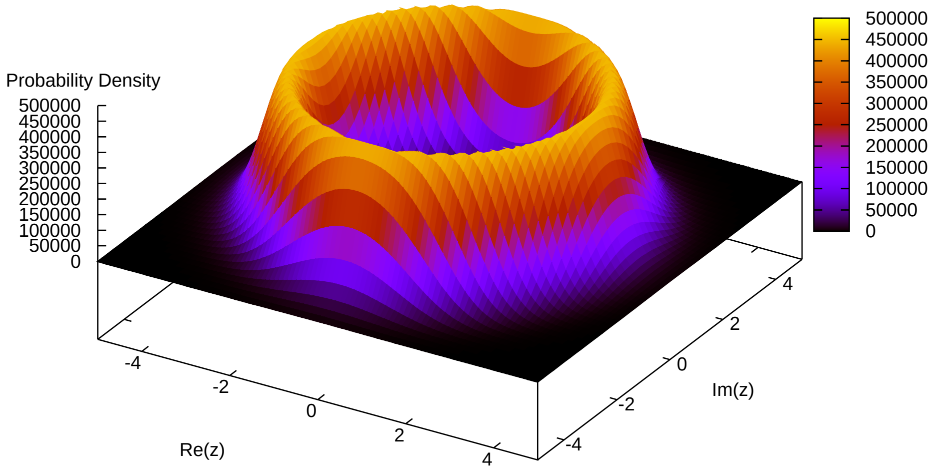

Figure 7.16: The unnormalised probability density (dimensionless) of the tenth excited state wave function of the Harmonic Oscillator in phase space.

The results obtained using an excited state wave function show differences to the previous results obtained. The total population increases to approximately 14 when $g = -0.3$ au and increases to approximately 13.5 when $g = -0.1$ au. The most interesting differences are observed in the $g = -0.3$ au results. Small oscillations which resemble

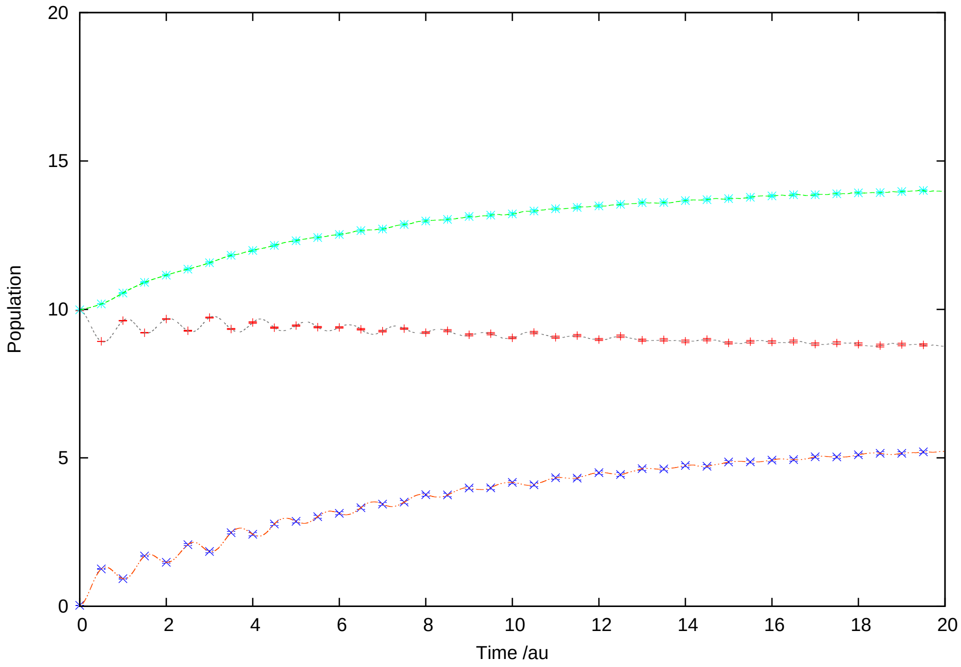

Figure 7.17: The time dependence of the populations of the two potential wells over time. The mean average was calculated using 100 simulations with $g = -0.3$ au. The basis sets consisted of two hundred basis functions that were randomly chosen from a Harmonic Oscillator wave function distribution, the centre of which was positioned at the centre of the wave function as before. For the first degree of freedom a tenth excited state wave function was used and for the second degree of freedom a ground state wave function was used. Initial population restrictions were used on choosing the basis functions. The first potential well is shown by the red line (with blue error bars), the second potential well is represented by the grey line (with red error bars) and the total population is shown by the green line (with turquoise error bars).

the oscillations in the semi-classical results when $g = -0.3$ au are found in both potential wells. The renormalised results in figure 7.19 confirm that the small oscillations found when $g = -0.3$ au are not due to fluctuations in the total population. However, unlike the semi-classical results, potential well 2 increases to approximately 5.5 and the smaller oscillations no longer remain. The most noticeable difference for the $g = -0.1$ au results is that the amplitudes of the oscillations due to tunnelling become less consistent after $t = 12$ au. It is also after this point where the total population is at its highest.

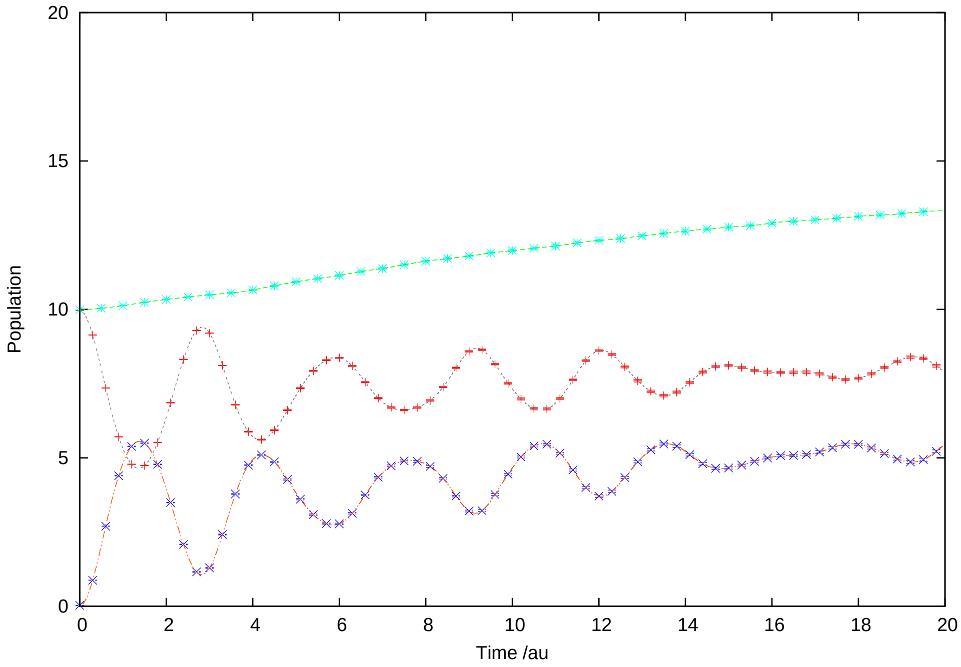

Figure 7.18: The time dependence of the populations of the two potential wells over time as shown in figure 7.13 but with $g = -0.1$ au.

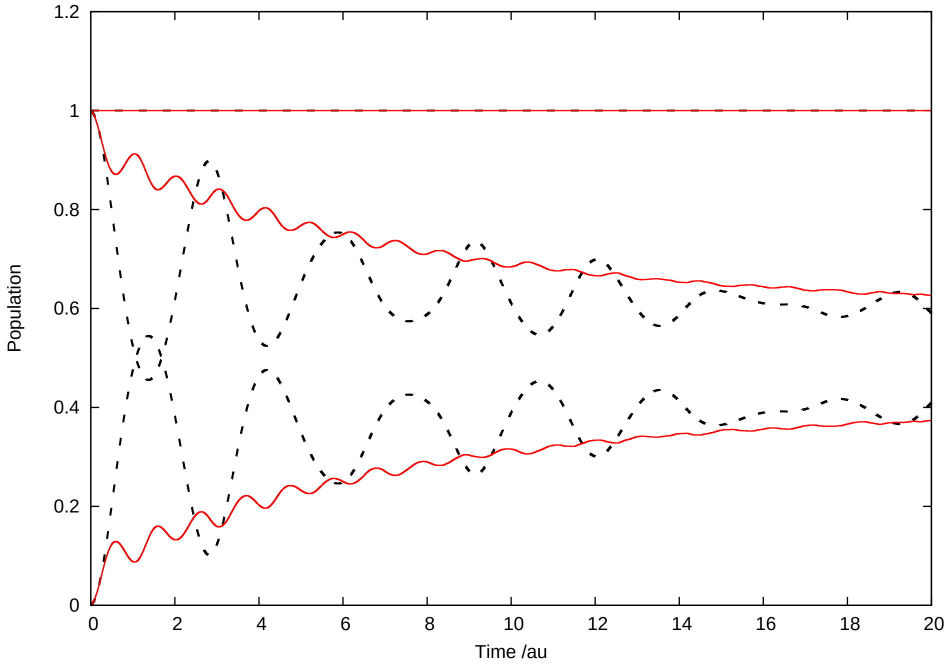

Figure 7.19: The time dependence of the populations shown in figures 7.17 ($g = -0.3$ au, shown in red) and 7.18 ($g = -0.1$ au, shown in black dashed lines) but renormalised to give total populations of 10.

7.8 Summary

In this chapter CCS simulations of the Bose-Hubbard model have been investigated. The semi-classical results show good conservation of the total population and the tunnelling between the potential wells is periodic. CCS results in which the total population was conserved, without renormalising, were obtained when 20 basis functions were chosen randomly from a Gaussian distribution with energy and population constraints. The conserved results show that when $g = -0.3$ au a small degree of tunnelling occurs and so self-trapping is described. The results with $g = -0.1$ au showed a significantly greater degree of tunnelling but this decreases over time. These results differ to the semi-classical results and further work may provide an explanation for this.

The calculations become significantly more computationally demanding when larger basis sets are used. The energy and population restrictions improve the conservation of the total population but overall the total population was not well conserved. Using an excited state wave function instead of a Gaussian wave function, with population restrictions, did not improve the conservation of the total population but small oscillations were observed when $g = -0.3$ au, which resemble the semi-classical results. Further work on imposing energy restrictions on the excited state basis functions may show interesting results. Unfortunately, due to the strict time restrictions of this project combined with the relatively high computational cost of the calculations with large basis sets, it is not possible to investigate this further.

Chapter 8

Future Work and Conclusions

This final chapter first discusses interesting future research that would extend the work carried out in this project. A summary of the conclusions of the project is then given.

8.1 Future Work

The strict time restrictions on this project have meant that some investigations could not be carried out. These include investigations of different numerical techniques that could be used. In addition to this it is possible to extend the work investigated, in particularly using the Morse Oscillator and the Hubbard Model.

The fourth order Runge-Kutta integration technique has been used throughout this work. This integration technique was chosen as previous studies using the Runge-Kutta technique have shown to give accurate results. However this technique is not computationally efficient and there have been many more modern techniques developed. According to Press *et al.* the standard Runge-Kutta routine is used when: $a)$ the person programming does not know any better, $b)$ other methods fail or $c)$ the problem is trivial.[44] Clearly the problem investigated here is not trivial and a thorough investigation that determines whether other methods fail is required.

The simplest way to improve the standard routine is by using an 'adaptive stepsize' integration technique. In this technique the deviation from the gradient is calculated and the size of the time step is changed to give a better description of the gradient. Although there is an initial cost in calculating the deviation it is overcome by using a larger step when the deviations to the gradient are small.

A more advanced numerical integration method uses the 'Richardson Extrapolation' method, called the 'Bulirsch-Stoer' method. Richardson Extrapolation considers the numerical calculation as an analytical function of the stepsize. The approximate result of the calculation is found by using a function that fits the results when the stepsize is equal to 0. The Bulirsh-Stoer method uses this method with rational functions to fit the results. In general, the computational cost of this accurate numerical integration technique is significantly lower than the Runge-Kutta method.

The numerical technique for the Fourier transform is not computationally efficient as it scales as N^2 but was relatively simple to implement. However, there are many techniques known as 'Fast Fourier Techniques' which scale as $N \log_2 N$. The difference between the scaling is significant. However, the disadvantages of these techniques is that they are more difficult to implement and that they require more memory for the calculations. In addition to this, existing 'Fast Fourier Techniques' often have specific requirements on the data that is being analysed. A constant time step is often required, which means that the output of the autocorrelation function must be written at the same time interval step, which is not guaranteed in the current code and is not likely to occur if an adaptive stepsize integration method is used. Furthermore for a large system the time it takes to calculate the Fourier transform will be small compared to the time it takes to run the simulation.

The largest computational cost associated with solving the CCS equations is when the LAPACK routines are used to solve the system of linear equations for the coefficients D. A computationally more efficient routine to do this would be very useful.

The total quantum expectation energy was discussed in section 2.5 but was not used in this work. Comparing the total quantum expectation energy with the total classical energy, throughout the simulation, indicates any differences between using classical trajectories and fully quantum trajectories for the basis function in the simulations. This is because if the equations of motion were derived by using the quantum Hamiltonian operator for the amplitudes and the basis functions the total quantum expectation energy would be conserved. Therefore further work that investigates the conservation of both of these properties is required for the systems investigated in this work.

The Morse Oscillator was used as a simple model for this project and the calculations based on symmetry were focused on this. This model is useful for describing potential energy surfaces of chemical systems. One well-known example of the Morse Oscillator is using it to describe the potential energy function of diatomic molecules. The Morse Oscillator gives a particularly good description of the energy levels of a diatomic molecule, where the spacing of the levels decreases as the energy rises closer to the dissociation energy. It is also used as part of a potential energy surface for more complicated systems. For example the Morse Oscillator was used to describe one degree of freedom for HO-SO_2.[19,32] Furthermore it is possible to couple many Morse Oscillator potentials together to describe a large chemical system. It would be interesting to calculate the autocorrelation function, using the symmetry arguments discussed, and the Franck-Condon spectrum of such a system and compare with other methods or experimental data. In addition to this, a potential energy surface often uses coupling between the degrees of freedom and it would be beneficial to add this coupling to the existing software.

Simulations of the Hubbard model, using a Gaussian wave function showed interesting results. The conservation of the total population was particularly difficult to achieve, particularly with the large basis sets. Further calculations with different restrictions on the basis set may be able to achieve this conservation. The central idea that motivated the work behind these calculations was to see if tunnelling and self-trapping of a Bose-Einstein condensate can be modelled. Similar calculations may be used to describe other quantum systems that require the Bose Hubbard Model such as multiple potential wells.

Using an excited state wave function for one degree of freedom in the Bose-Hubbard calculations showed interesting results but the total population was not as well conserved compared with other calculations. One possible reason for this is that no energy restrictions were used when selecting the basis functions. In order to introduce energy restrictions the program must calculate the Hamiltonian function. The difficulty arises due to the fact that this program is not part of the original software that was written for this project and linking this to the existing modules that describe the Hamiltonian was found to be difficult, in the time available. It would be beneficial to write an option that allows the wave function that is used in the original software to be an excited state wave function so as to allow the option of energy restrictions. This option would not only be useful for the Hubbard calculations but also for comparing the Franck-Condon absorption spectra for various excited state systems.

The computer software that has been written is flexible in which system to calculate. To add a new system the reordered Hamiltonian function must be defined along with its derivative in the HamiltonianSpec module. The appropriate parameters for the Hamiltonian must also be defined and these must be read from inham.dat file. However, an addition of a new multidimensional Hamiltonian is relatively straight forward and this flexibility allows the software to be used for a variety of different potential energy surfaces. This clearly emphasises the reasoning behind the design of the software and in particular the use of the GenHam and HamiltonianSpec modules. It also illustrates how adaptable and useful the software written during this project is. The next step is to use the software for a larger chemical system.

8.2 Conclusions

The initial aim of this project was to write a computer program that implements the CCS method using the Fortran programming language and to apply this to an interesting system.

One main program was written that simulates the wave function and two smaller subsidiary programs were written to initialise the basis set and analyse the results. This allows the same basis sets to be used for a number of different calculations and the outputs of the main program could be analysed in different ways without the need to redo the entire simulation.

The modular structure, which is inherent in Fortran, is used so that a change in one module would not result in large changes in the other modules or the main program. In addition to this new and advanced features of the Fortran programming language were used to make the main program easy to follow. Some key routines of the program have been shown in Appendix C and the three programs are available on the CD provided.

Testing the software showed that properties such as the classical energy and the norm were conserved to almost machine precision for simple systems. The conservation of these properties is expected from the equations of motion that are used in the CCS method. The tests also showed that increasing the number of basis functions in a specified region of phase space gave a norm value that was closer to one but that numerical errors were also found if the overlap of the basis functions was too high.

The software was used to simulate a Gaussian wave function positioned in an excited state in phase space for the Harmonic and Morse Oscillator systems. These results were analysed and clear Franck-Condon absorption spectra were obtained when a window function was applied to the autocorrelation function before calculating the Fourier transform. The positions of the large peaks in the spectra matched the eigenvalues of the systems exactly. This shows that a Gaussian wave function can be used to obtain a Franck-Condon spectra with accurate eigenvalues, when positioned appropriately in phase space.

A symmetrical basis set was then used for the Morse Oscillator system and it was shown that the symmetry of the system was retained in the trajectories of the basis functions and the amplitudes. These arguments were used to rewrite the equation of the autocorrelation function such that the autocorrelation function at time t was calculated using the positions of the basis functions and the amplitudes at time $\frac{t}{2}$. The autocorrelation functions calculated using this method were shown to calculate the eigenvalues of the Morse Oscillator accurately but the intensity of the peaks in the spectra were lower.

The particularly interesting problem that was investigated in this project was the simulations of a wave function for the Bose-Hubbard Model. Overall it was found that the wave function tunnels between two potential energy wells if the self on-site interaction is low and the wave function self-traps when the magnitude of the on-site interaction is high. Despite a good conservation in the classical energy and the norm, conservation of the total population was difficult to achieve. Restrictions on choosing the basis functions helped to improve this conservation. Interestingly using an excited state Harmonic Oscillator wave function for first degree of freedom rather than a Gaussian shaped wave packet, positioned in at an excited state in phase space, gave slightly different results which are more similar to the semi-classical results. Further experimental and theoretical studies will be useful so as to provide a physical description of the system.

The initial aim of this project has been achieved, where a well organised program has been written and applied to different systems. The next step forward would be to use the software for other larger and interesting chemical systems.

Appendix A

Software Diagrams

This appendix consists of two flow charts that give a better description of the software and the modules used in the main program. Figure A.1 shows how the main program must always be used for a simulation but that if a new basis set is not required the basis set program does not need to be run. Similarly, if the Frank-Condon absorption spectra results are not required the Fourier transform program does not need to be used.

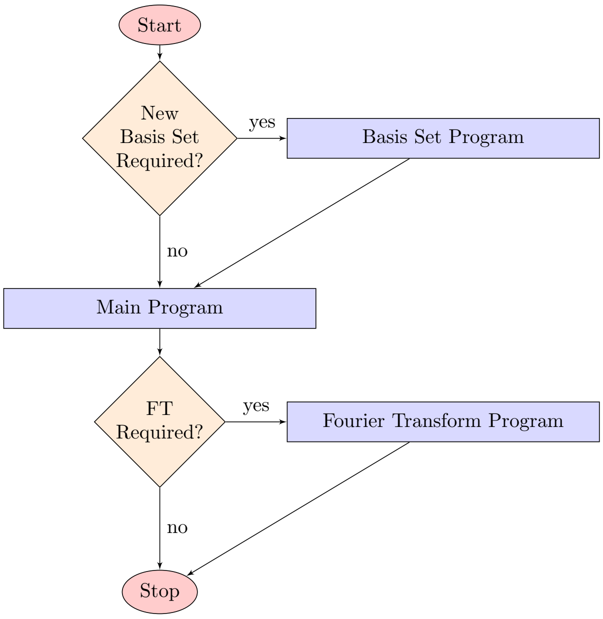

Figure A.1: A flow chart that shows how the three computer programs can be used to set up, simulate and analyse a system. FT is the Fourier transform

An overview of the main simulation program is given in figure A.2. The names of the modules correspond to those described in section 3.6. Figure A.2 clearly shows that the HamiltonianSpec module is never called directly by the main program. Instead it is called by the GenHam module, which acts as an interface between them. The link between OutMod and BSET has also been shown. Here, subroutines in the BSET module

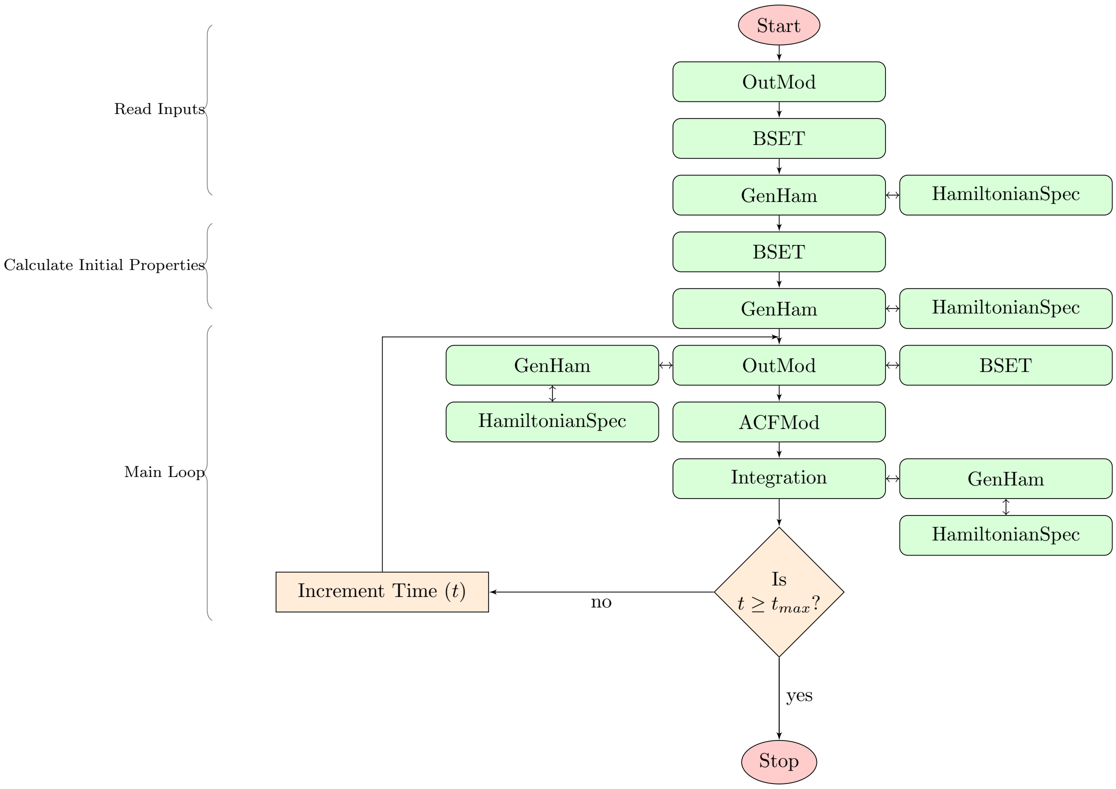

Figure A.2: A flow chart that shows an overview of how the main program calls the individual modules (shown in green blocks) to initialise, simulate the wave function and output the results. The maximum time of the simulation is given by t_{max}.

calculate properties, such as the norm. Subroutines in the OutMod module call these subroutines in BSET to calculate the properties and then to write the results to the output files. There are many other links between modules, such as the links between the defined type in BSET with the other modules. However, including all the links would make the flow chart difficult to follow and so only the two selected links have been shown. For the calculations that use the Bose-Hubbard system, the ACFMod must be replaced by the Hubbard module.

Appendix B

Software Instructions

The instructions for compiling and using the software, on the CD provided, are given in the order in which the three different programs are likely to be used.

B.1 Basis Set

To compile the basis set program the source files randgen.f, basisset.f90, HamiltonianSpec.f90, GenHam.f90, Hubbard.f90, Checks.f90 and mkgbasisprog.f90 must be compiled in this order. The 'compile' script provided in the directory 'basis' uses the gfortran compiler, with some additional options, to compile the basis set program. The script can be run by typing './compile'.

The basis set program requires an 'input.dat' for it to run. The basis set program has been written so that any lines that begin with '#' are comments, which are ignored by the program. One possible 'input.dat' file that can be used is shown below. The comments in this file explain what is being read.

```
#!!!!!!!Input file!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#number of basis functions given by nbf:
nbf 10
#number of degrees of freedom given by:
ndim 1
# ! Give the value of hbar to define the units
hbar 1.0d0
#Do you want a symmetric basis set (YES/NO)?:
sym YES
# The gamma for each degree of freedom
# syntax is :
# word(gamma) (degree number) gamma(value)
# e.g:
# if you have more than one degree of freedom put it directly below the first
gamma 1 1.0d0
# the same syntax is used for the mass of the system
mass 1 1.0d0
# and the angular frequency:
wfreq 1 1.0d0
# give a seed value for the random number generator, 0 =automatically generate 1
```

```
#SEED 0
# output unit value, where to print the seed,
outp 130
#give the mean p and q of the wave function
#use the same syntax as for gamma
meanp 1 0.0d0
meanq 1 5.0d0
# give the standard deviation of the Gaussian wave function in p and q
sigp 1.0d0
sigq 1.00d0
```

If energy or population restrictions are required an ‘inham.dat’ file must be provided. This is shown below for the Bose-Hubbard Model:

```
# System name, for Hubbard type Hubbard
System: Hubbard
# epsilon value for Hubbard model
HubEps -0.5d0
# on-site g value
HubG   -0.1d0
# coupling value nu
HubNu   1.d0

# gives an energy min point for basis function
#Ebfmin  -32.0d0
# gives an energy max for basis function
#Ebfmax  -8.0d0
#PopChk will determine if the pop check function will be on (YES/NO)
#PopChk YES
# gives a total pop cut off
#Popmin 9.5
#Popmax 10.5
#gives a max number of tries to initialise basis function with restrictions
Ntries 200
```

The inham.dat file is a system specific input file. The input for the Free Particle is:

```
System: Free
```

For the Harmonic Oscillator the lines required are:

```
System: Harmonic
# give force constant for each degree of freedom
ForceK 1 2.d0
```

The Morse Oscillator requires:

```
System: Morse
# alpha is parameter in the exponential
alpha 1 0.22090d0
# Morse D is the dissociation energy
MorseD 1 10.250d0
```

The program will generate a ‘General.out’ file with information used in the program and will show any errors produced. A ‘basisset.dat’ file is also produced and this is shown for one basis function with one degree of freedom below:

```
 ndof             1                                   #number of degrees of freedom
 nbasisfns        1      #number of basis functions
 basis            1      #basis function number
 C -8.68118578419914827E-002  0.50284013276624240      #amplitude C of basis function
    5.00000000000000000       0.00000000000000000   # z in first degree of freedom
```

B.2 Main Program

The same input files are required for the main program but the ‘inham.dat’ must also contain the lines:

```
# maximum time of simulation
tmax 20.0d0
# initial time
t0 0.0d0
# time increment
dt 0.001d0
# change the frequency of the output produced, dn is the maximum
dn 0
# the following lines are optional depending on what is required from the simulation
# name of file that shows conservation of energy and norm
Consv name.conv
#name of file that shows trajectories of basis functions
Traj  name.tra
#name of file that shows the population for the Hubbard model in 2 degree of freedom
Pop   name.pop
# name of file that shows a general output of the file with inputs that were used
OutPname name.outp
# name of autocorrelation file
#ACF nameACF.dat
```

If one of the files is not required to be produced then the option may be commented out using ‘#’ as before. The easiest way to compile the program is to use the ‘compile’ script in the ‘mainprog’ directory shown below.

```
FC=gfortran
cd LAPACK
$FC -c dep.f driv.f
cd ../
$FC -O3 -o ReadBSProgO3 -m64 -static-libgfortran -fbounds-check
./LAPACK/dep.o ./LAPACK/driv.o basissetread.f90 ACF2.f90 HamiltonianSpec.f90 GenHam.f90
Integration.f90 Hubbard.f90 OutMod.f90 prog3test.f90
```

In this script the LAPACK routines are compiled first and then these are linked with the other source files when the main program is compiled together.

B.3 Fourier Transform

The program that calculates the Fourier transform requires the ‘inham.dat’ file with the following information:

```
#name that ACF file that is read in
ACF nameACF.dat
#FT parameters
# window function? YES/NO?
Winop YES
#Maximum value of E for FT(E)
FTEmax 30.0d0
#Minimum value of E for FT(E)
E0   0.0d0
#The file with ACF*window function results
DecOut name.dec
#Name of FToutput file
FTOut  name.ft
#T_cut for the window function
FTcut 20.0d0
#Number of lines that need to be read in from ACF
#To find this look at the end of the ACF file
MAXJ 50001
```

The program must be compiled with the files ‘FT2.f90’ and ‘Fprog.f90’ in this order.

Appendix C

Code

The Fortran software, written for this project, consists of many subroutines within the modules described. The software is available on the CD provided and in addition to this some key features of the main program is shown in this appendix. The new basis set type and the functions that describe how to use this are shown. A modified version, due to formatting differences, of the key routines in the 'Integration' module, which benefit from the basis set type are also given.

C.1 Basis Set Type

The basis set type is defined as type 'csbasisfn' and consists of a complex allocatable array, z, which labels the basis function, a complex number (D) for the amplitudes and a real number S for the classical action. The operator $+$ adds the same components of two basis functions together and the operator × multiplies a real number by each component of the basis function. These are defined in the functions 'add' and 'mult'.

```
!$*************************************************************************!!
!$                          Derived Type: csbasisfn
!$
!$      This is a derived type which will define each individual basis function
!$      An array of these types (basis functions) will therefore give the basis
!$      set.
!$      The type consists of a complex array of zs which will depend on n*
!$      dimensions (ndim).
!$      It will also consist of an amplitude(complex) and an action variable
!$      (real).
!$*************************************************************************!!

  type csbasisfn
     complex(kind=8),dimension(:),allocatable::z
     complex(kind=8)::d
     real   (kind=8)::s
  end type csbasisfn
  interface operator (+)
     Module procedure add
  end interface

  interface operator (*)
     Module procedure mult
  end interface

!$******************************************************************************!!
!$                        Functions: OVERLOAD
!$      The functions add and mult are required to overload the operators
!$      The function add, adds two basis functions togther
!$      It adds the corresponding components of the basis functions
!$      The function mult, multiplies a real number by a basis function
!$      These functions are required for the integration routine
!$
!$******************************************************************************!!
  Function add(bs1,bs2)
    Implicit none
    type(csbasisfn),dimension(:),intent(in)::bs1,bs2
    type(csbasisfn),dimension(size(bs1))::add
    integer::i
    call IBASISSET(add)! allocates memory for the z array
    If((.not. size(bs1)==size(bs2)) .OR. (size(bs1)==0 .OR. size(bs2)==0)) then
            Print *, 'basis sets added arent equal sizes'
    End If
    do i=1,nbf
       add(i)%s=bs1(i)%s+bs2(i)%s
       add(i)%d=bs1(i)%d+bs2(i)%d
       !Print *, size(add%z), 'size'
       add(i)%z(1:ndim)=bs1(i)%z(1:ndim)+bs2(i)%z(1:ndim)
    end do
  END FUNCTION add

  Function mult(r,bs1)
    Implicit none
    type(csbasisfn),intent(in),dimension(:)::bs1
    real(kind=8),intent(in)::r
    ! this will help check kind is always the same
    type(csbasisfn),dimension(size(bs1))::mult
    integer::i
    call IBASISSET(mult)! allocates memory for the z array
    do i=1,nbf
       mult(i)%s=bs1(i)%s*r
       mult(i)%d=bs1(i)%d*r
       mult(i)%z(1:ndim)=bs1(i)%z(1:ndim)*r
    end do
  End Function mult
```

C.2 Integration Routine

The fourth order Runge-Kutta routine that uses the csbasisfn type, is given here. The key feature of this routine is how the csbasisfn type is used as a set data structure, which keeps the routine simple. This routine is very similar to the Runge-Kutta routine given by Press *et al.*[44]

```
!$******************************************************************************!!
!$                        Subroutine RK4ccs
!$      The arguments are:
!$      y - this is bs
!$      dydx- this is bsdot (the deriv of bs)
!$      x- time t
!$      h- time step, I will call this dt
!$      yout- the integrated output of the bs
!$******************************************************************************!!
    Subroutine RK4ccs(y,dydx,x,h,yout)
      Implicit none
      type(csbasisfn),dimension(:),intent(in)::y,dydx
      type(csbasisfn),dimension(:),intent(out)::yout
      real(kind=8),intent(in)::x,h
      real(kind=8)::h6,hh,xh
      type(csbasisfn),dimension(:),allocatable::dym,dyt,yt
      ! ' starting derivs'
      CALL ALCBASIS(dym)
      CALL ALCBASIS(dyt)
      CALL ALCBASIS(yt)
      ! 'allocated memory in derivs'
      hh=h*0.5d0
      h6=h/6.0d0
      xh=x+hh
      yt(:)=hh*dydx(:)
      yt(:)=yt(:)+y(:)
      !yt=y+tt*dydx ! first step
      !Print *, 'in rk4 calling derivs'
      call derivs(xh,yt,dyt)!second
      ! equation was: yt=y+tt*dyt but had to split it up
      yt(:)=hh*dyt(:)!+y
      yt(:)=yt(:)+y(:)
      !Print *, 'in rk4 calling derivs'
      call derivs(xh,yt,dym)! third step
      !equation was: yt= y+tt*dym
      yt(:)=h*dym(:)
      yt(:)=yt(:)+y(:)
      dym(:)=dyt(:)+dym(:)
      call derivs(x+h,yt,dyt)! fourth step
      ! equation was: yout=y+t6*(dydx+dyt+2.0d0*dym)
      ! accumulate the proper weights
      dym(:)=2.0d0*dym(:)
      yout(:)=dydx(:)+dyt(:)
      yout=yout(:)+dym(:)
      yout=h6*yout(:)
      yout=yout(:)+y(:)
      DEALLOCATE(dym)
      DEALLOCATE(dyt)
      DEALLOCATE(yt)
    End Subroutine RK4ccs
```

C.3 Derivatives

The ‘Derivs’ routine returns the derivatives for the **Z**, D and S values of the basis function. The derivatives for the amplitudes are more difficult to write and so have been written in the ‘Amp2d’ routine, which is called here.

```
!$******************************************************************************!!
!$                        Subroutine Derivs
!$      This subroutine will take in the basis set and give out the derivative
!$      of the z,s and ds.
!$      The arguments are:
!$      t- time
!$,     bs- basis set(input)
!$      dbsdt- derivative of basis set(output)
!$
!$      Equation for zdot:
!$      zdot=-i/hbar*dhord(zic,zi)/dzic
!$      Equation for Sdot=L
!$      Sdot=L=ihbar/2.0sum_ndim((zc*zdot-zdotc*z)-Hord(zc,z))
!$*****************************************************************************!!
    Subroutine Derivs(t,bs,dbsdt)
      Implicit none
      type(csbasisfn),dimension(:),intent(in)::bs
      type(csbasisfn)::bf1cz,bf2, bf1
      type(csbasisfn),dimension(:),intent(inout)::dbsdt
      type(csbasisfn),dimension(:),allocatable::dbsdtc
      complex(kind=8),dimension(:),allocatable::ddot
      real(kind=8)::t
      integer::i,j
      !!!!!!!!!!!!!!!!!!!!!Z derivs below:!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      call INBASISFN(bf1cz)  ! assigns memory
      call INBASISFN(bf2)
      call ALCBASIS(dbsdtc)
      do i=1,nbf
         bf1=bs(i)                        ! assign the zi and zic
         bf1cz=bf1
         bf1cz%z(1:ndim)=dconjg(bf1%z(1:ndim))
         do j=1,ndim
            dbsdt(i)%z(j)=-(Img/hbar)*DH_ord_Dz1c(bf1cz,bf1,j)
            !DHord takes  in a j for a degree of freedom
         end do
      end do
      !!!!!!!!!!!!!!!!!!!!!S derivs below:!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      do i=1,nbf
         bf1=bs(i)
         bf1cz%z(1:ndim)=dconjg(bf1%z(1:ndim))
         dbsdt(i)%s=Img*0.5d0*sum(bf1cz%z(1:ndim)*dbsdt(i)%z(1:ndim)&
             -dconjg(dbsdt(i)%z(1:ndim))*bf1%z(1:ndim))
         dbsdt(i)%s=dbsdt(i)%s - H_ord(bf1cz,bf1)
      end do
!     !!!!!!!!!!!!!!!!!!!!!D derivs below:!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      allocate(ddot(nbf))
      call Amp2d(bs,ddot)
      dbsdt(1:nbf)%d=ddot(1:nbf)
      deallocate(ddot)
    End Subroutine Derivs
```

C.4 Amplitude Derivatives

The ‘Amp2d’ routine calculates the derivatives for the amplitudes, D. This routine is noticeably longer than the derivatives to calculate $\mathbf{Z}$ and S due to the complex of the equation for the derivatives of D, which uses external LAPACK routines.

```
!$*************************************************************************!!
!$                        Subroutine Amp2d
!$      This routine is not in the derivs routine for simplicity
!$      Arguments:
!$      bs- input-  bs(i)%d will be used
!$      ddot-output- which will go into dbsdt(i)%d
!$      external:
!$      need to call overlap matrix and  Hord (Delta2Hord)
!$      need to call LAPACK
!$      Equation solved in element form:
!$      Sum_i D_i^dot *e^i/hbar*S_i*Omega_ji
!$          = i/hbar*Sum_i d_i e^(i/hbar*S_i)*Omega_ji*D2Hordji
!$      In LAPACK I will solve
!$      ==> Omega*x=b
!$      where Omega is the overlap matrix
!$      x is the vector ddot*exp(i/hbar*Si)  - ddot *exp is a scalar mult
!$      b is the vector:
!$      b=OmegaD2*diexp(i/hbar*Si)
!$      Matrix vector multiplication
!$      where OmegaD2 is a matrix where the elements are constructed from
!$      a scalar multiplication of the ij elements
!$      diexp(i/hbar*si) is a vector where the element di is multiplied by
!$      exp(i/hbar*si) - also called vector a!
!$      After solved with LAPACK need to divide x by exp(i/hbar*Si) to get ddot
!$*************************************************************************!!
    Subroutine Amp2d(bs,ddot)
      Implicit none
      type(csbasisfn),dimension(:),intent(in)::bs
      type(csbasisfn)::bf1,bf2
      complex(kind=8),dimension(:),intent(out)::ddot
      complex(kind=8),dimension(:),allocatable::a
      complex(kind=8),dimension(:),allocatable::b
      complex(kind=8),dimension(:,:),allocatable::OMEGA
      complex(kind=8),dimension(:,:),allocatable::D2
      complex(kind=8),dimension(:,:),allocatable::OMD2
      complex(kind=8),dimension(nbf,nbf)::E ! new phase matrix
      integer::i,j
      ! variables for LAPACK
      integer, dimension(nbf) ::IPIV
      integer:: LDA,LDB, N, NRHS, INFO
      complex(kind=8),dimension(nbf,nbf)::Omegain
      complex(kind=8),dimension(nbf)::bout
      !INTERFACE
       include "./LAPACK/ZGESV.f90"
      !END INTERFACE
      allocate(OMEGA(nbf,nbf))
      allocate(OMD2(nbf,nbf))
      allocate(D2(nbf,nbf))
      allocate(a(nbf))
      allocate(b(nbf))
      call OVRLMAT(OMEGA,bs)
      CALL INBASISFN(bf1)
      CALL INBASISFN(bf2)
!!!!!!!!!!!!!!!!!!!!Right Hand Side!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      ! right hand side matrix
      do i=1,nbf
         bf2=bs(i)
         do j=1,nbf
            bf1=bs(j)
            IF ( i .eq. j) then
               D2(j,i)=cmplx(0.0d0, 0.0d0)
            ELSE IF (i .ne. j) then
                D2(j,i)=Delta2Hord(bf1,bf2)
            END IF
         end do
      end do
      do i=1,nbf
         do j=1,nbf
                IF (j .ne. i) then
                        E(j,i)=0.0d0
                Else if (j .eq. i) then
                        E(i,j)=exp(Img/hbar*(bs(i)%s))
                End IF
         end do
      end do
      do i=1,nbf
         do j=1,nbf
            OMD2(j,i)=OMEGA(j,i)*D2(j,i)
         end do
      end do
      ! right hand side vector a
      do j=1,nbf
        a(j)=bs(j)%d*E(j,j)
      end do
      ! right hand side vector b that goes into lapack
      b(:)=MATMUL(OMD2,a)
      b(:)=-1.0d0*(Img/hbar)*b(:)
      ! matrix equation now is : Omega * x=b , where x is
      !ddot*exp(i/hbar*si) and b is vector above- solve with LAPACK
 ! ****variables for LAPACK*******************************
      omegain(:,:)=OMEGA(:,:)
      LDA=nbf
      LDB=nbf
      NRHS=1
      N=nbf
      INFO=0
      IPIV=0
!*************************************************************
      CALL ZGESV(N, NRHS, omegain , LDA, IPIV, b , LDB, INFO )
      bout(:)=b(:)
      If (INFO .ne. 0) then
         Print *, 'WARNING INFO =', INFO
      End If
      ! ddot=b/exp(i/hbar*si)
      do j=1,nbf
        ddot(j)=bout(j)/E(j,j)
      end do
      b(:)=MATMUL(Omega,bout)! omegain gets changed by LAPACK
      deallocate(D2)
      deallocate(OMD2)
      deallocate(OMEGA)
      deallocate(b)
      deallocate(a)
      deallocate(bf1%z)
      deallocate(bf2%z)
    End Subroutine Amp2d
```

Bibliography

[1] P. Atkins and R. Friedman, *Molecular Quantum Mechanics*, Oxford University Press, 4th edn., 2005.

[2] C. L. Tang, *Fundamentals of Quantum Mechanics For Solid State Electronics and Optics*, Cambridge University Press, 2005.

[3] D. J. Tannor, *Introduction To Quantum Mechanics A Time-Dependent Perspective*, Sausilito, CA: University Science Books, 2006.

[4] E. Heller, *J. Chem. Phys.*, 1975, **62**, 1544.

[5] E. Heller, *Acc. Chem. Res.*, 1981, **14**, 368–375.

[6] W. Hase and E. J. Heller, *Advances In Classical Trajectory Methods*, JAI Press Inc, 1992.

[7] M. F. Herman and E. Kluk, *Chem. Phys.*, 1984, **91**, 27 – 34.

[8] E. Kluk, M. F. Herman and H. L. Davis, *J. Chem. Phys.*, 1986, **84**, 326–334.

[9] D. Shalashilin and M. Child, *J. Chem. Phys.*, 2001, **114**, 9296.

[10] F. Grossmann and M. Herman, *J. Phy. A: Math. Gen.*, 2002, **35**, 9489–9492.

[11] H.-D. Meyer, U. Manthe and L. Cederbaum, *Chem. Phys. Lett.*, 1990, **165**, 73 – 78.

[12] M. Beck and H. Meyer, *Z. Phys. D. Atom. Mol. Cl.*, 1997, **42**, 113–129.

[13] I. Burghardt, M. Nest and G. A. Worth, *J. Chem. Phys.*, 2003, **119**, 5364–5378.

[14] D. V. Shalashilin and I. Burghardt, *J. Chem. Phys.*, 2008, **129**, 084104.

[15] M. Ben-Nun and T. Martínez, *J. Chem. Phys.*, 1998, **108**, 7244.

[16] T. Martinez and R. Levine, *J. Chem. Phys.*, 1996, **105**, 6334.

[17] S. Yang, J. Coe, B. Kaduk and T. Martínez, *J. Chem. Phys.*, 2009, **130**, 134113.

[18] B. Levine, J. Coe, A. Virshup and T. Martínez, *Chem. Phys.*, 2008, **347**, 3–16.

[19] S. K. Reed, D. R. Glowacki and D. Shalashilin, *Submitted to Chem. Phys.*

[20] M. Schröder and A. Brown, *J. Chem. Phys.*, 2009, **131**, 034101.

[21] L. J. Doriol, F. Gatti, C. Iung and H.-D. Meyer, *J. Chem. Phys.*, 2008, **129**, 224109.

[22] I. Burghardt, K. Giri and G. A. Worth, *J. Chem. Phys.*, 2008, **129**, 174104.

[23] G. A. Worth, H.-D. Meyer and L. S. Cederbaum, *J. Chem. Phys.*, 1998, **109**, 3518–3529.

[24] A. Raab, G. A. Worth, H.-D. Meyer and L. S. Cederbaum, *J. Chem. Phys.*, 1999, **110**, 936–946.

[25] S. Mahapatra, G. Worth, H. Meyer, L. Cederbaum and H. Koppel, *J. Phys. Chem. A*, 2001, **105**, 5567–5576.

[26] M. D. Coutinho-Neto, A. Viel and U. Manthe, *J. Chem. Phys.*, 2004, **121**, 9207–9210.

[27] F. Matzkies and U. Manthe, *J. Chem. Phys.*, 1999, **110**, 88–96.

[28] H. Köppel, M. Döscher, I. Bâldea, H.-D. Meyer and P. G. Szalay, *J. Chem. Phys.*, 2002, **117**, 2657–2671.

[29] D. V. Shalashilin and M. S. Child, *J. Chem. Phys.*, 2000, **113**, 10028–10036.

[30] P. A. Sherratt, D. V. Shalashillin and M. S. Child, *Chem. Phys.*, 2006, **322**, 127–134.

[31] Y. Wu and V. S. Batista, *J. Chem. Phys.*, 2004, **121**, 1676–1680.

[32] D. Glowacki, S. Reed, M. Pilling, D. Shalashilin and E. Martínez-Núñez, *Phys. Chem. Chem. Phys.*, 2009, **11**, 963–974.

[33] D. Shalashilin and M. Child, *J. Chem. Phys.*, 2004, **121**, 3563.

[34] D. V. Shalashilin and M. S. Child, *J. Chem. Phys.*, 2008, **128**, 054102.

[35] D. V. Shalashilin and M. S. Child, *J. Chem. Phys.*, 2001, **115**, 5367–5375.

[36] D. V. Shalashilin and M. S. Child, *J. Chem. Phys.*, 2003, **119**, 1961–1969.

[37] W. Louisell, *Quantum statistical properties of radiation*, John Wiley & Sons New York, 1973.

[38] S. L. Salas, E. Hille and G. J. Etgen, *Calculus One And several Variables*, John Wiley And Sons, INC, 9th edn., 2003.

[39] M. Metcalf and J. Reid, *Fortran 90/95 explained*, Oxford University Press, Inc. New York, NY, USA, 1999.

[40] R. Chandler and P. Northrop, *Free subroutine that is available from the website:*, http://www.homepages.ucl.ac.uk/ ucakarc/work/software/randgen.txt, 2009.

[41] S. K. Reed, *Private Communication*, Unpublished, 2009.

[42] E. M. Graefe and H. J. Korsch, *Phys. Rev. A.*, 2007, **76**, 032116.

[43] M. Albiez, R. Gati, J. Fölling, S. Hunsmann, M. Cristiani and M. K. Oberthaler, *Phys. Rev. Lett.*, 2005, **95**, 010402.

[44] W. Press, S. Teukolsky, W. Vetterling and B. Flannery, *Numerical recipes in FORTRAN: the art of scientific computing*, Cambridge Univ Pr, 1992.