Title

Numerical methods for high dimensional Hamilton-Jacobi equations using radial basis functions

Journal

Journal of Computational Physics, 196(1)

ISSN

0021-9991

Authors

Cecil, T
Qian, J L
Osher, S

Publication Date

2004-05-01

Peer reviewed

ARTICLE IN PRESS

Available online at www.sciencedirect.com
SCIENCE DIRECT®

Journal of Computational Physics xxx (2003) xxx–xxx

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Numerical methods for high dimensional Hamilton–Jacobi equations using radial basis functions

Tom Cecil *, Jianliang Qian, Stanley Osher

Department of Mathematics, University of California, P.O. Box 951555, Los Angeles, CA 90095-1555, USA

Received 18 September 2003; received in revised form 4 November 2003; accepted 4 November 2003

Abstract

We utilize radial basis functions (RBFs) to construct numerical schemes for Hamilton–Jacobi (HJ) equations on unstructured data sets in arbitrary dimensions. The computational setup is a meshless discretization of the physical domain. We derive monotone schemes on unstructured data sets to compute the viscosity solutions. The essentially nonoscillatory (ENO) mechanism is combined with radial basis function reconstruction to obtain high order schemes in the presence of gradient discontinuities. Numerical examples of time dependent HJ equations in 2, 3 and 4 dimensions illustrate the accuracy of the new methods.

1. Introduction

In the numerical solution of time dependent conservation laws such as

$$u_t + \nabla \cdot f(u) = 0, \tag{1}$$

a method for solving the PDE is by dividing the spatial domain into grid cells and solving a Riemann problem for each cell forward in time. For Hamilton–Jacobi equations of the form

$$\varphi_t + H(\nabla \varphi) = 0, \tag{2}$$

we can think of our problem as being a conservation law such as (1) in the variable $u = \varphi_x$ in one spatial dimension. This becomes precise in one dimension as we can see by taking the x derivative of (2). In this way there is a direct link between conservation laws and HJ equations, with the solution to (1) being a derivative of the solution to (2). Although this analogy fails in multiple spatial dimensions it guides us towards the numerical methods of conservation laws when finding solutions to (2).

* Corresponding author. Tel.: +1-310-403-6198.
E-mail addresses: tcecil@math.ucla.edu (T. Cecil), qian@math.ucla.edu (J. Qian), sjo@math.ucla.edu (S. Osher).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2003.11.010

On uniform grids in any dimension [21,22] and proposed extending the essentially nonoscillatory (ENO) schemes of [16,26] for conservation laws to HJ equations by reconstructing locally smooth polynomial interpolants of φ in each individual spatial dimension, x_i, and then taking the derivative, φ_{x_i} of that interpolant for use in $H(\nabla\varphi)$. These methods have shown good results on uniform grids, avoiding the oscillations typically associated with high order methods in the presence of discontinuities.

On nonuniform grids in higher dimensions there has been work done on extending the ENO type of smooth polynomial interpolant reconstruction, see [1,12,31]. These methods have shown some success, but also have some drawbacks. For example, when using divided differences as approximations to the higher derivatives needed to obtain the Newton polynomial in 1D, we see that each time the polynomial is raised by one degree, we need one extra evaluation point. In 2D on an arbitrary triangulated grid there are no Newton divided differences to aid us in reconstruction. If we would like to use polynomial reconstruction we now have the added burden of needing at least $[(n+1)(n+2)]/2$ nodes to construct a degree n polynomial. In K dimensions we would need at least $\binom{n+K}{K}$ nodes, and even with this many nodes there may still be problems resulting from the ill conditioning of the linear system for the coefficients if the nodes are not well spaced [1,17,31]. Attempts have been made to rectify these problems, but multidimensional polynomial reconstruction is still far from being a "black box" procedure.

In this work we propose a new evolution procedure based on reconstruction using radial basis functions (RBFs). Because of the discontinuites present in the gradient of the solutions of HJ equations we will introduce monotone evolution schemes and ENO interpolations. The dimension independent framework allows the methods presented to be generalized to higher dimensions.

We will begin in Section 2 by introducing radial basis function (RBF) interpolation. We then move in Section 3 to a brief description of how we handle neighbor access in a meshless computational framework. Next we cover the construction of monotone schemes in Section 4. We follow this in Section 5 by introducing a Roe with entropy fix scheme which minimizes artificial diffusion. In Section 6 we describe spatial discretization to achieve higher order accuracy, followed by temporal discretization in Section 7. Finally, we give a summary of the implementation procedure in Section 8.

2. Function reconstruction using RBFs

Instead of using polynomial reconstruction for a function Φ, which has been used successfully in 1D, we will use a type of multidimensional spline [13,28]

$$\Phi(x) := \sum_{j=1}^{M} \gamma_j \phi(x - y_j) + \sum_{j=1}^{Q} \beta_j p_j(x), \tag{3}$$

where ϕ is a RBF, M is the number of cells in the reconstruction stencil, and the second sum is over polynomials $\{p_j\}$ which form a basis of the kernel of the seminorm $[\cdot,\cdot]$ of the *native* space in which ϕ lives [18]. In general a spline, Φ, in a semi-Hilbert space, V, interpolating data, $\{u_i\}$, satisfies $|\Phi|_V = \min_{u\in A} |u|_V$ where $A = \{v \in V \,|\, \langle \lambda_i, v\rangle = u_i\}$. So in this norm we are finding an optimal recovery function. The functions ϕ are assumed to have radial symmetry. Φ is forced to have the property that on a given stencil $\{x_i\}_{i=1:M}$,

$$\Phi(x_i) = u(x_i) \quad \text{and} \quad \sum_{j=1}^{M} \gamma_j p_s(x_j) = 0, \qquad s = 1, \ldots, Q. \tag{4}$$

So to find $\{\gamma_j\}_{j=1:M}$ we need to solve the linear system

$$A = \begin{bmatrix} V & N \\ N^t & 0_{(Q,Q)} \end{bmatrix} \begin{bmatrix} \gamma \\ \beta \end{bmatrix} = \begin{bmatrix} u \\ 0_{(Q,1)} \end{bmatrix},$$

where

$$v_{i,j} = \phi(x_i - x_j), \quad n_{i,j} = p_j(x_i). \tag{5}$$

For HJ equations of the form (2) we need to calculate $\nabla\Phi(x)$, so we assume the RBF $\phi(x)$ is well behaved and differentiate (3).

The RBFs ϕ can be compactly or globally supported, and because we must solve (5) it is best if they are positive definite in some sense, implying unique solvability of (5) [20]. In [20] it was shown that there is a direct relationship between ϕ being positive definite and the function $t \to \phi(\sqrt{t})$ being completely monotone, i.e. $f(t) := \phi(\sqrt{t})$ is smooth and satisfies

$$(-1)^m f^{(m)}(t) \geqslant 0, \quad m \in \mathbb{N}_0, \ \ t > 0.$$

If a function is completely monotone, then it is positive definite, thus it does not need to be augmented by any polynomials in (3). However, if we only have that $(-1)^k f^{(k)}(t)$ is completely monotone for some $k > 0$ then ϕ is said to be conditionally positive definite of order k, and requires augmentation by polynomials of degree $k-1$. Using these types of tools analysts have proven the conditional positivity of many RBFs over the years, and so there are numerous ϕ from which to choose. Table 1 shows some useful RBFs and their positive definite order, k.

Wu has also constructed a family of positive definite, compactly supported RBFs [28,30]. All of these functions can be scaled by taking $r \to (r/\theta)$ with θ problem dependent. Because of its radial construction, if a basis function ϕ can be used in n dimensions then it can be used just as well in any dimension less than n. This allows for algorithms and theory to be developed and tested in low dimensions with easy extension to problems in higher dimensions.

For our tests so far we have used the positive definite Gaussian and inverse multiquadric RBFs of the form

$$e^{-\alpha r^2} \quad \text{and} \quad (r^2 + \alpha)^{-1/2},$$

respectively. These do not require augmentation by polynomials when solving (3).

In order to achieve better reconstructions we will attempt to optimize the parameter (e.g. α in $e^{-\alpha r^2}$) on each stencil. It has been shown that the accuracy of a RBF interpolant is inversely related to the condition number of the linear system in (5) [24]. Our optimization consists of choosing a maximum acceptable condition number, $\kappa_{\max}$, and performing an iterative procedure to determine the value of α that yields $0 < \kappa_{\max} - \kappa_\alpha < \beta$ for some tolerance β. In practice $\kappa_{\max}$ is only related to the machine-ε of the given computing system, and we choose $\kappa_{\max} = 10^8$, with $\beta = 10^6$. This need only be done prior to evolution and allows for optimization on different parts of the domain where mesh spacing may vary greatly. In the case of ENO interpolations it is not feasible to test all possible stencils and store the optimal α on each, so we

Table 1
Sample radial basis functions

RBF	$\phi(r)$	k
Polynomials	$r^\beta, \beta > 0, \beta \notin 2\mathbb{N}$	$k > \beta/2$
Thin plate splines	$r^{2\beta}\log r, \beta \in \mathbb{N}$	$k > \beta$
Gaussians	$e^{-\alpha r^2}, \alpha > 0$	$k \geqslant 0$
Multiquadrics	$(c^2 + r^2)^{\beta/2}, \beta > 0, \beta \notin 2\mathbb{N}$	$k > \beta/2$
Inverse multiquadrics	$(c^2 + r^2)^{\beta/2}, \beta < 0$	$k \geqslant 0$

only store a single α that will be acceptable for all the local stencils near a given data point. As long as the mesh is well behaved this α should work adequately on local stencils.

It should be noted that the procedure for choosing an optimal RBF parameter is an area of current research. There are other ways to optimize the parameter of the RBF [8,13,23], and any of these can be incorporated into our framework.

2.1. Example of location dependent RBF parameter optimization

Let us take an example to show both the accuracy and method of parameter optimization. To keep things simple we will work on a 2 point stencil, $\{a, b\}$ in 1D. Obviously the best interpolation of our function, $f(x)$, that we can hope for is linear. In this case the best approximation to $f'(x)$ is

$$\frac{f(b) - f(a)}{b - a},$$

or two-point finite differencing. Without loss of generality let $a = 0$ and $f(a) = 0$. Using a general RBF interpolation with basis function $\phi(r)$ we would like to find a condition on its parameter α such that for our RBF approximation to $f'(a)$, called *app*, satisfies

$$app = (\phi_x(0) \quad \phi_x(-b)) \begin{pmatrix} \phi(0) & -\phi(-b) \\ -\phi(-b) & \phi(0) \end{pmatrix} \Big/ (1 - \phi(-b)^2) \begin{pmatrix} 0 \\ f(b) \end{pmatrix} = \frac{f(b)}{b},$$

using

$$\begin{pmatrix} \phi(0) & -\phi(-b) \\ -\phi(-b) & \phi(0) \end{pmatrix} \Big/ (1 - \phi(-b)^2) = A^{-1} \text{ from (5)}$$

and differentiating (3). By multiplying we see that if we assume $\phi_x(0) = 0$, which it should for all smooth RBF basis functions, and scale ϕ so that $\phi(0) = 1$, then we have

$$app = \frac{-y'y}{1 - y^2} f(b), \tag{6}$$

where $y' = \phi_x(-b)$ and $y = \phi(-b)$. Solving the ODE

$$\frac{-y'y}{1 - y^2} = \frac{1}{b} \tag{7}$$

will give solutions $y(b) = \phi(b)$ that yield an equivalent RBF interpolation to two-point finite differencing. If we let $\phi(b) = e^{-\alpha b^2}$ and let $\alpha \to 0$, then we have a solution to (7). If we would like $\phi(r) = \alpha r^2$, then we need

$$\alpha = \sqrt{\frac{1}{3b^4}}.$$

Other basis functions will have different restrictions on α.

Note that for the Gaussian α does not depend on b, which makes its optimization straightforward. In practice if we wanted to optimize α when $\phi = e^{-\alpha r^2}$ all we would need to do is let $\alpha \to 0$ until we find that $|\kappa_{\max} - \kappa_\alpha| < \epsilon$. Even for higher order approximations (with more nodes in the stencil) it can be shown that letting $\alpha \to 0$ with $\phi = e^{-\alpha r^2}$ approaches the optimal solution.

For other ϕ whose optimal parameter may depend on the data locations (e.g. b above), we run a root finding method on the equation $\kappa_{\max} - \kappa_\alpha = 0$ and iterate in α until $|\kappa_{\max} - \kappa_\alpha| < \epsilon$. For many basis func-

tions there exist a priori local error estimates in terms of α which clearly indicate limiting values of α for which to strive during optimization [24,25].

We also note that a given $\Phi(x)$ can be represented in a Lagrange type fashion

$$\Phi(x) := \sum_{j=1}^{M} \Psi_j(x) u_j. \tag{8}$$

We can find the coefficient $\Psi_i(x_0)$ in (8) by setting $u_j = \delta_{i,j}$ for $j = 1, \ldots, M$ and solving

$$\Phi(x_0) = \sum_{j=1}^{M} \Psi_j(x_0) u_j = \Psi_i(x_0) u_i. \tag{9}$$

3. Data access in the meshless computational domain

In this section we present the way in which we give some structure to our unstructured data set, allowing access of nodes in a reasonable amount of time. If our data set, X, were to be stored as simply an unordered list of points, then each time we needed to access a neighbor of a given node, x_i, we would need to search through the entire list giving a $O(N)$ algorithm versus $O(1)$ on a uniform grid. Although the storage required for this method is minimal, the access time is much too slow for practical use.

Instead, we use a binning method. This method divides the entire domain, $\Omega \supset X$, into a coarse, structured grid C. Then for each coarse grid cell $c_j \in C$ we create a list of all the nodes of X that lie inside c_j. When a neighbor of $x_i \in c_j$ needs to be accessed we only need to search the lists of the coarse neighbors of c_j. So the total neighbor access time is $O(1)$ to access the coarse neighbor list times $O(list_j)$ to search the list and find the neighbor. Of course this procedure can be iterated over multiple coarse levels so that the list sizes are smaller, and other optimizations can be done such as noting which coarse cells are nonempty. Similar ideas have been explored in the context of local level set methods [29].

For the evolution procedure we can find all appropriate stencils prior to time evolution if we would like, and then the problem of neighbor access is only relevant in the preprocessing step and does not slow the evolution down. This idea was used in [31].

4. Monotone fluxes

4.1. Introduction to monotone schemes

In solving equations of the form (2) an important class of numerical methods are monotone schemes [10]. When they are also consistent these schemes have been shown to converge to the physically correct viscosity solution of (2).

For uniform data in 1D there are numerous schemes available [22], and these schemes can be generalized for uniform data in higher dimensions. In 2D on triangulated data there has been progress as well [2,9,19]. However, in higher dimensions there has not been as much progress for scattered data. One drawback is that as the dimension grows, the triangulation becomes very complex and storage consuming ($O(M^{\lceil d/2 \rceil})$ simplices for M points in d dimensions), and the number of neighbors of a given node grows very large.

In this section we will present some new monotone schemes for scattered data in an arbitrary dimension that is not required to be triangulated. We will also discuss some details on implementation.

4.2. Derivation of schemes

Given a Hamilton–Jacobi equation of the form (2), i.e.

$$\varphi_t + H(\nabla\varphi) = 0 \tag{10}$$

to be solved on a point set, we would like to derive a first-order in time monotone scheme, see [2] for a similar analysis. We will use two dimensions for simplicity here, and let i and j be 2D multi-indices. The scheme will be of the form

$$\varphi_i^{n+1} = \varphi_i^n - \mathrm{d}t\hat{H}_i(\varphi^n), \tag{11}$$

where φ_i^n is the numerical approximation to the solution of (2) at $(t = t_n, x = x_{i_1}, y = y_{i_2})$, and $\hat{H}_i$ is the numerical Hamiltonian there. The requirement for a method to be monotone [10] is that

$$u_i^n \geqslant v_i^n \ \forall i \Rightarrow u_i^{n+1} \geqslant v_i^{n+1} \ \forall i.$$

For our scheme of the form (11) this means that if we fix an index i_0, then at x_{i_0},

$$\frac{\partial\hat{H}_{i_0}}{\partial\varphi_j} \leqslant 0 \quad \text{and} \quad 0 \leqslant \mathrm{d}t \leqslant \left(\frac{\partial\hat{H}_{i_0}}{\partial\varphi_{i_0}}\right)^{-1} \qquad \forall j \neq i_0. \tag{12}$$

Thus our goal will be to find a numerical Hamiltonian satisfying (12).

Guided by the fact that some of the standard monotone schemes on uniform grids, such as Lax–Friedrichs, are approximations to solving the vanishing viscosity equation

$$u_t + H(\nabla u) = \epsilon\Delta u \quad \text{as } \epsilon \to 0,$$

we will construct our numerical Hamiltonian as an approximation of

$$H(\nabla u) - \epsilon\Delta u.$$

The procedure will be to reconstruct u near a given point, x_i, using an interpolation method and then differentiate the interpolant to get ∇u and Δu. The interpolation method we will use is RBF reconstruction.

4.2.1. Monotonicity

If the basic time evolution procedure at node i can be written as

$$u_i^{n+1} = u_i^n - \mathrm{d}t\{H(\nabla u^n) - \epsilon_i\Delta u^n\} = u_i^n - \mathrm{d}t\, G_i(u_{j1}, \ldots, u_{N_{\text{sten}}}), \tag{13}$$

where N_{sten} is the number of nodes used in the stencil approximating ∇u^n and Δu^n, then we will need to find an ϵ_i that satisfies all the inequalities in (12). Thus for each node i we should be able to calculate a minimal diffusion constant ϵ_i that guarantees monotonicity there. If we decide to evolve our solution using (13) with a unique ϵ_i at each node i then the method will be called a local Lax–Friedrichs scheme. If we decide to take $\epsilon_{\max} = \max_i \epsilon_i$ and evolve (13) using $\epsilon_i = \epsilon_{\max}\ \forall i$, then the scheme will be called simply Lax–Friedrichs.

To find the appropriate size of ϵ_i we begin by writing our reconstructed partial derivatives in the Lagrange form (8). If we know that

$$\frac{\partial u}{\partial x_k}(x_i) \approx \sum_{j=1}^{N_{\text{sten}}} c_{k,j}u_j, \ \ k = 1, 2, \quad \text{and} \quad \Delta u(x_i) \approx \sum_{j=1}^{N_{\text{sten}}} d_j u_j$$

then

$$\frac{\partial G_i}{\partial u_j} = \nabla H(Z) \cdot (c_{1,j}, \ldots, c_{N_{\dim},j}) - \epsilon_1 d_j,$$

where $Z_k \in [\min(\nabla u)_k, \max(\nabla u)_k]$.

Note that in the construction of our stencil if we do not have that

$$d_j > 0 \quad \forall j \neq i \text{ and } d_i < 0, \tag{14}$$

then we have a bad stencil which cannot yield a monotone scheme. In practice we have many more restrictions than (14), because this restriction can allow arbitrarily large diffusion terms which smear our solution. So we make sure to enforce restrictions on the relative sizes of c_j and d_j to keep diffusion to a minimum. Details of how this is done will be presented later.

We will now construct an ϵ_1 that satisfies the first $N_{\text{sten}} - 1$ inequalities in (12), and an ϵ_2 that satisfies the $\mathrm{d}t$ inequality, then finally set $\epsilon_i = \max(\epsilon_1, \epsilon_2)$. Thus we need

$$\left(\sum_{k=1}^{N_{\dim}} \max_{x \in \Omega} |H_k(x)||c_{k,j}| \right) - \epsilon_1 d_j \leqslant 0, \quad j \neq i.$$

The ϵ_1 for the scheme at x_i satisfies

$$\max_{j \neq i} \left[\frac{\sum_{k=1}^{N_{\dim}} \max_x |H_k(x)||c_{k,j}|}{d_j} \right] \leqslant \epsilon_1, \tag{15}$$

so to minimize the viscosity we choose ϵ_1 to satisfy the equality in (15).

Next we find ϵ_2 that satisfies

$$0 \leqslant \mathrm{d}t \leqslant \left(\frac{\partial G_i}{\partial u_i} \right)^{-1} \tag{16}$$

where

$$\frac{\partial G_i}{\partial u_i} = \nabla H(Z) \cdot (c_{1,i}, \ldots, c_{N_{\dim},i}) - \epsilon_2 d_i.$$

So we need that ϵ_2 satisfies

$$\left[\frac{\sum_{k=1}^{N_{\dim}} \max_x |H_k(x)||c_{k,i}|}{-d_i} \right] \leqslant \epsilon_2 \tag{17}$$

as $d_i < 0$. As this must hold for all x we choose ϵ_2 to satisfy the equality in (17), and finally choose $\epsilon_i = \max(\epsilon_1, \epsilon_2)$.

The CFL condition is then given by

$$\mathrm{d}t \leqslant \frac{1}{\sum_{k=1}^{N_{\dim}} \max_x |H_k(x)||c_{k,i}| - \epsilon_i d_i}.$$

4.2.2. Consistency and convergence

The consistency of schemes using RBF interpolants has not been fully explored as of yet. There is a large amount of research demonstrating the error bounds of RBF interpolants and their convergence properties, but the strict definition of consistency where we require that

$$\hat{H}(p) = H(p) \quad \text{if } u(x) \equiv \mathbf{p}\cdot\mathbf{x} + k_0$$

has not been proven for general basis functions ϕ. However, it is only machine precision that limits us from getting an interpolant, Φ such that

$$|\nabla\Phi - p| \leqslant \delta_1 \quad \text{and} \quad |\Phi_{x_i x_i} - 0| \leqslant \delta_2 \qquad \text{for } i = 1 : N_{\text{dim}}, \tag{18}$$

for arbitrarily small δ_1, δ_2, given an underlying linear function $u(x) \equiv \mathbf{p}\cdot\mathbf{x} + k_0$.

Thus, if consistency is wanted we can adjust the Lagrange coefficients of Φ_{x_i} and $\Phi_{x_i x_i}$ so that they exactly reproduce u_{x_i} and $u_{x_i x_i}$ when $u(x) \equiv \mathbf{p}\cdot\mathbf{x} + k_0$. We must satisfy (18) with $\delta_1, \delta_2 = 0$. Assume at a given node we have a Lagrange representation for $\Phi_{x_1} \equiv \sum_{j=1}^{N} c_j u_j = \mathbf{c}\cdot\mathbf{u}$, and that the underlying function that we are trying to reconstruct is $u(x) \equiv \mathbf{p}\cdot\mathbf{x} + k_0$. If our interpolation is not consistent, then

$$\mathbf{c}\cdot\mathbf{u} - p_1 = \sum_{j=1}^{N} c_j \left(\sum_{i=1}^{N_{\text{dim}}} (p_i x_{j,i}) + k_0 \right) - p_1 = r, \tag{19}$$

where $x_{j,i}$ is the ith coordinate of the jth stencil node.

To eliminate this residual r we need to find new coefficients d_j that satisfy

$$\sum_{j=1}^{N} d_j = 0, \quad \sum_{j=1}^{N} d_j x_{j,1} = 1, \quad \sum_{j=1}^{N} d_j x_{j,i} = 0 \quad \text{for } i = 2 : N_{\text{dim}}. \tag{20}$$

To find these d_j we solve

$$\sum_{j=1}^{N} b_j = -\sum_{j=1}^{N} c_j, \tag{21}$$

$$\sum_{j=1}^{N} b_j x_{j,1} = 1 - \sum_{j=1}^{N} c_j x_{j,1}, \tag{22}$$

$$\sum_{j=1}^{N} b_j x_{j,i} = -\sum_{j=1}^{N} c_j x_{j,i} \quad \text{for } i \neq 1, \tag{23}$$

and take $\mathbf{d} = \mathbf{c} + \mathbf{b}$ to replace $\mathbf{c}$. If the system determined by (21)–(23) is underdetermined (which it will be if $N > N_{\text{dim}} + 1$), then we take the least squares solution $\mathbf{b}$, thus moving our new approximation as little as possible from the original Φ. In practice we have found that $\|\mathbf{b}\|$ is usually very small.

Similarly, if the approximation for $\Phi_{x_i x_i}$ yields

$$\mathbf{c}\cdot\mathbf{u} = \sum_{j=1}^{N} c_j \left(\sum_{i=1}^{N_{\text{dim}}} (p_i x_{j,i}) + k_0 \right) = r, \tag{24}$$

where $r \neq 0$, we need

$$\sum_{j=1}^{N} d_j = 0, \quad \sum_{j=1}^{N} d_j x_{j,i} = 0 \quad \text{for } i = 1 : N_{\text{dim}}. \tag{25}$$

So we solve a similar system to (21)–(23) and take $\mathbf{d} = \mathbf{c} + \mathbf{b}$.

It should be noted that the diffusion coefficient ϵ needs to be chosen after finding these new consistent, monotone Lagrange formulations for $\nabla\Phi$ and $\Delta\Phi$.

Having a consistent scheme, in order to obtain a convergence estimate we could construct a doubling-variable function and proceed along the lines of the proof in [3,9,10]. However, we focus on the numerical implementation here.

4.3. Implementation details of schemes

We can see that there are some inequalities which must be satisfied by our RBF reconstruction at a point x_i before our scheme is deemed monotone. Firstly, we must satisfy (14). Once this is done it is straightforward to calculate the diffusion terms ϵ_i and CFL condition.

Thus we explain how we find a stencil at x_i that satisfies (14). Actually we will tighten the restrictions on (14) significantly, as it allows for diffusion terms that are too large. What we require is that (14) holds, but also that $\epsilon_i < \epsilon_{\max}$ for a specified $\epsilon_{\max}$ which is usually dependent on the spacing of the local mesh. Generally what this requires is that any Lagrange coefficient, l_j, of φ_j in the approximation to $\varphi_{x_i x_i}$ will have magnitude $\approx c_j \mathrm{d}x$, as it would if we were using finite difference approximations.

Once we have decided on the bounds for ϵ and d we can begin searching for acceptable stencils at a given node, x_i. The problem that usually arises is that for a given candidate stencil, S_C, one or more of the coefficients d_j of the $\Delta\varphi$ approximation are too small in magnitude or the wrong sign because x_j is either collinear or almost collinear with another node in S_C. Thus we will try to make our stencil as isotropic as possible. To do this we will decide on a stencil size, $N+1$ (the stencil will always include the node x_i), define N equispaced rays, $x_i + v_k t,\ t > 0$ emanating from x_i and find the neighbor x_j of x_i that maximizes

$$V = \frac{x_j - x_i}{\|x_j - x_i\|} \cdot \frac{v_k}{\|v_k\|} \tag{26}$$

by searching through the bins of the coarse grid that are near x_i, and only allowing points x_j such that $\|x_j - x_i\| < \xi(x_i)$, where the radius $\xi(x_i)$ is a function of the local density of nodes, the desired stencil size, and the dimension.

For example in 2D if the local density of nodes is $\rho = 10/h^2$, and the desired stencil size is 5, then we choose ξ such that $\pi\xi^2/h^2 \approx 5/10$. Then the vectors v_k are chosen as

$$(\cos, \sin)(\theta_0 + 2\pi k/N) \quad \text{for } k = 0 : N-1.$$

We search over a few different orientations (θ_0 in 2D) of the axes for a fixed stencil size, and stop when we find an acceptable stencil that satisfies our conditions on ϵ and d. As noted in Section 2 we optimize the RBF interpolation on each candidate stencil by trying to find the best function parameter for the basis function. If none of the candidates satisfy our bounds for ϵ and d then we change the stencil size by either increasing it or decreasing it by 1, and repeat the search for a prespecified number of increments, I, until we find an acceptable stencil. If we cannot find an acceptable stencil then our mesh is very bad and we must use the best of the candidate stencils we have examined. However, this has not yet occurred in our computations. If this were to occur it would mean that our scheme would not be guaranteed to be monotone under the current restrictions on ϵ and d. Thus, we can examine the possible stencils that failed to satisfy the original criteria for ϵ and d, and see if any of them satisfy the minimal requirements of (14). Then we choose the stencil that has the smallest diffusion coefficient ϵ, and also satisfies (14). While this last procedure may yield a large diffusion term, it does guarantee a monotone scheme.

The only ambiguous point in the description above is the definition of a *neighbor* of x_i. Unless a triangulation of the data is constructed we do not have a rigorous definition of what a neighbor is. One method is to search through the coarse cells near x_i for the M closest points, where M is arbitrary but on the

order of the stencil size. The neighbors of x_i are then said to be these M closest points. However, this may not work well for stencils with large discrepancies in distances between nodes near x_i. Therefore it is possible to adjust the stencil choosing algorithm so that given an axis v_k as described above we maximize a function $f(x_j - x_i, v_k)$ instead of (26), where f could penalize $\|x_j - x_i\|$ and perhaps place increased weight on the value V obtained in (26), such as taking $f \approx V^2$ as $\|V\| \leqslant 1$. In our implementation we use (26), which corresponds to $f(V) = V$, and we have run tests with $f(V) = V^2, V^3$ and have not found significant differences in the results. This topic of defining and finding a neighbor of a node on a meshless grid without creating a triangulation warrants further research.

Another option is to create a local triangulation of the M closest points to x_i and use this to define neighbors. We have not implemented this procedure yet. This method will take more time, but will in general give a smaller number of neighbors and more compact candidate stencils. As long as this local triangulation is not stored permanently this method is acceptable. It is when triangulations of large data sets in high dimensions must be stored that we exhaust memory restrictions.

Again it should be noted that this search for acceptable stencils need only be done prior to evolution if we are willing to store the nodes of the stencil at x_i.

5. A Roe–Fix scheme

Given that we are able to construct a prototypical Lax–Friedrichs scheme, we are tempted to push further and find a monotone scheme with even less diffusion. For uniform grids in 1d and even triangulated grids in higher dimensions there are upwind schemes which can be proved to be monotone [5]. These are based specifically on linear reconstructions (standard two-point upwinding in 1D). When upwinding is used at all non-sonic points, combined with a vanishing viscosity approximation such as LF or LLF at sonic points we have a method known as Roe–Fix or RF. This would be readily implementable were we to have a definition of upwinding that applies to our RBF reconstructions, but unfortunately we do not. However, we can construct a RF method using RBF interpolation and make an argument as to its convergence properties.

If we are advancing the solution at a node x_i the first thing we must determine is whether or not we are at a sonic point. Assume we have constructed a suitable stencil S_i at x_i, adhering to the constraints of Section 4.2. At each node $x_j \in S_i$ we calculate $\nabla H(\nabla\varphi(x_j))$ using the stencil S_j. If $\partial H/\partial\varphi_{x_k} \equiv H_k$ changes sign for any $k = 1, \ldots N_{\text{dim}}$ when searching over $j = 1, \ldots, |S_i|$ then we are at a sonic point and we advance the solution using either LF or LLF schemes. If H_k does not change sign then we are not at a sonic point, so we would like to use upwinding. Since we do not have a triangulation of the nodes surrounding x_i we cannot choose a triangle, T_c, from which the characteristics are flowing and then use the nodes of T_c to linearly reconstruct the function yielding a monotone, upwind scheme. However, as long as the nodes of S_i surround x_i sufficiently we have encompassed the domain of dependence for φ_i^{n+1}, assuming the CFL condition is small enough. Here, surrounding x_i means that the convex hull of S_i contains x_i. Thus, if φ is smooth near x_i then our RBF reconstruction can be interpreted as a higher order reconstruction extended from the linear interpolant φ_L on T_c. This reconstruction for φ should then only differ from φ_L by terms of order $O(dx^p)$ where $p \geqslant 2$. Using this interpolant for φ and dropping the artificial diffusion terms of the LF and LLF schemes should then give us a scheme that differs from a monotone scheme by $O(dt\,dx)$, similar to the argument given in [22] for high order ENO schemes. Because of the lack of the artificial diffusion term we should see better resolution.

Note that we require φ to be smooth near x_i for this argument to be valid. This is usually the case when we are far away from sonic points, but will not be the case when we are at a moving kink, which is a moving discontinuity in first derivatives. In that case the RBF interpolant may extend over the discontinuity and differ from φ_L significantly. So our scheme may differ from a monotone scheme by more than $O(dt\,dx)$

there. If a monotone method is desired at points of this type then we can easily insert a check into our algorithm such that when the jump in derivatives of φ near x_i is too large it will trigger the use of LF or LLF schemes even if the signs of H_k indicate upwinding.

6. High order ENO reconstruction

While the ability of monotone schemes to correctly converge to the viscosity solution of Hamilton–Jacobi equations makes them desirable, they do have an undesirable property: they are at most first order accurate [9,11,15]. In one dimension on uniform grids this drawback is overcome by taking ENO polynomial function reconstructions that avoid using interpolants which cross discontinuities, causing spurious oscillations. The familiarity of polynomials and the ability to simply construct their derivatives using divided differences made the ENO methods for conservation laws and Hamilton–Jacobi equations very popular [16,22,26]. In multiple dimension on nonuniform grids there has been some progress using polynomials [4,31], and RBFs [18]. Here we present an incremental stencil selection method which exploits the LU factorization of the RBF coefficient matrix. We also introduce a self-similar smoothness indicator that allows the ENO stencil to be chosen.

Our ENO reconstruction will involve extending an existing reconstruction in a smooth fashion. Assuming we have a reconstruction $\Phi_M(x)$ on an existing M point stencil $S_M = \cup_{i=1}^{M}\{P_i\}$, where $P_i \in X$, that has been constructed starting with $S_r \ni x_k = P_1$ for $r \leqslant M$, we want to extend it to $M+1$ nodes in an ENO fashion.

There are two pieces of information we need which are somewhat arbitrary. One is the choice of nodes from which to choose P_{M+1}, and the other is the measure of smoothness of our reconstructed function, $\Phi_{M+1}(x)$. A suggestion for choosing the candidates $\{P_{M+1}^{cand_j}\}$ for P_{M+1} is that we choose the N (again arbitrary, but finite) nodes $P_{M+1}^{cand_j} \in X$ that make the center of gravity of the stencil $S_M \cup P_{M+1}^{cand_j}$ closest to x_k, or closest to the center of gravity of S_M. In practice we choose $\{P_{M+1}^{cand_j}\}$ from a small selection ($N < 5$) of nodes that make the center of gravity of the stencil $S_M \cup P_{M+1}^{cand_j}$ closest to x_k. The number of choices we have for candidates depends on the surface area of the existing stencil, S_M, so in higher dimensions the potential cost incurred by maximizing $card(\{P_{M+1}^{cand_j}\})$ becomes prohibitive. There are many strategies to extend stencils [4,14,27,31], however, these are usually based on polynomial reconstruction and take steps so as to ensure the interpolation coefficient matrix has a good condition number. For RBF reconstruction the condition number depends on ϕ and the stencil, and in practice we have not found any problems with it. Since ϕ is radially symmetric there should not be any directional bias which causes polynomials to have badly conditioned coefficient matrices, see [1] for details about this problem.

For the measure of smoothness we use the self-similar indicator

$$\beta = \sum_{2 \leqslant |\alpha| \leqslant s} \int_{P_k} |P_k|^{(2|\alpha|-N)/N} (D^\alpha \Phi(x))^2 \mathrm{d}x, \tag{27}$$

where $|P_k|^{(2|\alpha|-N)/N}$ makes β invariant under grid scaling in N dimensions when $|P_k|$ is the area of a grid cell containing P_k, α is a multi-index, and s is proportional to the size of the stencil [31]. For polynomial interpolants, we can take s is the order of the interpolant, but with RBFs Φ can be a weighted average of C^∞ functions. Therefore we take s proportional to stencil size because we cannot expect that $D^\alpha u$ is influencing $D^\alpha \Phi$ for derivatives of order $\geqslant |\alpha|$ if we are using far fewer than $|\alpha|$ points. In practice we choose s as the largest n such that $\#\ Sten \approx \binom{n+N}{N}$.

Once we have made the above decisions we can proceed systematically to obtain the $M+1$st stencil. When constructing the Mth stencil it was necessary to solve the system of equations (4), that we will write $A\gamma = \bar{u}$ which is usually small enough to be done by Gaussian elimination/LU factorization. Noting that A is symmetric we have an $M \times M$ LL^t factorization (here we assume that $Q = 0$ in (3)). For the $M+1$st cell we must solve a new $A\gamma = \bar{u}$ that can be written as

$$A = \begin{bmatrix} LL^t & \alpha \\ \alpha^t & d \end{bmatrix} = \begin{bmatrix} \bar{u} \\ \bar{u}_{M+1} \end{bmatrix},$$

where α and d are found as in (5). To obtain L_{M+1} we compute the Schur complement $S = d - \alpha^t (LL^t)^{-1}\alpha$ of LL^t and get

$$L_{M+1} = \begin{bmatrix} L & 0_{(M,1)} \\ \alpha^t (L^t)^{-1} & \sqrt{S} \end{bmatrix}.$$

Having the new L_{M+1} we can find the new set of $\{\gamma_j\}$ and compute β for each candidate stencil. The stencil with the smallest β is chosen as the $M+1$st stencil.

We note that in practice all L's can be found and stored before the time evolution begins as long as data point set doesn't change during the calculation. Thus, the above Schur complement procedure does not save as much time as when it is applied to an adaptive mesh, where the linear system inversions must be done at each timestep. However, for large fixed data sets in higher dimensions the storage of the L matrices can become too large to be practical, and any acceleration to the matrix inversion procedure is helpful.

7. Time derivatives

We use total variation diminishing (TVD) Runge–Kutta (RK) methods for time advancement [26]. The procedure is as follows: given a node x_i and function values at time t_n we define the operator

$$L_i = -\mathrm{d}t\hat{H}(\varphi^n),$$

where $\hat{H}$ is the numerical Hamiltonian. We then advance the solution using a Runge–Kutta procedure of the form

$$\varphi_i^{(k)} = \sum_{m=0}^{k-1} [\alpha_{km}\varphi_i^{(m)} + \beta_{km}L_i^{(m)}], \quad k = 1, \ldots, r,$$

where $\varphi_i^{(0)} = \varphi_i^n$, $\varphi^{(r)} = \varphi_i^{n+1}$. If the forward Euler version (i.e. $r = 1$, $\alpha_{1,0} = 1$, $\beta_{1,0} = 1$) is TVD under the CFL condition

$$\mathrm{d}t/\mathrm{d}x \leqslant \lambda_0,$$

then the RK method can be proven to be TVD under the CFL condition

$$\mathrm{d}t/\mathrm{d}x \leqslant C_r\lambda_0.$$

Coefficients for the popular second- and third-order TVD-RK methods are shown in Table 2.

Together the ENO and TVD-RK methods give highly accurate solutions and can be quickly adapted to almost any Hamiltonian H.

8. Outline of evolution procedure

In this section we will outline the procedure for solving (2) given initial values $\varphi_0(x_j)$ on a dataset $X = \{x_j\}$ of points contained in the computational domain, Ω.

1. Construct a coarse mesh C over Ω and for each coarse grid cell $c_i \in C$ create a list of all the nodes of X that lie within in c_i. If C is uniform then this should take $O(|X|)$ time. An iterated coarse mesh can also

Table 2
TVD RK coefficients

Order	α_{kl}			β_{kl}			C_r
2	1			1			1
	1/2	1/2		0	1/2		
3	1			1			1
	3/4	1/4		0	1/4		
	1/3	0	2/3	0	0	2/3	

be constructed or any other mechanism which allows the user to determine the $M(\ll |X|)$ closest points to a given node in less than $O(|X|)$ time.

2. For each $x_i \in X$
   do
   - Construct a new candidate stencil S_C, using the guidelines of Section 4.3.
   - Optimize RBF parameter α on S_C.
   - Determine if d_j's and ϵ are acceptable for S_C,

   while (d_j's and ϵ are unacceptable).
   Set the chosen stencil $S_i = S_C$.

   It is actually a matter of memory versus time as to what the user stores here. If memory is abundant and its access is fast then for each stencil all the Lagrange coefficients (c_j's, d_j's) and ϵ_i can be stored, making the evolution procedure faster. If memory is scarce then just the nodes of the stencil and the optimal RBF parameters α should be stored.
3. For $t_n = 0 : T$,
   do
   - Compute $\nabla\varphi$ at all nodes using stencil from step 2.
   - If higher order accuracy of $\nabla\varphi$ is desired then use an ENO reconstruction for φ as described in Section 6.
   - For each node, if using RF scheme determine if sonic fix is necessary using ∇H. If so, or if the scheme is LF or LLF, then compute $\Delta\varphi$ and diffusion weight ϵ.
   - For each node, advance solution one step in time using $\varphi^{n+1} = \alpha\varphi^n - \beta\,\mathrm{d}t\hat{H}(\varphi^n)$.

   If RK method is being used then go to beginning of this do loop as many times as appropriate.

end do loop.

If adaptive grid is being used repeat stencil finding procedure in step 2, otherwise go to the beginning of this for loop.

9. Numerical examples

Unless otherwise noted the examples are calculated on a domain of $[-1, 1]^d$ in d dimensions. Figures showing multiple evolution frames should be read from left to right and then top to bottom chronologically.

We begin with a level set evolution of the form

$$\varphi_t - |\nabla\varphi| = 0,$$

calculated on a grid of points that lie on concentric circles as in Fig. 1. The nodes used lie at the vertices of the triangulation shown. Note that this triangulation is not necessary for our calculation and is only used in

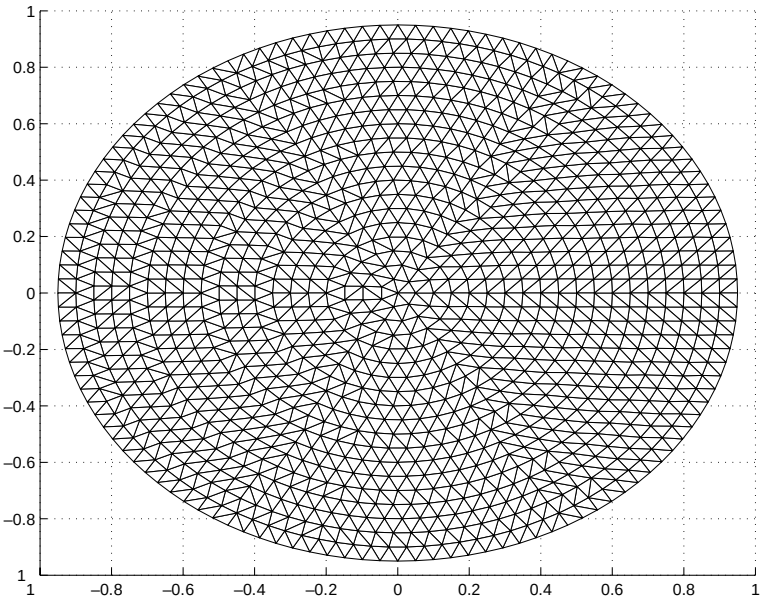

Fig. 1. 2D concentric grid, nodes at vertices of triangles.

the visualization. As the characteristics flow outward we use "upwind" reconstruction stencils at the boundary consisting of nodes within the domain. Fig. 2 shows how our method captures the vanishing viscosity solution.

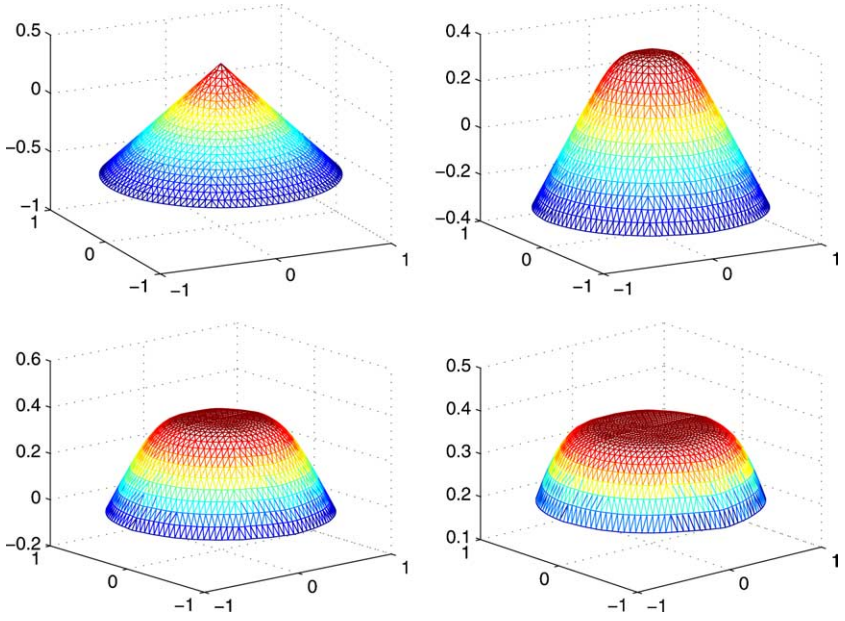

Fig. 2. Evolution of (2) with $H(p) = -|p|$.

The accuracy and convergence order of this method are shown in Table 3. The error is measured using the data points that lie on the concentric circle with radius = 0.4. Each reconstruction stencil used contains six nodes.

In Table 4 we show an accuracy and convergence analysis of a smooth solution on a uniform grid using periodic BCs where we solve

$$\varphi_t + \varphi_x + \varphi_y = 0, \quad \varphi(x, y, t = 0) = \cos(\pi(x + y)).$$

The error of is measured at time $t = 1.0$.

Fig. 3 shows the solution of

$$u_t + \sin(u_x + u_y) = 0,$$

calculated on a uniform grid, but using our meshless method. Periodic BCs are imposed in both directions. For this example the ENO reconstruction is used. The initial stencil at $x_{i,j}$ is a five-point centered stencil consisting of points

$$x_{i,j},\ x_{i\pm1,j},\ x_{i,j\pm1},$$

and is extended in an ENO fashion by adding one of the diagonal points $x_{i\pm1,j\pm1}$.

In Fig. 4 the solution to Burgers' equation

$$u_t + 0.5(u_x + u_y + 1)^2 = 0,$$

with periodic BCs, is shown.

In Figs. 6 and 7 we show level set solutions of

$$\varphi_t - |\nabla\varphi| = 0,$$

for initial conditions of a sphere and torus. Again characteristics flow outward and boundary reconstructions use interior point stencils. The computational domain consists of nodes that are approximately equispaced, lying on concentric spheres as in Fig. 5.

Figs. 8–10 show evolution sequences of the level set of a four-dimensional hypertorus initialized as

Table 3
2D convergence order analysis for $H(p) = -|p|$ on concentric grid

$\approx dx$	L^1 error	L^1 rate	L^∞ error	L^∞ rate
0.1	0.013637	–	0.020314	–
0.05	0.002740	2.315	0.006998	1.537
0.025	0.000425	2.686	0.002554	1.454
0.0125	0.000089	2.243	0.000971	1.395

Table 4
2D convergence order analysis for $H(\nabla\phi) = \phi_x + \phi_y$ on a uniform grid with periodic BCs

dx	L^1 error	L^1 rate	L^∞ error	L^∞ rate
0.1	0.0539332	–	0.0860662	–
0.05	0.012960489	2.057053543	0.0205767	2.064435145
0.025	0.002687455	2.269807805	0.00423452	2.280741279
0.0125	0.0000936797	4.842360447	0.0001479	4.839504486

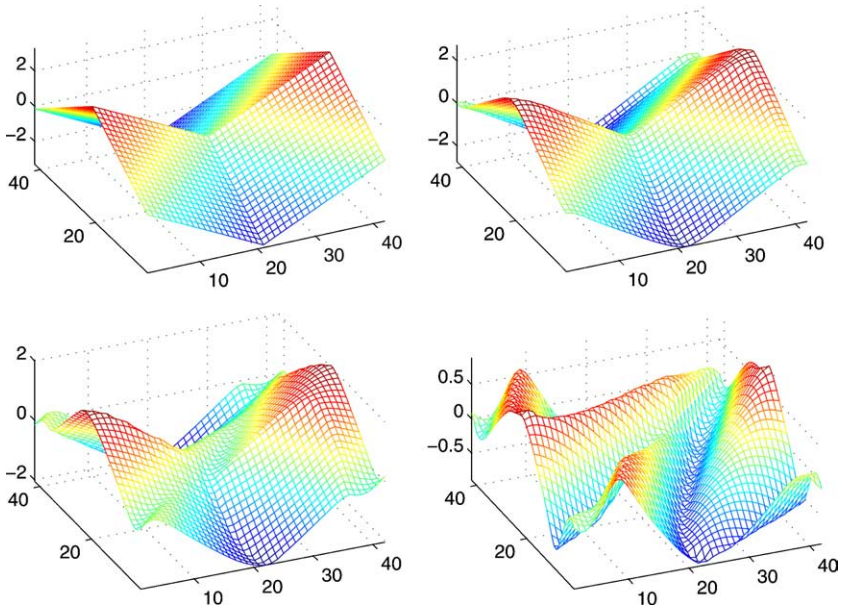

Fig. 3. Evolution of (2) with $H(u_x, u_y) = \sin(u_x + u_y)$.

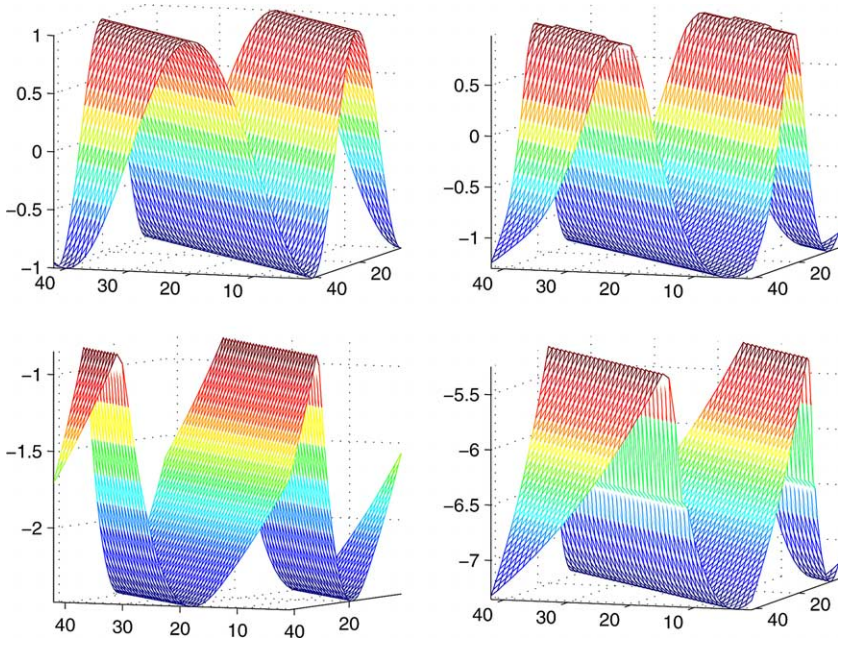

Fig. 4. Evolution of (2) with $H(u_x, u_y) = 0.5(u_x + u_y + 1)^2$.

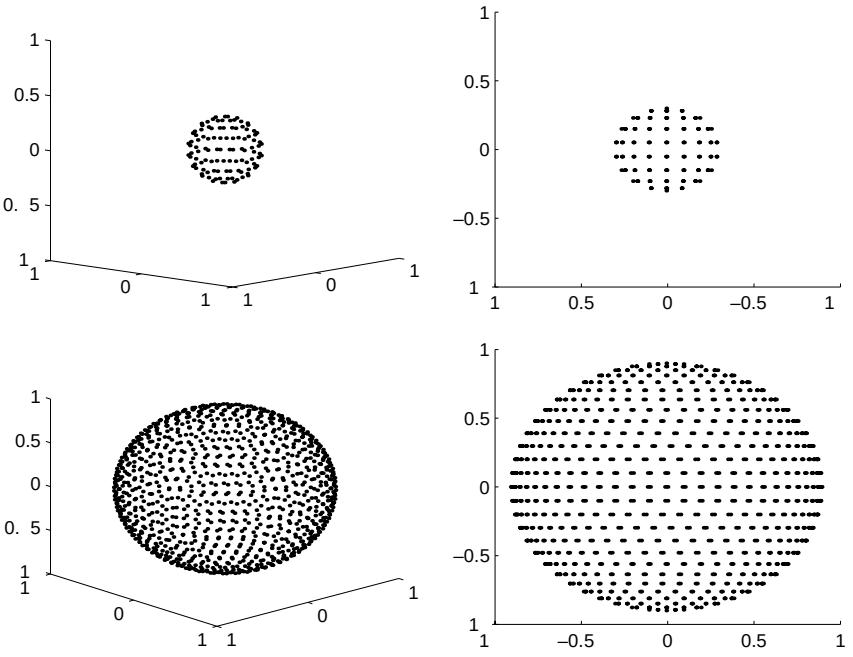

Fig. 5. Points on smaller (top) and larger (bottom) concentric spheres used for 3D unstructured grid calculation.

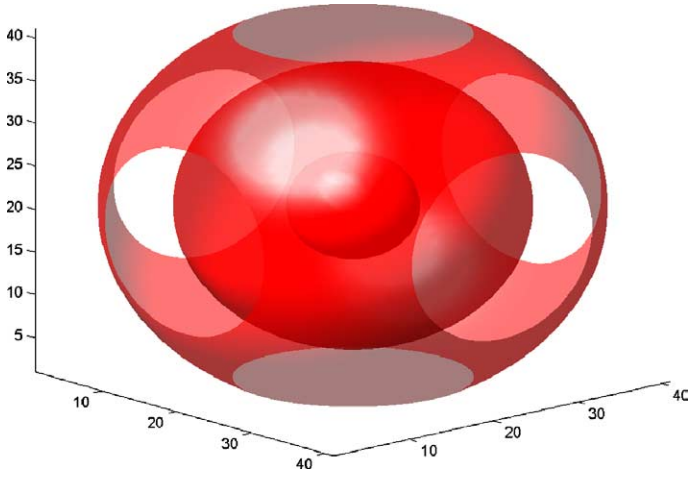

Fig. 6. Level set evolution of (2) with $H(p) = -|p|$. Initial time view and later time views as sphere expands.

$$\varphi(x,y,z,w) = r_3 - \sqrt{w^2 + \left[\sqrt{z^2 + \left[\sqrt{y^2+x^2} - r_1\right]^2} - r_2\right]^2},$$

where $r_1 = 0.2, r_2 = 0.4, r_3 = 0.8$, subject to the PDE

$$\varphi_t - |\nabla\varphi| = 0.$$

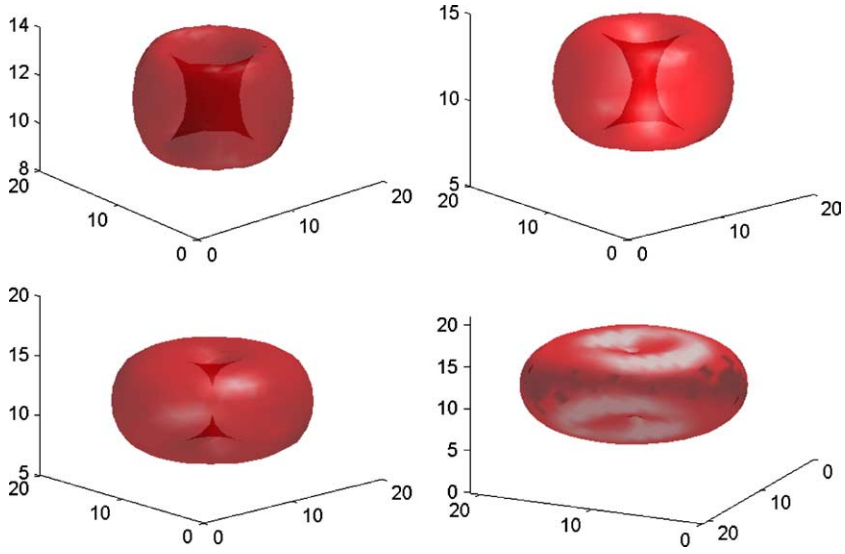

Fig. 7. Level set evolution of (2) with $H(p) = -|p|$.

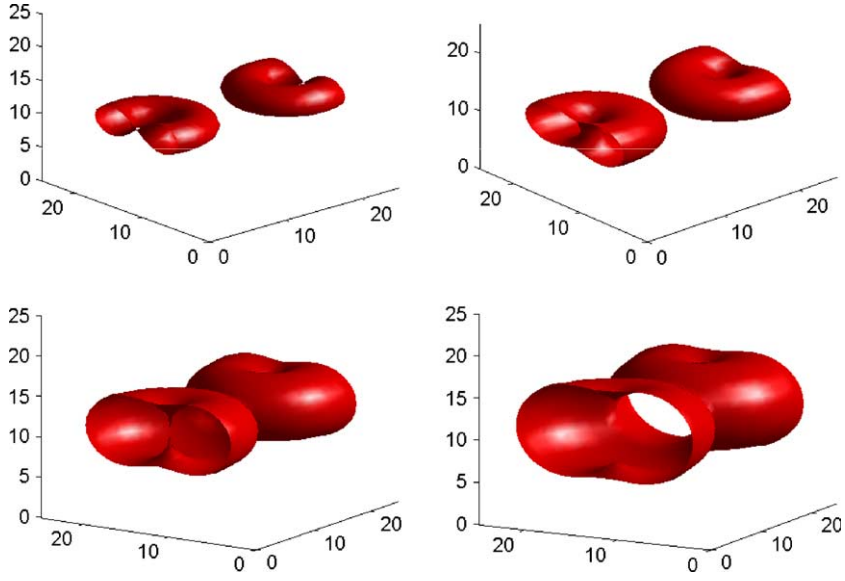

Fig. 8. Level set of fixed dimension 1 slice of a 4D hypertorus evolving under (2) with $H(p) = -|p|$.

In each figure we take a 3D slice of the data keeping the coordinate x_i fixed for the indicated ith dimension, and then plot the level set $\{x|\varphi(x) = 0\}$ as a surface in 3D. In the future we will implement a local level set framework allowing for high dimensional computations of this type to be performed without storing gridpoints that are far away from the interface. The storage for this method would then be on the order of the size of the interface (an interface that is of codimension $\geqslant 1$ with respect to the dimension of

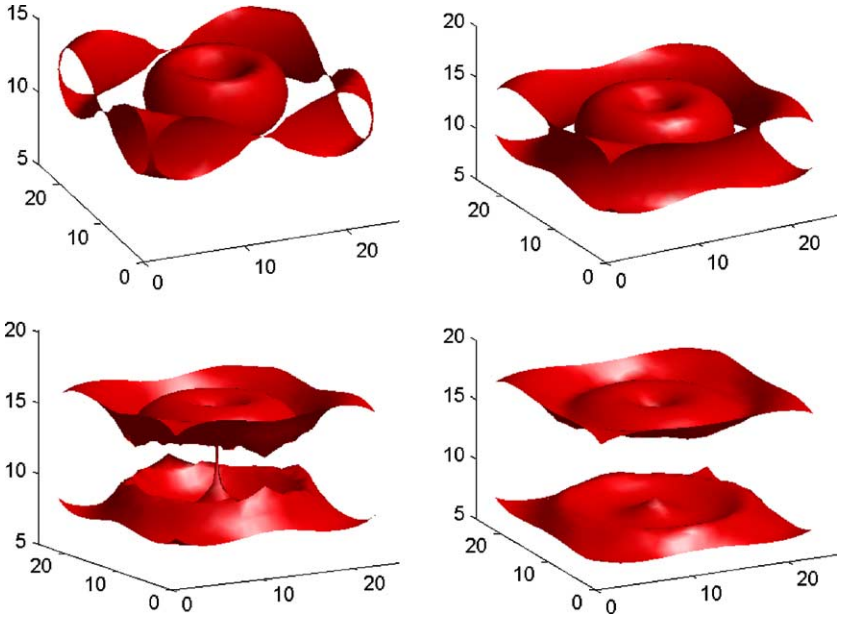

Fig. 9. Level set of fixed dimension 3 slice of a 4D hypertorus evolving under (2) with $H(p) = -|p|$.

the computational domain). The speed would then be dependent on whether the user decided to precalculate the Lagrange coefficients of $\nabla\varphi$ or decided to do this at each timestep. In the first case the speed would be comparable to existing level set methods on uniform grids, with a penalty for data access speed only. In the second case the speed would be penalized by the need to invert the coefficient matrix at each

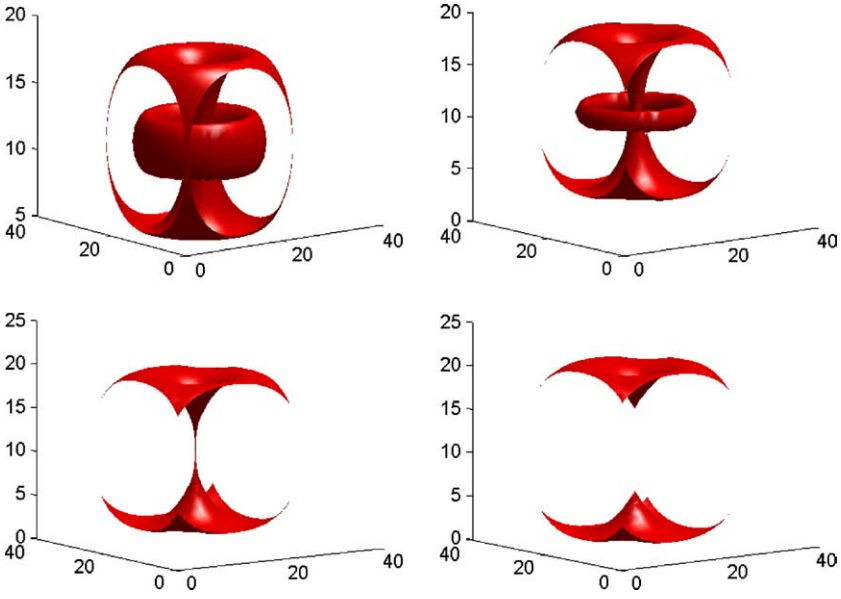

Fig. 10. Level set of fixed dimension 4 slice of a 4D hypertorus evolving under (2) with $H(p) = -|p|$.

data node for each timestep. For small stenciled reconstructions this is not too slow, and is unavoidable if the unstructured mesh is adaptive in time, no matter what reconstruction procedure is used.

10. Conclusion

The numerical solution of Hamilton–Jacobi equations on unstructured grids is becoming increasingly important. As higher dimensional problems are encountered we would like to isolate the important features of the solution and resolve them locally instead of globally. Examples such as local level set computations are already pushing the computational boundaries on coarse grids in 5D [7]. Minimization and control theory problems on irregularly shaped domains also call for scattered meshes to save space.

The methods presented here yield solutions which converge to the vanishing viscosity solution of HJ equations of the form (2). Error estimates for RBF interpolations and estimates on their partial derivatives have been proved, and are an area of current research. Optimal node choice, RBF parameter choice, and RBF basis function form are also areas that are being studied, and demand further theoretical results and a more intuitive description.

It should be noted that the arguments made for monotonicity and convergence of the LF and LLF schemes constructed in Section 4 can be applied to other interpolation schemes for meshless numerical methods such as moving least squares, kernel based approximations, and partition of unity methods by writing these methods in Lagrange form [6]. However, we cannot hope to apply our monotone construction to global interpolation schemes, as they will not satisfy the restrictions on the signs of d_j.

Other questions to be addressed concern the optimal way to handle neighbor access on a meshless data set, and what is the best way to automate the stencil selection process. We have addressed these problems here, but as they often consume the bulk of the computational time we suggest further study.

Acknowledgements

This work is supported by NSF Grant DMS-0312222, ACI 0113439, and ONR MURI Grant N00014-02-1-0720.

References

[1] R. Abgrall, On essentially non-oscillatory schemes on unstructured meshes: analysis and implementation, J. Comput. Phys. 114 (1) (1994) 45–58.
[2] R. Abgrall, Numerical discretization of the first-order Hamilton–Jacobi equation on triangular meshes, Commun. Pure Appl. Math. 49 (12) (1996) 1339–1373.
[3] S. Albert, B. Cockburn, D.A. French, T.E. Peterson, A posteriori error estimates for general numerical methods for Hamilton–Jacobi equations. I. The steady state case, Math. Comput. 71 (237) (2002) 49–76 (electronic).
[4] S. Augoula, R. Abgrall, High order numerical discretization for Hamilton–Jacobi equations on triangular meshes, J. Sci. Comput. 15 (2) (2000) 197–229.
[5] T.J. Barth, J.A. Sethian, Numerical schemes for the Hamilton–Jacobi and level set equations on triangulated domains, J. Comput. Phys. 145 (1) (1998) 1–40.
[6] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, P. Krysl, Meshless methods: an overview and recent developments, Comput. Methods Appl. Mech. Eng. 139 (1996) 3–47.
[7] P. Burchard, Li-Tien Cheng, B. Merriman, S. Osher, Motion of curves in three spatial dimensions using a level set approach, J. Comput. Phys. 170 (2) (2001) 720–741.
[8] R.E. Carlson, T.A. Foley, The parameter R^2 in multiquadric interpolation, Comput. Math. Appl. 21 (9) (1991) 29–42.
[9] B. Cockburn, J. Qian, Continuous dependence results for Hamilton–Jacobi equations, in: D. Estep, S. Tavener (Eds.), Collected Lectures on the Preservation of Stability under Discretization, SIAM, Philadelphia, PA, 2002.

[10] M.G. Crandall, P.-L. Lions, Two approximations of solutions of Hamilton–Jacobi equations, Math. Comput. 43 (167) (1984) 1–19.
[11] M.G. Crandall, A. Majda, Monotone difference approximations for scalar conservation laws, Math. Comput. 34 (149) (1980) 1–21.
[12] O. Friedrich, Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids, J. Comput. Phys. 144 (1) (1998) 194–212.
[13] R.L. Hardy, Multiquadric equations of topography and other irregular surfaces, J. Geophys. Res. 76 (1971) 1905–1915, Radial basis functions and their applications.
[14] A. Harten, S. Chakravarthy, Multi-dimensional ENO schemes for general geometries, ICASE Report 91-76, 1991.
[15] A. Harten, J.M. Hyman, P.D. Lax, On finite-difference approximations and entropy conditions for shocks, Commun. Pure Appl. Math. 29 (3) (1976) 297–322, With an appendix by B. Keyfitz.
[16] A. Harten, S. Osher, B. Engquist, S.R. Chakravarthy, Some results on uniformly high-order accurate essentially nonoscillatory schemes, Appl. Numer. Math. 2 (3–5) (1986) 347–377.
[17] Changqing Hu, Chi-Wang Shu, Weighted essentially non-oscillatory schemes on triangular meshes, J. Comput. Phys. 150 (1) (1999) 97–127.
[18] A. Iske, T. Sonar, On the structure of function spaces in optimal recovery of point functionals for ENO-schemes by radial basis functions, Numer. Math. 74 (2) (1996) 177–201.
[19] G. Kossioris, Ch. Makridakis, P.E. Souganidis, Finite volume schemes for Hamilton–Jacobi equations, Numer. Math. 83 (3) (1999) 427–442.
[20] C.A. Micchelli, Interpolation and scattered data: distance matrices and conditionally positive definite functions, Constr. Approx. 2 (1986) 11–22.
[21] S.J. Osher, J.A. Sethian, Fronts propagating with curvature dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.
[22] S. Osher, Chi-Wang Shu, High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations, SIAM J. Numer. Anal. 28 (4) (1991) 907–922.
[23] Shmuel Rippa, An algorithm for selecting a good value for the parameter c in radial basis function interpolation, Adv. Comput. Math. 11 (2–3) (1999) 193–210, Radial basis functions and their applications.
[24] R. Schaback, Error estimates and condition numbers for radial basis function interpolation, Adv. Comput. Math. 3 (3) (1995) 251–264.
[25] R. Schaback, Reconstruction of Multivariate Functions from Scattered Data. Electronic publication, available from: <www.num.math.uni-goettingen.de/schaback>.
[26] Chi-Wang Shu, S. Osher, Efficient implementation of essentially nonoscillatory shock-capturing schemes. II, J. Comput. Phys. 83 (1) (1989) 32–78.
[27] T. Sonar, Optimal recovery using thin plate splines in finite volume methods for the numerical solution of hyperbolic conservation laws, IMA J. Numer. Anal. 16 (4) (1996) 549–581.
[28] T. Sonar, On families of pointwise optimal finite volume ENO approximations, SIAM J. Numer. Anal. 35 (6) (1998) 2350–2369.
[29] J. Strain, Tree methods for moving interfaces, J. Comput. Phys. 151 (2) (1999) 616–648.
[30] Z.-M. Wu, Multivariate compactly supported positive definite radial basis functions, Adv. Comput. Math. 4 (1995) 389–396.
[31] Yong-Tao Zhang, Chi-Wang Shu, High-order WENO schemes for Hamilton–Jacobi equations on triangular meshes, SIAM J. Sci. Comput. 24 (3) (2002) 1005–1030 (electronic).