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: 21 (12a0a54c2052)
author: Chris Billington <chrisjbillington@gmail.com>
date: Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work

This page intentionally left blank

Contents

Contents 1

3 Quantum mechanics on a computer 3

3.1 From the abstract to the concrete: neglect, discretisation and representation . . . 3
3.2 Solution to the Schrödinger equation by direct exponentiation . . . 4
3.2.1 Matrix exponentiation by diagonalisation . . . 5
3.2.2 Time-ordered exponentials and time-ordered products . . . 6
3.3 Discrete degrees of freedom . . . 8
3.3.1 The interaction picture . . . 8
3.3.2 Unitary integration . . . 8
3.4 Continuous degrees of freedom . . . 8
3.4.1 Finite differences . . . 9
3.4.2 The Fourier basis . . . 9
3.4.3 Harmonic oscillator basis functions . . . 9
3.5 Finding ground states . . . 9
3.5.1 Imaginary time evolution . . . 9
3.5.2 Successive over-relaxation . . . 9
3.5.3 Generalisation to excited states via Gram–Schmidt orthonormalisation . . . 9
3.6 The finite-element discrete variable representation . . . 9
3.7 Fourth order Runge–Kutta in an instantaneous local interaction picture 10
3.7.1 Algorithm . . . 12
3.7.2 Domain of improvement over other methods . . . 13
3.7.3 Results . . . 15
3.7.4 Discussion . . . 18

References 19

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

This page intentionally left blank

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 simplicitly 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:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

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:

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

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

[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

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

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:

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

where

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

is the unitary evolution operator for time evolution from the initial time t to time $t + \Delta t$, and is computed using a matrix exponential of $-\frac{i\Delta t}{\hbar}H$. 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

$$e^H = Ue^D U^\dagger. \tag{3.10}$$

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:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

```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

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

with

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

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]:

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

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:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

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

$$U(t, t+\Delta t) = \mathcal{T}\left\{\sum_{n=0}^{\infty} \frac{1}{n!}\left[-\frac{i}{\hbar}\int_t^{t+\Delta t} H(t')\,\mathrm{d}t'\right]^n\right\} \tag{3.15}$$

$$= \sum_{n=0}^{\infty} \frac{(-i)^n}{n!\,\hbar^n}\mathcal{T}\left\{\int_t^{t+\Delta t} H(t')\,\mathrm{d}t'\right\}^n . \tag{3.16}$$

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:

$$U(t, t+\Delta t) = \sum_{n=0}^{\infty} \frac{(-i)^n}{n!\,\hbar^n}\int_t^{t+\Delta t}\cdots\int_t^{t+\Delta t} \mathcal{T}\left\{H(t'_0)\cdots H(t'_n)\right\}\mathrm{d}t'_0\cdots\mathrm{d}t'_n\,, \tag{3.17}$$

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 $U(t, t+\Delta t)$ 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:

$$U(t, t+\Delta t) = \sum_{n=0}^{\infty}\left(-\frac{i}{\hbar}\right)^n\int_t^{t+\Delta t}\cdots\int_t^{t'_n} H(t'_0)\cdots H(t'_n)\,\mathrm{d}t'_0\cdots\mathrm{d}t'_n\,. \tag{3.18}$$

[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.

- Descriptions of howf to perform simulations of atomic systems and Bose–Einstein condensates. Include numerical methods, interaction pictures, Monte–Carlo wavefunction methods and other acquired wisdom that would be relevant to anyone implementing similar simulations.
- Fourier method: usefulness, use with RK4IP, limitations (i.e. first derivs no good near vortices)
- absorbing boundary conditions, reflective boundary conditions, periodic boundary conditions

Operators and their matrix representation The general case: Need to integrate the equations of motion. What does this mean in a computer? Formal solution of the Schrodinger equation. Is not so formal - can be typed into a programming language thusly: [example code - time ordered product of matrix exponentials].

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.

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

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.

3.3 Discrete degrees of freedom

3.3.1 The interaction picture

- 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.

3.3.2 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!]

3.4 Continuous degrees of freedom

- 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*? 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

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

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.

3.4.1 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.

3.4.2 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

3.4.3 Harmonic oscillator basis functions

3.5 Finding ground states

3.5.1 Imaginary time evolution

3.5.2 Successive over-relaxation

3.5.3 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.

3.6 The finite-element discrete variable representation

[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

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

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.]

3.7 Fourth order Runge–Kutta in an instantaneous local interaction picture

Consider the differential equation for the components of a state vector $|\psi(t)\rangle$ in a particular basis with basis vectors $|n\rangle$. This might simply be the Schrödinger equation, or perhaps some sort of nonlinear or other approximate, effective or phenomenological equation not corresponding to pure Hamiltonian evolution. Though they may have additional terms, such equations are generally of the form:

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle n|\psi(t)\rangle = -\frac{i}{\hbar}\sum_m \langle n|\hat{H}(t)|m\rangle\,\langle m|\psi(t)\rangle\,, \tag{3.19}$$

where $\langle n|\hat{H}(t)|m\rangle$ are the matrix elements in that basis of the Hamiltonian $\hat{H}(t)$, which in general can be time dependent, or even a function of $|\psi(t)\rangle$, depending on the exact type of equation in use. If $\hat{H}(t)$ is almost diagonal in the $|n\rangle$ basis, then the solution to (3.19) is dominated by simple dynamical phase evolution, that is:

$$|\psi(t)\rangle \approx \sum_m e^{-\frac{i}{\hbar}E_m t}\,|m\rangle\,, \tag{3.20}$$

where E_m is the energy eigenvalue corresponding to the eigenstate $|m\rangle$.

A transformation into an interaction picture (IP) [1, p318] is commonly used to treat this part of the evolution analytically, before solving the remaining dynamics with further analytics or numerics. For numerical methods, integration in the interaction picture allows one to use larger integration timesteps, as one does not need to resolve the fast oscillations around the complex plane due to this dynamical phase.

Choosing an interaction picture typically involves diagonalising the time-independent part of a Hamiltonian, and then proceeding in the basis in which that time-independent part is diagonal. However, often one has a good reason to perform computations in a different basis, in which the time independent part of the Hamiltonian is only approximately diagonal,[8] and transforming between bases may be computationally expensive (involving large matrix-vector multiplications). Furthermore, the Hamiltonian may change sufficiently during the time interval being simulated that the original time-independent Hamiltonian no longer dominates the dynamics at later times. In both these cases it would still be useful to factor out the time-local oscillatory dynamics in whichever basis is being used, in order to avoid taking unreasonably small timesteps.

To that end, suppose we decompose $\hat{H}(t)$ into diagonal and non-diagonal (in the $|n\rangle$ basis) parts at each moment in time:

$$\hat{H}(t) = \hat{H}_{\text{diag}}(t) + \hat{H}_{\text{nondiag}}(t), \tag{3.21}$$

and use the diagonal part at a specific time $t = t'$ to define a time-independent Hamiltonian:

$$\hat{H}_0^{t'} = \hat{H}_{\text{diag}}(t'), \tag{3.22}$$

[8] For example, a spatial basis which allows for partitioning the integration region over multiple nodes on a cluster or cores on a GPU.

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

which is diagonal in the $|n\rangle$ basis. We can then use then use $\hat{H}_0^{t'}$ to define an interaction picture state vector:

$$|\psi_{\mathrm{I}}^{t'}(t)\rangle = e^{\frac{i}{\hbar}(t-t')\hat{H}_0^{t'}} |\psi(t)\rangle\,, \tag{3.23}$$

which obeys the differential equation:

$$\frac{\mathrm{d}}{\mathrm{d}t}|\psi_{\mathrm{I}}^{t'}(t)\rangle = e^{\frac{i}{\hbar}(t-t')\hat{H}_0^{t'}} \frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle + \frac{i}{\hbar}\hat{H}_0^{t'}|\psi_{\mathrm{I}}^{t'}(t)\rangle\,, \tag{3.24}$$

where:

$$|\psi(t)\rangle = e^{-\frac{i}{\hbar}(t-t')\hat{H}_0^{t'}} |\psi_{\mathrm{I}}^{t'}(t)\rangle \tag{3.25}$$

is the original Schrödinger picture (SP) state vector.

This transformation is exact, no approximations or assumptions have been made. If indeed the dynamics of $|\psi(t)\rangle$ in the given basis are dominated by fast oscillating dynamical phases, that is, the diagonals of $\hat{H}_{\mathrm{diag}}(t)$ are much greater than all matrix elements of $\hat{H}_{\mathrm{nondiag}}(t)$ in the $|n\rangle$ basis, then solving the differential equation (3.24) for $|\psi_{\mathrm{I}}^{t'}(t)\rangle$ should allow one to use larger integration timesteps than solving (3.19) directly. And if not, then it should do no harm other than the (small) computational costs of computing some extra scalar exponentials.

Equation (3.23) defines an *instantaneous* interaction picture, in that it depends on the dynamics at a specific time $t = t'$, and can be recomputed repeatedly throughout a computation in order to factor out the fast dynamical phase evolution even as the oscillation rates change over time. It is *local* in that $H_0^{t'}$ is diagonal in the $|n\rangle$ basis, which means that transformations between Schrödinger picture and interaction picture state vectors involves ordinary, elementwise exponentiation of vectors, rather than matrix products. Thus (3.23), (3.24) and (3.25) can be written componentwise as:

$$\langle n|\psi_{\mathrm{I}}^{t'}(t)\rangle = e^{i(t-t')\omega_n^{t'}} \langle n|\psi(t)\rangle\,, \tag{3.26}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle n|\psi_{\mathrm{I}}^{t'}(t)\rangle = e^{i(t-t')\omega_n^{t'}} \frac{\mathrm{d}}{\mathrm{d}t}\langle n|\psi(t)\rangle + i\omega_n^{t'}\langle n|\psi_{\mathrm{I}}^{t'}(t)\rangle\,, \tag{3.27}$$

and:

$$\langle n|\psi(t)\rangle = e^{-i(t-t')\omega_n^{t'}} \langle n|\psi_{\mathrm{I}}^{t'}(t)\rangle\,, \tag{3.28}$$

where we have defined:

$$\omega_n^{t'} = \frac{1}{\hbar}\langle n|\hat{H}_0^{t'}|n\rangle \tag{3.29}$$

This is in contrast to fourth order Runge–Kutta in the interaction picture (RK4IP) [2], in which the interaction picture uses the Fourier basis and thus transforming to and from it involves fast Fourier transforms (FFTs). RK4IP was developed to augment computations in which FFTs were already in use for evaluating spatial derivatives, and so its use of FFTs imposes no additional cost. Nonetheless, an interaction picture based on the kinetic term of the Schrödinger equation (which is the term of the Hamiltonian that RK4IP takes as its time-independent part) may not be useful if that term does not dominate the Hamiltonian, as in the case of a Bose–Einstein condensate in the Thomas–Fermi limit. We compare the two methods below.

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

3.7.1 Algorithm

The *fourth order Runge–Kutta in an instantaneous local interaction picture* RK4ILIP algorithm is now obtained by using (3.23) to define a new interaction picture at the beginning of each fourth-order Runge–Kutta (RK4) integration timestep. The differential equation and initial conditions supplied to the algorithm are in the ordinary Schrödinger picture, and the interaction picture is used only within a timestep, with the Schrödinger picture state vector returned at the end of each timestep. Thus differential equations need not be modified compared to if ordinary RK4 were being used, and the only modification to calling code required is for a function to compute and return $\omega_n^{t'}$.

Being based on fourth order Runge–Kutta integration, this new method enjoys all the benefits of a workhorse method that is time-proven, and—as evidenced by its extremely widespread use—at a sweet-spot of ease of implementation, accuracy, and required computing power [3].

Below is the resulting algorithm for performing one integration timestep. It takes as input the time t_0 at the start of the timestep, the timestep size Δt, an array $\boldsymbol{\psi}_0$ containing the components $\{\langle n|\psi(t_0)\rangle\}$ of the state vector at time t_0, a function $F(t, \boldsymbol{\psi})$ which takes a time and (the components of) a state vector and returns an array containing the time derivative of each component, and a function $G(t, \boldsymbol{\psi})$ which takes the same inputs and returns an array containing the interaction picture oscillation frequency ω_n for each component at that time.

For example, for the case of the Gross–Pitaevskii equation [4] in the spatial basis $\psi(\boldsymbol{r}, t) = \langle \boldsymbol{r}|\psi(t)\rangle$, these would be:

$$F(t, \psi(\boldsymbol{r}, t)) = -\frac{i}{\hbar}\Big[\underbrace{-\frac{\hbar^2}{2m}\nabla^2}_{\hat{H}_{\text{nondiag}}} + \underbrace{V(\boldsymbol{r}, t) + g|\psi(\boldsymbol{r}, t)|^2}_{\hat{H}_{\text{diag}}}\Big]\psi(\boldsymbol{r}, t), \tag{3.30}$$

and

$$G(t, \psi(\boldsymbol{r}, t)) = \frac{1}{\hbar}\Big[\underbrace{V(\boldsymbol{r}, t) + g|\psi(\boldsymbol{r}, t)|^2}_{\hat{H}_{\text{diag}}}\Big]. \tag{3.31}$$

Note that each symbol in bold in the algorithm below denotes an array containing one element for each basis vector $|n\rangle$, subscripts denote the different stages of RK4, and all arithmetic operations between arrays are elementwise[9] The only opportunity for non-elementwise operations to occur is within F, which contains the details (via $\hat{H}_{\text{nondiag}}$) of any couplings between basis states for whatever system of equations is being solved, for example, using FFTs or finite differences to evaluate the Laplacian in (3.30).

[9] For example, the expression $\boldsymbol{a} \leftarrow e^{-i\boldsymbol{\omega}\Delta t}\boldsymbol{b}$ indicates that for all n, $a_n \leftarrow e^{-i\omega_n \Delta t} b_n$, where a_n denotes the n^{th} element of $\boldsymbol{a}$ etc.

Algorithm 1 RK4ILIP

function RK4ILIP(t_0, Δt, $\boldsymbol{\psi}_0$, F)
$\boldsymbol{f}_1 \leftarrow F(t_0, \boldsymbol{\psi}_0)$ ▷ First evaluation of Schrödinger picture DE
$\boldsymbol{\omega} \leftarrow G(t_0, \boldsymbol{\psi}_0)$ ▷ Oscillation frequencies: $\hbar\omega_n = \langle n|\hat{H}_{\text{diag}}(t_0)|n\rangle$
$\boldsymbol{k}_1 \leftarrow \boldsymbol{f}_1 + i\boldsymbol{\omega}\boldsymbol{\psi}_0$ ▷ Evaluate (3.27) with $t - t' = 0$
$\boldsymbol{\phi}_1 \leftarrow \boldsymbol{\psi}_0 + \boldsymbol{k}_1 \frac{\Delta t}{2}$ ▷ First RK4 estimate of IP state vector, at $t = t_0 + \frac{\Delta t}{2}$
$\boldsymbol{\psi}_1 \leftarrow e^{-i\boldsymbol{\omega}\frac{\Delta t}{2}}\boldsymbol{\phi}_1$ ▷ Convert first estimate back to SP with (3.28)
$\boldsymbol{f}_2 \leftarrow F(t_0 + \frac{\Delta t}{2}, \boldsymbol{\psi}_1)$ ▷ Second evaluation of Schrödinger picture DE
$\boldsymbol{k}_2 \leftarrow e^{i\boldsymbol{\omega}\frac{\Delta t}{2}}\boldsymbol{f}_2 + i\boldsymbol{\omega}\boldsymbol{\phi}_1$ ▷ Evaluate (3.27) with $t - t' = \frac{\Delta t}{2}$
$\boldsymbol{\phi}_2 \leftarrow \boldsymbol{\psi}_0 + \boldsymbol{k}_2 \frac{\Delta t}{2}$ ▷ Second RK4 estimate of IP state vector, at $t = t_0 + \frac{\Delta t}{2}$
$\boldsymbol{\psi}_2 \leftarrow e^{-i\boldsymbol{\omega}\frac{\Delta t}{2}}\boldsymbol{\phi}_2$ ▷ Convert second estimate back to SP with (3.28)

rev: 21 (12a0a54c2052)
author: Chris Billington <chrisjbillington@gmail.com>
date: Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work

11: $\boldsymbol{f}_3 \leftarrow F(t_0 + \frac{\Delta t}{2}, \boldsymbol{\psi}_2)$ ▷ Third evaluation of Schrödinger picture DE
12: $\boldsymbol{k}_3 \leftarrow e^{i\boldsymbol{\omega}\frac{\Delta t}{2}}\boldsymbol{f}_3 + i\boldsymbol{\omega}\boldsymbol{\phi}_2$ ▷ Evaluate (3.27) with $t - t' = \frac{\Delta t}{2}$
13: $\boldsymbol{\phi}_3 \leftarrow \boldsymbol{\psi}_0 + \boldsymbol{k}_3\Delta t$ ▷ Third RK4 estimate of IP state vector, at $t = t_0 + \Delta t$
14: $\boldsymbol{\psi}_3 \leftarrow e^{-i\boldsymbol{\omega}\Delta t}\boldsymbol{\phi}_3$ ▷ Convert third estimate back to SP with (3.28)
15: $\boldsymbol{f}_4 \leftarrow F(t_0 + \Delta t, \boldsymbol{\psi}_3)$ ▷ Fourth evaluation of Schrödinger picture DE
16: $\boldsymbol{k}_4 \leftarrow e^{i\boldsymbol{\omega}\Delta t}\boldsymbol{f}_4 + i\boldsymbol{\omega}\boldsymbol{\phi}_3$ ▷ Evaluate (3.27) with $t - t' = \Delta t$
17: $\boldsymbol{\phi}_4 \leftarrow \boldsymbol{\psi}_0 + \frac{\Delta t}{6}(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4)$ ▷ Fourth RK4 estimate, at $t = t_0 + \Delta t$
18: $\boldsymbol{\psi}_4 \leftarrow e^{-i\boldsymbol{\omega}\Delta t}\boldsymbol{\phi}_4$ ▷ Convert fourth estimate back to SP with (3.28)
19: **return** $\boldsymbol{\psi}_4$ ▷ Return the computed SP state vector at $t = t_0 + \Delta t$
20: **end function**

Note on imaginary time evolution

When RK4ILIP is used for imaginary time evolution (ITE) [5], the oscillation frequencies $\boldsymbol{\omega}$ may have a large imaginary part. If the initial guess is different enough from the ground state, then the exponentials in (3.26), (3.27) and (3.28) may result in numerical overflow. To prevent this, one can define a clipped copy of $\boldsymbol{\omega}$,

$$\boldsymbol{\omega}_{\text{clipped}} = \text{Re}(\boldsymbol{\omega}) + i\begin{cases} -\frac{\log X}{\Delta t} & \text{Im}(\boldsymbol{\omega})\Delta t < -\log X \\ \text{Im}(\boldsymbol{\omega}) & -\log X \leq \text{Im}(\boldsymbol{\omega})\Delta t \leq \log X \\ \frac{\log X}{\Delta t} & \text{Im}(\boldsymbol{\omega})\Delta t > \log X \end{cases}, \tag{3.32}$$

where X is very large but less than the largest representable floating-point number, and use $\boldsymbol{\omega}_{\text{clipped}}$ in the exponents instead. In the below results I used RK4ILIP with ITE to smooth initial states of a Bose–Einstein condensate after a phase printing, and performed clipping with [10] $\log X = 400$.

This clipped version of $\boldsymbol{\omega}$ should be used in all exponents in the above algorithm, but only in exponents—not in the second term of (3.27). If it is used everywhere then all we have done is chosen a different (less useful) interaction picture, and the algorithm will still overflow. By clipping only the exponents, we produce temporarily "incorrect" evolution[11], limiting the change in magnitude of each component of the state vector to a factor of X per step (remembering that X is very large). This continues for the few steps that it takes ITE to get all components of the state vector to within a factor of X of the ground state, after which no clipping is necessary and convergence to the ground state proceeds as normal, subject to the ordinary limitations on which timesteps may be used with ITE.

[10] 400 being about half the largest (base e) exponent representable in double-precision floating point.

[11] Of no concern since we are using ITE as a relaxation method, and are not interested in intermediate states. Only the final state's correctness concerns us.

3.7.2 Domain of improvement over other methods

For simulations in the spatial basis, RK4ILIP treats the spatially local part of the Hamiltonian analytically to first order, and hence can handle larger potentials than ordinary RK4. However, since a global energy offset can be applied to any potential with no physically meaningful change in the results, ordinary RK4 can also handle large potentials — if they are large due a a large constant term which can simply be subtracted off.

So RK4ILIP is only of benefit in the case of large *spatial variations* in the potential. Only one constant can be subtracted off potentials without changing the physics — subtracting a spatially varying potential would require modification of the differential equation in the manner of a gauge transformation in order to leave the system physically unchanged[12].

However that's not quite all: large spatial variation in potentials often comes with the prospect of the potential energy turning into kinetic energy, in which case RK4ILIP is

[12] Though a numerical solution based on analytically gauging away potentials at each timestep might be equally as fruitful as RK4ILIP.

rev: 21 (12a0a54c2052)
author: Chris Billington <chrisjbillington@gmail.com>
date: Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work

Method	RK4	RK4IP	RK4ILIP	FSS
Error	$\mathcal{O}(\Delta t^4)$	$\mathcal{O}(\Delta t^4)$	$\mathcal{O}(\Delta t^4)$	$\mathcal{O}(\Delta t^2)$
FFTs per step	4	4	4	2
Large ΔV	No	No	Yes	Yes
Large kinetic term	No	Yes	No	Yes
Arbitrary operators	Yes	Yes†	Yes	No
Locally parallelisable	Yes	No	Yes	No
Arbitrary boundary conditions	Yes	No	Yes	No

Table 3.1: Advantages and disadvantages of four timestepping methods for simulating Bose–Einstein condensates. *Large* ΔV refers to whether the method can simulate potentials that vary throughout space by an amount larger than the energy scale $2\pi\hbar/\Delta t$ associated with the simulation timestep Δt. *Arbitrary operators* refers to whether the method permits operators that are not diagonal in either the spatial or Fourier basis, such as angular momentum operators. *Locally parallelisable* means the method can be formulated so as to use only spatially nearby points in evaluating operators, and thus is amenable to parallelisation by splitting the simulation over multiple cores in the spatial basis. † Whilst one can include arbitrary operators within the RK4IP method, only operators diagonal in Fourier space can be analytically treated the way RK4IP treats the kinetic term, and so there is no advantage for these terms over ordinary RK4.

also of little benefit, since in order to resolve the dynamical phase due to the large kinetic term, it would require timesteps just as small as those which ordinary RK4 would need to resolve the dynamical phase evolution from the large potential term.

This leaves RK4ILIP with an advantage only in the case of large spatial variations in the potential that do not lead to equally large kinetic energies. Hence the examples I show in the next section are ones in which the condensate is trapped in a steep potential well—the trap walls are high and hence involve large potentials compared to the interior, but do not lead to large kinetic energies because the condensate is trapped close to its ground state.

The Fourier split-step (FSS) method [6] (see section [TODO]) also models dynamical phases due to the potential analytically to low order. As such it is also quite capable of modeling large potentials. However, it requires that all operators be diagonal in either the spatial basis or the Fourier basis [6]. Therefore BECs in rotating frames, due to the Hamiltonian containing an angular momentum operator, are not amenable to simulation with FSS[13].

[13] Split-step with more than these two bases is however possible in other schemes such as the finite element discrete variable representation [7]—each operator can be diagonalised and exponentiated locally in each element and applied as a (relatively small) matrix multiplication rather than using FFTs.

This use of FFTs in both the FSS and RK4IP methods necessarily imposes periodic boundary conditions on a simulation, which may not be desirable. By contrast, if different boundary conditions are desired, finite differences instead of FFTs can be used to evaluate spatial derivatives in the RK4 and RK4ILIP methods, so long as a sufficiently high-order finite difference scheme is used so as not to unacceptably impact accuracy.

Along with the ability to impose arbitrary boundary conditions, finite differences require only local data, that is, only points spatially close to the point being considered need be known in order to evaluate derivatives there. This makes finite differences amenable to simulation on cluster computers [8, p100], with only a small number of points (depending on the order of the scheme) needing to be exchanged at node-boundaries each step. By contrast, FFT based derivatives require data from the entire spatial region. Whilst this can still be parallelised on a GPU, where all the data is available, it cannot be done on a cluster without large amounts of data transfer between nodes [9]. Thus, RK4 and RK4ILIP, being implementable with finite difference schemes, are considerably friendlier to cluster computing.

Table 3.1 summarises the capabilities of the four methods considered in the following results section. RK4ILIP is the only method capable of modelling a large spatial variation in the potential term whilst being locally parallelisable, and supporting arbitrary operators

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

and boundary conditions.

3.7.3 Results

Here I compare four numerical methods: Fourier split-step (FSS), fourth order Runge–Kutta in the interaction picture (RK4IP), ordinary fourth order Runge–Kutta (RK4), and my new method — fourth order Runge–Kutta in an instananeous local interaction picture (RK4ILIP).

The example chosen is a 2D simulation of a turbulent Bose–Einstein condensate, in both a rotating and nonrotating frame. For the nonrotating frame the differential equation simulated was equation (3.30), and for the rotating frame the same equation was with an additional two terms added to the Hamiltonian:

$$\hat{H}_{\text{rot}} + \hat{H}_{\text{comp}} = -\boldsymbol{\Omega} \cdot \hat{\boldsymbol{L}} + \frac{1}{2}\hbar m^2 \Omega^2 r^2 \tag{3.33}$$

$$= i\hbar\Omega\left(x\frac{\partial}{\partial y} - y\frac{\partial}{\partial x}\right) + \frac{1}{2}\hbar m^2 \Omega^2 r^2. \tag{3.34}$$

The addition of the first term transforms the original Hamiltonian into a frame rotating at angular frequency Ω in the (x, y) plane, and is equivalent to the the Coriolis and centrifugal forces that appear in rotating frames in classical mechanics [10]. The second term is a harmonic potential that exactly compensates for the centrifugal part of this force. In this way the only potential in the rotating frame is the applied trapping potential, and the only effect of the rotating frame is to add the Coriolis force.

Four trapping potentials were used, all radial power laws with different powers. These examples were chosen to demonstrate the specific situation in which RK4ILIP provides a benefit over the other methods for spatial Schrödinger-like equations, as discussed above.

The results of 120 simulation runs are shown in Figure 3.1. Each simulation was of a ^{87}Rb condensate in the $|F = 2, m_F = 2\rangle$ state, in which the two-body s-wave scattering length is $a = 98.98$ Bohr radii [11]. The simulation region was 20 μm in the x and y directions, and the Thomas–Fermi radius of the condensate was $R = 9$ μm. The chemical potential was $\mu = 2\pi\hbar \times 1.91$ kHz, which is equivalent to a maximum Thomas–Fermi density $\rho_{\text{max}} = 2.5 \times 10^{14}$ cm^{-3} and a healing length $\xi = 1.1$ μm. There were 256 simulation grid points in each spatial dimension, which is 14 points per healing length.

Four different potentials were used, all of the form $V(r) = \mu(r/R)^\alpha$ with $\alpha = 4, 8, 12, 16$. For the rotating frame simulations, the rotation frequency was $\Omega = 2\pi \times 148$ Hz. This is 89% of the effective harmonic trap frequency, defined as the frequency of a harmonic trap that would have the same Thomas–Fermi radius given the same chemical potential.

All ground states were determined using successive over-relaxation (See section [TODO]) with sixth-order finite differences for spatial derivatives. For the nonrotating simulations, convergence was reached with $\Delta\mu/\mu < 1 \times 10^{-13}$, with:

$$\Delta\mu = \sqrt{\frac{\langle\psi|(\hat{H} - \mu)^2|\psi\rangle}{\langle\psi|\psi\rangle}}, \tag{3.35}$$

where $\hat{H}$ is the nonlinear Hamiltonian and $\langle \boldsymbol{r}|\psi\rangle$ is the condensate wavefunction, which does not have unit norm. For the rotating frame simulations the ground states converged to $\Delta\mu/\mu \approx 9\times10^{-7}, 2\times10^{-6}, 3\times10^{-6}$ and 2×10^{-6} for $\alpha = 16, 12, 8$, and 4 respectively.

After each ground state was found, it was multiplied by a spatially varying phase factor corresponding to the phase pattern of a number of randomly positioned vortices:

$$\psi_{\text{vortices}}(x, y) = \psi_{\text{groundstate}}(x, y) \prod_{n=1}^{N} e^{\pm_n i \arctan2(y-y_n, x-x_n)} \tag{3.36}$$

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

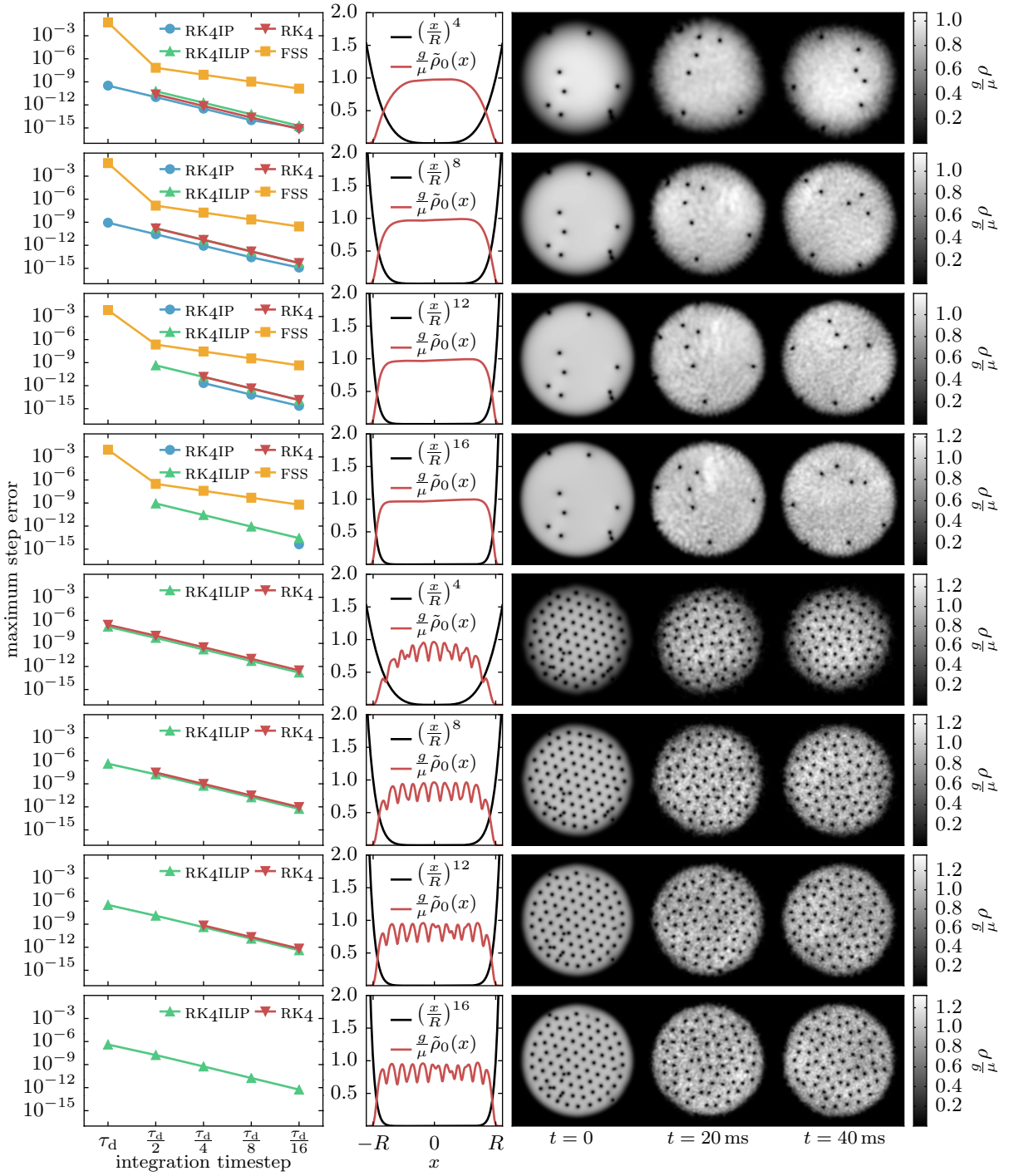

Figure 3.1: Results of simulations to compare RL4ILIP to other timestepping methods. Top four rows: Nonrotating frame simulations with four different radial power-law potentials. Bottom four rows: Rotating frame simulations with same four potentials. Left column: maximum per-step error $\int |\psi - \tilde{\psi}|^2 \,\mathrm{d}\boldsymbol{r} / \int |\tilde{\psi}|^2 \,\mathrm{d}\boldsymbol{r}$ of fourth order Runge–Kutta (RK4), its interaction picture variants (RK4IP and RK4ILIP) and Fourier split-step (FSS) as a function of timestep. Solutions were checked every 100 timesteps against a comparison solution $\tilde{\psi}$ computed using half sized steps for RK4 methods, and quarter sized steps for FSS. Simulations encountering numerical overflow not plotted. Centre column: potential (black) and average density $\tilde{\rho}_0$ of the initial state (red) over a slice of width $R/5$ in the y direction. Right column: Density of solution at initial, intermediate and final times for each configuration simulated (taken from RK4ILIP results). RK4ILIP is the only method usable in rotating frames and not encountering overflow in the steeper traps for the timesteps considered.

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

where arctan2 is the two-argument arctan function,[14] $N = 30$, $\pm_n$ is a randomly chosen sign, and (x_n, y_n) are vortex positions randomly drawn from a Gaussian distribution centred on $(0, 0)$ with standard deviation equal to the Thomas–Fermi radius R. The same seed was used for the pseudorandom number generator in each simulation run, and so the vortex positions were identical in each simulation run.

[14] Defined as the principle value of the argument of the complex number $x + iy$: $\text{arctan2}(y, x) = \text{Arg}(x + iy)$.

After vortex phase imprinting, the wavefunctions were evolved in imaginary time [5]. For the nonrotating frame simulations, imaginary time evolution was performed for a time interval equal to the chemical potential timescale $\tau_\mu = 2\pi\hbar/\mu$, and for the rotating frame simulations, for $\tau_\mu/10$. This was done to smooth out the condensate density in the vicinity of vortices, producing the correct density profile for vortex cores. However, since imaginary time evolution decreases the energy of the state indiscriminately, it also had the side effect of causing vortices of opposite sign to move closer together and annihilate. This decreased the number of vortices, and is the reason the smoothing step in the rotating frame simulations was cut short to $\tau_\mu/10$, as otherwise all vortices had time to annihilate with one of the lattice vortices. A vortex pair in the process of annihilating is visible in Figure 3.1 as a partially filled hole in the initial density profile near the top of the condensate in the $\alpha = 4$, 12, and 16 rotating frame simulations.[15]

[15] The initial states for the four different potentials are not identical, so by chance the corresponding vortex in the $\alpha = 8$ case was not close enough to a lattice vortex to annihilate.

The smoothed, vortex imprinted states were then evolved in time for 40 ms. For each simulation, five different timesteps were used: $\Delta t = \tau_\text{d}, \tau_\text{d}/2, \tau_\text{d}/4, \tau_\text{d}/8, \tau_\text{d}/16$, where $\tau_\text{d} = m\Delta x^2/\pi\hbar \approx 2.68$ µs is the dispersion timescale associated with the grid spacing Δx, defined as the time taken to move one gridpoint at the phase velocity of the Nyquist mode.

For the nonrotating frame simulations, spatial derivatives for the RK4 and RK4ILIP methods were determined using the Fourier method [see section TODO]. This was to ensure a fair comparison with the other two methods, which necessarily use Fourier transforms to perform computations pertaining due to the kinetic term.

For the rotating frame simulations, sixth-order finite differences with zero boundary conditions were used instead for the kinetic terms of the RK4 and RK4ILIP methods, which were the only two methods used for those simulations (due to the other methods being incompatible with the angular momentum operator required for a rotating frame). This choice was fairly arbitrary, but did allow the condensate to be closer to the boundary than is otherwise possible with the periodic boundary conditions imposed by use of the Fourier method for spatial derivatives. This is because the rotating frame Hamiltonian is not periodic in space, and so its discontinuity at the boundary can be a problem if the wavefunction is not sufficiently small there.

As shown in Figure 3.1, all methods tested generally worked well until they didn't work at all, with the per-step error of RK4-based methods being either small and broadly the same as the other RK4-based methods, or growing rapidly to the point of numerical overflow (shown as missing datapoints). The break down of FSS was less dramatic, though it too had a clear jump in its per-step error for larger timesteps. Comparing methods therefore came down to mostly whether or not a simulation experienced numerical overflow during the time interval being simulated.

The main result was that RK4ILIP and FSS remained accurate over the widest range of timesteps and trap steepnesses, with RK4 and RK4IP requiring ever smaller timesteps in order to not overflow as the trap steepness increased.

For the rotating frame simulations, which were only amenable to the RK4 and RK4ILIP methods, the same pattern was observed, with RK4 only working at smaller timesteps as the trap steepness was increased, and ultimately diverging for all timesteps tested at the maximum trap steepness. By contrast, RK4ILIP remained accurate over the entire range of timesteps at the maximum trap steepness.

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

3.7.4 Discussion

As mentioned, RK4ILIP is mostly useful for continuum quantum mechanics only when there are large spatial differences in the potential, which cannot give rise to equally large kinetic energies[16]. Furthermore, the advantage that RK4ILIP has over other methods with that same property is that it is does not require a particular form of Hamiltonian or a particular method of evaluating spatial derivatives. The former means is is applicable in rotating frames or to situations with unusual Hamiltonians, and the latter means is can be used with finite differences or FEDVR [7] and thus is amenable to parallelisation on a cluster computer.

[16] This is essentially due to such a situation violating the condition we laid out at the beginning of this section — that the simulation basis must be nearly an eigenbasis of the total Hamiltonian.

The ability to model large spatial variations in the potential provides only a narrow domain of increased usefulness over other methods. If a large kinetic energy results from the large potential, then the method requires just as small timesteps as any other. And if the large potential is supposed to approximate an an infinite well, then an actual infinite well may be modelled using zero boundary conditions, negating the need for something like RK4ILIP. However, when potential wells are steep, but not infinitely steep, here RK4ILIP provides a benefit. The only other model that can handle these large potentials—Fourier split-step—has the disadvantage that it cannot deal with arbitrary operators such as those arising from a rotating frame, and is not parallelisable with local data. The benefits of parallelisability are obvious, and the above results demonstrate RK4ILIP's advantage at simulating BECs in tight traps and rotating frames.

For systems with discrete degrees of freedom, RK4ILIP may be useful in the case where an approximate diagonalisation of the Hamiltonian is analytically known, and when the Hamiltonian's eigenvalues vary considerably in time (making a single interaction picture insufficient to factor out dynamical phases throughout the entire simulation). In this situation an analytic transformation into the diagonal basis can be performed at each timestep (or the differential equation analytically re-cast in that basis in the first place), and RK4ILIP can be used to factor out the time-varying dynamical phase evolution at each timestep. An example may be an atom with a magnetic moment in a time-varying magnetic field which varies over orders of magnitude. The transformation into the spin basis in the direction of the magnetic field can be analytically performed, and if the field varies by orders of magnitude, so do the eigenvalues of the Hamiltonian. Although the eigenvalues in this case and other similar cases can be computed analytically too, unless all time dependence of the Hamiltonian is known in advance of the simulation, it would be difficult to incorporate this into a re-casting of the differential equation in a time-dependent interaction picture. RK4ILIP may be useful in these cases to automate this process and evolve the system in the appropriate interaction picture at each timestep.

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

References

[1] J. J. Sakurai. *Modern quantum mechanics*. Addison-Wesley Pub. Co, Reading, Mass (1994). [p 10]

[2] Benjamin M. Caradoc Davies. *Vortex dynamics in Bose–Einstein condensates*. PhD thesis, University of Otago, Dunedin, New Zealand (2000). [p 11]

[3] W.H. Press. *The art of scientific computing*. Cambridge University Press (1992). [p 12]

[4] C.J. Pethick and H. Smith. *Bose–Einstein condensation in dilute gases*. Cambridge University Press (2002). [p 12]

[5] M. L. Chiofalo, S. Succi, and M. P. Tosi. *Ground state of trapped interacting Bose–Einstein condensates by an explicit imaginary-time algorithm*. Phys. Rev. E **62**, 7438 (2000). DOI: 10.1103/PhysRevE.62.7438. [pp 13 and 17]

[6] G.M. Muslu and H.A. Erbay. *Higher-order split-step Fourier schemes for the generalized nonlinear Schrödinger equation*. Mathematics and Computers in Simulation **7**, 581 (2005). DOI: 10.1016/j.matcom.2004.08.002. [p 14]

[7] Barry I. Schneider, Lee A. Collins, and S. X. Hu. *Parallel solver for the time-dependent linear and nonlinear Schrödinger equation*. Phys. Rev. E **73**, 036708 (2006). DOI: 10.1103/PhysRevE.73.036708. [pp 14 and 18]

[8] M.A. Heroux, P. Raghavan, and H.D. Simon. *Parallel processing for scientific computing*. Society for Industrial and Applied Mathematics, Philadelphia, PA (2006). [p 14]

[9] Anshul Gupta and Vipin Kumar. *The scalability of FFT on parallel computers*. IEEE Transactions on Parallel and Distributed Systems **4**, 922 (1993). [p 14]

[10] P. Gulshani and D. J. Rowe. *Quantum mechanics in rotating frames. I. The impossibility of rigid flow*. Canadian Journal of Physics **56**, 468 (1978). DOI: 10.1139/p78-060. [p 15]

[11] E. G. M. van Kempen, S. J. J. M. F. Kokkelmans, D. J. Heinzen, and B. J. Verhaar. *Interisotope determination of ultracold rubidium interactions from three high-precision experiments*. Phys. Rev. Lett. **88**, 093201 (2002). DOI: 10.1103/PhysRevLett.88.093201. [p 15]


```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```

This page intentionally left blank

Word count

```
Total
Words in text: 8151
Words in headers: 239
Words outside text (captions, etc.): 949
Number of headers: 57
Number of floats/tables/figures: 6
Number of math inlines: 216
Number of math displayed: 47
Files: 9
Subcounts:
  text+headers+captions (#headers/#floats/#inlines/#displayed)
  89+48+0 (11/0/1/0) File(s) total: atomic_physics.tex
  106+19+0 (5/0/0/0) File(s) total: experiment.tex
  50+0+0 (0/0/0/0) File(s) total: front_matter.tex
  1355+19+65 (3/2/39/13) File(s) total: hidden_variables.tex
  88+1+0 (1/0/0/0) File(s) total: introduction.tex
  6311+113+766 (26/2/174/34) File(s) total: numerics.tex
  64+10+0 (5/0/0/0) File(s) total: software.tex
  52+16+118 (3/2/2/0) File(s) total: velocimetry.tex
  36+13+0 (3/0/0/0) File(s) total: wave_mixing.tex
```

```
rev:     21 (12a0a54c2052)
author:  Chris Billington <chrisjbillington@gmail.com>
date:    Thu Aug 18 17:45:30 2016 +1000
summary: More numerics chapter work
```