The direct midpoint method as a quantum mechanical integrator III *

Ulrich Mutze †

This talk is about a method for solving the numerical initial value problem of the time-dependent Schrödinger equation with a Hamiltonian which may depend on time. It is only for time independent Hamiltonians that I was able to describe the accuracy and stability properties in the form of proven theorems. All practical experience shows, that the time-dependent case behaves equally well. In this method, the Hamiltonian comes in only as a hermitian linear operator and not as a mathematical expression that can be reordered or otherwise transformed in order to speed up the necessary calculations. In all cases where it is a feasible computational task to apply the Hamilton operator to a generic state vector, my suggestion is to try this method as a workhorse. It can be considered a 'Simpson's rule' for quantum dynamics.

There is no way for me to explain the matter from a technical point of view in 10 minutes, so I'll try to convey the basic idea of the method.

This idea is to treat quantum dynamics in closer analogy to classical dynamics as one usually does. One may argue, that there is a fundamental difference between classical and quantum dynamics since the Schrödinger equation is first order with respect to the time derivative, whereas classical mechanics is inherently second order, since forces determine accelerations and not velocities. This is not so clear a distinction as one might think, since also classical mechanics can be given the first-order form of canonical equations.

I propose the opposite of such an approach, namely to give quantum dynamics a second-order form by simply differentiating the time-dependent Schrödinger equation with respect to time. To be sure, this is only a heuristic device for defining a computationally feasible approximation and not a proposal to change quantum mechanics. For the resulting equation for the second time derivative of the wave function, I then carry over and use the integration methods from the field of *granular matter dynamics* which I have been using in an industrial project concerning simulation of the toning process in electro-photographic copiers/printers over some years [4]. Such methods come in

*Presented at the DPG Physics School 2007 on Quantum Information and Quantum Simulation as a ten minutes blackboard talk

†ulrichmutze@aol.com , former Heidelberger Druckmaschinen AG, former Kodak AG, former LMU Munich, habilitation there 1984 (theoretical physics)

many variants and may be referred to as *midpoint methods*, *leap-frog methods* or *Verlet methods*. A particular useful such variant is defined and studied in detail in [1].

Let us write the time-dependent Schrödinger equation as

$$\dot{\psi}(t) = -\,\mathrm{i}\,H(t)\,\psi(t) =: D(t)\,\psi(t) \tag{1}$$

and ask for a discrete-time state updating algorithm

$$(t,\psi) \;\;\mapsto\;\; (\underline{t},\underline{\psi})\,, \tag{2}$$

where $\underline{t}$ is a point in time close to t. It could be not only later than t (which is the most natural situation) but could also be earlier. Before we see the algorithm, we should understand its intended usage: it allows to construct by iterated application a sequence of states — a time-discrete trajectory — which, if the time-values associated with the trajectory points are sufficiently close together, give an approximate representation of the time-continuous trajectory as determined by (1). To start this trajectory-creating process we need a simple additional step which introduces and initializes an additional state-like quantity, the 'quantum velocity' ϕ which then for all further steps will serve as an efficient mediator between the important algorithmic steps. This being said, it is clear that our algorithm actually is of the form

$$(t,\psi,\phi) \;\;\mapsto\;\; (\underline{t},\underline{\psi},\underline{\phi})\,, \tag{3}$$

with the natural definition of a velocity

$$\phi := D(t)\psi \tag{4}$$

for the time t to which the first state ψ of our initial value problem belongs.

Optimizing formal elegance, I express the algorithm in programming style which allows changing values of quantities without changing their name: Changing the time from t to $\underline{t} = t + h = t + 2\tau$ induces a state change given by

$$\begin{aligned}
t &\mathrel{+}= \tau \\
\psi &\mathrel{+}= \tau\,\phi \\
\phi &\mathrel{+}= h\,D(t)\,D(t)\,\psi + D(t+\tau)\,\psi - D(t-\tau)\,\psi \\
\psi &\mathrel{+}= \tau\,\phi \\
t &\mathrel{+}= \tau\,.
\end{aligned} \tag{5}$$

Here, of course, $a \mathrel{+}= b$ means that a is to be changed to $a+b$ and it is understood that the values of t, ψ, and ϕ at the end of the algorithm just define $(\underline{t},\underline{\psi},\underline{\phi})$ of (3).

Let us now assume that the Hamiltonian does not depend on time. Then (5) simplifies to

$$\begin{aligned}
\psi &\mathrel{+}= \tau\,\phi \\
\phi &\mathrel{+}= h\,D^2\,\psi \\
\psi &\mathrel{+}= \tau\,\phi
\end{aligned} \tag{6}$$

which is a very natural scheme when interpreted in terms of velocity ϕ and acceleration $D^2\psi$. The obvious decomposition of (6) into three steps can be written as a composition of linear maps:

$$\begin{pmatrix} \underline{\psi} \\ \underline{\phi} \end{pmatrix} = \begin{pmatrix} 1 & \frac{h}{2} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ hD^2 & 1 \end{pmatrix} \begin{pmatrix} 1 & \frac{h}{2} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \psi \\ \phi \end{pmatrix} =: U_h \begin{pmatrix} \psi \\ \phi \end{pmatrix} . \tag{7}$$

The basic properties of the method can be expressed as properties of the linear operator U_h defined by (7):

1. reversibility, i.e. $U_h \circ U_{-h} = 1$
2. symplecticity, i.e. U_h commutes with $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$
3. second order method, meaning considered clear from numerical analysis
4. exact energy conservation
5. unitarity up to corrections of order h^2

To define what the exact meaning of the last two items is, we consider a discrete trajectory of constant time step h.

$$\begin{pmatrix} \psi_h^0 \\ \phi_h^0 \end{pmatrix} := \begin{pmatrix} \psi \\ D\psi \end{pmatrix} , \quad \begin{pmatrix} \psi_h^{n+1} \\ \phi_h^{n+1} \end{pmatrix} := U_h \begin{pmatrix} \psi_h^n \\ \phi_h^n \end{pmatrix} , \quad n \in \mathbb{N} , \quad h \in \mathbb{R} . \tag{8}$$

The quantities related to unitarity and to energy conservation are the scalar products $\langle \psi_h^n | \psi_h^n \rangle$ and $\langle \psi_h^n | \phi_h^n \rangle$, which correspond to the constant quantities $\langle \psi(nh) | \psi(nh) \rangle$ and $-\mathrm{i} \langle \psi(nh) | H \psi(nh) \rangle$ in an exact solution of (1), and thus should be approximately constant. As descriptors for the deviations from constancy, I introduce the quantities

$$\nu(\psi, n, h) := \langle \psi_h^n | \psi_h^n \rangle - \langle \psi_h^0 | \psi_h^0 \rangle , \quad \varepsilon(\psi, n, h) := \langle \psi_h^n | \phi_h^n \rangle - \langle \psi_h^0 | \phi_h^0 \rangle . \tag{9}$$

It it clear from (8) that the right-hand sides of these two defining equations are in fact determined by ψ in (8) and n, and h, so that ν and ε are well-defined functions. Strict energy conservation means that the imaginary part of ε is zero (remember that ε is related to matrix elements of $D = -\mathrm{i}H$). This is a direct consequence of symplecticity. Unitarity would imply that ν would be zero. It turns out that ν vanishes only in the limit h towards zero and is of order h^2. The same is true for the real part of ε. These two functions turn out ([2]) to be related by the remarkable (and cryptic) equation

$$nh\,\varepsilon(\psi, n, h) = \left(\frac{h}{2} \frac{\mathrm{d}}{\mathrm{d}h} - 1 \right) \nu(\psi, n, h) . \tag{10}$$

For details see [2], [3]. The basic method of investigation is to use a spectral decomposition of H to derive an explicit representation of the n-th power of U_h.

There is some irony at work here: we make heavy use of the spectral decomposition of H to derive the properties of an integration method that is made to avoid spectral representation (diagonalization) of the Hamiltonian in its own implementation.

In pedagogical computer experiments concerning a single particle in one dimension, the diagonalization of a general Hamiltonian is no problem for, say, 100 spatial discretization points. The need to avoid diagonalization occurred to me when I turned to a particle in two spatial dimensions, or — probably more instructive — to two particles, each living on a linear lattice. If these lattices cross each other, a particularly instructive system ('crossway-system') results, which is studied [2]. Independent from questions of computational burden, the diagonalization method can't handle time dependent Hamiltonians, which very naturally occur when the interaction picture is being employed.

Finally, I would like to warmly recommend the programming language C++ for flexible and fast development (implementation and test) of quantum dynamical algorithms. Ref [2] contains some arguments to support this. Further, I offer to all of you the possibility to use my broad computational physics C++ class system (2500 printed pages of C++ source code). I call this system C+- since it provides within C++ the logic which Bjarne Stroustrup, the creator C++, may have envisioned when he wrote: 'Within C++, there is a much smaller and cleaner language struggling to get out' ([5], p. 207). Please, contact me if you are interested in such stuff.

References

[1] Ulrich Mutze: A Simple Variational Integrator for General Holonomic Mechanical Systems, Mathematical Physics Preprint Archive 2003–491

`www.ma.utexas.edu/mp_arc/c/03/03-491.pdf`

[2] Ulrich Mutze: The direct midpoint method as a quantum mechanical integrator, Mathematical Physics Preprint Archive 2006–356

`www.ma.utexas.edu/mp_arc/c/06/06-356.pdf`

[3] Ulrich Mutze: The direct midpoint method as a quantum mechanical integrator II, Mathematical Physics Preprint Archive 2007–176

`www.ma.utexas.edu/mp_arc/c/07/07-176.pdf`

[4] Ulrich Mutze: Rigidly connected overlapping spherical particles: a versatile grain model, Granular Matter Vol. 8, 185-194, 2006

[5] Bjarne Stroustrup: The Design and Evolution of C++, Addison-Wesley 1994