State-dependent forces in cold quantum gases

Christopher Billington

Submitted in total fulfilment of the requirements
of the degree of Doctor of Philosophy

Supervisory committee:
Prof Kristian Helmerson
Dr Lincoln Turner
Dr Russell Anderson

School of Physics and Astronomy
Monash University

August, 2016

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

This page intentionally left blank

Contents

rev: 28 (9bcc7631183f)
author: Chris Billington <chrisjbillington@gmail.com>
date: Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom

This page intentionally left blank

rev: 28 (9bcc7631183f)
author: Chris Billington <chrisjbillington@gmail.com>
date: Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom

Chapter 3

Quantum mechanics on a computer

This chapter comprises a summary of some of the methods used by cold atom physicists to compute numerical results pertaining to cold atom systems. Many a problem in quantum mechanics is not analytically solvable, especially when the real world of experimental physics rears its ugly head, violating theorists' assumptions of simplicity left and right. In particular, atomic physics experiments are time-dependent, with each run of an experiment generally proceeding in stages. Lasers may turn on and off, magnetic fields may vary in magnitude and direction, RF pulses may be chirped [CITE LISA?] to reliably induce particular atomic transitions. Much of the numerical calculations performed by researchers in cold atom physics groups such as ours are accordingly of the time-dependent variety, and are fairly literal simulations of specific experiments that may be carried out in the lab.

3.1 From the abstract to the concrete: neglect, discretisation and representation

To numerically simulate a quantum mechanical system, one must evolve a state vector in time according to the Schrödinger equation:

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t} |\psi(t)\rangle = \hat{H}(t) |\psi(t)\rangle \tag{3.1}$$

To do this on a computer, one must first decide which degrees of freedom are to be simulated. We necessarily neglect many degrees of freedom as a matter of course; which ones can be neglected is warranted by the specific situation and we do it so often we barely notice. For example, simulating a single component Bose–Einstein condensate entails neglecting the internal degrees of freedom of the atoms—as well as reducing the atom-light interaction to a simple potential such as an optical dipole trap or magnetic dipole interaction (neglecting the quantum degrees of freedom in the electromagnetic field). We may ignore one or more spatial degrees of freedom as well, say, if we are simulating an experiment in which the condensate is confined to two dimensions by way of a tight trapping potential in one direction [CITE AN EXAMPLE]. Or, when simulating laser cooling [SEE SECTION TODO], we may care very much about the electronic state of the atom, but treat its motional state classically. In these cases we are essentially imposing the assumption that the system will only occupy one state with respect to those degrees of freedom ignored (the condensate will remain in lowest excitation level in the direction of the tight trap, the atoms will remain in one specific Zeeman sublevel), or we are assuming those degrees of freedom can be treated classically (the electromagnetic field is well described by classical electromagnetism, the atoms' motional state is described

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

well by Newtonian mechanics). Which degrees of freedom can be neglected and which cannot requires knowledge of the situation at hand, often informed by best-practices of the research community in question and ultimately justified by experiment.[1]

Once the degrees of freedom are known, one must decide on a basis in which to represent them concretely. The basis often cannot be complete, since for many degrees of freedom this would require an infinite number of basis states—for example the electronic state of an atom contains a countably infinite number of states, and a spatial wavefunction in free space has an uncountable number of states (one for each position in R3 [TODO BLACKBOARD BOLD]). For the internal state of an atom, therefore, we restrict ourselves to only the states we expect can become non-negligibly occupied, given the initial conditions and transitions involved. For example, at low temperature we can expect atoms to be almost completely in their electronic ground states, since energy gaps between ground and excited states are large compared to the atoms' thermal energy.[2]. We need only include the small number of excited states that might become occupied as a result of optical transitions present in the situation being simulated. This can still be a large number of states if one is studying Rydberg atoms [CITE] or using ultrafast (and therefore broad-band) laser pulses, but is otherwise fairly small. For example, including both the D1 and D2 lines of Rubidium 87, with all hyperfine levels and Zeeman sublevels gives 32 states (see section [TODO]).

For spatial degrees of freedom, one can either discretise space on a grid, or use a set of orthogonal basis functions[3].

Once the degrees of freedom and basis vectors have been chosen, the state vector is then represented on a computer as an array of complex numbers, giving the coefficients of each basis vector required to represent a particular state vector. Matrix elements of the Hamiltonian in the same basis must be calculated, and the Schrödinger equation can then be written:

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t} \langle n|\psi(t)\rangle = \sum_m \langle n|\hat{H}(t)|m\rangle \langle m|\psi(t)\rangle\,, \tag{3.2}$$

or in standard matrix/vector notation (without Dirac notation):

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t} \psi_n(t) = \sum_m H_{nm}(t)\psi_m(t), \tag{3.3}$$

where $\psi_n(t) = \langle n|\psi(t)\rangle$ and $H_{nm}(t) = \langle n|\hat{H}(t)|m\rangle$. This is now something very concrete that can be typed into a computer. Programming languages generally don't know about Dirac kets and operators, and so everything that is to be computed must be translated into matrices and vectors in specific bases. This may seem so obvious as to not be worth mentioning, but was nonetheless a stumbling block in my own experience of getting to grips with quantum mechanics. Once realising that every operator has a matrix representation in some basis, at least in principle, and that every ket is just a list of vector components in some basis, similarly at least in principle, expressions dense in bras and kets become much more concrete as the reader has a feel for exactly how they would type it into a computer. Without this realisation, doing quantum mechanics on paper can seem like an exercise in abstract mumbo-jumbo.

3.2 Solution to the Schrödinger equation by direct exponentiation

As an example of something seemingly abstract being more concrete than first appearances, it is sometimes said that the 'formal' solution to the Schrödinger equation (3.1) is:

$$|\psi(t_f)\rangle = e^{-\frac{i}{\hbar}\int_t^{t_f} \hat{H}(t')\,\mathrm{d}t'} |\psi(t)\rangle\,. \tag{3.4}$$

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

[1] A classic example in the cold atom community of neglected degrees of freedom leading to *disagreement* with experiment is the discovery of polarisation gradient cooling (PGC) [CITE THE LIGHT SHEET EXPERIMENT], the explanation for which requires consideration of Zeeman sublevels of the atoms. The experiment that discovered PGC was designed to measure the effect of Doppler cooling, which does not involve Zeeman sublevels, and it was not until afterwards that theorists determined [CITE DALIBARD ETC] that transitions between Zeeman sublevels cannot be neglected and indeed are crucial in explaining the lower than predicted temperatures observed.

[2] The D-line of rubidium 87 has an energy gap of [TODO], requiring a temperature of [TODO] or higher in order for the Boltzmann factor $e^{\frac{\Delta E}{kT}}$ describing the excited state population to be greater than 1×10^{-6}.

[3] Strictly speaking, these can be the same thing—discretising space on a grid is declaring one's basis functions to be a set of functions, one for each gridpoint, each of which is zero at all gridpoints .

Saying that this is the 'formal' solution rather than just 'the solution' is presumably intended to emphasise that the arithmetic operations involved in (3.4) might not make immediate sense for the types of mathematical objects they are operating on, and that we have to be careful in defining the operations such that they produce a result that is not only sensible, but also the solution to the Schrödinger equation. If both $\hat{H}(t)$ and $|\psi(t)\rangle$ were single-valued functions of time rather than an operator valued function of time (the values at different times of which don't necessarily commute) and a vector valued function of time, then we would have no problem. However, (3.4) as written with operators and vectors is ambiguous, and we need to elaborate on it in order to ensure it is correct. I will come back to this after considering a simpler case.

If the Hamiltonian is time-independent, then (3.4) reduces to

$$|\psi(t_f)\rangle = e^{-\frac{i}{\hbar}(t_f-t)\hat{H}}\,|\psi(t)\rangle\,. \tag{3.5}$$

Given the matrix representation H of $\hat{H}$ and vector representation $\psi(t)$ of $|\psi(t)\rangle$ in a particular basis, this can now be directly typed into a computer as the matrix multiplication:

$$\psi(t_f) = U(t, t_f)\psi(t), \tag{3.6}$$

where

$$U(t, t_f) = e^{-\frac{i}{\hbar}(t_f-t)H} \tag{3.7}$$

is the unitary evolution operator for time evolution from the initial time t to time t_f, and is computed using a matrix exponential of $-\frac{i}{\hbar}(t_f - t)H$. Exponentiation of matrices is defined via the Taylor series of the exponential,

$$e^A = \sum_{n=0}^{\infty} \frac{A^n}{n!}, \tag{3.8}$$

which reduces exponentiation to the known operations of matrix multiplication and addition. However, any linear algebra programming library worth the bytes it occupies will have a matrix exponentiation function that should be used instead, as there are other methods of computing matrix exponentials that are more computationally efficient and numerically stable, such as the Padé approximant [CITE].

3.2.1 Matrix exponentiation by diagonalisation

Regardless of which method is used, matrix exponentiation is computationally expensive. It can be sped up however if a diagonalisation of H is known, since if

$$H = UDU^\dagger, \tag{3.9}$$

where D is a diagonal matrix and U is a unitary matrix[4], then

$$e^{-\frac{i}{\hbar}(t_f-t)H} = Ue^{-\frac{i}{\hbar}(t_f-t)D}U^\dagger. \tag{3.10}$$

This is simple to evaluate because the exponentiation of a diagonal matrix can be performed by exponentiating each diagonal matrix element individually[5]

Even if a diagonalisation of H is not analytically known, numerically diagonalising H (using a linear algebra library function or otherwise) can form the basis for writing your own matrix exponentiation function, if needed. I found this necessary for efficiently exponentiating an array of matrices in Python, since the `scipy` and `numpy` scientific and numeric libraries at the present time lack matrix exponentiation functions that can act on arrays. Writing a `for` loop in an interpreted language such as Python to exponentiate the matrices individually in many cases is unacceptably slow, so for these cases[6] I use a function such as the below:

[4] Note that the diagonals of D are the eigenvalues of H, and the columns of U are its eigenvectors.

[5] The reason for this is clear from the Taylor series definition of matrix exponentiation, since matrix multiplication and addition can both be performed elementwise for diagonal matrices.

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

```python
import numpy as np
from numpy.linalg import eigh

def expmh(M):
    """compute exp(M), where M, shape (..., N, N) is an array of N by N
    Hermitian matrices, using the diagonalisation method. Made this function
    because scipy's expm can't take an array of matrices as input, it can only
    do one at a time."""

    # Diagonalise the matrices:
    evals, evecs = eigh(M)

    # Now we compute exp(M) = U exp(D) U^\dagger where U is the matrix of
    # eigenvectors (as columns) and D is the diagonal matrix of eigenvalues:

    U = evecs
    U_dagger = U.conj().swapaxes(-1, -2) # Only transpose the matrix dimensions
    exp_D_diags = np.exp(evals)

    # Compute the 3-term matrix product U*exp_D_diags*U_dagger using the
    # einsum function in order to specify which array axes of each array to
    # sum over:
    return np.einsum('...ik,...k,...kj->...ij', U, exp_D_diags, U_dagger)
```

This concludes all I have to say for the moment on evolving systems with time-independent Hamiltonians using direct exponentiation. Sections [TODO - refer to operator product sections] will say a bit more about how to *approximately* exponentiate certain Hamiltonians, which is sometimes useful. In addition, it turns out that the time-dependent case can be reduced to the repeated application of evolution with time-independent Hamiltonians over small timesteps, as we will see in the next section.

3.2.2 Time-ordered exponentials and time-ordered products

As mentioned, the solution (3.4) is not the whole picture. It can only be taken at face value if the Hamiltonian at each moment in time commutes with itself at all other times [CITE SOMETHING]. In this case, once represented in a specific basis, the solution to the Schrödinger equation is again the matrix multiplication

$$\psi(t_f) = U(t, t_f)\psi(t), \tag{3.11}$$

with

$$U(t, t_f) = e^{-\frac{i}{\hbar}\int_t^{t_f} H(t')\,\mathrm{d}t'}. \tag{3.12}$$

Since matrix addition can be performed elementwise, so can the integral in the exponent, yielding a matrix which once exponentiated will give the evolution operator $U(t, t + \Delta t)$ for the solution to the Schrödinger equation. If the Hamiltonian at each moment in time does not commute with itself at all other times, however, then the unitary evolution operator for the solution to the Schrödinger equation is instead given by the following *time-ordered exponential* [CITE]:

$$U(t, t_f) = \mathcal{T}\left\{e^{-\frac{i}{\hbar}\int_t^{t_f} H(t')\,\mathrm{d}t'}\right\}. \tag{3.13}$$

In this expression, $\mathcal{T}$ denotes the *time-ordering operator*. The time ordering operator reorders terms within products that contain a time parameter (for us, the time parameter is the argument of the matrix H), such that the value of the time parameter is smallest on the rightmost term, largest on the leftmost term, and monotonically increasing right-to-left in between. For example:

$$\mathcal{T}\{H(4)H(1)H(2)H(5)H(3)\} = H(5)H(4)H(3)H(2)H(1). \tag{3.14}$$

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

Despite appearances, this time-ordered exponential is perfectly concretely defined via the definitions of all the operations involved that we have described so far, and can—with some effort—be typed into a computer and evaluated directly. Even though this is not how I evaluate time-ordered exponentials in my simulations of atomic systems, I'll quickly elaborate on this just to emphasise the concreteness of all these operations.

"What products is $\mathcal{T}$ reordering?" you might ask, as (3.13) doesn't appear to contain any products of $H(t)$. On the contrary, it does, since exponentiation is defined by its Taylor series, and so

$$U(t,t_f) = 1 + \mathcal{T}\left\{\sum_{n=1}^{\infty}\frac{1}{n!}\left[-\frac{i}{\hbar}\int_t^{t_f} H(t')\,\mathrm{d}t'\right]^n\right\} \tag{3.15}$$

$$= 1 + \sum_{n=1}^{\infty}\frac{(-i)^n}{n!\,\hbar^n}\mathcal{T}\left\{\left[\int_t^{t_f} H(t')\,\mathrm{d}t'\right]^n\right\}. \tag{3.16}$$

Each term in this series contains the n^{th} power (and hence a product) of an integral of $H(t)$. The time ordering operator doesn't allow us to evaluate each term by computing the matrix integral once and then raising it to a power—to do so would violate time-ordering since each integral involves evaluating $H(t)$ at all times. Instead we have to write each product of integrals as the integral of a product:

$$U(t,t_f) = 1 + \sum_{n=1}^{\infty}\frac{(-i)^n}{n!\,\hbar^n}\int_t^{t_f}\cdots\int_t^{t_f}\mathcal{T}\left\{H(t_1')\cdots H(t_n')\right\}\mathrm{d}t_1'\cdots\mathrm{d}t_n', \tag{3.17}$$

from which we can see exactly which product of matrices the time ordering operator is acting on.

Now we are close to seeing one might evaluate $U(t,t_f)$ numerically by summing each term in the Taylor series up to some order set by the required accuracy. For the n^{th} term, one needs to evaluate an n-dimensional integral over n time coordinates, with each coordinate having the same limits of integration. This can be computed in the usual way an integral is numerically computed[7], with the minor change that each time the integrand is evaluated, the terms within it must be re-ordered to respect the required time-ordering. Alternatively, the integration region can be restricted to the region in which the terms are already time-ordered, and then the total integral inferred by symmetry, which gives:

$$U(t,t_f) = 1 + \sum_{n=1}^{\infty}\left(-\frac{i}{\hbar}\right)^n\int_t^{t_f}\cdots\int_t^{t_{n-1}'} H(t_1')\cdots H(t_n')\,\mathrm{d}t_1'\cdots\mathrm{d}t_n'. \tag{3.18}$$

This is now a perfectly concrete expression, with each term comprising an integral over an n-simplex[8] of a product of n matrices.

This expression for the unitary evolution operator is called the Dyson series [CITE]. It is not generally used in cold atom physics; I certainly have never evaluated it. However it does see use in high energy physics [CITE] for computing transition amplitudes between incoming and outgoing waves in scattering problems (in which U is called the S-matrix). In these problems, H is an interaction Hamiltonian containing terms for all particle interactions being considered. Accordingly, the integrand for the n^{th} term, being a product of n copies of H at different times, contains one term for each possible sequence of n particle interactions. The integral itself can be considered a sum of transition amplitudes over all possible times that each interaction could have occurred. Indeed, each term corresponds to a Feynman diagram with n nodes [CITE].

The Dyson series isn't really suited to the problems we face in cold atom physics. For one, the series must be truncated at some point, and the result won't be a U that is actually unitary.[9]. Also, we are usually interested in the intermediate states, not just the final state of a system as in a scattering problem.

[7] By sampling the integrand on a uniform grid, using a quadrature method, or Monte-Carlo integration [CITE], which is widely used for high dimensional integrals such as these.

[8] A simplex is the generalisation of a triangle to higher dimensions, i.e a 3-simplex is a tetrahedron.

[9] Although unitarity is not often a strict requirement - we also frequently solve [EQREF TDSE] directly with fourth order Runge–Kutta, which is also not unitary.

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

In any case, typically when solving the Schrödinger equation by exponentiation we use the following, alternate expression for a time-ordered exponential:

$$U(t, t_f) = \mathcal{T}\left\{e^{-\frac{i}{\hbar}\int_t^{t_f} H(t')\,\mathrm{d}t'}\right\} \tag{3.19}$$

$$= \lim_{N\to\infty} \mathcal{T}\left\{\prod_{i=0}^{N} e^{-\frac{i}{\hbar}H(t_i)\Delta t}\right\}, \tag{3.20}$$

where $\Delta t = (t - t_f)/N$ and $t_i = t + i\Delta t$. You can convince yourself that [EQREF INTEGRAL DEFINITION] is equivalent to this by replacing the integral in the exponent with a sum—as per the Riemann definition of an integral—and expanding the exponential according to its Taylor series. Expanding each exponential in [EQREF PRODUCT DEFINITION] as a Taylor Series and collecting terms with the same number of copies of H then reveals that the two Taylor series are identical.

In any case, [EQREF PRODUCT DEFINITION] paints an intuitive picture of solving the Schrödinger equation: one evolves the initial state vector in time by evolving it according to constant Hamiltonians repeatedly over small time intervals. This has the desirable property that all intermediate state vectors are computed at the intermediate steps, meaning one can study the dynamics of the system and not just obtain the final state. This is of course useful for comparison with experiments, plenty of which involve time-dependent data acquisition and not just post-mortem analysis of some evolution.

Numerically, we can't actually take $N \to \infty$, or equivalently $\Delta t \to 0$, and so we instead choose a Δt smaller than the timescale of any time-dependence of H, and step through time using [EQREF].

Thus the case of a time-dependent Hamiltonian reduces to repeated application of the solution [EQREF] for a time-independent Hamiltonian. Furthermore, if the Hamiltonian is analytically diagonalisable at each moment in time, each of these steps can be taken with the diagonalisation method of matrix exponentiation. If it is not, and the system is small, then resorting to numerical diagonalisation may be acceptable. Otherwise, if the Hamiltonian can be decomposed into a *sum* of terms which are individually diagonalisable, the split-step method can be used (see sec [SECREF]). Failing that, a general purpose integration method such as fourth-order Runge-Kutta may have to be resorted to.[10] I will elaborate on these comments with examples in the following sections.

[10] Though if I admit it, RKR4 is usually what I reach for first.

- absorbing boundary conditions, reflective boundary conditions, periodic boundary conditions

3.3 Continuous degrees of freedom

The single-particle, non-relativistic, scalar Schrödinger wave equation, as distinct from the general Schrödinger equation [EQREF], is:

[EQUATION].

As mentioned in [SECREF], the equation for the single-particle wavefunction of an atom in a single-component Bose–Einstein condensate is the Gross–Pitaevskii equation

[EQUATION],

where $\psi(\boldsymbol{r}) = \sqrt{N}\,\langle \boldsymbol{r}|\psi\rangle$ is the single-particle wavefunction scaled by the square root number of atoms N.

Both these equations are partial differential equations involving both spatial and temporal derivatives. But in quantum mechanics all state vectors can be mapped to column vectors and all operators to matrices. Spatial wavefunctions are no exception to the former and differential operators such as ∇^2 are no exception to the latter. So what do these vectors and operators look like? That depends on whether we choose to

rev: 28 (9bcc7631183f)
author: Chris Billington <chrisjbillington@gmail.com>
date: Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom

discretise space on a grid, or use a functional basis (and on which functional basis we choose). As we'll see below, however, spatial discretisation is actually just a particular choice of functional basis, namely the Fourier basis.

3.3.1 Spatial discretisation

Imagine a two dimensional spatial region within which we are solving the single-component Gross–Pitaevskii equation, evolving an initial condensate wavefunction in time. Having specified which degrees of freedom we want to simulate (two continuous degrees of freedom, one for each spatial dimension), the next step according to the method outlined in [SECREF ABOVE] is to choose a basis in which to represent this state vector.

Lets say we discretise space in an equally-spaced $N_x \times N_y = 5 \times 5$ rectangular grid,[11] with spacing Δx, and only represent the wavefunction at those 25 points in space. The state vector can then be represented by a list of 25 complex numbers, each taken to be the wavefunction's value at the spatial position corresponding to one gridpoint. This 25-vector is now a concrete representation of our state vector.

[11] For the sake of example—256 × 256 is a more realistic minimum.

[FIGURE SHOWING UNWRAPPING OF 2D REGION INTO COLUMN VECTOR. PERHAPS 2D REGION IS CONTINUOUS WITH A GRID SUPERIMPOSED ON IT, AND LINES JOIN EACH GRIDPOINT TO THE VECTOR AT RIGHT]

But at what point did we choose a basis just now—what are the basis vectors? This just looks like discretising space at a certain resolution, rather than the formal process of choosing a basis and projecting the state vector and operators onto each basis vector, as outlined in [SECREF ABOVE]. Assuming what we've done is equivalent to choosing a basis, that basis has a finite number (25) of basis vectors, which means it cannot be complete, since state vectors we're approximately representing with it require an infinite number of complex numbers to be described exactly.[12] So what do the basis functions look like, and what state vectors have we implicitly excluded from simulation by choosing a basis that is incomplete?

[12] One for each position within the two dimensional space we're representing.

As a sidenote, spatial wavefunctions are often described as the representation of wavefuctions in the "spatial basis"—a basis in which the basis vectors $\{|\boldsymbol{r}\rangle\}$ are Dirac-deltas centred on each point in space [CITE DIRAC'S BOOK MAYBE]. The wavefunction $\psi(\boldsymbol{r})$ is then simply a coefficient saying how much of the basis vector $|\boldsymbol{r}\rangle$ (the spatial representation of which is a Dirac delta centred on the position $\boldsymbol{r}$) to include in the overall state vector. What we have *not* done is chosen a subset of these Dirac delta basis functions[13] as our basis. This would be very strange—our representation of the wavefunction would allow it to have a value at one gridpoint, and at the next gridpoint, but not in between. Spatially separated Dirac deltas do not spatially overlap at all; the matrix elements of the kinetic energy operator:

[13] Strictly, distribtuions, not functions, but "basis distributions" just doesn't have the right ring to it.

$$\langle \boldsymbol{r}_i | \hat{K} | \boldsymbol{r}_j \rangle = \int \delta(\boldsymbol{r} - \boldsymbol{r}_i) \left(-\frac{\hbar^2}{2m} \nabla^2 \right) \delta(\boldsymbol{r} - \boldsymbol{r}_j) \, \mathrm{d}\boldsymbol{r} \tag{3.21}$$

would all be zero for $i \neq j$, disallowing any flow of amplitude from one point in space to another by virtue of it not being able to pass through the intervening points. Neither have we chosen a set of two-dimenstional boxcar functions centred on each gridpoint with width Δx in each direction. These cover all space in between gridpoints, but are no good because they are not twice differentiable everywhere, and hence the kinetic energy operator's matrix elements cannot be evaluated. No, neither of these bases will do. To interpret our spatial grid as a basis, we need a set of functions $[FUNC_{ij}(r)]$ (where i and j are the indices of the gridpoints in the x and y directions respectively) that have unit norm, are nonzero only at one gridpoint and are zero at all others, and are twice differentiable everywhere in our spatial region. Infinite choices present themselves to us,

rev: 28 (9bcc7631183f)
author: Chris Billington <chrisjbillington@gmail.com>
date: Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom

differing only in their incompleteness—the choice of which state vectors they will and won't be able to represent. A sensible choice is that we want to be able to represent the state vectors whose wavefunctions do not change much between adjacent gridpoints, and we are happy for the necessary incompleteness of our basis to exclude wavefunctions with any sort of structure in between gridpoints.

The discrete Fourier transform to the rescue

It turns out that discretising space in this way can indeed be equivalent to choosing an entirely sensible basis. This is made clearer by thinking in terms of what we have done in Fourier space, which I'll quickly go through now.

One possible basis for representing all possible state vectors is the Fourier basis $\{|\boldsymbol{k}_{i,j}\rangle\}$. With it, any state vector (whose wavefunction is nonzero only within the $2D$ region)can be represented as the sum of basis vectors whose wavefunctions are 2D plane waves, also localised to the 2D region:

$$\langle \boldsymbol{r}|\boldsymbol{k}_{i,j}\rangle = \begin{cases} \frac{1}{\sqrt{A}} e^{i\boldsymbol{k}_{i,j}\cdot\boldsymbol{r}} & (\boldsymbol{r} \text{ within 2D region}) \\ 0 & (\boldsymbol{r} \text{ not within 2D region}), \end{cases} \tag{3.22}$$

where A is the area of the $2D$ region and the wavevector of each plane wave is

$$\boldsymbol{k}_{i,j} = \left[\frac{2\pi i}{L_x}, \frac{2\pi j}{L_y}\right]^{\mathrm{T}}, \tag{3.23}$$

where i and j are (possibly negative) integers. Any state vector localised to the 2D region can then be written as the infinite sum:

$$|\psi\rangle = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \langle \boldsymbol{k}_{i,j}|\psi\rangle \; |\boldsymbol{k}_{i,j}\rangle \tag{3.24}$$

$$\Rightarrow \psi(\boldsymbol{r}) = \begin{cases} \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \langle \boldsymbol{k}_{i,j}|\psi\rangle \frac{1}{\sqrt{A}} e^{i\boldsymbol{k}_{i,j}\cdot\boldsymbol{r}} & (\boldsymbol{r} \text{ within 2D region}) \\ 0 & (\boldsymbol{r} \text{ not within 2D region}). \end{cases} \tag{3.25}$$

So $\{\langle \boldsymbol{k}_{i,j}|\psi\rangle\}$ are simply the coefficients of the 2D Fourier series of $\psi(\boldsymbol{r})$.

What does this have to do with our discretised space? These basis functions $\{\langle \boldsymbol{r}|\boldsymbol{k}_{i,j}\rangle\}$ don't have the required properties for a discrete basis. For one, there are an infinite number of them, and we require 25. Secondly, all of them are nonzero everywhere within the 2D region, whereas we require each basis function to be nonzero at exactly one of out 25 gridpoints.

We can solve the first problem by truncating the Fourier series. By only including basis vectors $|\boldsymbol{k}_{i,j}\rangle$ for which:

$$\begin{cases} i \in [-\frac{N_x}{2}, \frac{N_x}{2} - 1] & (N_x \text{ even}) \\ i \in [-\frac{N_x-1}{2}, \frac{N_x-1}{2}] & (N_x \text{ odd}) \end{cases} \tag{3.26}$$

and

$$\begin{cases} j \in [-\frac{N_y}{2}, \frac{N_y}{2} - 1] & (N_y \text{ even}) \\ j \in [-\frac{N_y-1}{2}, \frac{N_y-1}{2}] & (N_y \text{ odd}) \end{cases} \tag{3.27}$$

we include only the Nx and N_y (both equal to 5 in our example) longest wavelengths in each respective spatial dimension. This is a sensible truncation with a physically meaningful interpretation. By making it, we are no longer able to represent state vectors with

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

short wavelength components. Because the kinetic energy operator, when represented in the Fourier basis, is:

$$\langle \boldsymbol{k}_{i,j}|\hat{K}|\boldsymbol{k}_{i'j'}\rangle = \frac{\hbar^2 k^2}{2m}\delta_{ii'}\delta_{jj'}, \tag{3.28}$$

where $k = |\boldsymbol{k}_i|$, by excluding basis vectors with larger wavevectors, we are excluding state vectors with large kinetic energy. Thus the truncation is a kinetic energy cutoff, and is an accurate approximation whenever a simulation is such that the system is unlikely to obtain kinetic energies above the cutoff.[14]

[14] Because a *square* region in Fourier space is being carved out, by limiting each of k_x and k_y to finite ranges rather than the total wavenumber $k = \sqrt{k_x^2 + k_y^2}$, there is no single kinetic energy cutoff so to speak. Nonetheless there is a maximum wavenumber $k_{\max} = \min(\{|k_x|\} \cup \{|k_y|\})$ defining a kinetic energy cutoff $K_{\max} = \hbar^2 k_{\max}^2/(2m)$ below which kinetic energies definitely are representable and above which they may not be.

Now we have 25 basis vectors—a discrete Fourier basis—but their spatial wavefunctions still don't have the property of being nonzero only at a single gridpoint each. On the contrary, each plane wave has has a constant amplitude everywhere in space. But consider the following superposition of Fourier basis vectors:

$$|\boldsymbol{r}_{i,j}\rangle = \sum_{i',j'} e^{-i\boldsymbol{k}_{i',j'}\cdot\boldsymbol{r}_{i,j}}\,|\boldsymbol{k}_{i',j'}\rangle \tag{3.29}$$

with $\boldsymbol{r}_{i,j} = (i\Delta x, j\Delta y)^{\mathrm{T}}$ and $i,j \in [0,4]$. This is simply a unitary transformation of the truncated Fourier basis, with the unitary transformation matrix elements:

$$\hat{U}_{\mathrm{DFT2};i',j';i,j} = \langle \boldsymbol{k}_{i',j'}|\boldsymbol{r}_{i,j}\rangle = e^{-i\boldsymbol{k}_{i',j'}\cdot\boldsymbol{r}_{i,j}}. \tag{3.30}$$

This transformation is in fact a discrete Fourier transform (hence the subscript), and the basis vectors $\{|\boldsymbol{r}_{i,j}\rangle\}$ have spatially localised wavefunctions that are nonzero only at one of the spatial gridpoints, so we refer to them as a discrete real-space basis. State vectors and operators can be transformed from their discrete Fourier space representation to their discrete real-space representation and back using the unitary $\hat{U}_{\mathrm{DFT2}}$:

$$\langle \boldsymbol{r}_{i,j}|\psi\rangle = \sum_{i',j'} \hat{U}^{\dagger}_{\mathrm{DFT2};i,j;i',j'}\,\langle \boldsymbol{k}_{i',j'}|\psi\rangle \tag{3.31}$$

$$\langle \boldsymbol{k}_{i,j}|\psi\rangle = \sum_{i',j'} \hat{U}_{\mathrm{DFT2};i',j';i,j}\,\langle \boldsymbol{r}_{i',j'}|\psi\rangle \tag{3.32}$$

[TODO OPERATOR TRANSFORMATIONS ONCE INDEXING IMPROVED AND VECTOR REPRESENTATIONS INTRODUCED]

The spatial representation of the basis vectors $\{|\boldsymbol{r}_{i,j}\rangle\}$ can be computed using (3.22) as:

$$\phi_{i,j}(\boldsymbol{r}) = \langle \boldsymbol{r}|\boldsymbol{r}_{i,j}\rangle = \sum_{i',j'} e^{-i\boldsymbol{k}_{i',j'}\cdot\boldsymbol{r}_{i,j}}\,\langle \boldsymbol{r}|\boldsymbol{k}_{i',j'}\rangle \tag{3.33}$$

$$\Rightarrow \phi_{i,j}(\boldsymbol{r}) = \begin{cases} \sum_{i',j'} \frac{1}{\sqrt{A}} e^{i\boldsymbol{k}_{i',j'}\cdot(\boldsymbol{r}-\boldsymbol{r}_{i,j})} & (\boldsymbol{r} \text{ within 2D region}) \\ 0 & (\boldsymbol{r} \text{ not within 2D region}), \end{cases} \tag{3.34}$$

and are plotted in [TODO MAKE FIGURE].

These functions are sometimes called *periodic sinc functions*, or *band-limited delta functions* [CITE]. Each of them is zero at all of our gridpoints except one, and and they form an orthonormal basis set. They satisfy all of our requirements to be a basis corresponding to our gridded discretisation of space. Thus, the approach of discretising space on a grid is indeed equivalent to choosing a (necessarily incomplete) orthonormal basis in which to work.

One thing to note is that these functions are periodic. By using the Fourier basis in the way we have to restrict our basis to cover only a finite region of both Fourier space and real space, we have necessarily imposed periodicity on the problem. This periodicity shows

rev: 28 (9bcc7631183f)
author: Chris Billington <chrisjbillington@gmail.com>
date: Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom

itself when we compute matrix elements of operators in this basis - if we compute the kinetic energy operator's matrix elements for example, it will couple basis states across the boundary of the region, resulting in spatial periodicity—a wavepacket moving rightward through the right boundary will emerge moving rightward from the left boundary.

Perhaps less obviously, the basis is also periodic in Fourier space, and so a wavepacket moving out of the region of Fourier space simulated will also wrap around to the opposite side of Fourier space. In real space, this may appear as a wavepacket undergoing acceleration only to suddenly reverse its velocity as if reflecting off a barrier [SEE FIG TODO FOR EXAMPLE]. This effect is entirely unphysical[15] and should be taken as a sign that the spatial grid is not fine enough for the dynamics being simulated.

[15] With the possible exception of the region of Fourier space being simulated corresponding to the first Brillouin zone of a lattice potential, in which case these velocity reversals are Bloch oscillations.

[16] Or rather, a state vector's projection onto the subspace being simulated.

Matrix elements using DFT basis

We now have a finite basis $\{|\boldsymbol{r}_{i,j}\rangle\}$ that matches our intuitions somewhat for representing a wavefunction at a set of gridpoints. A state vector[16] can be represented as a linear sum of these basis vectors, with the coefficient for each one simply being equal to the value of that state vector's wavefunction at the gridpoint where that basis vector is nonzero:

$$|\psi\rangle = \sum_{i,j} c_{i,j} |\boldsymbol{r}_{i,j}\rangle , \tag{3.35}$$

where

$$c_{i,j} = \langle \boldsymbol{r}_{i,j} | \psi \rangle = \psi(\boldsymbol{r}_{i,j}). \tag{3.36}$$

Armed with a basis, we can now proceed to calculate matrix elements of the Hamiltonian, and then proceed to solve the differential equation [TODO EQREF 3.3] to determine how the coefficients $\{c_{i,j}\}$ evolve in time.

The specific properties of our basis make it quite useful for a range of common Hamiltonians. For example, let's take the single particle Schrödinger Hamiltonian:

$$\hat{H}_{\text{Schrö}} = -\frac{\hbar^2 \hat{k}^2}{2m} + V(\hat{\boldsymbol{r}}), \tag{3.37}$$

where $\hat{k} = |\hat{\boldsymbol{k}}|$.

The two terms, kinetic and potential, are each diagonal in different bases. The kinetic term is diagonal in the Fourier basis:

$$\langle \boldsymbol{k}' | -\frac{\hbar^2 \hat{k}^2}{2m} | \boldsymbol{k} \rangle = -\frac{\hbar^2 k^2}{2m} \delta(\boldsymbol{k} - \boldsymbol{k}'), \tag{3.38}$$

where $k = |\boldsymbol{k}|$, and the potential term is diagonal in the spatial basis:

$$\langle \boldsymbol{r}' | V(\hat{\boldsymbol{r}}) | \boldsymbol{r} \rangle = V(\boldsymbol{r}) \delta(\boldsymbol{r} - \boldsymbol{r}'). \tag{3.39}$$

Equivalent relations hold for our discrete Fourier and spatial basis:

$$\langle \boldsymbol{k}_{i'j'} | -\frac{\hbar^2 \hat{k}^2}{2m} | \boldsymbol{k}_{i,j} \rangle = -\frac{\hbar^2 k_{i,j}^2}{2m} \delta_{i'i} \delta_{j'j}, \tag{3.40}$$

$$\langle \boldsymbol{r}_{i'j'} | V(\hat{\boldsymbol{r}}) | \boldsymbol{r}_{i,j} \rangle = V(\boldsymbol{r}_{i,j}) \delta_{i'i} \delta_{j'j}. \tag{3.41}$$

But to obtain our differential equation, we need all the matrix elements of $\hat{H}_{\text{Schrö}}$ in a single basis.

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

3.4 Discrete degrees of freedom

3.4.1 The interaction picture

A common situation in atomic physics is to be simulating the internal state of an atom, armed with the knowledge that only a small number of atomic states are able to become occupied.

- Sometimes called a "rotating frame"
- Is equivalent to basis change where new basis functions differ by a time-dependent phase factor
- Is defined by a time-independent Hamiltonian
- This has the effect of moving some time dependence into the operators (demonstrate, by writing some operators with the unitary in front of them. As you can see it is simply a change of basis - but a time-dependent one.)
- No need to remain in the same interaction picture - can be redefined arbitrarily often throughout a simulation and state vectors transformed into new basis.

3.4.2 Unitary integration

Direct exponentiation via diagonalisation of Hamiltonian

- Unitary - doesn't mean it's accurate but means it won't explode. Great for the bits of your simulation that are explosion-prone but don't matter (like regions of space where the wavefunction is near zero but the potential is large or steep)
- error is order [whatever it is] per timestep, not great compared to RK4
- can be combined with RK4 to improve accuracy (see later subsection)

Approximate exponentiation by operator product

[Comment in this section how the approximate total unitary can be used at each timestep to define an interaction picture, and the remaining dynamics simulated with RK4 like RKILIP does in the spatial basis. Interaction pictures are really useful!]

3.5 Continuous degrees of freedom

Every symbol on the paper has a representation in a computer. State vectors are arrays of complex numbers, operators are matrices - differential operators are no exception. Operators must have a concrete representation, their matrix elements can be computed and then things solved with linear algebra. For discrete degrees of freedom, the matrix representation of the operators may be known, for continuous ones you can find the matrix elements once you define what basis functions you will use [show how]. Or, for the DVR it is a little more subtle (because it's not a basis) but still basically the same process.

- Have to be discretised in some way to simulate on a computer - need basis functions. Often a spatial basis is used. Any spatial basis must be combined with assumptions about what the wavefunction is doing at points in between the basis points, in order to define differential operators. Finite differences approximates wavefunction as low-order polynomial in between points (is this equivalent to a polynomial *basis*?

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

Probably not.). Fourier method assume the Fourier series of the wavefunction at the given points can be used to interpolate between points (or the wavefuntion can be Fourier transformed and calculations can be done directly in the Fourier basis). DVR is not actually a spatial basis despite appearances. It assumes the wavefunction is a sum of polynomial 'shape functions', but these shape functions are not basis functions as they are not orthonormal. This is why it is called a representation rather than a basis. Regardless, the shape functions can be used to define an interpolation of the wavefunction between points and thus define differential operators.

3.5.1 Finite differences

Show a matrix representation of a few different finite differences, to show that differential operators really are just matrices. They approximate the function as low-order polynomials about each point. You can take them to arbitrarily high order.

3.5.2 The Fourier basis

Because of properties of Fourier transforms, derivitives can be taken in Fourier space as simple multiplication. This is essentially because differential operators are diagonal in the Fourier basis. So you can use this fact to define a differential operator in the spatial basis [show matrix] ...or, you could just implement it with Fourier transforms, since FFTs are faster than matrix-vector multiplication ($O(n \log(n))$ rather than $O(n^2)$)

Split operator method

- Equivalent to approximate exponentiation via operator product with the discrete case

3.5.3 Harmonic oscillator basis functions

3.6 Finding ground states

3.6.1 Imaginary time evolution

3.6.2 Successive over-relaxation

3.6.3 Generalisation to excited states via Gram–Schmidt orthonormalisation

Directly diagonalising a Hamiltonian can be costly in a spatial basis. Another approach is to find the ground state using one of the above techniques, and then repeat the process, subtracting off the wavefunction's projection onto the already found ground state at every step. This yields the lowest energy state that is orthogonal to the first - i.e the first excited state. Repeating the process, but subtracting off *both* eigenstates found so far, then yields the second excited state and so forth. This is simply the Gram-Schmidt process for finding orthonormal vectors, with the additional step of relaxing each vector to the lowest possible energy for each one - this ensures the eigenstates of the Hamiltonian are produced, rather than a different orthogonal basis. Extra conditions can be imposed on the wavefunction at each relaxation step in order to obtain particular solutions in the case of degenerate eigenstates. For example, a phase winding can be imposed in order to obtain a particular harmonic oscillator state - otherwise this process produces an arbitrary superposition of basis states that have equal energy.

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```

3.7 The finite-element discrete variable representation

- Plots of representation of sine wave as function of number of points between FEDVR and FD. RMS error of a derivative operator perhaps.

[explanation of how it works, comparison of implementation with RSP4 vs something like RK4. RK4 is more general purpose, method does not need to be modified for different Hamiltonians. Main limitation is inability to factor out fast dynamical phases, see RK4ILIP for solution to this. MPI implementation and scaling properties with increasing cores. Superscaling at low number of cores. Mention how my implementation can tolerate high network latency due to early sending od data before all local computations are complete. Mention that it is ripe for GPU processing. Limitations: vulnerable to Runge's phenomenon for sharp potentials. Can't increase the order of the polynomials much because small spacing at the edges requires tiny timesteps. Possible solution: preconditioning the potential to be an approximation better representable in the DVR basis.]

rev: 28 (9bcc7631183f)
author: Chris Billington <chrisjbillington@gmail.com>
date: Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom

This page intentionally left blank

References

rev: 28 (9bcc7631183f)
author: Chris Billington <chrisjbillington@gmail.com>
date: Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom

This page intentionally left blank

Word count

```
Total
Words in text: 21632
Words in headers: 270
Words outside text (captions, etc.): 3328
Number of headers: 70
Number of floats/tables/figures: 26
Number of math inlines: 399
Number of math displayed: 78
Files: 9
Subcounts:
  text+headers+captions (#headers/#floats/#inlines/#displayed)
  1753+28+325 (9/2/26/8) File(s) total: atomic_physics.tex
  725+17+385 (4/4/7/0) File(s) total: experiment.tex
  50+0+0 (0/0/0/0) File(s) total: front_matter.tex
  1370+19+65 (3/2/39/13) File(s) total: hidden_variables.tex
  2043+7+145 (3/1/4/0) File(s) total: introduction.tex
  8503+131+958 (30/2/245/53) File(s) total: numerics.tex
  4435+33+760 (11/9/1/0) File(s) total: software.tex
  2717+22+690 (7/6/77/4) File(s) total: velocimetry.tex
  36+13+0 (3/0/0/0) File(s) total: wave_mixing.tex
```

```
rev:     28 (9bcc7631183f)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Wed Nov 30 15:03:55 2016 -0500
summary: Numerics continuous degrees of freedom
```