Quantum Dynamics with Bohmian Trajectories†

D.-A. Deckert* and D. Dürr

Mathematisches Institut der LMU München, Theresienstrasse 39, D-80333 Munich, Germany

P. Pickl

Institut für Theoretische Physik, Universität Wien, Austria

Received: February 12, 2007; In Final Form: May 13, 2007

We describe the advantages and disadvantages of numerical methods when Bohmian trajectory grids are used for numerical simulations of quantum dynamics. We focus on the crucial noncrossing property of Bohmian trajectories, which, numerically, must be given careful attention. Failure to do so causes instabilities or leads to false simulations.

1. Introduction

Bohmian Mechanics. Bohmian mechanics[1] is a mechanical theory for the motion of particles. We describe the theory in units of mass $m = \hbar = 1$. Given the Schrödinger wave function ψ_t of an N-particle system, the trajectories of the N particles $\boldsymbol{q}_k(\boldsymbol{q}_k^0,\psi^0;\ t) \in \mathbb{R}^3$, $1 \le k \le N$, are solutions of

$$\frac{\mathrm{d}\boldsymbol{q}_k(t)}{\mathrm{d}t} = \mathfrak{Im}\left[\frac{\psi_t^*\nabla_{\boldsymbol{q}_k}\psi_t}{\psi_t^*\psi_t}(\boldsymbol{q}_1, ..., \boldsymbol{q}_N)\right] \tag{1}$$

with $\boldsymbol{q}_k(t = 0) = \boldsymbol{q}_k^0$ as the initial conditions and ψ^* denoting the complex conjugate of ψ, so that $\psi^*\psi = |\psi|^2$. [Note: By interpreting $\psi^*\psi$ as an inner product, the generalization to spin wave functions is straightforward.] We denote the configuration point $x = (\boldsymbol{x}_1, ..., \boldsymbol{x}_N) \in \mathbb{R}^{3N}$, $\nabla_x = (\nabla_{\boldsymbol{x}_1}, ..., \nabla_{\boldsymbol{x}_N})$ where $\nabla_{\boldsymbol{x}_k}$ is the gradient with respect to $\boldsymbol{x}_k \in \mathbb{R}^3$. ψ_t is the solution of the Schrödinger equation,

$$i\frac{\mathrm{d}\psi_t(x)}{\mathrm{d}t} = \left(-\frac{\nabla_x^2}{2} + V(x)\right)\psi_t(x) \tag{2}$$

with the initial condition $\psi_{t=0}(x) = \psi_{t=0}(\boldsymbol{x}_1, ..., \boldsymbol{x}_N) = \psi^0(\boldsymbol{x}_1, ..., \boldsymbol{x}_N)$. Using configuration space language, the Bohmian trajectory of an N particle system is an integral curve $q(t) \in \mathbb{R}^{3N}$ of the following velocity field on configuration space:

$$v^\psi(q,t) = \mathfrak{Im}\left[\frac{(\psi_t^*\nabla_q\psi_t)(q)}{(\psi_t^*\psi_t)(q)}\right] \tag{3}$$

i.e.,

$$\frac{\mathrm{d}q(t)}{\mathrm{d}t} = v^\psi(q,t) \tag{4}$$

Under general conditions, one has the global existence and uniqueness of Bohmian trajectories; i.e., the integral curves do not run into nodes of the wave functions and they cannot cross.[2] From this point forward, we shall only discuss the trajectories as integral curves in configuration space. Note that for one particle, the configuration space is equal to physical space; for more than one particle, this is not the case.

The empirical import of Bohmian mechanics results from equivariance of the $|\psi|^2$ measure: One readily sees that, by virtue of eq 3, the continuity equation for the Bohmian flow on configuration space is identically fulfilled by the density, $\rho_t = |\psi_t|^2$, then known as the quantum flux equation,

$$\frac{\partial|\psi(x,t)|^2}{\partial t} + \nabla_x\cdot(|\psi(x,t)|^2 v^\psi(x,t)) = 0$$

This means that, if the configuration of particles $q^0 = (\boldsymbol{q}_1^0, ..., \boldsymbol{q}_N^0)$ is distributed according to $\rho_0 = |\psi^0(\boldsymbol{x}_1, ..., \boldsymbol{x}_N)|^2$ at time $t = 0$, then the configuration $q(t) = (\boldsymbol{q}_1(\boldsymbol{q}_1^0,\psi^0;\ t), ..., \boldsymbol{q}_N(\boldsymbol{q}_N^0,\psi^0;\ t))$ is distributed according to $\rho_t = |\psi_t(\boldsymbol{x}_1, ..., \boldsymbol{x}_N)|^2$ at any time t. Because of this observation, Bohmian mechanics agrees with all predictions made by orthodox quantum mechanics whenever the latter are unambiguous.[3,4]

Hydrodynamic Formulation of Bohmian Mechanics. Write ψ in the Euler form:

$$\psi(x,t) = R(x,t)\mathrm{e}^{iS(x,t)}$$

where R and S are given by real-valued functions. Equation 3, together with eq 2, separated in their real and complex parts, gives the following set of differential equations:

$$\frac{\mathrm{d}q}{\mathrm{d}t} = \nabla_q S(q,\ t)$$

$$\frac{\mathrm{d}R(x,t)}{\mathrm{d}t} = -\frac{1}{2}\nabla_x(R(x,\ t)\nabla_x S(x,\ t))$$

$$\frac{\mathrm{d}S(x,t)}{\mathrm{d}t} = -\frac{1}{2}(\nabla_x S(x,t))^2 - V(x) + \frac{1}{2}\frac{\nabla_x^2 R(x,t)}{R(x,t)}$$

* Author to whom correspondence should be addressed. Tel.: 49 (0)89 2180 4394. Fax: 49 (0)89 2180 4466. E-mail address: dirk.deckert@mathematik.uni-muenchen.de.

† Part of the special issue "Robert E. Wyatt Festschrift".

10.1021/jp0711996 CCC: $37.00
Published on Web 07/13/2007

This set of equations can be examined along the Bohmian trajectory $q(q^0, \psi^0; t)$. We then obtain

$$\frac{dq}{dt} = \nabla_q S(q,t) \tag{5}$$

$$\frac{\partial R(q,t)}{\partial t} = -\frac{1}{2} R(q,t) \nabla_q{}^2 S(q,t) \tag{6}$$

$$\frac{\partial S(q,t)}{\partial t} = \frac{1}{2}\left(\frac{dq}{dt}\right)^2 - V(q) + \frac{1}{2}\frac{\nabla_q{}^2 R(q,t)}{R(q,t)} \tag{7}$$

Numerical Integration. Numerically, this set of equations (eqs 5−7) can readily be integrated[5] and offers some advantages over standard techniques for solving the Schrödinger equation (eq 2) numerically. The basic idea, according to Wyatt,[5] is that the Bohmian configuration space trajectories define a co-moving grid (Bohmian grid) in configuration space, which is best adapted for the computation of ψ. If initially the grid points (n points in $\mathbb{R}^{3N}$) are $|\psi^0|^2$ distributed, they will remain $|\psi_t|^2$ distributed for all times t. Thus, the co-moving grid spreads dynamically, according to the spreading of $|\psi_t|^2$ and the grid points will primarily remain in regions of space where $|\psi_t|^2$ is large while avoiding regions of nodes or tails of $|\psi_t|^2$, which are numerically problematic. Therefore, Wyatt's idea is this: Integrate the eqs 5−7 *simultaneously*, i.e., get the best-adapted moving grid along with ψ, instead of integrating the Schrödinger equation on some fixed or otherwise determined grid. This is why the algorithm is particularly interesting for long-time simulations, which usually demand huge computational effort on fixed grids. Therefore, if one aims at the relevant parts of the wave function (where probabilities are high), one can use a fixed number n of $|\psi|^2$ distributed Bohmian grid points in $\mathbb{R}^{3N}$, so that the Bohmian grid simulation scales with the number N of particles,[5] while conventional grid methods mostly scale exponentially with N. Section 3 has greater discussion on how to distribute n grid points in a $|\psi|^2$ manner.

To perform the numerical integration of the set of differential equations given as eqs 5−7, we follow the straightforward method described by Wyatt.[5] The only crucial part in this methodology is computing the derivatives involved in the set of differential equations. Several techniques are known, and Wyatt's work[5] gives a comprehensive overview. Among them, one technique—called least-squares fitting—is commonly used. In this letter, we argue that this method is inappropriate for integrating eqs 5−7 for general initial conditions and potentials. Bad situations occur whenever Bohmian trajectories move toward each other, because least-squares fitting will allow crossings of the simulated trajectories that are not allowed for Bohmian trajectories. When a crossing is encountered in a numerical simulation, further computation can be aborted, because this numerical wave function would differ immensely from the solution of the Schrödinger equation (eq 2). This will happen generically, i.e., for non-Gaussian wave functions. To illustrate our argument, we shall present a typical numerical example in one dimension with one particle. Therefore, in what follows, $N = 1$ and the n trajectories that we shall consider (making up the grid) are the possible trajectories of this one particle only.

2. Least-Squares Fitting versus Polynomial Fitting

To compute the derivatives encountered in eqs 5−7, we only consider two different types of fitting: the least-squares fitting and the polynomial fitting. Most other fitting algorithms are descendants of one or the other. Both provide an algorithm for finding a polynomial (or more general an element of a m dimensional vector space of functions; the coefficients of the design matrix X determine which basis functions are used) of degree, here $(m - 1)$, which is, in some sense to be specified, close to a given function f: $\mathbb{R} \to \mathbb{R}$, known only on a set of data points, $(x_i, f(x_i))_{1\le i\le n}$, for pairwise distinct $x_{i,1\le i\le n}$. The derivative can then be computed from the fitting polynomial by algebraic means. For the further discussion, let

$$y := (f(x_i))_{1\le i\le n}$$

$$X := (x_i^{(j-1)})_{1\le i\le n, 1\le j\le m}$$

$$a = (a_i)_{1\le i\le m} \in \mathbb{R}^m$$

$$\delta = (\delta_i)_{1\le i\le n} \in \mathbb{R}^n$$

Using this notation, the problem of finding a fitting polynomial to f, on the basis of n pairwise distinct data points of the graph of f, reduces to finding the coefficients of the vector a that obeys the equation

$$y = X\cdot a + \delta$$

such that the error term δ is, in some sense, small. Here, the center dot ($\cdot$) denotes matrix multiplication. The fitting polynomial is then given by

$$p(x) = \sum_{j=1}^{m} a_j x^{j-1}$$

Polynomial Fitting. For the case of $n = m$, we can choose the error term δ to be identical zero, because X is an invertible square matrix and $a = X^{-1}\cdot y$ can be computed in a straightforward manner.

Least-Square Fitting. For arbitrary $n \ge m$, there is no unique solution anymore and one needs a new criterion to find a unique vector a. Therefore, the algorithm of least-squares fitting uses the minimum value of the accumulated error

$$\Delta := \sum_{i=1}^{n} w(x - x_i)\delta_i^2$$

where $w: \mathbb{R} \to \mathbb{R}$ specifies a weight dependent on the distance between the ith data point x_i and some point x where the fitting polynomial shall be evaluated. The vector a can now be determined by minimizing Δ as a function of a, by solving

$$\left(\frac{\partial \Delta(a)}{\partial a_j}\right)_{1\le j\le m} = \left(-2\sum_{i=1}^{n} w(x - x_i)(y - X\cdot a)_i x_i^{j-1}\right)_{1\le j\le m} = 0$$

Note that, for $m = n$ and any nonzero and positive weight w, the algorithm of least-squares fitting produces the same a as the polynomial fitting would, because, for the a determined by polynomial fitting, the non-negative function $\Delta(a)$ is zero and, thus, a naturally minimizes the error term.

Why Least-Squares Fitting is Inappropriate for Bohmian Grids. The algorithm of least-squares fitting is well-known for its tendency to stabilize numerical simulations by averaging out numerical errors. However, this averaging makes it difficult to keep the grid points from crossing each other. To understand what happens during a numerical simulation, recall the form of the equations of motion (eqs 5−7), which must be integrated step by step. The change in time of the phase S is determined by three terms in eq 7. The first two terms form the classical

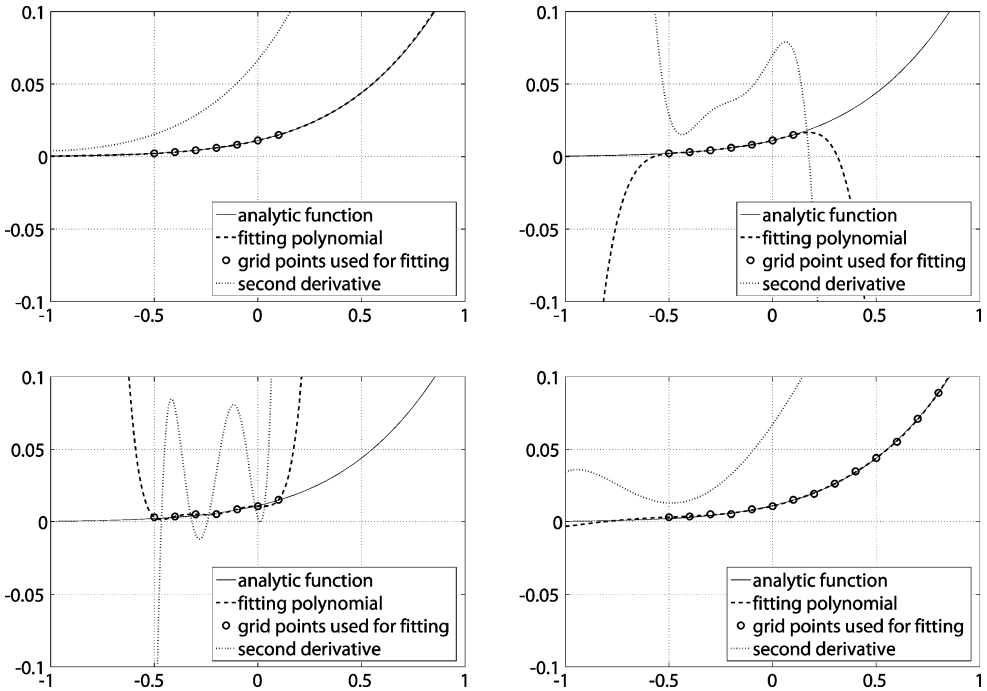

Figure 1. Fitting of a sixth-degree polynomial and its second derivative near the boundary. In the upper left-hand panel, the function to fit smoothly decays; both least-squares fitting and polynomial fitting give identical fitting polynomials mimicking this decay. In the upper right-hand panel, polynomial fitting was used while the function value at the fourth grid point was shifted upward by 10^{-4} (not visible in plot) to simulate a numerical error, to examine how sensitive polynomial fitting reacts to such an error. In the lower left-hand panel, all values at the grid points were shifted by an individual random amount of the order of 10^{-3} and polynomial fitting was used. In the lower right-hand panel, all values at the grid points were shifted by the same random numbers as those in the lower left-hand panel, while the additional grid points were also shifted by individual random amounts of the order of 10^{-3} and least-squares fitting was used. By comparison with the plot shown in the upper left-hand panel, one observes the robustness of least-squares fitting to such numerical errors at the boundary.

Lagrangian and are dependent on the current velocities of the grid points and on the potential *V*. The third term is the so-called quantum potential and is dependent only on *R* and its second derivative. This quantum potential is what prevents Bohmian trajectories from crossing each other. Therefore, it requires special attention during the numerical integration. Now imagine initial conditions such that two grid points approach each other. An increase of the density of the grid points in a region where the two grid points move toward each other will cause a small bump in *R*. By "small bump", we mean a bump of the wave function shape on a microscopic scale, i.e., a scale defined by few grid points. Such a small bump of *R* may have large derivatives changing the quantum potential in eq 7. Thus, it is absolutely vital for a numerical simulation to implement a fitting algorithm that reconstructs not only *R*, respectively *S*, in an accurate way but also its second derivative. The tendency of the least-squares fitting algorithm is to average out those small bumps in *R*, respectively *S*. Hence, the simulation is blind to recognize grid points moving toward each other and, hence, does not prevent a crossing of these trajectories.

Note that, because the averaging occurs on a microscopic scale, situations in which grid points move only very slowly or do generically not move toward each other are often numerically doable with least-squares fitting. [For example, a free Gaussian wave packet or one approaching a potential, creating only soft reflections.] On the other hand, numerical simulations with least-squares fitting in general situations (such as the one just discussed) are bound to break down as soon as two grid points get too close to each other.

Why Polynomial Fitting is More Appropriate for Bohmian Grids. We previously stressed that small bumps may increase numerical instability exponentially. A good numerical method must take note of the small bumps and prevent their increase. Such a numerical method is provided by polynomial fitting, because, there, the polynomials go through all grid points. Therefore, polynomial fitting recognizes the bumps of *R* and/or *S* and, hence, the resulting quantum potential recognizes the approaching grid points. Now recall the physics of the Bohmian evolution, which, as we stressed in the introduction, prevents the trajectories from crossing each other. Therefore, we expect that this method is self-correcting and, hence, stabilizing.

The Boundary Problem. It must be remarked that polynomial fitting creates a more severe problem at the boundary of the supporting grid than does least-squares fitting. This problem is of the conceptual type, because, at the boundary, there is a generic lack of knowledge of how the derivatives of *R* or *S* behave (see Figure 1).

We describe briefly how this conceptual problem can cause severe numerical instability. For this, suppose that only the last grid point in the upper left plot of Figure 1 is lifted a little bit upward by, e.g., some numerical error. The resulting quantum potential then will cause the last grid point to move toward the second-to-last one. This increases the density of grid points and, thus, again, increases *R* in the next step such that this effect is self-amplifying (in fact, growing superexponentially) and will effect the entire wave function quickly.

At the boundary, the good property of polynomial fitting—namely, to recognize all small bumps—works against Bohmian grids techniques. In contrast, least-squares fitting simply averages these small numerical errors out (see the lower right plot in Figure 1). However, the conceptual boundary problem remains and will eventually manifest also with least-squares fitting.

The Numerical Simulation. To demonstrate our argument, we shall now give a numerical example for one particle in one

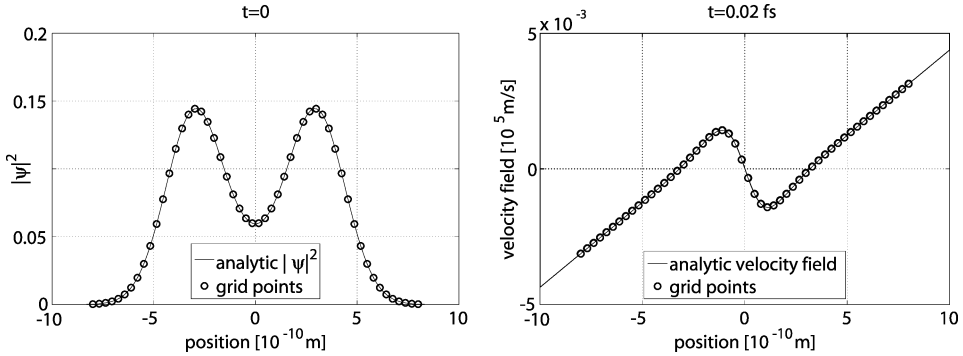

Figure 2. Initial wave function at time zero (left), and its velocity field after a very short time t (right).

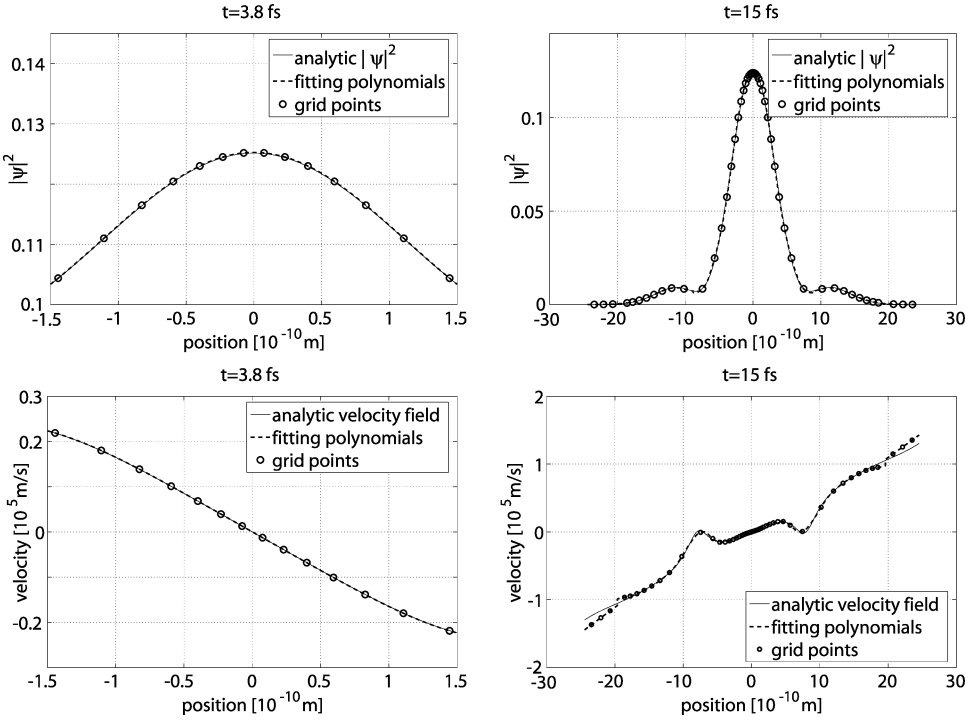

Figure 3. Simulated $|\psi|^2$, together with the velocity field at times $t = 3.8$ fs and $t = 15$ fs, using polynomial fitting between the boundaries. Panels on the left-hand side show the center part of the wave function, where grid points move toward each other. Panels on the right-hand side show the entire wave function at a later time. The grid points have all turned and move apart from each other. The kinks at $\pm 20 \times 10^{-15}$ m in the velocity field in the lower right plot are due to the transition from least-squares fitting at the boundary to polynomial fitting between the boundaries (see "The Boundary Problem" subsection in section 2 of the text).

dimension. We remark, as mentioned previously, that Gaussian wave packets are unfit for probing the quality of the numerical simulations. Therefore, we take, as initial conditions, two superposed free Gaussian wave packets with a small displacement, together with a velocity field identically to zero, and focus the attention on the region where their tails meet, i.e., where the Bohmian trajectories will move toward each other, according to the spreading of the wave packets (see Figure 2). The physical units given in the figures and the following discussion refer to a wave packet on a length scale of 1 Å $= 10^{-10}$ m and a Bohmian particle with the mass of an electron.

The numerical simulation is implemented as described in the work by Wyatt[5] with minor changes. It is written in MATLAB, using IEEE Standard 754 double precision. To ease the boundary problem, we use a very strong least-squares fitting at the boundary. Note that this only eases the boundary problem and is by no means a proper cure. Between the boundaries, however, the number of data points for the fitting algorithm and the degree of the fitting polynomial can be chosen in each run. In this way, it is possible to have either polynomial fitting or least-squares fitting between the boundaries. The simulation is run twice: first with the polynomial fitting, and then with the least-squares fitting algorithm between the boundaries.

We take seven basis functions for the fitting polynomial in both cases. Note that the choice should at least be greater or equal to 4 to have enough information about the third derivative of the fitting polynomial. In the first run, with polynomial fitting, the number of grid points used for fitting is seven and in the second run, for the least-squares fitting, we choose nine, which induces a mild least-squares behavior. For the numerical integration, we have used a time step of 10^{-2} fs with the total number of 51 grid points supporting the initial wave function. The source code is available in our previous work.[6]

Results. The first run with polynomial fitting yields accurate results and does not allow for trajectory crossing way beyond 5000 integration steps, i.e., 50 fs (see Figures 3 and 4). The second run with least-squares fitting reports a crossing of trajectories already after 430 integration steps, i.e., 4.3 fs, and aborts (see Figures 5 and 6). The numerical instability can already be observed earlier (see the left-hand side of Figure 5). An adjustment of the time step of the numerical integration does not lead to better results in the second run. Of course, the

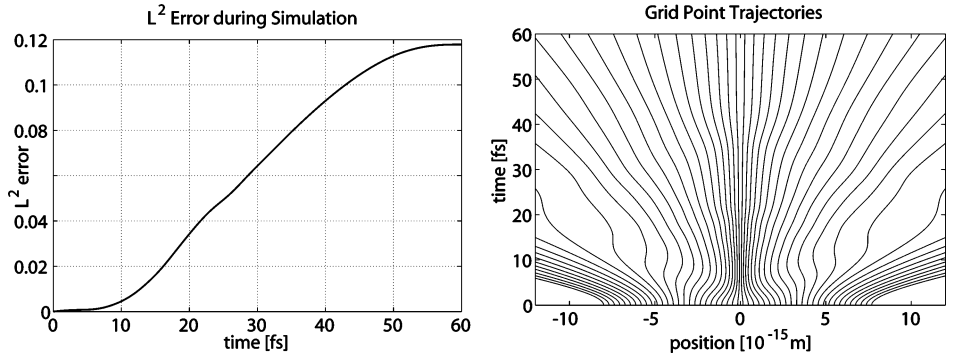

Figure 4. Both plots belong to the simulation using polynomial fitting between the boundaries. The panel on the left-hand side indicates the distance L^2 between the simulated wavefunction and the analytic solution of the Schrödinger equation (norm equal one), and the panel on the right-hand side shows a plot of the trajectories of the grid points. Note how some trajectories initially move toward each other, decelerate, and finally move apart.

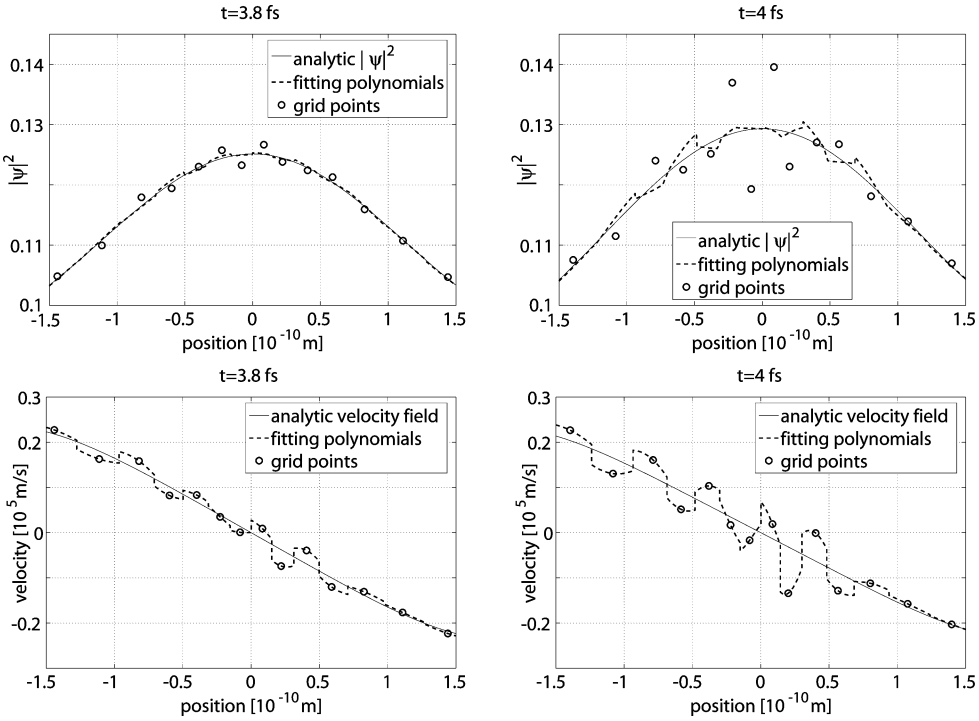

Figure 5. The left-hand side is the same as the left-hand side of Figure 3 but using least-squares fitting throughout. The right-hand side shows the same a short time later. Note, in the upper left plot, that the fitting polynomials of least-squares fitting fail to recognize the relatively big ordinate change of the grid points. Therefore, the grid points that are moving toward each other are not decelerated by the quantum potential. Finally, after $t = 4.3$ fs, a crossing of the trajectory occurs.

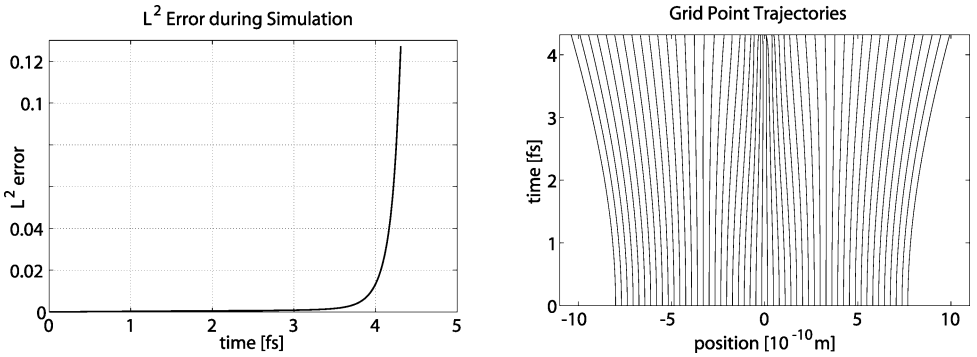

Figure 6. Both plots belong to the simulation using least-squares fitting throughout. The left-hand panel shows the distance L^2 between the simulated wavefunction and the analytic solution of the Schrödinger equation (norm equal one). The right-hand panel shows a plot of the trajectories of the grid points. Note, by comparison with the right-hand side of Figure 4, the failure of least-squares fitting to recognize grid point trajectories moving toward each other until they finally cross and the numerical simulation aborts.

crossing of the trajectories occurs exactly in the region where the grid points move fastest toward each other. Referring to the previous discussion, the small bumps in R created by the increase of the grid point density in this region are not recognized by the least-squares algorithm and, thus, are not recognized by the numerical integration of eqs 5−7 (compare

Figures 3 and 5). The approaching grid points are not decelerated by the quantum potential and finally cross each other (see Figure 5), whereas in the first run, they begin to decelerate and turn to the opposite direction during the time between 3 fs and 5 fs (see Figure 3). To visualize how the two algorithms "see" R^2 and the velocity field during numerical integration, these entities have been plotted in Figures 3 and 5 by merging all fitting polynomials in the neighborhood of every grid point together (from halfway to the left neighboring grid point to halfway to the right neighboring grid point). In Figure 5, one clearly observes the failure of the least-squares fitting algorithm to detect the small bumps in R.

Note that, in some special situations in which the time step of the numerical integration is chosen to be large, polynomial fitting may lead to trajectory crossing as well. This may happen when the number of time steps in which the simulation must decelerate two fast-approaching grid points is not sufficient. This effect is entirely due to the fact that the numerical integration uses discretized time. The choice of a smaller time step for the simulation will always remedy the problem, as long as other numerical errors do not accumulate too much.

The relevant measure of quality of the numerical simulation is naturally the L^2 distance between the simulated wavefunction Re^{iS} and the analytic solution of the Schrödinger equation ψ_t, i.e., $(\int dx|\psi_t(x) - R(x,t)e^{iS(x,t)}|^2)^{1/2}$, and is spelled out on the left-hand side of Figures 4 and 6 for both runs.

3. Using $|\psi^0|^2$ as the Initial Distribution

As discussed in the introduction, if the grid points are distributed according to $|\psi^0|^2$, they will remain so for all times. Doing so will bring two advantages but at the price of a more-severe conceptual boundary problem, as discussed in the last section. The first advantage is that this Bohmian grid avoids regions where R is very small and, thus, where the computation of the quantum potential becomes numerically tricky. The second advantage is that now the system is overdetermined and the actual density of grid points must coincide with R^2 for all times. This could be used as an on-the-fly check to determine whether the grid points behave as Bohmian trajectories or not. If not, the numerical integration does not give a good approximation to the solution of eqs 5−7. It could also be considered to stabilize the numerical simulation with a feedback mechanism balancing R^2 and the distribution of the particle positions, which may correct numerical errors of one or the other on the fly.

One way of choosing $|\psi^0|^2$ distributed initial positions for the grid points q_j^0 for $j = 1, ..., n$ for some large number n can be described as follows. Choose q_j^0 in such a way that

$$|\psi^0(q_j^0)|^2 \frac{q_{j+1}^0 - q_{j-1}^0}{2} = \frac{1}{n} \tag{8}$$

This formula can be used to compute the grid points iteratively, starting with some grid point near the maximum of $|\psi(q,0)|^2$. [This procedure of setting the initial grid points might run into a node of ψ. In that case one should simply start with a slightly shifted initial grid point.] Another way is to simulate $|\psi^0|^2$ distributed random variables q_j^0 via the commonly used techniques. Here, however, one must be careful that the randomly chosen q_j^0 do not lie too close together. If they do, either some of them must be deleted or the time step of the numerical integration must be adjusted carefully. A rule of thumb is that the distance of approaching grid points divided by their relative velocity should be much smaller than the chosen time step.

4. Conclusion

The use of polynomial fitting instead of least-squares fitting between the boundaries increases the stability of the numerical integration immensely. This is due to the fact that polynomial fitting, in contrary to least-squares fitting, does not average over the microscopic structure of the function to fit and therefore reconstructs the needed derivatives more accurately. The discussed boundary problem is more severe for polynomial fitting because of this fact. However, this problem is generic to the numerical integration considered here and also occurs using least-squares fitting but in a milder way. We suggest that the smoothness of the decay of the wave function near the boundary should be the guide for solving this problem, which requires a detailed study. Furthermore, we have discussed that the Bohmian grid is best-adapted to the problem of numerical integration of eqs 5−7, because the grid points naturally avoid regions where R becomes very small.

Acknowledgment. We thank Bob Wyatt for acquainting us with the Bohmian grid methods in theoretical chemistry. We further thank, in alphabetical order, Lara Hernando, Salvador Miret-Artes, Angel Sanz, and Clemens Woywod for discussions. This work was partly supported by DFG. Author D.-A.D. would like to cordially thank R.-R. Deckert and W. E. Göhde-Deckert for funding and support.

References and Notes

(1) Goldstein, S. Bohmian Mechanics. In *Stanford Encyclopedia of Philosophy*; Zalta, E. N., Ed.; Center for the Study of Language and Information, Standford University, Stanford, CA, 2001 (available via the Internet at the following URL: http://plato.stanford.edu/entries/qm-bohm/).

(2) Teufel, S.; Tumulka, R. Simple proof for global existence of bohmian trajectories. *Commun. Math. Phys.* **2005**, *258*, 349−365.

(3) Dürr, D.; Goldstein, S.; Zanghì, N. Quantum Equilibrium and the Origin of Absolute Uncertainty. *J. Stat. Phys.* **1992**, 67, 843−907.

(4) Dürr, D.; Goldstein, S.; Zanghì, N. Quantum equilibrium and the role of operators as observables in quantum theory. *J. Stat. Phys.* **2004**, *116*, 959−1055.

(5) Wyatt, R. E. *Quantum Dynamics with Trajectories*; Springer: Berlin, 2005.

(6) Deckert, D.-A.; Dürr, D.; Pickl, P. *Quantum Dynamics with Bohmian Trajectories*; 2007; 17 pp (available via the Internet at the following URL: http://arXiv.org/abs/quant-ph/0701190v2).