QUEENS UNIVERSITY BELFAST

MASTERS THESIS

# Computational Methods For Ultrafast Quantum Physics

*Author:*
Luke ROANTREE

*Supervisors:*
Dr. Andrew BROWN
Dr. Jack WRAGG

*A thesis submitted in fulfillment of the requirements for the degree of Masters in Mathematics*

*in the*

RMT Group
CTAMOP

April 4, 2019

# Declaration of Authorship

I, Luke ROANTREE, declare that this thesis titled, "Computational Methods For Ultrafast Quantum Physics" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

*"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."*

Dave Barry

QUEENS UNIVERSITY BELFAST

# *Abstract*

School of Mathematics and Physics
CTAMOP

Masters in Mathematics

**Computational Methods For Ultrafast Quantum Physics**

by Luke ROANTREE

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

# *Acknowledgements*

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **LAH** | **L**ist **A**bbreviations **H**ere |
| **WSF** | **W**hat (it) **S**tands **F**or |

# Physical Constants

Speed of Light $c_0 = 2.997\,924\,58 \times 10^8\,\mathrm{m\,s^{-1}}$ (exact)

# List of Symbols

| | | |
|---|---|---|
| $a$ | distance | m |
| $P$ | power | W ($\mathrm{J\,s^{-1}}$) |
| $\omega$ | angular frequency | rad |

*For/Dedicated to/To my…*

# Chapter 1

# Background Theory

## 1.1 Overview of Relevent Quantum Theory

In this thesis, the interactions of laser pulses with quantum systems are investigated. The systems investigated are limited to Hydrogenic atoms; that is spherically-symmetric systems with a central attractive potential field; the Coulomb potential. Such systems allow several simplifications to calculations based on symmetry, while still remaining useful models of many real molecules, atoms, or sub-atomic particles. Before describing the mathematical models used in the investigation, some relevant quantum theory, numerical analysis, and computational methods are revisited.

### 1.1.1 Schrodinger Equation In Multiple Forms

The 'Schrodinger Equation' describes the wavefunction of a quantum system, and how that wavefunction changes dynamically with time *TODO - add source for this*. In it's most general, time-dependent, form the Schrodinger equation is written;

$$i\hbar\frac{d}{dt}\left|\Psi\left(t\right)\right\rangle = \hat{H}\left|\Psi\left(t\right)\right\rangle$$

(where $\hbar$ is Planck's constant, $\Psi$ is the wavefunction of the system, and $\hat{H}$ is the Hamiltonian of the system)
*Should I go into more detail on the Hamiltonian?*
To get a 'snapshot' of the wavefunction at a given instant, the time-independent Schrodinger equation (TISE) can be considered;

$$E\left|\Psi\right\rangle = \hat{H}\left|\Psi\right\rangle$$

(where $\Psi$ is the wavefunction of the system at the particular 't' of the snapshot, and $E$ is the energy of the system)
This equation can be solved as an eigenvalue problem, where the wavefunction of the system can be described in terms of eigenfunctions of the Hamiltonian - each with associated energy corresponding to the eigenfunctions' eigenvalues.

For a system with a single non-relativistic particle, under the influence of an external potential $V$, the TISE can be written in differential form;

$$\left[\frac{-\hbar^2}{2m}\nabla^2 + V\left(\mathbf{r}\right)\right]\Psi\left(\mathbf{r}\right) = E\Psi\left(\mathbf{r}\right)$$

The radial time-independent Schrodinger equation (R-TISE) is a simplification of the TISE for the case where the system is spherically symmetric, which is the case

for a Hydrogenic atom.Making this simplification allows the, generally 3D, TISE to be re-written in spherical coordinates and the two angular terms to be discounted (as the wavefunction has no dependence on the angluar directions). The result is a more easily solvable, 1D, partial differential equation;

$$\left[\frac{-\hbar^2}{2m}\frac{d^2}{dr^2} + V(r)\right]\Psi(r) = E\Psi(r)$$

### 1.1.2 Multi-Electron Systems

No idea what I was planning to do here.
Statement that Schrodinger Eqn still valid, and description of new Hamiltonian? Possibly also statement about if it is/isn't analytically solvable?

## 1.2 Overview Of Finite Difference Methods

### 1.2.1 Relation to Taylor Series

a lkjnasd lka a la lkjasd la lksad lak a

### 1.2.2 Comparison To Basis Expansion Methods

Splines:
Fourier Expansions:

### 1.2.3 Stability Considerations

Von Neumann relation Defn. (Stability): A F.D. scheme is 'stable' if the errors at a given timestep do not cause subsequent errors to be magnified. Graphs displaying unstable and stable solutions?

### 1.2.4 Backwards & Central Finite Difference Methods & Stiff Equations

look into that report I wrote last year

### 1.2.5 Calculation of Finite Difference Coefficients

Show without proof? include code/pseudo-code?

## 1.3 Parallel Computation Overview

### 1.3.1 Types Of Parallelism

There are three main types of parallelism [1];

- **Shared Memory Systems**: These are systems consisting of several processors that are all able to access a shared memory
- **Distributed Systems**: These are systems with several seperate 'units', each with a processor and individual memory, which connect to each other over a network

- **Graphic Processing Units**: These are used as co-processors, to perform highly parallelised numerical tasks given to them by a main processor

Additionally, there are three main approaches to implementing parallelism, each roughly corresponding to one of the above 'types' of parallelism [1];

- **Multi-Threading**: TODO
- **Message Passing**: TODO
- **Stream Based**: TODO

### 1.3.2 OpenMP & MPI

OpenMP is a leading software utility used for implementing multi-threaded programs via an exposed API accessible for . As mentioned above, multi-threaded programs are usually based around a shared memory system,... TODO: dangers (race conditions) TODO: basic usage of OpenMP, advantages (easy parallelisation of for loops) / disadvantages

MPI (Message Passing Interface) is another software utility, used for communication across a distributed system... TODO: dangers of MPI (find out) TODO: basic usage, advantages / disadvantages, differences from OpenMP

### 1.3.3 Parallelisationability Of F.D. Methods

As described in chapter (?), spatial finite difference methods can approximate solutions to problems over a specified range, with given boundary conditions at the end points. The solution at each point can then be projected through time with a time propagator. If a finite difference time propagator is used, the initially specified range can be split up into several sub-ranges, each of which can be propagated independently - although they will lose accuracy at the end points of the sub-ranges over time. This can be rectified with message passing to update end point values based on neighbouring sub-range's end points periodically

# Chapter 2

# Baseline Solution of Radial TISE for Hydrogenic Atom with a Soft-Core Potential

## 2.1 Derivation of Radial TISE approximation

This 'baseline solution' finds the first $k$ eigenstates of the Hamiltonian associated with an elecron interacting with a Hydrogenic atom at an instant, with the Coulomb Potential being modelled instead by a Soft-Core potential - which at all but very small distances is an accurate approximation of the Coulomb Potential. Due to the Coulomb Potential tending towards $\infty$ as $r \to 0$ it is difficult to model it numerically, but with a Soft-Core Potential approximation, we avoid the issues at small $r$s yet obtain accurate results for everywhere other than at the center.

Is this an appropriate place for the above?

The Radial TISE is a simplification of the TISE for the case where the system is spherically symmetric, which is the case for a Hydrogenic atom - for both a Coulomb Potential or a Soft-Core Potential.

### 2.1.1 Derivation of Radial Equation from Time Independent Scrodinger Equation

lksdlkjasd kalsjdnf alksjd fakjn a lkjnasd lka a la lkjasd la lksad lak a as akjs ka lka alk lllasd l asdkjfnaslkdjfal lkajsnd a;kjndfajsnasdk aoa assdf as kjdnfasd

### 2.1.2 Derivation of Matrix Central Finite Difference Method (MCFD) from Taylor Series

Taylor expansion:

### 2.1.3 Useful Properties of MCFD Method

- Sparse:
  Krylov Subspace based Arnoldi methods are known to be extremely efficient at finding eigenvectors of sparse matrices

- Hermitian:
  asdfabsdflkajsnkja nds kljand akjs aksj lkajs nalkjs nalkjn lka nalk alk na ksjna knja kjna ;kjna;j na jn;jndav k;aj n

## 2.2 Comparison Between Soft-Core Potential and Coulomb Potential

* Move out of here * a lkjnasd lka a la lkjasd la lksad lak a as akjs ka lka alk lllasd l asdkjfnaslkdjfal lkajsnd a;kjndfajsnasdk aoa assdf as kjdnfasd lsdkfgdlkfmsdlfgm lk oijfoi laskdmaksd kadfoaskfv aksd

## 2.3 Implementation

### 2.3.1 Propagator Method and Limitations

### 2.3.2 Matrix Method and Benefits

### 2.3.3 Solving the Matrix Method

* Make sure this includes at least an outline of krylov subspace iteration and preconditioning

## 2.4 Results

### 2.4.1 Results with no Potential (Particle In A Box)

### 2.4.2 Results for Hydrogenic Atom with A Soft-Core Potential

## 2.5 Optimisations

### 2.5.1 Sparse Storage

The first improvement to the chosen implementation (matrix method) involved moving from storing the Hamiltonian in a standard storage format, in this case a numpy ndarray, to a more suitable data structure; a sparse matrix.

The motivation for this is that to obtain a solution to a very high accuracy a large number of grid points are needed and, with ndarray storage, the memory needed to store the Hamiltonian scales with the square of the number of grid points - meaning that the maximum possible accuracy is limited by the size of the Hamiltonian that can be held in memory.

From <ref earlier>, we see that the Hamiltonian is tri-diagonal for our 3 point CFD approach. As a result we know everything about the Hamiltonian if we know its tri-diagonal elements, and so switching to a sparse diagonal storage data structure (scipy's sparse.diags) we can throw away the unneeded zeros and obtain a more efficient Hamiltonian representation that scales linearly with the number of grid points.

Switching to this data structure expanded the maximum size of Hamiltonian that could be held in memory on my laptop (8GB RAM) from one using <original> grid points to one using over 10 million grid points.

Limitations of this data structure include that more specialised functions are needed to operate on sparse matrices; for example numpy's linalg.eig eigensolver is unable to operate on sparse data. Luckily, scipy's sparse library has a range of equivalent tools designed to replicate the effects of many of numpy's operations for sparse data structure, including several eigensolvers which we now look at as an additional optimisation method.

### 2.5.2 Eigensolver Choice

As seen in the above section, a sparse storage system allows for more accurate calculations to be performed. There are two main sparse eigensolvers available to use in place of numpy's linalg.eig here; scipy's sparse.linalg.eigs and sparse.linalg.eigsh. These both work similarly, using Arnoldi-based methods for subspace iteration to obtain the eigenpairs, with the difference being that the former uses standard Arnoldi iteration, while the latter uses Lanczos iteration * CITE *.

Lanczos iteration is a simplification of Arnoldi iteration for the case where the matrix in question is Hermitian; from * cite previous *, we know that a CFD-based Hamiltonian will be real and symmetric - and therefore Hermitian.

For a model TISE solution using 10k grid points, the relative speedup of using eigsh over eigs was (* TEST *)%

Additional optimisations involving this eigensolver are possible, including the optional parameter 'k' which allows the number of desired eigenstates to be specified. By default, the 'k' most dominant eigenstates will be returned (dominant implying largest eigenvalue magnitude).

As we only care about bound states, with eigenvalues below zero, we don't necessarily want the 'k' most dominant eigenstates; rather we want the 'k' most dominant eigenstates *corresponding to negative eigenvalues*.

Another optional parameter, 'sigma', can be used to achieve this. 'sigma' allows a 'target' eigenvalue value to be given and, through performing shift-invert preconditioning (* cite my L3? *), the 'k' eigenstates closest to that value be returned.

By passing in a suitably large negative number, we ensure that the returned eigenstates will correspond to the ground state and 'k-1'-first excited states. This allows a massive reduction in computational effort needed to obtain particular bound states as, without these, all eigenstates would need to be calculated and then sorted.

Additional optimisations involving the 'sigma' parameter are described below, which lead to further large reductions in computational expense.

### 2.5.3 FastBOI Improvement

An additional efficiency improvement that I developed makes use of results from (* cite previous *), where it is seen that the rate of convergence of an eigenstate during subspace iteration is proportional to the magnitude of the corresponding eigenvalue and that eigenstates of of a matrix are also eigenstates of that matrix's inverse (if it exists), with corresponding eigenvalues equal to the reciprocal of the original eigenvalues.

Combining these it can be seen that if an estimate of an eigenstate's eigenvalue is known, say $\lambda$, then $\lambda I$ can be subtracted from the original matrix to create a new matrix with the same eigenstates, but with eigenvalues equal to the original ones minus $\lambda$.

As a result, the eigenstate desired will have a 'new' eigenvalue of 0 (or very near, depending on accuracy of the estimate) - thus, if we invert the matrix, the result will have the same eigenvectors but reciprocal eigenvalues, which in the case of the desired eigenstate will be very large and tend towards $\infty$ with increasing accuracy.

Using the knowledge that the rate of convergence is proportional to eigenvalue magnitude, this shifted and inverted matrix will converge extremely rapidly to the desired eigenstate during subspace iteration, with increased speed if the accuracy to which the original estimate is known is increased.

Based on this result, I developed a 'Fast Boosted Optimiser Iteratiion' method to improve convergence rates for the eigensolver used to find bound states from my Hamiltonian. Another result used in the development of this approach is that a model with fewer grid points than another should still produce relatively correct eigenstates and eigenvalues, simply to a lower accuracy than a model with more grid points.

Based on these results, the general idea of this approach is to find the ground state eigenvalue for a model with a greatly reduced number of grid points, and then use this as an estimate in the 'sigma' parameter of the eigensolver for the model with full number of grid points - greatly increasing the convergence rate for the full model at the cost of having to solve an extra, much smaller, problem.

Additionally, although only useful for models with a very large number of grid points, this approach can be stacked; a small model can generate an estimate ground state eigenvalue for a medium sized system, which can then generate a better estimate for the full very large system. Pseudo-code for the basic approach is described below, followed by adaptions to allow several 'boosting' stages.

Basic FastBOI algorithm:
definition of variables:

- **x_full:** The grid points for the full accuracy model
- **x_red:** The grid points for the reduced accuracy model, it will have the same start- and end-points as x_full
- **boosting_factor:** The relative size of x_full to x_red

```
function basic_fastboi(x_full, boosting_factor){
  start, end, size = x_full[0], x_full[last], SIZE(x_full);
  inc = (end - start) / (size / boosting_factor);
  x_red = ARRAY();

  for i = 0..(size / boosting_factor){
    x_red[i] = start + i*inc;
  }

  H_red = generate_hamiltonian(x_red);
  est_GS_eigval, est_GS_eigvec = eigensolver(H_red, sigma = -B);

  /* where B is some arbitrarily large number such that B
     is definitely less than the real GS eigenvalue.
     Note: matching the output of ARPACK and LAPACK eigensolver outputs,
     only est_GS_eigval needed
  */

  H_full = generate_hamiltonian(x_full);
  acc_GS_eigval, acc_GS_eigvec = eigensolver(H_full, sigma = est_GS_eigval);

  return (acc_GS_eigval, acc_GS_eigvec);
}
```

As mentioned, this 'boosted optimiser' approach can be stacked iteratively allowing a more accurate estimate to be generated for the full accuracy model, at less than the cost of solving a medium size model with an arbitrarily large negative number that would otherwise be needed. The adaption to the above algorthm to allow several boosting stages is described as follows, using a recursive approach;

```
function fastboi(x_full, boosting_factor, num_boosts, estimate=-B){
  start, end, size = x_full[0], x_full[last], SIZE(x_full);
```

```
  inc = (end - start) / (size / (boosting_factor^num_boosts));
  x_red = ARRAY();

  for i = 0..(size / boosting_factor){
    x_red[i] = start + i*inc;
  }

  H_red = generate_hamiltonian(x_red);
  est_GS_eigval, est_GS_eigvec = eigensolver(H_red, sigma=estimate);

  if (num_boosts > 0){
    return fastboi(x_full, boosting_factor, num_boosts-1, est_GS_eigval);
  }
  else{
    H_full = generate_hamiltonian(x_full);
    acc_GS_eigval, acc_GS_eigvec = eigensolver(H_full, sigma = est_GS_eigval);
    return (acc_GS_eigval, acc_GS_eigvec);
  }
}
```

### 2.5.4 Pre-Trained Predictive Model Improvement

# Chapter 3

# TDSE Solution & Optimisation

## 3.1 Baseline: Analytic Solution for Field-Free Propagation

This 'baseline solution' finds the first $k$ eigenstates of the Hamiltonian associated with an elecron interacting with a Hydrogenic atom at an instant, with the Coulomb Potential being modelled instead by a Soft-Core potential - which at all but very small distances is an accurate approximation of the Coulomb Potential. Due to the Coulomb Potential tending towards $\infty$ as $r \to 0$ it is difficult to model it numerically, but with a Soft-Core Potential approximation, we avoid the issues at small $r$s yet obtain accurate results for everywhere other than at the center.

Is this an appropriate place for the above?

The Radial TISE is a simplification of the TISE for the case where the system is spherically symmetric, which is the case for a Hydrogenic atom - for both a Coulomb Potential or a Soft-Core Potential.

## 3.2 Time Propagators

### 3.2.1 Forward Finite Difference Propagator

lksdlkjasd kalsjdnf alksjd fakjn a lkjnasd lka a la lkjasd la lksad lak a as akjs ka lka alk lllasd l asdkjfnaslkdjfal lkajsnd a;kjndfajsnasdk aoa assdf as kjdnfasd

### 3.2.2 Krylov Subspace Propagator

- Sparse:
  Krylov Subspace based Arnoldi methods are known to be extremely efficient at finding eigenvectors of sparse matrices

- Hermitian:
  asdfabsdflkajsnkja nds kljand akjs aksj lkajs nalkjs nalkjn lka nalk alk na ksjna knja kjna ;kjna;j na jn;jndav k;aj n

### 3.2.3 Propagator Comparison

dfgadfg sdfga adfga adf ads as

## 3.3 Parallelisation of Propagators

### 3.3.1 Using Shared Memory

dafgadfadg sdfga adfg

### 3.3.2 Fully Independent Threads

dsfgasdf

## 3.4 Comparison Propagator Efficiencies

### 3.4.1 On A Laptop

dfgadg kjads akja akjn aknkj kjnk;jn akjn akjn

### 3.4.2 On A Server Cluster

a lkjnasd lka a la lkjasd la lksad lak a as akjs ka lka alk lllasd l asdkjfnaslkdjfal lkajsnd a;kjndfajsnasdk aoa assdf as kjdnfasd lsdkfgdlkfmsdlfgm lk oijfoi laskdmaksd kadfoaskfv aksd

# Chapter 4

# TDSE Solutions for Time-Dependent Potentials

## 4.1 DC Drift

### 4.1.1 Potential Description

adfgadfga ag a ads a a sd fasd tr df

### 4.1.2 Effet On State Evolution

Is this an appropriate place for the above?

The Radial TISE is a simplification of the TISE for the case where the system is spherically symmetric, which is the case for a Hydrogenic atom - for both a Coulomb Potential or a Soft-Core Potential.

## 4.2 Simulated 'Laser Pulse' Potential

### 4.2.1 Potential Description

lksdlkjasd kalsjdnf alksjd fakjn a lkjnasd lka a la lkjasd la lksad lak a as akjs ka lka alk lllasd l

### 4.2.2 Effect On State Evolution

asdkjfnaslkdjfal lkajsnd a;kjndfajsnasdk aoa assdf as kjdnfasd

## 4.3 TODO: model electron transitions & photon emmision?

asdjfb kjad kjnas kjna skljn as k as as ldlkn

# Chapter 5

# RMT

## 5.1 R-Matrix Theory

### 5.1.1 TODO 1

df h jd nd re kfd nljd da nk kjnds Is this an appropriate place for the above?

### 5.1.2 TODO 2

The Radial TISE is a simplification of the TISE for the case where the system is spherically symmetric, which is the case for a Hydrogenic atom - for both a Coulomb Potential or a Soft-Core Potential.

## 5.2 Overview Of Project

### 5.2.1 TODO 3

lksdlkjasd kalsjdnf alksjd fakjn a lkjnasd lka a la lkjasd la lksad lak a as akjs ka lka alk lllasd l

### 5.2.2 TODO 4

asdkjfnaslkdjfal lkajsnd a;kjndfajsnasdk aoa assdf as kjdnfasd

## 5.3 Optimisation Of **TODO**

### 5.3.1 Overview Of Potential Inefficiency

sdf gd jd kns doijsa o;noire jknf jndf

### 5.3.2 Method Of Attempted Improvement

dsf lkdf lkndf lksn lkna lk dsf gskfdn sdf dsf kn

### 5.3.3 Analysis Of Effectiveness & Usefulness Of Changes

dsfg kjd klds lk sd klndf lk s sd dfg sdfgdsfgsd

### 5.3.4 Insights

## 5.4 Optimisation Of **TODO**

### 5.4.1 Overview Of Potential Inefficiency

sdf gd jd kns doijsa o;noire jknf jndf

### 5.4.2 Method Of Attempted Improvement

dsf lkdf lkndf lksn lkna lk dsf gskfdn sdf dsf kn

### 5.4.3 Analysis Of Effectiveness & Usefulness Of Changes

dsfg kjd klds lk sd klndf lk s sd dfg sdfgdsfgsd

### 5.4.4 Insights

## 5.5 Mid-Region Investigation To Improve Parallelisation

### 5.5.1 Communication Costs For Parallelisation In Current Inner/Outer Region Model

sd fkjs aksn ;ksdn ;aksd sakld dsa sd d df df fds

### 5.5.2 Conjecture For How Addition Of Mid-Region Could Improve Speeds

sd gfjknds lkjsdf nlkjdf nskjdf nsdkfj

### 5.5.3 Description Of Method Used In Attempt

df gkl slkd flkdnf dsf df kjdnf

### 5.5.4 Analysis Of Speed Improvements In Different Use Cases

dsf jdflkm ldskfmlkfd sdflkn

### 5.5.5 Review & Insights Of Addition Of Mid-Region

sd kjns dlkjnds skdj ns sdak nsd

# Appendix A

# Frequently Asked Questions

## A.1 How do I change the colors of links?

The color of links can be changed to your liking using:

`\hypersetup{urlcolor=red}`, or
`\hypersetup{citecolor=green}`, or
`\hypersetup{allcolor=blue}`.

If you want to completely hide the links, you can use:

`\hypersetup{allcolors=.}`, or even better:
`\hypersetup{hidelinks}`.

If you want to have obvious links in the PDF but not the printed text, use:

`\hypersetup{colorlinks=false}`.

# Bibliography

[1] Roman Trobec Boštjan Slivnik Patricio Bulić Borut Robič. *Introduction To Parallel Computing*. Springer Nature Switzerland, 2018. URL: https://doi.org/10.1007/978-3-319-98833-7.