

A fast eikonal equation solver using the Schrödinger wave equation

Karthik S. Gurumoorthy^{a,*}, Anand Rangarajan^a

^a*Department of Computer and Information Science and Engineering, University of Florida, Gainesville, Florida, USA*

Abstract

We use a Schrödinger wave equation formalism to solve the eikonal equation. We show that a solution to the eikonal equation is obtained in the limit as Planck's constant \hbar (treated as a free parameter) tends to zero of the solution to the corresponding linear Schrödinger equation. The Schrödinger equation corresponding to the eikonal turns out to be a *generalized, screened Poisson equation*. Despite being linear, it does not have a closed-form solution for arbitrary forcing functions. We use a standard perturbation analysis approach to derive a new algorithm which is guaranteed to converge provided the forcing function is bounded and positive. The perturbation technique requires a sequence of discrete convolutions which can be performed in $O(N \log N)$ using the Fast Fourier Transform (FFT) where N is the number of grid points. For the Euclidean distance function problem—a special case of the eikonal equation where the forcing function is everywhere identically equal to one—a major advantage of our approach over most other methods is that we do not require a spatial discretization of gradient operators and this contributes to the increased accuracy of our technique. The solution to the eikonal solution is recovered from the exponent of the wave function. Since the wave function is computed for a small but non-zero \hbar , the obtained solution is an approximation. We provide evidence for the usefulness of our technique by comparing the results of our approach with those obtained from popular Hamilton-Jacobi solvers such as the fast sweeping algorithm as well as with

*Corresponding Author

Address:

E301, CSE Building, University of Florida,
P.O. Box 116120, Gainesville, FL 32611-6120, USA.
Ph: 001-352-392-1200, Fax: 001-352-392-1220

Email addresses: ksg@cise.ufl.edu (Karthik S. Gurumoorthy),
anand@cise.ufl.edu (Anand Rangarajan)

exact solutions (when available). The latter can be easily computed for the Euclidean distance function problem.

Keywords: eikonal equation, Schrödinger wave equation, perturbation theory, Fast Fourier Transform (FFT), screened Poisson equation, Green's function

1. Introduction

The *eikonal* (from the Greek word *εικον* or “image”) equation is traditionally encountered in the wave and geometric optics literature where the principal concern is the propagation of light rays in an inhomogeneous medium [1]. Its twin roots are in wave propagation theory and in geometric optics. In wave propagation theory, it is obtained when the wave is approximated using the Wentzel–Kramers–Brillouin (WKB) approximation [2]. In geometric optics, it can be derived from Huygen’s principle [3]. In the present day, the eikonal equation has outgrown its humble optics origins and now finds application in far flung areas such as electromagnetics [2], robot motion path planning [4] and image analysis [5].

The eikonal equation is a nonlinear, first order, partial differential equation [6] of the form

$$\|\nabla S(X)\| = f(X), X \in \Omega \tag{1}$$

subject to the boundary condition $S|_{\partial\Omega} = U(X)$, where Ω is an open subset of \mathbb{R}^D . The forcing function $f(X)$ is a positive valued function and ∇ denotes the gradient operator. In the special case where $f(X)$ equals one everywhere, the solution to the eikonal equation is the Euclidean distance function [5]. Detailed discussions on the existence and uniqueness of the solution can be found in [7].

While the eikonal equation is venerable and classical, it is only in the last twenty years that we have seen the advent of numerical methods aimed at solving this problem. To name a few are the pioneering fast marching [8] and fast sweeping [9] methods. Algorithms based on discrete structures such as the well known Dijkstra single source shortest path algorithm [10] can also be adapted to solve this problem. When we seek solutions on a discretized spatial grid with N points, the complexity of the fast marching method is $O(N \log N)$ while that of the fast sweeping method is $O(N)$ and therefore both of these efficient algorithms have seen widespread use since their inception. Recently, the ingenious work of Sapiro *et al.* provided an $O(N)$ implementation of the fast marching method [11]. Typically, eikonal

solvers grow the solution from a set of K seed points at which the solution is known.

The eikonal equation can also be derived from a variational principle, namely, Fermat’s principle of least time which states that “Nature always acts by the shortest paths” [12]. From this variational principle, the theoretical physics developmental sequence proceeds as follows: The first order Hamilton’s equations of motion are derived using a Legendre transformation of the variational problem wherein new momentum variables are introduced. Subsequently, a canonical transformation converts the time varying momenta into constants of the motion. The Hamilton-Jacobi equation emerges from the canonical transformation [13]. In the Hamilton-Jacobi formalism specialized to the eikonal problem, we seek a surface $S(X, t)$ such that its increments are proportional to the speed of the light rays. This is closely related to Huygen’s principle and thus marks the *rapprochement* between geometric and wave optics [3]. It is this nexus that drives numerical analysis methods [8, 9] (focused on solving the eikonal equation) to base their solutions around the Hamilton-Jacobi formalism.

So far, our development has followed that of classical physics. Since the advent of quantum theory—specifically the Schrödinger wave equation—in the 1920s, the close relationship between the Schrödinger and Hamilton-Jacobi equations has been intensely studied [14]. Of particular importance here is the quantum to classical transition as $\hbar \rightarrow 0$ where the nonlinear Hamilton-Jacobi equation emerges from the phase of the Schrödinger wave equation. This relationship has found very few applications in the numerical analysis literature despite being well known. In this paper, we leverage the important distinction between the Schrödinger and Hamilton-Jacobi equations, namely, that the former is *linear* whereas the latter is not. We take advantage of the linearity of the Schrödinger equation while exploiting its relationship to Hamilton-Jacobi and derive computationally efficient solutions to the eikonal equation.

A time-independent Schrödinger wave equation at the energy state E has the form $\hat{H}\phi(X) = E\phi(X)$ [15], where $\phi(X)$ —the stationary state wave function—is the solution to the time-independent wave equation and \hat{H} is the Hamiltonian operator. When the Hamilton-Jacobi scalar field S^* is the exponent of the stationary state wave function, specifically $\phi(X) = \exp(\frac{-S^*(X)}{\hbar})$, and if $\phi(X)$ satisfies the Schrödinger equation, we show that as $\hbar \rightarrow 0$, S^* satisfies the Hamilton-Jacobi equation. Note that in the above, a nonlinear Hamilton-Jacobi equation is obtained in the limit as $\hbar \rightarrow 0$ of a linear Schrödinger equation which is novel from a numerical

analysis perspective. Consequently, instead of solving the Hamilton-Jacobi equation, one can solve its Schrödinger counterpart (taking advantage of its linearity), and compute an approximate S^* for a suitably small value of \hbar . This computational procedure is approximately equivalent to solving the original Hamilton-Jacobi equation.

Since the efficient solution of a linear wave equation is the cornerstone of our approach, we now briefly describe the actual computational algorithm used. We derive the static Schrödinger equation for the eikonal problem. The result is a generalized, *screened Poisson* equation [16] whose solution is known at K seed points. This linear equation does not have a closed-form solution and therefore we resort to a perturbation method [17] of solution—which is related to the Born expansion [18]. The perturbation method comprises a sequence of multiplications with a space-varying forcing function followed by convolutions with a Green’s function (for the screened Poisson operator) which we solve using an efficient $[O(N \log N)]$ fast Fourier transform (FFT)-based technique [19, 20]. Perturbation analysis involves a geometric series approximation for which we show convergence for all bounded forcing functions independent of the value of \hbar .

The paper is organized as follows. In section (2), we provide the Hamilton-Jacobi formulation for the eikonal equation as adopted by the fast sweeping and fast marching methods. We restrict to the special case of the eikonal equations involving constant forcing functions in section (3) and derive its corresponding Schrödinger wave equation. Section (4) considers the more general version, where we derive and provide an efficient *arbitrary precision* FFT-based method for solving the Schrödinger equation using techniques from perturbation theory. In section 5, we provide results and comparisons (to fast sweeping and the exact solution) where we differentiate between experiments focused on the Euclidean distance function problem (where the forcing function is a constant) and experiments devoted to the general eikonal equation. We conclude in section (6).

2. Hamilton-Jacobi formulation for the eikonal equation

2.1. Fermat’s principle of least time

It is well known that the Hamilton-Jacobi equation formalism for the eikonal equation can be obtained by considering a variational problem based on Fermat’s principle of least time [3] which in $2D$ is

$$I[q] = \int_{t_0}^{t_1} f(q_1, q_2, t) \sqrt{1 + \dot{q}_1^2 + \dot{q}_2^2} dt. \quad (2)$$

We take an idiosyncratic approach to the eikonal equation by considering a different variational problem which is still very similar to Fermat's least time principle. The advantage of this variational formulation is that the corresponding Schrödinger wave equation can be easily obtained.

Consider the following variational problem namely,

$$I[q] = \int_{t_o}^{t_1} \frac{1}{2} (\dot{q}_1^2 + \dot{q}_2^2) f^2(q_1, q_2) dt \quad (3)$$

where the forcing term f is assumed to be independent of time and the Lagrangian L is defined as

$$L(q_1, q_2, \dot{q}_1, \dot{q}_2, t) \equiv \frac{1}{2} (\dot{q}_1^2 + \dot{q}_2^2) f^2(q_1, q_2). \quad (4)$$

Defining

$$p_i \equiv \frac{\partial L}{\partial \dot{q}_i} = f^2(q_1, q_2) \dot{q}_i \quad (5)$$

and applying the Legendre transformation [3], we can obtain the Hamiltonian of the system in $2D$ as

$$H(q_1, q_2, p_1, p_2, t) = \frac{1}{2} \frac{(p_1^2 + p_2^2)}{f^2(q_1, q_2)}. \quad (6)$$

From a canonical transformation of the Hamiltonian [13], we obtain the following Hamilton-Jacobi equation

$$\frac{\partial S}{\partial t} + \frac{1}{2} \frac{\left(\frac{\partial S}{\partial q_1}\right)^2 + \left(\frac{\partial S}{\partial q_2}\right)^2}{f^2(q_1, q_2)} = 0 \quad (7)$$

Since the Hamiltonian in (6) is a constant *independent* of time, equation (7) can be simplified to the static Hamilton-Jacobi equation. By separation of variables, we get

$$S(q_1, q_2, t) = S^*(q_1, q_2) - Et \quad (8)$$

where E is the total energy of the system and $S^*(q_1, q_2)$ is called Hamilton's characteristic function [3]. Observing that $\frac{\partial S}{\partial q_i} = \frac{\partial S^*}{\partial q_i}$, equation (7) can be rewritten as

$$\frac{1}{2} \left[\left(\frac{\partial S^*}{\partial q_1}\right)^2 + \left(\frac{\partial S^*}{\partial q_2}\right)^2 \right] = Ef^2. \quad (9)$$

Choosing the energy E to be $\frac{1}{2}$, we obtain

$$\| \nabla S^* \|^2 = f^2 \quad (10)$$

which is the original eikonal equation (1). S^* is the required Hamilton-Jacobi scalar field which is efficiently obtained by the fast sweeping [9] and fast marching methods [8].

3. Eikonal equations with constant forcing functions

We begin the quantum formulation of the eikonal equation by considering its special case where the forcing function is constant and equals \tilde{f} everywhere.

3.1. Deriving the Schrödinger wave equation

The time independent Schrödinger wave equation is given by [15]

$$\hat{H}\phi(X) = E\phi(X) \quad (11)$$

where $\phi(X)$ is the time-independent wave function and \hat{H} is the Hamiltonian operator obtained by first quantization¹—where the momentum variables p_i are replaced with the operator $\frac{\hbar}{i}\frac{\partial}{\partial x_i}$. E denotes the energy of the system.

For this special case where the forcing functions is constant and equals \tilde{f} everywhere, the Hamiltonian of the system is given by (in 2D)

$$H(q_1, q_2, p_1, p_2, t) = \frac{1}{2} \frac{(p_1^2 + p_2^2)}{\tilde{f}^2(q_1, q_2)}. \quad (12)$$

Its first quantization then yields the wave equation

$$-\frac{\hbar^2}{2\tilde{f}^2} \nabla^2 \phi = E\phi. \quad (13)$$

When $E > 0$ we get oscillatory solution and when $E < 0$ we get exponential solutions in the sense of distributions. In [21] we have shown that for the Euclidean distance function problem where $\tilde{f} = 1$, the exponential solution for ϕ obtained by setting $E = -\frac{1}{2}$ which is then used to recover S^* using the relation $\phi = \exp(\frac{-S^*}{\hbar})$, guarantees convergence of S^* to the true solution as $\hbar \rightarrow 0$. Following along similar lines, we propose to solve (13) at $E = -\frac{1}{2}$, for which ϕ satisfies the differential equation

$$-\hbar^2 \nabla^2 \phi + \tilde{f}^2 \phi = 0. \quad (14)$$

¹First quantization is still mysterious. For an informal but illuminating treatment, please see <http://math.ucr.edu/home/baez/categories.html>.

3.2. Solving the Schrödinger wave equation

We now provide techniques for efficiently solving the Schrödinger equation (14). Note that we are interested in the computing the solution only on the specified set of N discrete grid locations.

Since the Hamiltonian operator $-\hbar^2 \nabla^2 + \tilde{f}^2$ is positive definite, the above equation do not have a solution in the classical sense. Hence we look for a solution in the distributional sense by considering the forced version of the equation, namely

$$-\hbar^2 \nabla^2 \phi + \tilde{f}^2 \phi = \sum_{k=1}^K \delta(X - Y_k). \quad (15)$$

The points $Y_k, k \in \{1, \dots, K\}$ can be considered to be the set of locations which encode initial knowledge about the scalar field S^* , say for example $S^*(Y_k) = 0, \forall Y_k, k \in \{1, \dots, K\}$.

For the forced equation (15), closed-form solutions for ϕ can be obtained in $1D$, $2D$ and $3D$ [21] using the Green's function approach [22]. Since $S^*(X)$ goes to infinity for points at infinity, we can use Dirichlet boundary conditions $\phi(X) = 0$ at the boundary of an unbounded domain. The form of the solution for the Green's function G is given by,

1D: In $1D$, the solution for G [22] is

$$G(X) = \frac{1}{2\hbar\tilde{f}} \exp\left(\frac{-\tilde{f}|X|}{\hbar}\right). \quad (16)$$

2D: In $2D$, the solution for G [22] is

$$\begin{aligned} G(X) &= \frac{1}{2\pi\hbar^2} K_0\left(\frac{\tilde{f}\|X\|}{\hbar}\right) \\ &\approx \frac{\exp\left(\frac{-\tilde{f}\|X\|}{\hbar}\right)}{2\hbar\sqrt{2\pi\hbar\tilde{f}\|X\|}}, \frac{\tilde{f}\|X\|}{\hbar} \gg 0.25 \end{aligned} \quad (17)$$

where K_0 is the modified Bessel function of the second kind.

3D: In $3D$, the solution for G [22] is

$$G(X) = \frac{1}{4\pi\hbar^2} \frac{\exp\left(\frac{-\tilde{f}\|X\|}{\hbar}\right)}{\|X\|}. \quad (18)$$

The solutions for ϕ can then be obtained by convolution

$$\phi(X) = \sum_{k=1}^K G(X) * \delta(X - Y_k) = \sum_{k=1}^K G(X - Y_k) \quad (19)$$

from which S^* can be recovered using the relation (32). S^* can explicitly be shown to converge to the true solution $\tilde{f}r$, where $r = \min_k \|X - Y_k\|$ as $\hbar \rightarrow 0$ [21].

3.2.1. Modified Green's function

Based on the nature of the Green's function we would like to highlight on the following very important point. In the limiting case of $\hbar \rightarrow 0$,

$$\lim_{\hbar \rightarrow 0} \frac{\exp\left\{\frac{-\tilde{f}\|X\|}{\hbar}\right\}}{c\hbar^d \|X\|^p} = 0, \text{ for } \|X\| \neq 0 \quad (20)$$

for c, d and p being constants greater than zero and therefore we see that if we define

$$\tilde{G}(X) = C \exp\left(\frac{-\tilde{f}\|X\|}{\hbar}\right) \quad (21)$$

for some constant C ,

$$\lim_{\hbar \rightarrow 0} |G(X) - \tilde{G}(X)| = 0, \text{ for } \|X\| \neq 0 \quad (22)$$

and furthermore, the convergence is *uniform* for $\|X\|$ away from zero. Therefore, $\tilde{G}(X)$ provides a very good approximation for the actual Green's function as $\hbar \rightarrow 0$. For a fixed value of \hbar and X , the difference between the Green's functions is $O\left(\frac{\exp\left(\frac{-\tilde{f}\|X\|}{\hbar}\right)}{\hbar^2}\right)$ which is relatively insignificant for small values of \hbar and for all $X \neq 0$. Moreover, using \tilde{G} also avoids the singularity at the origin that G has in the 2D and 3D case. The above observation motivates us to compute the solutions for ϕ by convolving with \tilde{G} , namely

$$\phi(X) = \sum_{k=1}^K \tilde{G}(X) * \delta(X - Y_k) = \sum_{k=1}^K \tilde{G}(X - Y_k) \quad (23)$$

instead of the actual Green's function G and recover S^* using the relation (32), given by

$$S^*(X) = -\hbar \log \left[\sum_{k=1}^K \exp\left(\frac{-\tilde{f}\|X - Y_k\|}{\hbar}\right) \right] + \hbar \log(C). \quad (24)$$

Since $\hbar \log(C)$ is a constant independent of X and converges to 0 as $\hbar \rightarrow 0$, it can be ignored while computing S^* at small values of \hbar —it is equivalent to setting C to be 1. Hence the Schrödinger wave function for a constant force \tilde{f} can be approximated by

$$\phi(X) = \sum_{k=1}^K \exp\left(\frac{-\tilde{f}\|X - Y_k\|}{\hbar}\right). \quad (25)$$

It is worth emphasizing that the above defined wave function $\phi(X)$ (25), contains all the desirable properties that we need. Firstly, we notice that as $\hbar \rightarrow 0$, $\phi(Y_k) \rightarrow 1$ at the given point-set locations Y_k . Hence from (32) $S(Y_k) \rightarrow 0$ as $\hbar \rightarrow 0$ satisfying the initial conditions. Secondly as $\hbar \rightarrow 0$, $\sum_{k=1}^K \exp\left(\frac{-\tilde{f}\|X - Y_k\|}{\hbar}\right)$ can be approximated by $\exp\left(\frac{-\tilde{f}r}{\hbar}\right)$ where $r = \min_k \|X - Y_k\|$. Hence $S^*(X) \approx -\hbar \log \exp\left(\frac{-\tilde{f}r}{\hbar}\right) = \tilde{f}r$, which is the true value. When $\tilde{f} = 1$, we get the Euclidean distance function. Thirdly, ϕ can be easily computed using the fast Fourier transform as described under section (4.3). Hence for computational purposes we consider the wave function defined in (25).

4. General eikonal equations

Armed with the above set up, we can now solve the eikonal equation for arbitrary, positive-valued, bounded forcing functions f . We first show that even for general f , when ϕ satisfies the *same* differential equation as in the case of constant forcing equation (replacing \tilde{f} by f), namely

$$-\hbar^2 \nabla^2 \phi + f^2 \phi = 0, \quad (26)$$

and is related to S^* by $\phi = \exp\left(\frac{-S^*}{\hbar}\right)$, S^* asymptotically satisfies the eikonal equation (1) as $\hbar \rightarrow 0$. We show this for the 2D case but the generalization to higher dimensions is straightforward.

When $\phi(x_1, x_2) = \exp\left(\frac{-S^*(x_1, x_2)}{\hbar}\right)$, the first partials of ϕ are

$$\frac{\partial \phi}{\partial x_1} = -\frac{1}{\hbar} \exp\left(\frac{-S^*}{\hbar}\right) \frac{\partial S^*}{\partial x_1}, \quad \frac{\partial \phi}{\partial x_2} = -\frac{1}{\hbar} \exp\left(\frac{-S^*}{\hbar}\right) \frac{\partial S^*}{\partial x_2}. \quad (27)$$

The second partials required for the Laplacian are

$$\begin{aligned} \frac{\partial^2 \phi}{\partial x_1^2} &= \frac{1}{\hbar^2} \exp\left(\frac{-S^*}{\hbar}\right) \left(\frac{\partial S^*}{\partial x_1}\right)^2 - \frac{1}{\hbar} \exp\left(\frac{-S^*}{\hbar}\right) \frac{\partial^2 S^*}{\partial x_1^2}, \\ \frac{\partial^2 \phi}{\partial x_2^2} &= \frac{1}{\hbar^2} \exp\left(\frac{-S^*}{\hbar}\right) \left(\frac{\partial S^*}{\partial x_2}\right)^2 - \frac{1}{\hbar} \exp\left(\frac{-S^*}{\hbar}\right) \frac{\partial^2 S^*}{\partial x_2^2}. \end{aligned} \quad (28)$$

From this, equation (26) can be rewritten as

$$\left(\frac{\partial S^*}{\partial x_1}\right)^2 + \left(\frac{\partial S^*}{\partial x_2}\right)^2 - \hbar \left(\frac{\partial^2 S^*}{\partial x_1^2} + \frac{\partial^2 S^*}{\partial x_2^2}\right) = f^2 \quad (29)$$

which in simplified form is

$$\|\nabla S^*\|^2 - \hbar \nabla^2 S^* = f^2. \quad (30)$$

The additional $\hbar \nabla^2 S^*$ term [relative to (1)] is referred to as the *viscosity term* [7, 8] which emerges naturally from the Schrödinger equation derivation—an intriguing result. Since $|\nabla^2 S^*|$ is bounded, as $\hbar \rightarrow 0$, (30) tends to

$$\|\nabla S^*\|^2 = f^2 \quad (31)$$

which is the original eikonal equation (1). This relationship motivates us to solve the linear Schrödinger equation (26) instead of the non-linear eikonal equation and then compute the scalar field S^* via

$$S^*(X) = -\hbar \log \phi(X). \quad (32)$$

4.1. Perturbation theory

Since the linear system (26) (and its forced version) doesn't have a closed-form solution for non-constant forcing functions, we propose to solve it using *perturbation* theory [17]. Assuming that f is close to a constant non-zero forcing function \tilde{f} , equation (26) can be rewritten as

$$(-\hbar^2 \nabla^2 + \tilde{f}^2) \left[1 + (-\hbar^2 \nabla^2 + \tilde{f}^2)^{-1} \circ (f^2 - \tilde{f}^2)\right] \phi = 0. \quad (33)$$

Now, defining the operator L as

$$L \equiv (-\hbar^2 \nabla^2 + \tilde{f}^2)^{-1} \circ (f^2 - \tilde{f}^2) \quad (34)$$

and ϕ_0 as

$$\phi_0 \equiv (1 + L)\phi \quad (35)$$

we see that ϕ_0 satisfies

$$(-\hbar^2 \nabla^2 + \tilde{f}^2)\phi_0 = 0 \quad (36)$$

and

$$\phi = (1 + L)^{-1}\phi_0. \quad (37)$$

Notice that in the differential equation for ϕ_0 (36), the forcing function is constant and equals \tilde{f} everywhere. Hence ϕ_0 behaves like the wave function corresponding to the constant forcing function \tilde{f} —described under section (3.2) and can be approximated by

$$\phi_0(X) = \sum_{k=1}^K \exp\left(\frac{-\tilde{f}\|X - Y_k\|}{\hbar}\right). \quad (38)$$

We now solve for ϕ in (37) using a geometric series approximation for $(1 + L)^{-1}$. Firstly, observe that the approximate solution for ϕ_0 in (38) is a square-integrable function which is necessary for the subsequent steps.

Let \mathcal{H} denote the space of square integrable functions on \mathbb{R}^D , i.e, $g \in \mathcal{H}$ iff

$$\int g^2 d\mu < \infty. \quad (39)$$

The function norm $\|g\|$ for a function $g \in \mathcal{H}$ is given by

$$\|g\|^2 = \int g^2 d\mu. \quad (40)$$

Let \mathcal{B} denote a closed unit ball in the Hilbert space \mathcal{H} , i.e

$$\mathcal{B} = \{g \in \mathcal{H} : \|g\| \leq 1\}. \quad (41)$$

Let $c_0 \equiv \|L\|_{op}$ —the operator norm—defined by

$$c_0 = \sup\{\|Lg\|, \forall \text{functions } g \in \mathcal{B}\}. \quad (42)$$

If $c_0 < 1$, we can approximate $(1 + L)^{-1}$ using the first few $T + 1$ terms of the geometric series to get

$$(1 + L)^{-1} \approx 1 - L + L^2 - L^3 + \dots + (-1)^T L^T \quad (43)$$

where the operator norm of the difference can be bounded by

$$\|(1 + L)^{-1} - \sum_{i=0}^T (-1)^i L^i\|_{op} \leq \sum_{i=T+1}^{\infty} \|L^i\|_{op} \leq \sum_{i=T+1}^{\infty} c_0^i = \frac{c_0^{T+1}}{1 - c_0} \quad (44)$$

which converges to 0 *exponentially* in T . We would like to point out that the above geometric series approximation is similar to a Born expansion used in scattering theory [18]. We now derive an upper bound for c_0 .

Let $L = A_1 \circ A_2$ where $A_1 \equiv (-\hbar^2 \nabla^2 + \tilde{f}^2)^{-1}$ and $A_2 \equiv f^2 - \tilde{f}^2$. We now provide an upper bound for $\|A_1\|_{op}$.

For a given $g \in \mathcal{B}$, let $z = A_1(g)$, i.e z satisfies the relation

$$(-\hbar^2 \nabla^2 + \tilde{f}^2)z = g \quad (45)$$

with vanishing Dirichlet boundary conditions at ∞ . Then

$$\begin{aligned} \|(-\hbar^2 \nabla^2 + \tilde{f}^2)z\|^2 &= \|-\hbar^2 \nabla^2 z\|^2 + \|\tilde{f}^2 z\|^2 + 2\hbar^2 \tilde{f}^2 \langle -\nabla^2 z, z \rangle \\ &= \|g\|^2 \leq 1. \end{aligned} \quad (46)$$

We now use the relation

$$\nabla \cdot (z \nabla z) = z \nabla^2 z + |\nabla z|^2 \quad (47)$$

to compute

$$\langle -\nabla^2 z, z \rangle = - \int z \nabla^2 z d\mu = - \int \nabla \cdot (z \nabla z) d\mu + \int |\nabla z|^2 d\mu. \quad (48)$$

Now from the divergence theorem we have

$$- \int \nabla \cdot (z \nabla z) d\mu = 0 \quad (49)$$

and hence

$$\langle -\nabla^2 z, z \rangle = \int |\nabla z|^2 d\mu \geq 0. \quad (50)$$

Using the above relation in (46), we then observe that

$$\|z\| = \|A_1(g)\| \leq \frac{1}{\tilde{f}^2}, \forall g \in \mathcal{B}. \quad (51)$$

Since we showed $A_1(\mathcal{B})$ is bounded and less than or equal to $\frac{1}{\tilde{f}^2}$, we immediately have

$$\|A_1\|_{op} \leq \frac{1}{\tilde{f}^2}. \quad (52)$$

Now, let $M = \sup\{|f^2 - \tilde{f}^2|\}$. Then, for any $g \in \mathcal{B}$

$$\|(f^2 - \tilde{f}^2)g\|^2 = \int (f^2 - \tilde{f}^2)^2 g^2 d\mu \leq M^2 \|g\|^2 \leq M^2 \quad (53)$$

and hence from (42),

$$\|A_2\|_{op} \leq M = \sup\{|f^2 - \tilde{f}^2|\}. \quad (54)$$

Since $\|L\|_{op} \leq \|A_1\|_{op}\|A_2\|_{op}$, from equations (52) and (54), we observe that

$$c_0 = \|L\|_{op} \leq \frac{\sup\{|f^2 - \tilde{f}^2|\}}{\tilde{f}^2}. \quad (55)$$

It is worth commenting that the bound for c_0 is *independent* of \hbar . So, if we guarantee that $\frac{\sup\{|f^2 - \tilde{f}^2|\}}{\tilde{f}^2} < 1$, the geometric series approximation for $(1 + L)^{-1}$ (43) converges for *all* values of \hbar .

4.2. Deriving a bound for convergence of the perturbation series

Interestingly, for any positive, upper bounded forcing function f bounded away from zero, i.e $f(X) > \epsilon$ for some $\epsilon > 0^2$, by defining $\tilde{f} = \sup\{f(X)\}$, we observe that $|f^2 - \tilde{f}^2| < \tilde{f}^2$. From equation (55), we immediately see that $c_0 < 1$. This proves the existence of \tilde{f} for which the geometric series approximation (43) is *always* guaranteed to converge for any positive bounded forcing function f bounded away from zero. The choice of \tilde{f} can then be made prudently by defining it to be the value that *minimizes*

$$F(\tilde{f}) = \frac{\sup\{|f^2 - \tilde{f}^2|\}}{\tilde{f}^2}. \quad (56)$$

This in turn minimizes the operator norm c_0 , thereby providing a better geometric series approximation for the inverse (43).

Let $f_{min} = \inf\{f(X)\}$ and let $f_{max} = \sup\{f(X)\}$. We now show that $F(\tilde{f})$ attains its minimum at

$$\tilde{f} = \nu = \sqrt{\frac{f_{min}^2 + f_{max}^2}{2}}. \quad (57)$$

Case(i): If $\tilde{f} < \nu$, then $\sup\{|f^2 - \tilde{f}^2|\} = f_{max}^2 - \tilde{f}^2$. Clearly,

$$\frac{f_{max}^2 - \tilde{f}^2}{\tilde{f}^2} > \frac{f_{max}^2 - \nu^2}{\nu^2}. \quad (58)$$

Case(ii): If $\tilde{f} > \nu$, then $\sup\{|f^2 - \tilde{f}^2|\} = \tilde{f}^2 - f_{min}^2$. It follows that

$$\frac{\tilde{f}^2 - f_{min}^2}{\tilde{f}^2} = 1 - \frac{f_{min}^2}{\tilde{f}^2} > 1 - \frac{f_{min}^2}{\nu^2}. \quad (59)$$

²If $f(X) = 0$, then the velocity $v(X) = \frac{1}{f(X)}$ becomes ∞ at X . Hence it is reasonable to assume $f(X) > 0$.

We therefore see that $\tilde{f} = \nu = \sqrt{\frac{f_{min}^2 + f_{max}^2}{2}}$ is the optimal value.

Now, using the above approximation for $(1+L)^{-1}$ (43) and the definition of L from (34), we obtain the solution for ϕ as

$$\phi = \phi_0 - \phi_1 + \phi_2 - \phi_3 + \dots + (-1)^T \phi_T \quad (60)$$

where ϕ_i satisfies the recurrence relation

$$(-\hbar^2 \nabla^2 + \tilde{f}^2)\phi_i = (f^2 - \tilde{f}^2)\phi_{i-1}, \forall i \in \{1, 2, \dots, T\}. \quad (61)$$

Observe that (61) is an inhomogeneous, screened Poisson equation with a constant forcing function \tilde{f} . Following a Green's function approach [22], each ϕ_i can be obtained by convolution

$$\phi_i = G * [(f^2 - \tilde{f}^2)\phi_{i-1}] \quad (62)$$

where G is given by equations (16), (17) or (18) depending upon the spatial dimension.

Once the ϕ_i 's are computed, the wave function ϕ can then be determined using the approximation (60). The solution for the eikonal equation can be recovered using the relation (32). Notice that if $f = \tilde{f}$ everywhere, then all ϕ_i 's except ϕ_0 is identically equal to zero and we get $\phi = \phi_0$ as described under section (3).

4.3. Efficient computation of the wave function

In this section, we provide numerical techniques for efficiently computing the wave function ϕ . Recall that we are interested in solving the eikonal equation only at the given N discrete grid locations. Consider the solution for ϕ_0 given in (38). In order to obtain the desired solution for ϕ_0 computationally, we must replace the δ function by the Kronecker delta function

$$\delta_{kron}(X) = \begin{cases} 1 & \text{if } X = Y_k; \\ 0 & \text{otherwise} \end{cases}$$

that takes 1 at the point-set locations ($\{Y_k\}$) and 0 at other grid locations. Then ϕ_0 can be *exactly* computed at the grid locations by the discrete convolution of \tilde{G} (setting $C = 1$) with the Kronecker-delta function.

To compute ϕ_i , we replace each of the convolutions in (62) with the discrete convolution between the functions computed at the N grid locations. By the convolution theorem [20], a discrete convolution can be obtained as the inverse Fourier transform of the product of two individual transforms

Table 1: Algorithm for the approximate solution of the eikonal equation

1. Compute the function $\tilde{G}(X) = \exp\left(\frac{-f\|X\|}{\hbar}\right)$ at the grid locations.
 2. Define the function $\delta_{kron}(X)$ which takes the value 1 at the point-set locations and 0 at other grid locations.
 3. Compute the FFT of \tilde{G} and δ_{kron} , namely $\tilde{G}_{FFT}(U)$ and $\delta_{FFT}(U)$ respectively.
 4. Compute the function $H(U) = \tilde{G}_{FFT}(U)\delta_{FFT}(U)$.
 5. Compute the inverse FFT of H to obtain $\phi_0(X)$ at the grid locations.
 6. Initialize $\phi(X)$ to $\phi_0(X)$.
 7. Consider the Green's function G corresponding to the spatial dimension and compute its FFT, namely $G_{FFT}(U)$.
 8. For $i = 1$ to T do
 9. Define $P(X) = [f^2(X) - \tilde{f}^2] \phi_{i-1}(X)$.
 10. Compute the FFT of P namely $P_{FFT}(U)$.
 11. Compute the function $H(U) = G_{FFT}(U)P_{FFT}(U)$.
 12. Compute the inverse FFT of H and multiply it with the grid width area/volume to compute $\phi_i(X)$ at the grid locations.
 13. Update $\phi(X) = \phi(X) + (-1)^i \phi_i(X)$.
 14. End
 15. Take the logarithm of $\phi(X)$ and multiply it by $(-\hbar)$ to get the approximate solution for the eikonal equation at the grid locations.
-

which for two $O(N)$ sequences can be performed in $O(N \log N)$ time [19]. Thus, the values of each ϕ_i at the N grid locations can be efficiently computed in $O(N \log N)$ making use of the values of ϕ_{i-1} determined at the earlier step. Thus, the overall time complexity to compute the approximate ϕ using the first few $T+1$ terms is then $O(TN \log N)$. Taking the logarithm of ϕ then provides an approximate solution to the eikonal equation. The algorithm is adumbrated in table 1.

We would like to emphasize that the number of terms (T) used in the geometric series approximation of $(1+L)^{-1}$ (43) is *independent* of N . Using more terms only improves the approximation of this truncated geometric series as shown in the experimental section. From equation (44), it is evident that the error incurred due to this approximation converges to zero *exponentially* in T and hence even with a small value of T , we should be able to achieve good accuracy.

4.4. Numerical issues

In principle, we should be able to apply our technique at very small values of \hbar and obtain highly accurate results. But we noticed that a naïve double precision-based implementation tends to deteriorate for \hbar values very close to zero. This is due to the fact that at small values of \hbar (and also at large values of \tilde{f}), $\exp\left(\frac{-\tilde{f}\|X\|}{\hbar}\right)$ drops off very quickly and hence for grid locations which are far away from the point-set, the convolution done using FFT may not be accurate. To this end, we turned to the GNU MPFR multiple-precision arithmetic library which provides arbitrary precision arithmetic with correct rounding [23]. MPFR is based on the GNU multiple-precision library (GMP) [24]. It enabled us to run our technique at very small values of \hbar giving highly accurate results. We corroborate our claim and demonstrate the usefulness of our method with the set of experiments described in the subsequent section.

4.5. Exact computational complexity

More the number of precision bits p used in the GNU MPFR library, better is the accuracy of our technique, as the error incurred in the floating point operations can be bounded by $O(2^{-p})$. But using more bits has an adverse effect of slowing down the running time. The $O(N \log N)$ time complexity of the FFT algorithm [19] for an $O(N)$ length sequence takes into account only the number of floating-point operations involved, barring any numerical accuracy. The accuracy of the FFT algorithm and our technique entirely depends on the number of precision bits used for computing elementary functions like \exp , \log , \sin and \cos and hence should be taken into account while calculating the time complexity of our algorithm. If p precision bits are used, the time complexity for computing these elementary functions can be shown to be $O(M(p) \log p)$ [25, 26, 27], where $M(p)$ is the computational complexity of multiplying two p -digit numbers. The Schönhage-Strassen algorithm [28] gives an asymptotic upper bound on the time complexity for multiplying two p -digit numbers. The run-time bit complexity is $M(p) = O(p \log p \log \log p)$. Then taking these p precision bits into account, the time complexity of our algorithm for computing S^* at the given N grid locations, using the first $T + 1$ terms in the geometric series approximation of ϕ (60), is $O(TN \log(N)p(\log p)^2 \log(\log p))$ bit-wise operations. For the following experiments we used $p = 512$ precision bits.

5. Experiments

5.1. Euclidean distance functions

In this section, we show the efficacy of our Schrödinger method by computing the approximate Euclidean distance function S^* and comparing it to the actual Euclidean distance function and the fast sweeping method, first on randomly generated $2D$ point-sets and then on a set of bounded $2D$ and $3D$ grid points.

Example 1: We begin by demonstrating the effect of \hbar on our Schrödinger method and show that as $\hbar \rightarrow 0$, the accuracy our method does improve significantly. To this end, we considered a $2D$ grid consisting of points between $(-0.121, -0.121)$ and $(0.121, 0.121)$ with a grid width of $\frac{1}{2^9}$. The total number of grid points is then $N = 125 \times 125 = 15,625$. We ran 1000 experiments each time randomly choosing 5000 grid locations as data points (point-set), for 9 different values of \hbar ranging from 5×10^{-5} to 4.5×10^{-4} in steps of 5×10^{-5} . For each run and each value of \hbar , we calculated the percentage error as

$$error = \frac{100}{N} \sum_{i=1}^N \frac{\Delta_i}{D_i}, \quad (63)$$

where D_i and Δ_i are respectively the actual distance and the absolute difference of the computed distance to the actual distance at the i^{th} grid point. The plot in figure 1 shows the *mean* percentage error at each value of \hbar . The *maximum* value of the error at each value of \hbar is summarized in table 2. The error is less than 0.6% at $\hbar = 0.00005$ demonstrating the algorithm's ability to compute accurate Euclidean distances.

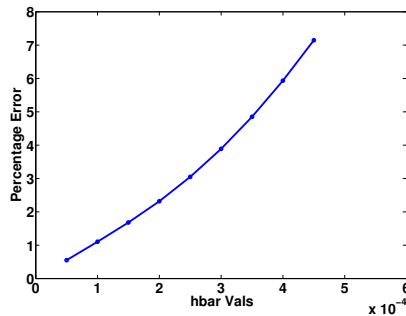


Figure 1: Percentage error versus \hbar in 1000 $2D$ experiments.

\hbar	Maximum error
0.00005	0.5728%
0.0001	1.1482%
0.00015	1.7461%
0.0002	2.4046%
0.00025	3.1550%
0.0003	4.0146%
0.00035	4.9959%
0.0004	6.1033%
0.00045	7.3380%

Table 2: Maximum percentage error for different values of \hbar in 1000 $2D$ experiments.

Example 2: We pitted the Schrödinger algorithm against the fast sweeping method [9] on a $2D$ grid consisting of points between $(-0.123, -0.123)$ and $(0.123, 0.123)$ with the grid with of $\frac{1}{2^{10}}$. The number of grid points equals $N = 253 \times 253 = 64,009$. We ran 100 experiments, each time randomly choosing 10,000 grid points as data points. We set $\hbar = 0.0001$ for the Schrödinger and ran the fast sweeping for 10 iterations sufficient for it to converge. The plot in figure 2 shows the average percentage error calculated according to equation (63) for both these techniques in comparison to the true Euclidean distance function. From the plot, it is clear that while the fast sweeping method has a percentage error of around 7%, Schrödinger method gave a percentage error of less than **1.5%** providing much better accuracy.

Example 3: In this example, we computed the Euclidean distance transform using the grid points of certain silhouettes (figure 3) [29]³, on a $2D$ grid consisting of points between $(-0.125, -0.125)$ to $(0.125, 0.125)$ with a grid width of $\frac{1}{2^{10}}$. The number of grid points equals $N = 257 \times 257 = 66049$. We set \hbar for the Schrödinger method 0.0003. For the sake of comparison, we ran the fast sweeping for 10 iterations which was sufficient for convergence. The percentage error for the Schrödinger and the fast sweeping (calculated as per equation (63)) when compared with the true Euclidean distance function for each of these shapes is adumbrated in table(3).

³We thank Kaleem Siddiqi for providing us the set of 2D shape silhouettes.

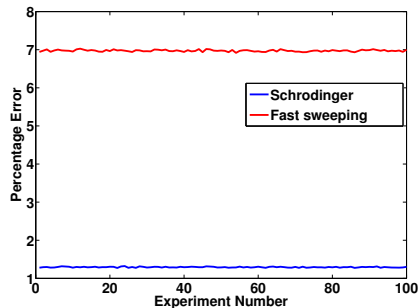


Figure 2: Percentage error between the true and computed Euclidean distance function for (i) Schrödinger (in blue) (ii) Fast sweeping (in red) in 100 2D experiments.

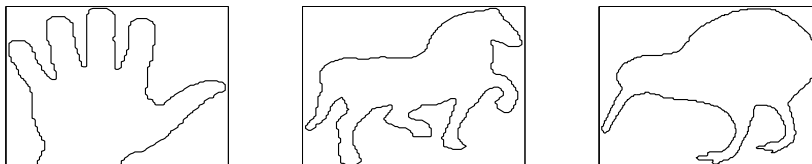


Figure 3: Shapes

Shape	Schrödinger	Fast sweeping
Hand	2.182%	2.572%
Horse	2.597%	2.549%
Bird	2.116%	2.347%

Table 3: Percentage error of the Euclidean distance function computed using the grid points of these silhouettes as data points

The true Euclidean distance function contour plot and those obtained from our method and fast sweeping is delineated in figure 4.

Example 4: We took the Stanford bunny dataset⁴ and used the coordi-

⁴This dataset is available at http://www.cc.gatech.edu/projects/large_models/bunny.html

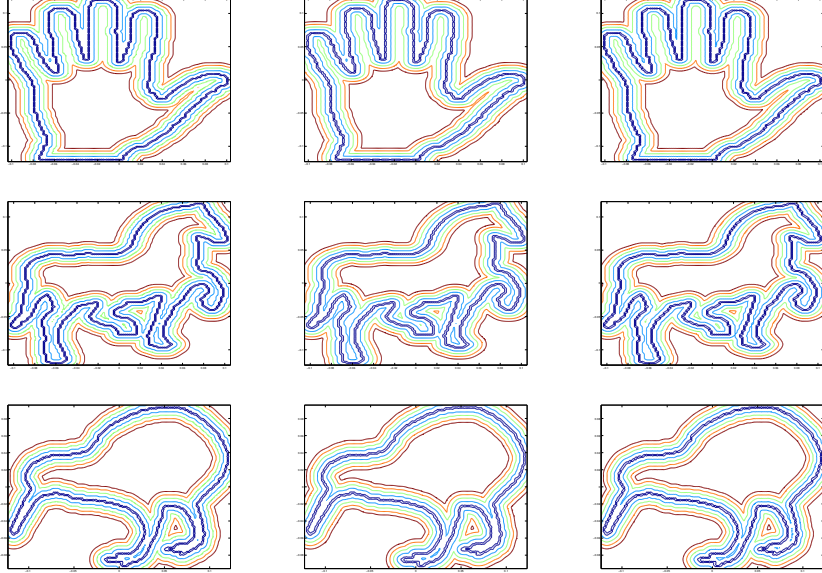


Figure 4: Contour plots: (i) Left: True Euclidean distance function, (ii) Center: Schrödinger, (iii) Right: Fast sweeping

nates of the data points on the model as our point-set locations. Since the input data locations need not conform to grid locations, we scaled the space uniformly in all dimensions and rounded off the data so that the data lies at grid locations. The input data was also shifted so that it was approximately symmetrically located with respect to the x , y and z axis. We should point out that shifting the data doesn't affect the Euclidean distance function value and uniform scaling of all dimensions is also not an issue, as the distances can be rescaled after they are computed. Moreover, the formula for calculating the percentage error [equation (63)] is invariant to shifting and scaling.

After these basic data manipulations, the cardinality of the point set was $K = 8948$ with the data confined to the cubic region $-0.117 \leq x \leq 0.117$, $-0.117 \leq y \leq 0.117$ and $-0.093 \leq z \leq 0.093$, with a grid width of $\frac{1}{2^8}$. The number of points on the grid equals $N = 182,329$. We ran the Schrödinger with $\hbar = 0.0004$ and the fast sweeping for 15 iterations and later calculated the percentage error for both these methods by comparing with the true distance function according to equation (63). Our method had a percentage error of only **1.23%** and which favorably compares to fast sweeping which had a percentage error of 4.75%.

The isosurface obtained by connecting the grid points at a distance of 0.005 from the point set, determined by the true Euclidean distance function, Schrödinger and the fast sweeping is shown in figure 5. Notice the similarity between the plots. It provides anecdotal visual evidence for the usefulness of our approach.



Figure 5: Isosurfaces: (i) Left: Actual Euclidean distance function, (ii) Center: Schrödinger and (iii) Right: Fast sweeping

5.2. The general eikonal equation

In this section, we demonstrate the usefulness of our approach by computing the approximate solution to the general eikonal equation (1) over a 2D grid.

5.2.1. Comparison with the true solution

Example 5: In this example, we solve the eikonal equation for the scenario where the exact solution is known *a priori* at the grid locations. The exact solution is

$$S(x, y) = |e^{\sqrt{x^2+y^2}} - 1|. \quad (64)$$

The boundary condition is, $S(x, y) = 0$ at the point source located at $(x_0, y_0) = (0, 0)$. The forcing function—the absolute gradient $|\nabla S|$ —is

$$f(x, y) = |\nabla S| = e^{\sqrt{x^2+y^2}} \quad (65)$$

specified on a 2D grid consisting of points between $(-0.125, -0.125)$ and $(0.125, 0.125)$ with a grid width of $\frac{1}{20}$. We ran the Schrödinger for 6 iterations at $\hbar = 0.006$ and the fast sweeping for 15 iterations sufficient enough for both the methods to converge. The percentage error—calculated according to equation (63)—and the maximum difference between the true and approximate solution for different iterations is summarized in the table (4).

Iter	% error	max diff
1	2.081042	0.002651
2	1.514745	0.002140
3	1.390552	0.002142
4	1.363256	0.002128
5	1.357894	0.002128
6	1.356898	0.002128

Table 4: Percentage error for the Schrödinger method for different iterations

The fast sweeping gave a percentage error of 1.135%. We believe that the error incurred in our Schrödinger approach can be further reduced by decreasing \hbar but at the expense of more computational power requiring higher precision floating point arithmetic.

The contour plots of the true solution and those obtained from Schrödinger and fast sweeping are displayed below (figure 6). We can immediately observe the similarity of our solution with the true solution. We do observe smoother isocontours in our Schrödinger method relative to fast sweeping.

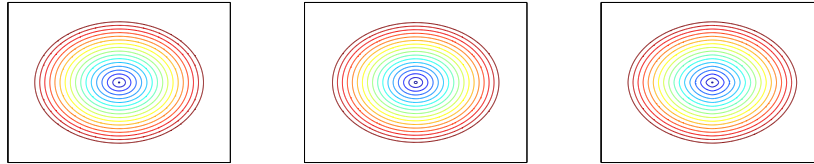


Figure 6: Contour plots: (i) Left: True solution, (ii) Center: Schrödinger, and (iii) Right: Fast sweeping

5.2.2. Comparison with the fast sweeping

In order to verify the accuracy of our technique, we compared our solution with fast sweeping for the following set of examples, using the latter as the ground truth as the true solution is not available in closed-form.

Example 6: In this example we solved the eikonal equation from a point source located at $(x_0, y_0) = (0, 0)$ for the following forcing function

$$f(x, y) = 1 + 2(e^{-2((x+0.05)^2+(y+0.05)^2)} - e^{-2((x-0.05)^2+(y-0.05)^2)}) \quad (66)$$

on a $2D$ grid consisting of points between $(-0.125, -0.125)$ and $(0.125, 0.125)$ with a grid width of $\frac{1}{2^{10}}$. We ran our method for 6 iterations with \hbar set at 0.015 and fast sweeping for 15 iterations sufficient for both techniques to converge. When we calculated the percentage error for the Schrödinger according to equation 63 (with fast sweeping as the ground truth), the error was just around 1.245%. The percentage error and maximum difference between the fast sweeping and Schrödinger solutions after each iteration are adumbrated in table (5).

Iter	%error	max diff
1	1.144632	0.008694
2	1.269028	0.008274
3	1.223836	0.005799
4	1.246392	0.006560
5	1.244885	0.006365
6	1.245999	0.006413

Table 5: Percentage error for the Schrödinger method in comparison to Fast sweeping

We believe that the fluctuations both in the percentage error and the maximum difference are due to repeated approximations of the integration involved in the convolution with discrete convolution and summation, but nevertheless stabilized after 6 iterations. The contour plots shown in figure (7) clearly demonstrate the similarities between these methods.



Figure 7: Contour plots: (i) Left: Schrödinger, (ii) Right: Fast sweeping

Example 7: Here we solved the eikonal equation for the sinusoidal forcing function

$$f(x, y) = 1 + \sin(\pi(x - 0.05)) \sin(\pi(y + 0.05)) \quad (67)$$

on the same $2D$ grid as in the previous example. We randomly chose 4 grid locations namely,

$$\{0, 0\}, \{0.0488, 0.0977\}, \{-0.0244, -0.0732\}, \{0.0293, -0.0391\}$$

as data locations and ran our method for 6 iterations with \hbar set at 0.0085 and ran fast sweeping for 15 iterations. The percentage error between the Schrödinger solution (after 6 iterations) and fast sweeping was 4.537% with the maximum absolute difference between them being 0.0109.

The contour plots are shown in figure (8). Notice that the Schrödinger contours are more smoother in comparison to the fast sweeping contours.



Figure 8: Contour plots: (i) Left: Schrödinger, (ii) Right: Fast sweeping

Example 8: Here we compared with fast sweeping on a larger $2D$ grid consisting of points between $(-5, -5)$ and $(5, 5)$ with a grid width of 0.25. We again considered the sinusoidal forcing function

$$f(x, y) = 1 + 0.3 \sin(\pi(x + 1)) \sin(\pi(y - 2)) \quad (68)$$

and chose 4 grid locations namely $\{0, 0\}, \{1, 1\}, \{-2, -3\}, \{3, -4\}$ as data locations. Notice that the Green's function G and \tilde{G} goes to zero exponentially faster for grid locations away from zero for small values of \hbar . Hence for a grid location say $(-4, 4)$ which is reasonably far away from 0, the value of the Green's function say at $\hbar = 0.001$ may be zero even when we use a large number of precision bits p . This problem can be easily circumvented by first scaling down the entire grid by a factor τ , computing the solution S^* on the smaller denser grid and then rescaling it back again by τ to obtain the actual solution. It is worth emphasizing that scaling down the grid is tantamount to scaling down the forcing function as clearly seen from the fast sweeping method. In fast sweeping [9], the solution S^* is computed using the quantity $f_{i,j}\delta$ where $f_{i,j}$ is the value of forcing function at the $(i, j)^{th}$

grid location and δ is the grid width. Hence scaling down δ by a factor of τ is equivalent to fixing δ and scaling down f by τ . Since the eikonal equation (1) is linear in f , computing the solution for a scaled down f —equivalent to a scaled down grid—and then rescaling it back again is guaranteed to give the actual solution.

τ can be set to any desired quantity. For the current experiment we set $\tau = 100$, $\hbar = 0.001$ and ran our method for 6 iterations. Fast sweeping was run for 15 iterations. The percentage error between these methods was about 3.165%. The contour plots are shown in figure (9). Again, the contours obtained from the Schrödinger are more smoother than those obtained from fast sweeping.



Figure 9: Contour plots: (i) Left: Schrödinger, (ii) Right: Fast sweeping

6. Discussion

In this paper, we have introduced a new approach to solving the non-linear eikonal equation. We proved that the solution to the eikonal equation can be obtained as a limiting case of the solution to a corresponding linear Schrödinger wave equation. Instead of directly solving the eikonal equation, the Schrödinger formalism results in a generalized, screened Poisson equation which is solved at very small values of \hbar . Our Schrödinger-based approach follows the pioneering Hamilton-Jacobi solvers such as the fast sweeping [9] and fast marching [8] methods with the crucial difference being its linearity. We developed a fast and efficient perturbation series method for solving the wave equation (generalized, screened Poisson equation) which is guaranteed to converge provided the forcing function f is positive and bounded. Using the perturbation method and the relation (32), we obtained the solution to (30) without spatially discretizing the operators.

Acknowledgements

We thank Arunava Banerjee, Rama Chellappa, Alireza Entezari, Chiu-Yen Kao, Ajit Rajwade, Guillermo Sapiro, Kaleem Siddiqi, Baba Vemuri and Alan Yuille for helpful discussions.

- [1] G. Chartier, *Introduction to Optics*, Springer, 2005.
- [2] D. Paris, F. Hurd, *Basic Electromagnetic Theory*, McGraw-Hill Education, 1969.
- [3] V. Arnold, *Mathematical Methods of Classical Mechanics*, Springer, 1989.
- [4] J. Canny, *Complexity of robot motion planning*, The MIT Press, 1988.
- [5] S. Osher, R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Springer, 2002.
- [6] G. Whitham, *Linear and Nonlinear Waves*, Pure and Applied Mathematics, Wiley-Interscience, 1999.
- [7] M. Crandall, H. Ishii, P. Lions, User's guide to viscosity solutions of second order partial differential equations, *Bulletin of the American Mathematical Society* 27 (1) (1992) 1–67.
- [8] S. Osher, J. Sethian, Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations, *Journal of Computational Physics* 79 (1) (1988) 12–49.
- [9] H. Zhao, A fast sweeping method for eikonal equations, *Mathematics of Computation* (2005) 603–627.
- [10] T. Cormen, C. Leiserson, R. Rivest, C. Stein, *Introduction to Algorithms*, 2nd Edition, The MIT Press, 2001.
- [11] L. Yatziv, A. Bartesaghi, G. Sapiro, $O(N)$ implementation of the fast marching algorithm, *J. Comput. Phys.* 212 (2) (2006) 393–399.
- [12] J.-L. Basdevant, *Variational Principles in Physics*, Springer, 2007.
- [13] H. Goldstein, C. Poole, J. Safko, *Classical Mechanics*, 3rd Edition, Addison-Wesley, 2001.

- [14] J. Butterfield, On Hamilton-Jacobi theory as a classical root of quantum theory, in: A. Elitzur, S. Dolev, N. Kolenda (Eds.), Quo-Vadis Quantum Mechanics, Springer, 2005, Ch. 13, pp. 239–274.
- [15] D. Griffiths, Introduction to Quantum Mechanics, 2nd Edition, Benjamin-Cummings, 2004.
- [16] A. Fetter, J. Walecka, Theoretical Mechanics of Particles and Continua, Dover, 2003.
- [17] F. Fernandez, Introduction to Perturbation Theory in Quantum Mechanics, CRC Press, 2000.
- [18] R. Newton, Scattering Theory of Waves and Particles, 2nd Edition, Springer-Verlag, New York, 1982.
- [19] J. Cooley, J. Tukey, An algorithm for the machine calculation of complex Fourier series, Mathematics of Computation 19 (90) (1965) 297–301.
- [20] R. Bracewell, The Fourier Transform and its Applications, 3rd Edition, McGraw-Hill Science and Engineering, 1999.
- [21] K. Gurumoorthy, A. Rangarajan, A Schrödinger equation for the fast computation of approximate Euclidean distance functions, in: Scale Space and Variational Methods in Computer Vision (SSVM), Vol. LNCS 5567, Springer, 2009, pp. 100–111.
- [22] M. Abramowitz, I. Stegun, Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, Government Printing Office, USA, 1964.
- [23] L. Fousse, G. Hanrot, V. Lefèvre, P. Pélicier, P. Zimmermann, MPFR: A multiple-precision binary floating-point library with correct rounding, ACM Trans. Math. Software 33 (2007) 1–15.
- [24] G. Torbjörn, *et al.*, GNU multiple precision arithmetic library 5.0.1 (June 2010).
- [25] R. Brent, Fast multiple-precision evaluation of elementary functions, Journal of the ACM 23 (1976) 242–251.
- [26] T. Sasaki, Y. Kanada, Practically fast multiple-precision evaluation of $\log(x)$, Journal of Information Processing 5 (1982) 247–250.

- [27] D. Smith, Efficient multiple-precision evaluation of elementary functions, *Mathematics of Computation* 52 (185) (1989) 131–134.
- [28] A. Schönhage, V. Strassen, Schnelle multiplikation großer zahlen, *Computing* 7 (1971) 281–292.
- [29] K. Siddiqi, A. Tannenbaum, S. Zucker, A Hamiltonian approach to the eikonal equation, in: *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, Vol. LNCS 1654, Springer-Verlag, 1999, pp. 1–13.
- [30] C.-Y. Kao, S. Osher, Y.-H. Tsai, Fast sweeping methods for static Hamilton-Jacobi equations, *SIAM Journal on Numerical Analysis* 42 (6) (2004) 2612–2632.