Chapter 3

Time Dependent Schrödinger Equation

3.1 Revision of solution of partial differential equations

Our basic approach depends on whether we are solving an *boundary value* or *initial value* problem. As an example of a boundary value problem, consider solving Poisson's equation with a fixed array of charges, or for a potential specified on the bounding surface. Such problems are usually solved using various iterative grid-based methods, and are not relevant here. The time dependent Schrödinger equation is most usually cast into an initial value form, and so we shall first revise the basic approach to solving such problems.

As classic examples of initial value problems, we shall consider both the diffusion equation with constant diffusivity D:

$$\nabla^2 u(x,t) = \frac{1}{D}\frac{\partial u(x,t)}{\partial t} \tag{3.1}$$

and the wave equation with constant velocity v:

$$\nabla^2 u(x,t) = \frac{1}{v^2}\frac{\partial^2 u(x,t)}{\partial t^2} \tag{3.2}$$

Both can in general be written as one or more coupled first-order equations in terms of some *conserved flux* $\boldsymbol{F}(\mathbf{u})$:

$$\frac{\partial \mathbf{F}(\mathbf{u})}{\partial x} = -\frac{\partial \mathbf{u}(x,t)}{\partial t} \tag{3.3}$$

so for the wave equation we have

$$\mathbf{F}(\mathbf{u}) = \begin{pmatrix} 0 & -v \\ -v & 0 \end{pmatrix}.\mathbf{u} \tag{3.4}$$

3.1.1 Forward-Time Centred-Space algorithm

We start by constructing a finite difference approximation to equation 3.3 by choosing equally spaced points in t- and x- axes:

$$\begin{aligned} x_j &= x_0 + j\triangle x, \quad j = 0, 1, \ldots J \\ t_n &= t_0 + n\triangle t, \quad n = 0, 1, \ldots N \end{aligned} \tag{3.5}$$

and we introduce the following compact notation which will be used from hereon:

$$u\left(x_j, t_n\right) \rightarrow u_j^n \tag{3.6}$$

We know how to solve the spatial part of the differential equation by discretizing the derivatives, as seen in the earlier part of this course. We therefore choose a second-order "centred-space" representation for the first-order spatial derivative, using only quantities known at the current time-step n:

$$\left.\frac{\partial u}{\partial x}\right|_{j,n} = \frac{u_{j+1}^n - u_{j-1}^n}{2\triangle x} + O\left(\triangle x^2\right) \tag{3.7}$$

which is appropriate given the boundary conditions on the spatial part of the solution.

However, with the time part of the solution, we often have the boundary conditions expressed as initial value conditions. We cannot therefore use a centred-space algorithm in a straightforwards manner. The obvious way to propagate forwards in time then is to use the explicit first-order *forward Euler differencing* scheme:

$$\left.\frac{\partial u}{\partial t}\right|_{j,n} = \frac{u_j^{n+1} - u_j^n}{\triangle t} + O\left(\triangle t\right) \tag{3.8}$$

These two derivatives can then be combined to yield the Forwards-Time Centred-Space (FTCS) finite difference approximation to equation 3.3:

$$u_j^{n+1} \approx u_j^n - \frac{\triangle t}{\triangle x}\frac{u_{j+1}^n - u_{j-1}^n}{2}\frac{\partial F}{\partial u} \tag{3.9}$$

We represent this schematically in figure 3.9. Does this scheme work? As always, we need to do both a stability analysis and an error analysis ...

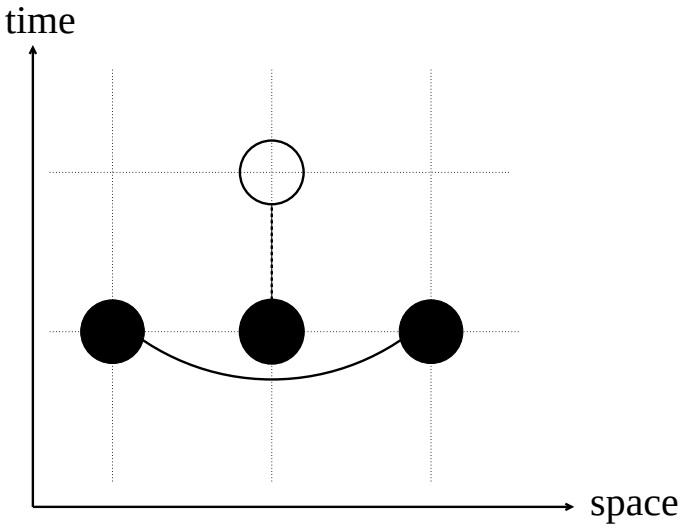

Figure 3.1: Schematic representation of the Forward-Time Centred-Space (FTCS) differencing method. The filled circles represent known points at the current time-step, and open circles represent new points at the next time-step. Solid lines are used to connect points used in the spatial derivatives, and dashed lines connect points used in time derivatives.

3.1.2 von Neumann stability analysis

The von Neumann stability analysis examines the behaviour of the eigenmode solutions in the finite difference equation. We assume all coefficients to be slowly varying in time and locally constant in space. So for the discretized wave equation we have the following general form for an eigenmode:

$$u_j^n(k) = \xi(k)^n e^{ikj\triangle x} \tag{3.10}$$

at some particular wavenumber k, at time $n\triangle t$ and spatial grid point $j\triangle x$.

In general, if $|\xi(k)| > 1$ then at that wavenumber k we have amplification of the corresponding eigenmode at successive times $n\triangle t$ and the solution will be unstable. This will then give an *amplitude error*. For *stability* therefore we must have:

$$|\xi(k)| \leq 1 \tag{3.11}$$

and for *accuracy* in the solution at wavenumber k we desire $|\xi(k)| = 1$.

What then is the effect of $|\xi(k)| < 1$? This corresponds to damping of part of the solution, which will lead to a loss of accuracy. However, we are only interested in solutions that cover many gridpoints (otherwise the grid is too coarse) i.e. in wavenumbers k s.t. $k\triangle x \ll 1$, and so damping of other modes will not (in general) be a problem.

Example

Let us perform the von Neumann stability analysis for the FTCS discretization of the wave equation. Equation 3.9 now becomes:

$$u_j^{n+1} \approx u_j^n - \frac{v\triangle t}{\triangle x}\frac{u_{j+1}^n - u_{j-1}^n}{2} \tag{3.12}$$

We therefore substitute the general form of an eigenmode (equation 3.10) into this equation and divide by ξ^n to get:

$$\xi(k) \approx 1 - i\frac{v\triangle t}{\triangle x}\sin(k\triangle x) \tag{3.13}$$

and so we see that $|\xi(k)| > 1 \forall k$. Therefore we have shown that the FTCS scheme is unconditionally unstable when applied to the wave equation.

This instability can be cured by modifying equation 3.9 using :

$$u_j^n \to \frac{1}{2}\left(u_{j+1}^n + u_{j-1}^n\right) \tag{3.14}$$

and the resulting scheme is known as the *Lax method*.

3.1.3 Courant condition

Why does the Lax modification make the FTCS scheme stable? The answer is that the original equation required information from u_{j-1}^n to propagate to u_j^{n+1} in every time time-step $\triangle t$, regardless of the velocity v of the wave in the medium! The Lax scheme however now satisfies the *Courant condition*:

$$\frac{|v|\triangle t}{\triangle x} \leq 1 \tag{3.15}$$

which ensures causality, and is therefore another requirement for stability. The Lax modification is numerically equivalent to adding dissipation into the continuum equation 3.3 unless $\frac{|v|\triangle t}{\triangle x} = 1$.

3.1.4 Other errors

Note that there are other sources of error as well as the amplitude errors highlighted by the von Neumann analysis. One example is that of *phase errors*. For example, even if we set $\frac{|v|\triangle t}{\triangle x} = 1$ in the equation 3.1 we get

$$\xi = e^{-ik\triangle x} \tag{3.16}$$

and so at each time-step, the eigenmode solution is multiplied by an arbitrary phase factor. Therefore, an initial wave packet which was a superposition of modes with different k will rapidly disperse and loose its coherence.

There are other possible sources of error in general, e.g. due to non-linear features, shock formation etc. but we shall not consider these here as our primary goal is the study of the (linear) Schrödinger equation!

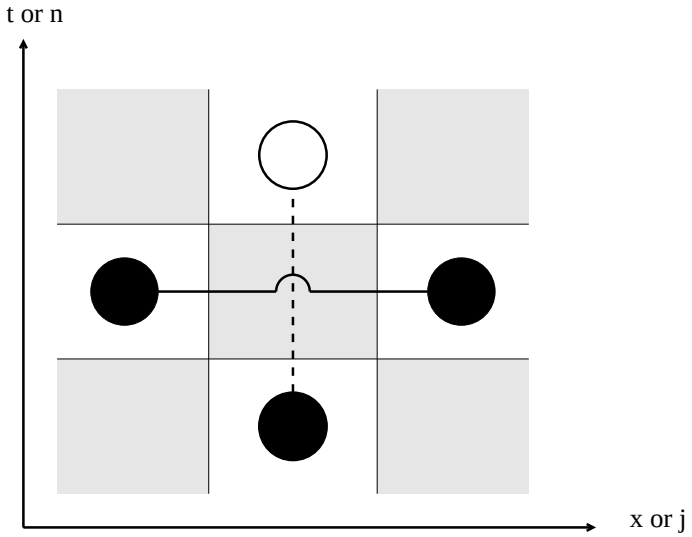

Figure 3.2: Schematic representation of the staggered-leapfrog method. The checkerboard pattern shows that there are two uncoupled meshes in this scheme, which might therefore result in a mesh-drift instability.

3.1.5 Higher order methods

The FTCS method is second-order in space but only first-order in time. It is possible to improve upon this, by going to a method such as *staggered-leapfrog*, which is second-order in time:

$$u_j^{n+1} = u_j^{n-1} - \frac{v\triangle t}{\triangle x}\left(u_{j+1}^n - u_{j-1}^n\right) \tag{3.17}$$

as illustrated in figure 3.2.

In this case, repeating the von Neumann analysis shows that this method has no amplitude dissipation ($|\xi(k)| = 1$) as long as the Courant condition is satisfied, and so is to be preferred. However, we can see from the checkerboard pattern in figure 3.2 that only alternate points are coupled in this scheme, resulting in two distinct meshes. This method may then result in *mesh-drifting*, which can be cured in a similar manner to the Lax method. The result is the *Two-Step Lax-Wendroff* scheme.

3.1.6 Implicit schemes

All the schemes discussed so far are *explicit* schemes, where all the information required to derive the next point is known at the beginning of the next time-step. This is not the only possibility! There also exist *implicit* (also known as *backward time*) schemes which are sometimes useful. For example, in solving the diffusion equation 3.1 we often have a competition

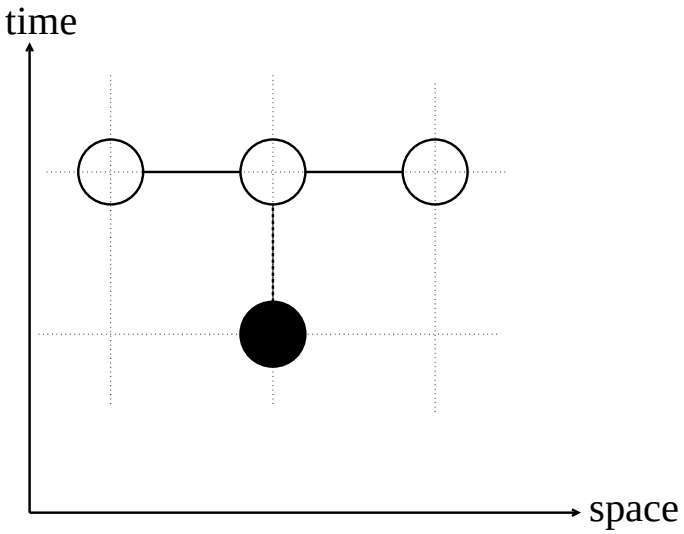

Figure 3.3: Schematic representation of the Backward-Time Centred-Space (BTCS) differencing method.

between behaviour on small and large scales. Using an explicit scheme, the time-step would be dominated by the need to ensure stability of the solution for these small scale features, which are often of secondary importance w.r.t. the large scale features. Therefore, we want a scheme that correctly evolves the large-scale features of interest using as large a time-step as possible, and leaves the small scale features "frozen in". Then at the end of the calculation we can switch to another scheme to finish off the small scale features.

An example of a fully implicit scheme (applied to equation 3.1) is the Backward-Time Centred-Space (BTCS) scheme :

$$\frac{u_j^{n+1} - u_j^n}{\triangle t} = D\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\triangle x)^2} \tag{3.18}$$

Note that with equation 3.1 we have the second-order spatial derivative, which results in the centred term appearing in the spatial derivative. This term is absent in equation 3.7 as we were only considering an effective first-order equation at that time. Both schemes are second-order in space and first-order in time. The BTCS scheme is illustrated in figure 3.3.

Applying the von Neumann stability analysis to the BTCS scheme shows that this is now unconditionally stable for any size time-step. This stability is a characteristic of implicit methods. To solve equation 3.18 requires solving a set of linear simultaneous equations at each time-step for the u_j^{n+1} which therefore increases the computational workload. However, this is a

tridiagonal set of equations,

$$\begin{pmatrix} BC_0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ -\alpha & 1+2\alpha & -\alpha & 0 & 0 & \cdots & 0 \\ 0 & -\alpha & 1+2\alpha & -\alpha & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \vdots & & -\alpha & 1+2\alpha & -\alpha & 0 \\ 0 & 0 & \cdots & 0 & -\alpha & 1+2\alpha & -\alpha \\ 0 & 0 & \cdots & 0 & 0 & 0 & BC_J \end{pmatrix} \begin{pmatrix} u_0^{n+1} \\ u_1^{n+1} \\ \\ \\ \vdots \\ \\ u_{J-1}^{n+1} \\ u_J^{n+1} \end{pmatrix} = \begin{pmatrix} u_0^n \\ u_1^n \\ \\ \\ \vdots \\ \\ u_{J-1}^n \\ u_J^n \end{pmatrix} \tag{3.19}$$

where $\alpha = \frac{D\triangle t}{(\triangle x)^2}$ and BC_0 and BC_J refer to appropriate boundary conditions. This is a *sparse matrix* equation which can be solved very efficiently using special methods.

Finally, to combine the accuracy of a fully second-order scheme with the stability of an implicit method, we average the FTCS and BTCS schemes, resulting in the *Crank-Nicholson scheme.*

3.2 Application to the Schrödinger equation

We start by writing the time dependent Schrödinger equation, in one dimension with atomic units:

$$-\frac{1}{2}\frac{\partial^2\psi}{\partial x^2} + V(x)\,\psi = i\frac{\partial\psi}{\partial t} \tag{3.20}$$

or in terms of the Hamiltonian operator $\widehat{H}$,

$$\widehat{H}\psi = i\frac{\partial\psi}{\partial t} \tag{3.21}$$

and the formal solution to equation 3.21 is:

$$\psi(x,t) = e^{-i\widehat{H}t}\psi(x,0) \tag{3.22}$$

where $e^{-i\widehat{H}t}$ is known as the *time evolution operator.* We may understand such an operator in terms of its power-series expansion:

$$e^{-i\widehat{H}\triangle t} = 1 - i\widehat{H}\triangle t + O\left(\triangle t^2\right)$$

The FTCS algorithm results in

$$\psi_j^{n+1} = \left(1 - i\widehat{H}\triangle t\right)\psi_j^n \tag{3.23}$$

and as expected, the von Neumann analysis with eigenmode solutions as in equation 3.10 shows this to be always unstable for all k at all time-steps.

We therefore consider the implicit BTCS algorithm:

$$\psi_j^{n+1} = \left(1 + i\widehat{H}\triangle t\right)^{-1} \psi_j^n \tag{3.24}$$

which as expected is unconditionally stable. We can then extend this to higher order schemes as before.

So is that it? Do we now have everything we need to solve the time dependent Schrödinger equation? NO!

3.2.1 Constraints

There is an fundamental additional constraint upon solutions to the Schrödinger equation, which is that the solutions must stay normalised at all times, i.e.

$$\int_{-\infty}^{\infty} |\psi|^2 \, dx = 1 \tag{3.25}$$

which none of the methods discussed satisfy. The reason for the failure is that the approximation we have made to the time evolution operator $e^{-i\widehat{H}t}$ is not *unitary*, although the operator is unitary.

Remember that in matrices, a unitary matrix U is one that satisfies $U^\dagger U = 1$, i.e.. the complex generalization of an orthogonal matrix. As such, it can be used as a rotation matrix as it does not change the norm of the matrix it is applied to. Similarly, in quantum mechanics, a unitary operator does not change the normalisation of the wavefunction. The problem is that the time evolution operator is unitary, but our simple first-order Taylor expansion of the operator is not.

The solution is to use *Cayley's form* for the finite-difference representation of the time evolution operator:

$$e^{-i\widehat{H}\triangle t} = \frac{1 - \frac{1}{2}i\widehat{H}\triangle t}{1 + \frac{1}{2}i\widehat{H}\triangle t} + O\left(\triangle t^3\right) \tag{3.26}$$

which is both unitary and also has the additional advantage of being exact to second-order in time.

We can now use this approximation for the time evolution operator in equation 3.24 which results in an unconditionally stable, second-order, unitary algorithm. This is the preferred finite difference method for solving the time dependent Schrödinger equation, and is actually equivalent to the Crank-Nicholson method! Note that the presence of the denominator in equation 3.26 means that a matrix inversion is required at every time step, which therefore dominates the overall computational time scaling of this algorithm. We discretize $\widehat{H}$ using standard spatial derivatives, which results in a tridiagonal matrix for $\widehat{H}$ which is similar to equation 3.19. Again, using special sparse matrix methods, we can efficiently invert this matrix in $O(J)$ operations and hence solve the time dependent Schrödinger equation.

3.3 Final comments

A few final points to highlight:

Remember that a higher-order algorithm is not necessarily better.

Remember the difference between stability and accuracy, and the different sorts of errors that can be introduced by your choice of algorithm (as well as round-off and truncation errors in the computer implementation).

Remember that you may have additional constraints (such as normalisation) which a "standard" algorithm may not handle.

3.4 Further reading

- "Computational Physics" by J.M. Thijssen, Chapter 3