MATHEMATICS OF COMPUTATION
Volume 76, Number 258, April 2007, Pages 647–668
S 0025-5718(06)01898-9
Article electronically published on October 30, 2006

BACK AND FORTH ERROR COMPENSATION AND CORRECTION METHODS FOR SEMI-LAGRANGIAN SCHEMES WITH APPLICATION TO LEVEL SET INTERFACE COMPUTATIONS

TODD F. DUPONT AND YINGJIE LIU

Abstract. Semi-Lagrangian schemes have been explored by several authors recently for transport problems, in particular for moving interfaces using the level set method. We incorporate the backward error compensation method developed in our paper from 2003 into semi-Lagrangian schemes with almost the same simplicity and three times the complexity of a first order semi-Lagrangian scheme but with improved order of accuracy. Stability and accuracy results are proved for a constant coefficient linear hyperbolic equation. We apply this technique to the level set method for interface computation.

1. Introduction

Semi-Lagrangian schemes, e.g., the Courant-Isaacson-Rees (CIR) scheme [2], have no CFL restriction for the time step size, and therefore local space refinement becomes more convenient. Recently several researchers have used and studied semi-Lagrangian schemes for transport equations, in particular for computing level sets (Osher and Sethian [19]) describing interface movement. Strain [25, 26, 27] has developed several fast semi-Lagrangian schemes for evolving level sets which incorporate techniques including essentially non-oscillatory (ENO [10, 23, 24]) spatial interpolation, predictor-corrector temporal approximation, velocity smoothing, and quad-tree meshes. Enright *et al.* [5] apply the CIR scheme to the hybrid particle level set method [4] to simplify the method with almost no loss of resolution.

For a hyperbolic equation $u_t + \mathbf{v} \cdot u_x = 0$, the CIR scheme computes the numerical solution defined on a mesh $\{x_i\}$ as $U(x_i, t_{n+1}) = U(\hat{x_i}, t_n)$, where $\hat{x_i} = \Gamma_i(t_n)$ and $\Gamma_i(t)$ is the approximate characteristic curve passing (x_i, t_{n+1}). Various approximations of $\hat{x_i}$ and $U(\hat{x_i}, t_n)$ (since U is only defined at grid points (x_i, t_n)) can be used. For example, one may choose $\hat{x_i} = x_i - \mathbf{v}(x_i, t_n)(t_{n+1} - t_n)$ and compute $U(\hat{x_i}, t_n)$ by linearly interpolating the U values at two nearest grid points x_j and x_{j+1} such that $\hat{x_i} \in [x_j, x_{j+1}]$, and obtain the first order CIR scheme which does not increase the L_∞ norm of the numerical solution with increasing time. If

Received by the editor February 1, 2005 and, in revised form, December 19, 2005.

2000 *Mathematics Subject Classification.* Primary 65M06, 65M12.

Key words and phrases. CIR scheme, front tracking, level set method, MacCormack scheme, semi-Lagrangian scheme.

The work of the first author was supported by the ASC Flash Center at the University of Chicago under DOE contract B532820, and by the MRSEC Program of the National Science Foundation under award DMR-0213745.

The work of the second author was supported by NSF grant DMS-0511815.

$t_{n+1} - t_n$ is small enough so that $\hat{x_i} \in [x_{i-1}, x_{i+1}]$ (the CFL condition), then the CIR scheme is actually the first order upwind scheme. Furthermore, if the linear interpolation of $U(\hat{x_i}, t_n)$ is from the U values at grid nodes x_{i-1} and x_{i+1}, then it becomes the Lax-Friedrich scheme. Therefore the CIR scheme removes the CFL condition by interpolating $U(\hat{x_i}, t_n)$ from the U values near the root of the characteristics $\hat{x_i}$, instead of from the U values near x_i. In order to achieve a higher order of accuracy, a higher degree (2nd degree or higher) polynomial interpolation can be applied and a corresponding order of temporal numerical integration is also necessary for computing the characteristics. Falcone and Ferretti [6] have analyzed the stability and convergence of a general class of semi-Lagrangian schemes.

In higher space dimensions, the first order CIR scheme is quite simple since it only uses a local linear interpolation. Is there a convenient way to manipulate the first order CIR scheme to achieve a higher order of accuracy simultaneously in both space and time without using a higher order polynomial interpolation? The MacCormack scheme [17] uses an upwind scheme followed by a downwind scheme to obtain an improved order of accuracy in both space and time for hyperbolic equations. For semi-Lagrangian schemes, the integration is along the approximate characteristics and the upwind discretization is not clearly defined at the root of the characteristics. We are interested in whether the backward error compensation algorithm introduced in [3] can be successfully applied to the CIR scheme. This algorithm is based on a simple observation that if one solves a hyperbolic system forward in time for one time step with a scheme (e.g., a first order scheme) and then backward in time for one time step with the same scheme, one obtains another copy of the solution at the initial time. The two copies of the solution should have been equal if there were no numerical errors (at least, away from singularities). Therefore comparing the two copies of the solution gives us information about the errors which we can use to improve the accuracy. In Shu and Osher [23], some TVD Runge-Kutta methods also incorporate downwind spatial discretizations in order to achieve the TVD property, which are implemented by discretizing the time reversed hyperbolic equation in certain middle time steps.

Two difficulties involved in the numerical computation of the level set method are (1) how to reduce diffusion; and (2) how to minimize artifacts near the singular points of the interface. Typically high order ENO or WENO ([16, 11]) schemes are used for solving the level set equation and redistancing. Sussman and Puckett [28] have combined the level set and volume-of-fluid method so that one has the interface represented by a smooth level set function for extracting information such as mean curvature, *etc.*, and also has the local volume conservation from the volume-of-fluid method. Enright *et al.* [4] have developed the hybrid particle level set method which takes advantage of the high resolution of Lagrangian schemes near interface singularities, and also has the convenience of the level set method which automatically resolves topological changes of the interface. Strain [25, 26, 27] addresses these difficulties by using semi-Lagrangian schemes to compute the level set equation so that local space refinement can be done near singular points of the interface without locally reducing the time step size. Here we incorporate the backward error compensation algorithm [3] with the CIR scheme and obtain an efficient and simple algorithm for the level set method. We will introduce this algorithm in Section 2, and discuss its stability and accuracy in Sections 3 and 4. In Section 5, we discuss its application to the level set method. We would like to refer to Kim *et al.* [12] for fluid simulations incorporating the backward error compensation algorithm with other methods.

2. Backward error compensation for semi-Lagrangian schemes

The level set method proposed by Osher and Sethian [19] uses a continuous function $\phi(\mathbf{x},t) \in R$ to represent evolving interfaces as the zero level set $\{(\mathbf{x},t) : \phi(\mathbf{x},t) = 0\}$, where $\mathbf{x} \in R^d$ is the spatial variable and $t \in R$ represents the time. For a given velocity field $\mathbf{v}(\mathbf{x},t) \in R^d$, the level set function ϕ satisfies

$$\frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \bigtriangledown \phi = 0. \tag{2.1}$$

We define a straightforward scheme based on the first-order CIR scheme with backward error compensation. For simplicity we use a uniform mesh and describe the scheme in all of R^d. We assume a uniform rectangular grid in R^d with the spatial mesh size

$$\Delta \mathbf{x} = (\Delta x_1, \Delta x_2, \ldots, \Delta x_d)$$

and let the time step size be $\Delta t_n = t_{n+1} - t_n$. Given the approximate level set function $\Phi(\cdot, t_n)$ at grid points

$$\{\mathbf{x}_\mathbf{i} = (i_1 \Delta x_1, i_2 \Delta x_2, \ldots, i_d \Delta x_d) : \mathbf{i} = (i_1, i_2, \ldots, i_d) \in Z^d\},$$

the first order CIR scheme can be formulated as

$$\Phi(\mathbf{x}_\mathbf{i}, t_{n+1}) = \Phi(\hat{\mathbf{x}}_\mathbf{i}, t_n), \tag{2.2}$$

where $\hat{\mathbf{x}}_\mathbf{i} = \mathbf{x}_\mathbf{i} - \mathbf{v}(\mathbf{x}_\mathbf{i}, t_n)\Delta t_n$. In one space dimension ($d = 1$), $\Phi(\hat{x_i}, t_n)$ is computed from the linear interpolation of $\Phi(x_j, t_n)$ and $\Phi(x_{j+1}, t_n)$ where $\hat{x_i} \in [x_j, x_{j+1}]$. In two space dimensions $\Phi(\hat{\mathbf{x}}_\mathbf{i}, t_n)$ can be approximated by the bilinear interpolation of the $\Phi(\cdot, t_n)$ values at the vertices (grid points) of a grid cell containing $\hat{\mathbf{x}}_\mathbf{i}$. For general space dimensions one can use the tensor product of one-dimensional linear polynomials to interpolate. Denote $\Phi^n_\mathbf{i} = \Phi(\mathbf{x}_\mathbf{i}, t_n)$.

The backward error compensation algorithm [3] can be applied to the CIR scheme as follows (see Figure 2.1):

Step 1. Solve equation (2.1) forward in time to obtain $\tilde{\Phi}^{n+1}$ by the CIR scheme (2.2), with Φ^n being the initial value at the time t_n.

Step 2. Solve equation (2.1) backward in time to obtain $\breve{\Phi}^n$ by the same method. This is equivalent to solving the time reversed equation $\frac{\partial \phi}{\partial t} - \mathbf{v} \cdot \bigtriangledown \phi = 0$ forward in time by (2.2), with $\tilde{\Phi}^{n+1}$ being the initial value.

Step 3. Let $\underline{\Phi}^n_\mathbf{i} = \Phi^n_\mathbf{i} + \frac{1}{2}(\Phi^n_\mathbf{i} - \breve{\Phi}^n_\mathbf{i})$ for all $\mathbf{i}$.

Step 4. Solve equation (2.1) forward in time to obtain Φ^{n+1} by (2.2), with $\underline{\Phi}^n$ being the initial value at the time t_n.

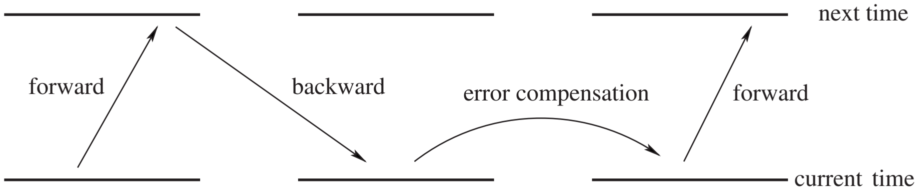

Figure 2.1. Backward error compensation algorithm.

The term $\frac{1}{2}(\Phi_{\mathbf{i}}^n - \breve{\Phi}_{\mathbf{i}}^n)$ is called the backward compensation term. It should be noted that the velocity field $\mathbf{v}$ is only evaluated at grid points at times t_n and t_{n+1} in the above algorithm and the that CIR scheme (2.2) involves only local linear interpolation of $\Phi(\hat{\mathbf{x}}_{\mathbf{i}}, \cdot)$. Therefore the implementation of the above algorithm is quite simple even for three space dimensions.

The dual of the above algorithm, called the forward error correction algorithm, can be applied to the CIR scheme as follows:

Step 1. Solve equation (2.1) forward in time to obtain $\tilde{\Phi}^{n+1}$ by the CIR scheme (2.2), with Φ^n being the initial value at the time t_n.

Step 2. Solve equation (2.1) backward in time to obtain $\breve{\Phi}^n$ by the same method. This is equivalent to solving the time reversed equation $\frac{\partial \phi}{\partial t} - \mathbf{v} \cdot \bigtriangledown \phi = 0$ forward in time by (2.2), with $\tilde{\Phi}^{n+1}$ being the initial value.

Step 3. Solve equation (2.1) forward in time to obtain $\underline{\Phi}^{n+1}$ by (2.2), with $\breve{\Phi}^n$ being the initial value at the time t_n.

Step 4. Let $\Phi_{\mathbf{i}}^{n+1} = \tilde{\Phi}_{\mathbf{i}}^{n+1} + \frac{1}{2}(\tilde{\Phi}_{\mathbf{i}}^{n+1} - \underline{\Phi}_{\mathbf{i}}^{n+1})$ for all $\mathbf{i}$.

If the velocity field $\mathbf{v}$ depends only on $\mathbf{x}$ and t, the above two algorithms are equivalent in the sense that they will result in the same Φ^{n+1}. If the velocity field $\mathbf{v}$ depends on ϕ, i.e., $\mathbf{v} = \mathbf{v}(\phi(\mathbf{x}, t), \mathbf{x}, t)$, in the backward error compensation algorithm we may use the same velocity field in Step 4 as in Step 1. Therefore the velocity field only needs to be computed twice at Steps 1 and 2. In the forward error correction algorithm, we may use the same velocity field in Step 3 as in Step 1 so that the velocity field only needs to be computed twice. When using this velocity approximation, we can easily see that the above two algorithms applied to the CIR scheme are equivalent.

3. Stability

In [3], we have proved the l_2 stability of the backward error compensation algorithm applied to the first order upwind scheme for the 1D equation $u_t + u_x = 0$. Here we prove some more general results in higher space dimensions. Throughout this and the next section, we consider equation (2.1) in the domain $[0,1]^d$ with periodic boundary conditions, and assume $\mathbf{v}$ is a constant vector in equation (2.1) unless specified otherwise. We use $i, j, k, l, s \in Z$ for indices and $\mathbf{i}, \mathbf{j}, \mathbf{k}, \mathbf{l}, \mathbf{s} \in Z^d$ for multi-indices. In particular, we use $\mathbf{k}$ to represent the dual index of the Fourier series. The symbol i will also represent $\sqrt{-1}$ when the meaning is clear from context. Let $\Delta x_j = 1/N_j$, $j = 1, 2, \ldots, d$, for some positive integers N_j. With $\mathbf{N} = (N_1, N_2, \ldots, N_d)$ set $\mathcal{D}_{\mathbf{N}} = Z^d \cap \Pi_j[0, N_j - 1]$. The $U_{\mathbf{i}}^n$ is defined outside $\mathcal{D}_{\mathbf{N}}$ by periodic extension. Similarly, take $\mathcal{F}_{\mathbf{N}} = Z^d \cap \Pi_j[1 - N_j, N_j - 1]$. Let $L:\ \ U^{n+1} = L(U^n)$ be a linear scheme for equation (2.1) such that $U_{\mathbf{i}}^{n+1} = \sum_{\mathbf{j} \in \mathcal{D}_{\mathbf{N}}} \alpha_{\mathbf{j}} U_{\mathbf{i}+\mathbf{j}}^n$, where the $\alpha_{\mathbf{j}}$'s depend on $\Delta t_n / \Delta x_l$, $l = 1, \ldots, d$. The $U_{\mathbf{j}}^n$ can be expressed uniquely as the finite Fourier series

$$U_{\mathbf{j}}^n = \sum_{\mathbf{k} \in \mathcal{F}_{\mathbf{N}}} C_{\mathbf{k}}^n e^{2\pi i \mathbf{k} \cdot \mathbf{x}_{\mathbf{j}}},$$

where $\mathbf{k} = (k_1, k_2, \ldots, k_d)$. Substituting this finite Fourier series into scheme L we obtain $C_{\mathbf{k}}^{n+1} = \rho_L C_{\mathbf{k}}^n$, where $\rho_L(\mathbf{k}) = \sum_{\mathbf{j} \in \mathcal{D}_{\mathbf{N}}} \alpha_{\mathbf{j}} e^{2\pi i \mathbf{k} \cdot \mathbf{x}_{\mathbf{j}}}$ is the Fourier symbol of L, and $\max\{|\rho_L(\mathbf{k})| : \mathbf{k} \in \mathcal{F}_{\mathbf{N}}\}$ is called the amplification factor of scheme L. Let $L^*:\ \ W^n = L^*(W^{n+1})$ be the corresponding linear scheme that solves equation (2.1) backward in time using scheme L. (Note that L^* is not defined to be the adjoint of L with respect to any inner product, although it may be in some cases.) Applying the backward error compensation algorithm to scheme L we obtain a linear scheme for equation (2.1),

$$F: \quad V^{n+1} = F(V^n) = L(I + \frac{1}{2}(I - L^*L))(V^n), \tag{3.1}$$

where I is the identity operator. Let ρ_{L^*} and ρ_F be the Fourier symbols of schemes L^* and F respectively; we then have the following theorem.

Theorem 1. *Suppose* $\rho_{L^*}(\mathbf{k}) = \overline{\rho_L}(\mathbf{k})$ *for all* $\mathbf{k} \in \mathcal{F}_{\mathbf{N}}$. *Then* $|\rho_F(\mathbf{k})| \le 1$ *for all* $\mathbf{k} \in \mathcal{F}_{\mathbf{N}}$ *if and only if* $|\rho_L(\mathbf{k})| \le 2$ *for all* $\mathbf{k} \in \mathcal{F}_{\mathbf{N}}$.

Proof. The Fourier symbol of (3.1) can be obtained as follows:

$$\rho_F = \rho_L(1 + \frac{1}{2}(1 - \overline{\rho_L}\rho_L)).$$

Let $\eta = |\rho_L|$, $G(\eta) = |\rho_F|$, then the theorem is proved by inspecting the function $G(\eta) = \eta|\frac{3}{2} - \frac{1}{2}\eta^2|$ for $\eta \in [0, \infty)$. □

Theorem 1 not only insures that the backward error compensation algorithm applied to a stable (in l^2) scheme is stable, but also implies that some unstable schemes can be turned into stable ones. Throughout this paper, we say a scheme is stable if it is stable in the l^2 sense, unless specified otherwise. It is easy to verify that the condition in Theorem 1 is satisfied when applying the backward error compensation algorithm to the following classical schemes.

Example 1. In one space dimension ($d = 1$), the first order upwind scheme for equation (2.1) has an amplification factor $|\rho| \le 2$ if the CFL factor $|\mathbf{v}|\Delta t/\Delta x$ is no more than 1.5 (the scheme is unstable for the CFL factor greater than 1). Therefore, applying the backward error compensation algorithm to it creates a scheme stable with CFL factor less than or equal to 1.5, and the new scheme is second order accurate [3].

Example 2. Using center spatial difference and forward Euler time difference for equation (2.1) will create an unstable scheme. When $d = 1$, the scheme has an amplification factor $|\rho| \le 2$ if the CFL factor is no more than $\sqrt{3}$. Therefore applying the backward error compensation algorithm to it creates a second order scheme (see Section 4) stable with the CFL factor less than or equal to $\sqrt{3}$.

Example 3. In one space dimension, the Lax–Friedrich scheme has an amplification factor no more than 2 if the CFL factor is less than or equal to 2 (it is stable only if the CFL factor is less than or equal to 1). Therefore applying the backward error compensation algorithm to it creates a second order scheme (see Section 4) stable with the CFL factor less than or equal to 2.

Remark. It is well known that for the linear Schrodinger equation

$$i\psi_t = -a \triangle \psi,$$

the simplest explicit scheme with forward Euler time discretization and center spatial discretization is unstable. In 1D, the amplification factor for such a scheme is

$$|\rho(k)|^2 = 1 + (4\lambda)^2 \sin^4(\pi k \Delta x), \quad \text{where} \quad \lambda = |a|\frac{\Delta t}{\Delta x^2}.$$

Therefore $|\rho| \leq 2$ if $\lambda \leq \frac{1}{4}$. By applying equivalent results of Theorem 1 and Theorem 4 (in the next section), we could also show that applying the backward error compensation algorithm to it creates a new second order scheme stable for $\lambda \leq \frac{1}{4}$.

Next we verify that for constant coefficients with periodic boundary conditions, the CIR scheme for equation (2.1) satisfies the condition of Theorem 1. Given Φ^n, the Φ^{n+1} computed by the CIR scheme can be written as

$$\Phi^{n+1}_{\mathbf{j}} = (L\Phi^n)_{\mathbf{j}} = \sum_{\mathbf{l}} \Phi^n_{\mathbf{l}} \Psi_{\mathbf{l}}(\mathbf{x_j} - \mathbf{v}\Delta t_n), \tag{3.2}$$

where $\Psi_{\mathbf{l}}$ is the Lagrange basis function which in each cell is a linear ($d = 1$) polynomial or a bilinear ($d = 2$) polynomial, *etc.*, satisfying $\Psi_{\mathbf{l}} \geq 0$ and $\Psi_{\mathbf{l}}(x_{\mathbf{j}}) = \delta_{\mathbf{lj}}$ ($= 1$ if $\mathbf{l} = \mathbf{j}$; 0 otherwise). The function $\Psi_{\mathbf{l}}$ has the property that

$$\Psi_{\mathbf{l}}(\mathbf{x}) = \Pi^d_{j=1} \psi_{\Delta x_j, l_j}(x_j),$$

where $\psi_{h,j}$ is the basis function associated with the mesh point jh in one-dimensional piecewise linear interpolation using the mesh of integer multiples of h.

It then follows that

$$\Phi^{n+1}_{\mathbf{j}} = (L_1 \otimes \cdots \otimes L_d \Phi^n)_{\mathbf{j}}, \tag{3.3}$$

where the operators L_j are the one-dimensional versions of L using v_j and Δx_j. From this it follows that

$$\rho_{CIR}(\mathbf{k}) = \Pi^d_{j=1} \rho_{L_j}(k_j). \tag{3.4}$$

Hence,

$$\rho_{CIR} = \overline{\rho_{CIR^*}}, \tag{3.5}$$

if the result holds in one space dimension. The one-dimensional result is easily checked. We are now in a position to obtain

Corollary 2. *The CIR scheme with backward error compensation algorithm for equation* (2.1) *with constant coefficients has an amplification factor less than or equal to* 1 *for any mesh size* $\Delta \mathbf{x}$ *and time step size* Δt_n.

Proof. It is well known (see, e.g., [6]) that the CIR scheme has an amplification factor less than or equal to 1. It then follows that it has an amplification factor less than or equal to one in d-space by (3.4). From (3.5) and Theorem 1 the proof is complete. □

4. Accuracy

We study the accuracy improvement of the backward error compensation algorithm for a general linear scheme for equation (2.1) with constant coefficients and periodic initial data (see the previous section). The result generalizes the accuracy improvement theorem in [3] for a linear ordinary differential equation and is based on the comparison of the Fourier symbols of the differential equation (2.1) and its corresponding numerical scheme; see Lax [14]. Let L, L^*, F be linear schemes defined as in Section 3, and let $\rho_L, \rho_{L^*}, \rho_F$ be their corresponding Fourier symbols, respectively. Expanding ϕ into Fourier series

$$\phi(\mathbf{x}, t) = \sum_{\mathbf{k} \in Z^d} c_{\mathbf{k}}(t) e^{2\pi i \mathbf{k} \cdot \mathbf{x}},$$

and plugging in equation (2.1), we obtain

$$\frac{dc_{\mathbf{k}}}{dt} = P(i\mathbf{k})c_{\mathbf{k}},$$

where P is a linear homogeneous polynomial with real coefficients. Therefore we can write

$$c_{\mathbf{k}}(t_n + \Delta t) = e^{\Delta t P(i\mathbf{k})} c_{\mathbf{k}}(t_n).$$

Assume $\Delta x_1 = \cdots = \Delta x_d = h$ and $\Delta t/h$ is fixed during the mesh refinement. A scheme $L_1 : \ \Phi^{n+1} = L_1(\Phi^n)$ is said to be accurate of order r if for any solution ϕ of equation (2.1) having continuous derivatives up to order $r+1$,

$$\phi(\mathbf{x_j}, t_{n+1}) - L_1(\phi(\cdot, t_n))|_{\mathbf{x_j}} = O(h^{r+1}).$$

We first state the theorem of Lax [14].

Theorem 3. *Scheme L is accurate of order r if and only if*

$$\rho_L(\mathbf{k}) = e^{\Delta t P(i\mathbf{k})} + O(|\mathbf{k}h|^{r+1}), \text{ as } h \to 0.$$

The "only if" part of the theorem is proved by Lax [14] for more general linear hyperbolic equations with variable coefficients. With constant coefficients, Lax's proof also implies the "if" part of this theorem. Using the Lax's Theorem 3, we can prove the following theorem.

Theorem 4. *Suppose $\rho_{L^*}(\mathbf{k}) = \overline{\rho_L(\mathbf{k})}$ for any $\mathbf{k} \in Z^d$ and that scheme L is accurate of order r for equation* (2.1) *with constant coefficients, where r is an odd positive integer. Then scheme F is accurate of order $r+1$.*

Proof. The accuracy of scheme L implies that

$$\rho_L(\mathbf{k}) = e^{\Delta t P(i\mathbf{k})} + Q_{r+1}(i\mathbf{k}h) + O(|\mathbf{k}h|^{r+2}),$$

where Q_{r+1} is a homogeneous polynomial of order $r+1$ with real coefficients (recalling that we assume the scheme coefficient $\alpha_{\mathbf{j}}$ depends on $\Delta t/h$ which is fixed). Since $r+1$ is even, we have

$$\rho_{L^*}(\mathbf{k}) = \overline{\rho_L(\mathbf{k})} = e^{-\Delta t P(i\mathbf{k})} + Q_{r+1}(i\mathbf{k}h) + O(|\mathbf{k}h|^{r+2}).$$

Therefore

(4.1)
$$\begin{aligned} \rho_F(\mathbf{k}) &= \rho_L(\mathbf{k})\{1 + \tfrac{1}{2}[1 - \rho_{L^*}(\mathbf{k})\rho_L(\mathbf{k})]\} \\ &= \rho_L(\mathbf{k})\{1 - \tfrac{1}{2}[e^{-\Delta t P(i\mathbf{k})} + e^{\Delta t P(i\mathbf{k})}]Q_{r+1}(i\mathbf{k}h) + O(|\mathbf{k}h|^{r+2})\} \\ &= [e^{\Delta t P(i\mathbf{k})} + Q_{r+1}(i\mathbf{k}h) + O(|\mathbf{k}h|^{r+2})][1 - Q_{r+1}(i\mathbf{k}h) + O(|\mathbf{k}h|^{r+2})] \\ &= e^{\Delta t P(i\mathbf{k})} + O(|\mathbf{k}h|^{r+2}). \end{aligned}$$

The proof is complete. □

Remark. When equation (2.1) has variable coefficients, the Fourier-Stieltjes transform (see, e.g., [14]) is used to replace the Fourier symbols, and a formula similar to (4.1) can be derived for the Fourier–Stieltjes transform of scheme F.

An interesting phenomenon is that the backward error compensation algorithm seems to improve the numerical result, even for a very irregular mesh. In the following example we use a first order upwind scheme with and without backward error compensation to compute the linear advection of a pyramid function: $u_t + u_x = 0$, $x \in [0,1]$ with periodic boundary condition. The grid points are distributed as

$$x_i = i * 0.01 + 0.003 * \sin[(i - 0.2) * (i + 6.1789) * i], \quad i = 0, 1, \ldots, 99.$$

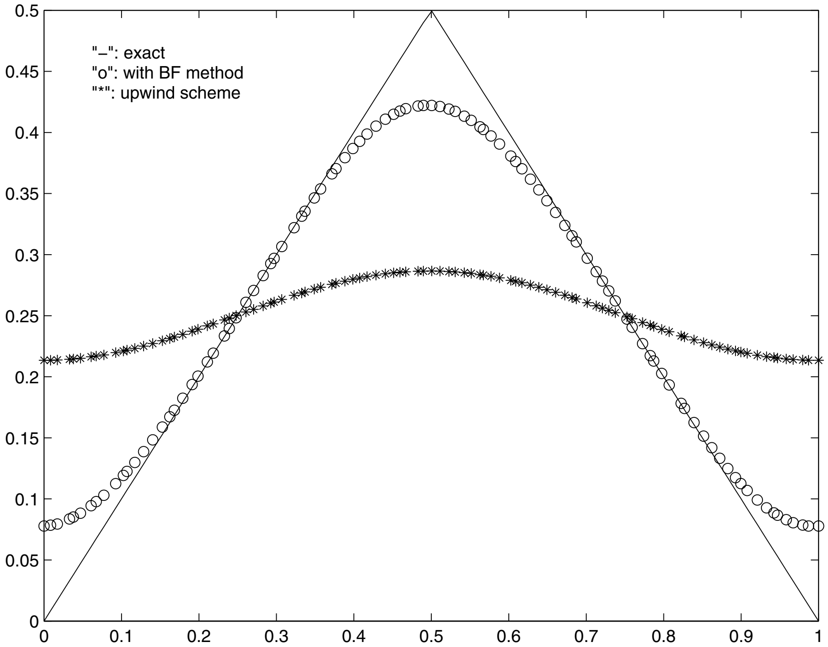

Figure 4.1. Linear advection of a pyramid function over 100 irregular cells on domain $[0, 1]$ with the size of the largest cell 4 times that of the smallest cell. $CFL = 0.5$, $T = 10$. Computed by the first order upwind scheme with and without backward error compensation (BF).

The solutions at the final time $T = 10$ are shown in Figure 4.1. The result for triangular mesh is shown in Section 5.

5. Application to the level set method

Besides equation (2.1), since the velocity field could create a large gradient in Φ, there is usually an auxiliary equation to solve until reaching the steady state at each time step [30],

$$\frac{\partial \Phi}{\partial \tau} + \mathrm{sign}(\Phi)(|\bigtriangledown \Phi| - 1) = 0. \tag{5.1}$$

This procedure is supposed to transform the Φ into a signed distance function without changing its zero level set. This step also helps clean the error pollution coming from the "skeleton", *i.e.*, the nonsmooth area of the level set function. As in [30], equation (5.1) can be written as

$$\tilde{\Phi}_\tau + W \cdot \bigtriangledown \tilde{\Phi} = S(\tilde{\Phi}^0), \tag{5.2}$$

where $W = S(\tilde{\Phi}^0) \bigtriangledown \tilde{\Phi}/| \bigtriangledown \tilde{\Phi}|$ and $S(\tilde{\Phi}^0)$ is the sign function of $\tilde{\Phi}^0$, $S(\tilde{\Phi}^0) = 1$, if $\tilde{\Phi}^0 > 0$; $S(\tilde{\Phi}^0) = -1$ if $\tilde{\Phi}^0 < 0$. $\tilde{\Phi}^0$ is the initial value for (5.2) and is the current level set function obtained by solving equation (2.1). We only discuss the cases in 2D, and the indices $\mathbf{i}$ or $(i, j) \in Z^2$ will be used throughout the rest of the paper.

5.1. **New modifications to the redistancing procedure.** We first compute equation (2.1) using the CIR scheme with backward error compensation to obtain the approximate level set function Φ^n at the time t_n. Then let $\tilde{\Phi}^0 = \Phi^n$ and solve equation (5.2) for a few time steps (e.g., m_1 steps). Then replace Φ^n by $\tilde{\Phi}^{m_1}$ and finish the redistancing procedure at this time.

We use a slightly modified center difference to approximate W. For example, $\frac{\partial\tilde{\Phi}}{\partial x}(\mathbf{x}_{i,j})$ is approximated by $(\tilde{\Phi}_{i+1,j}-\tilde{\Phi}_{i-1,j})/(2\Delta x)$ if $\tilde{\Phi}_{i+1,j}-\tilde{\Phi}_{i,j}$ and $\tilde{\Phi}_{i,j}-\tilde{\Phi}_{i-1,j}$ are of the same sign; by $\text{maxmod}\{(\tilde{\Phi}_{i+1,j}-\tilde{\Phi}_{i,j})/\Delta x, (\tilde{\Phi}_{i,j}-\tilde{\Phi}_{i-1,j})/\Delta x\}$ otherwise, where

$$\text{maxmod}\{a,b\} = \begin{cases} a, & \text{if } |a| > |b|, \\ b, & \text{otherwise}, \end{cases}$$

and similarly for the approximation of $\frac{\partial\tilde{\Phi}}{\partial y}(\mathbf{x}_{i,j})$. This modification gives a more accurate normal direction of the interface in the unresolved region of the interface, e.g., near the place where the interfaces are about to have topological changes.

At each time of solving equation (5.2), given $\tilde{\Phi}^m$ at the time τ_m, we compute equation (5.2) only at places, say $\mathbf{x}_{i,j}$, to obtain $\tilde{\Phi}^{m+1}_{i,j}$ where either

(A) the absolute value of the difference between $\tilde{\Phi}^m_{i,j}$ and one of its neighbors is greater than their distance Δx, i.e., at least one of the four statements is true,

$$|\tilde{\Phi}^m_{i,j} - \tilde{\Phi}^m_{i\pm1,j}| > 1.1\Delta x, \quad |\tilde{\Phi}^m_{i,j} - \tilde{\Phi}^m_{i,j\pm1}| > 1.1\Delta x; \quad \text{or}$$

(B) $\tilde{\Phi}^m_{i,j}$ is of the same sign with $\tilde{\Phi}^m_{k,l}$ for all integers k,l such that $|k-i| \le 1$ and $|l-j| \le 1$.

For other grid nodes, say $\mathbf{x}_{p,q}$, simply let $\tilde{\Phi}^{m+1}_{p,q} = \tilde{\Phi}^m_{p,q}$. This allows us to use a simple low cost first order upwind scheme to discretize equation (5.2) without generating large diffusion or distortion, yet keeps an upper bound for the norm of the gradient of $\tilde{\Phi}$ at the equilibrium state.

Remarks. 1. Note that for an exact signed distance function ϕ, the absolute value of the difference between two ϕ values at any two points is no more than the distance between the two points. Condition (A) detects the violation of this property and corrects it, which ensures (at equilibrium state) an upper bound for the Euclidean norm of the discrete gradient of the redistanced level set function (1.1 in 1D, $1.1\sqrt{2}$ in 2D if the discrete gradient is approximated by one-sided or center difference). For some problems, e.g., an expanding bubble with initial radius about Δx, the level set function Φ^n could become flatter and flatter near the interface without redistancing. Condition (B) addresses the problem.

This procedure is an improvement of the procedure proposed in [3], where condition (A) is replaced by (A1) $|\tilde{\Phi}^m_{\mathbf{i}}| > \Delta x$. The problem with (A1) is that when two interfaces are about to merge, the narrow region between them will not be redistanced, and the unbalanced redistancing on both sides of an interface could create an artificial movement of the interface and delay the merging process. Condition (A) has solved this problem as shown in the following numerical examples. In actual implementation, we use the condition ((A2)$|\tilde{\Phi}^m_{\mathbf{i}}| > 1.1\Delta x$ or (A) or (B)) to determine if the redistancing is necessary at a grid point or not in order to reduce the computational cost of the "if" statement, and the computational results are essentially the same as using the condition ((A) or (B)).

2. Russo and Smereka [22] seem to be the first to realize that not changing the values of the level set function at grid nodes adjacent to the interface produces

Table 5.1. Rotation of a circle: the maximum error between the computed and exact level set functions at grid nodes near the interface, computed by the CIR scheme with backward error compensation, $CFL = 3$.

Δx	error without redistancing	order	error with redistancing	order
2	0.623	-	0.454	-
1	0.110	2.50	0.154	1.56
0.5	0.0262	2.07	0.0536	1.52
0.25	0.00638	2.04	0.0208	1.37

good results in redistancing. In [22], they propose that the upwind discretization of equation (5.2) should not go across the interface. So the value of the level set function at a grid node adjacent to the interface is instead recomputed by its value divided by the norm of the approximated gradient of the level set function at the grid node. In one remark of [22], the approximated gradient is chosen such that the values of the level set function at the grid nodes adjacent to the interface are unchanged during the redistancing.

We first conduct a convergence test with and without the redistancing. We compute the rotation of a circle around the point $(50, 50)$ for one revolution in the domain $[0, 100] \times [0, 100]$. The circle is initially centered at $(50, 75)$ with radius 15. The velocity field is given as $(u, v) = (\frac{\pi}{314}(50 - y), \frac{\pi}{314}(x - 50))$. Every point of this circle is supposed to move along the local velocity field. One revolution occurs at the time $T = 628$. The initial level set function Φ is set to be a signed distance function which is negative inside the circle and positive outside. The maximum error between the computed and exact level set functions at grid nodes near the interface is shown in Table 5.1. Clearly we have the second order convergence for the CIR scheme with backward error compensation without redistancing, and the simple redistancing procedure causes the order of convergence to lie between 1 and 2. However, for interfaces containing singular points, the redistancing improves the resolution (with only a fraction of the cost for computing equation (2.1)) as shown in the following examples.

In the next example we replace the circle with a cutout circle. It is the so-called Zalesak's Problem [33], which is one of the difficult test problems for the level set method or volume of fluid method, because of their Eulerian representation of the interface. (Lagrangian-type methods, e.g., [8, 20, 7, 32] *etc.*, could perform better for this problem.) Initially the cutout circle is centered at $(50, 75)$ with radius 15. The slot being cut out has width 5 and length 25. The challenge for computation is that this disk has corner points, curves, straight lines, and a very narrow slot (when the mesh size is 1 or 0.5, the slot width is 5 or 10 times the mesh cell size, respectively). In the first test we compute this problem with $N = 100$ $(\Delta x = 1)$ and CFL factor 3. Equation (2.1) is computed by the CIR scheme with backward error compensation and redistancing. In all the following test examples, equation (5.2) is computed for only two time steps with the CFL factor 0.25 after solving equation (2.1) for each time step. In Figure 5.1 the computed disk (dash line) is drawn against the exact one (solid line) after one (left figure) and two revolutions (right figure). The results seem to match the ones computed by the coupled level set and volume-of-fluid method [29]. In Figure 5.2, the mesh is refined

Table 5.2. Rotation of a slotted disk: average distance between the exact interface and the one computed by the CIR scheme with backward error compensation and redistancing, CFL=3.

Δx	average distance	order
1	0.138	-
0.5	0.0497	1.47
0.25	0.0211	1.23

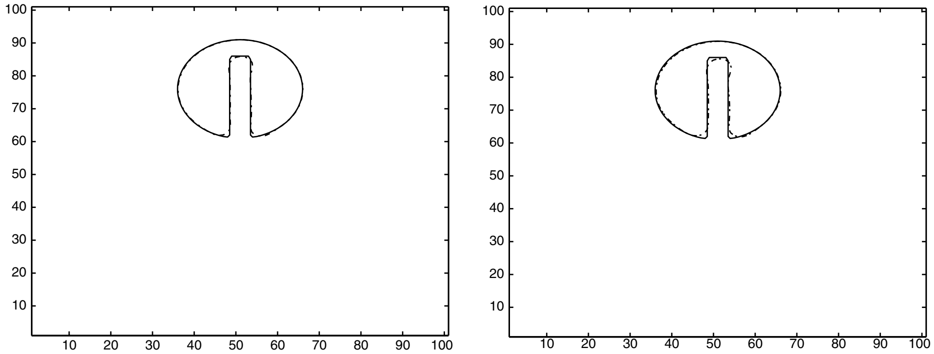

Figure 5.1. Zalesak's Problem. Comparison of a slotted disk that has been rotated one (left) and two revolutions (right). The level set equation is computed by the CIR scheme with backward error compensation and redistancing, CFL=3, 100×100 ($\Delta x = 1$).

with $N = 200$ ($\Delta x = 0.5$). The average distances (defined and computed as in [28]) between the exact and computed interfaces are shown in Table 5.2 for three meshes: 100×100, 200×200 and 400×400. The relative errors of the computed disk area A are plotted against time for the three meshes; see Figure 5.3.

The CIR scheme can be applied to irregular meshes. Applying the backward error compensation to it is essentially calling it 3 times. Therefore once a first order code is written down for an irregular mesh, the backward error compensation algorithm can be applied without too much work. In Kim *et al.* [13], the Zalesak's disk is initially put on the triangulated surface of a 3D sphere; see Figure 5.4. The slot is between 5 to 6 times the size of a triangular cell. There is a constant angular velocity field rotating around the Z-axis. In Figure 5.5, the level set method is computed by the first order CIR scheme. In Figure 5.6, the level set method is computed by the first order CIR scheme with backward error compensation. In both cases, the redistancing procedure is also implemented by the first order CIR scheme. Improved results similar to those on rectangular meshes can be observed in Figure 5.6, which clearly demonstrate the effectiveness of the backward error compensation algorithm on triangular meshes. Simulations of smoke advection on adaptive quad-tree meshes with backward error compensation algorithm have also been explored in [13] and are successful.

5.2. **Interfaces moving with nonsmooth velocity.** The velocity field directing the interface movement usually contains singularities. For example, if the interface is moving along its normal direction and contains corner points, the velocity

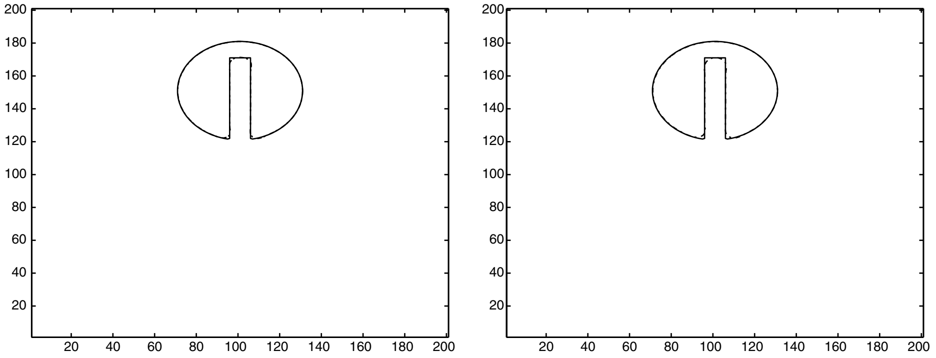

FIGURE 5.2. Zalesak's Problem. Comparison of a slotted disk that has been rotated one (left) and two revolutions (right). The level set equation is computed using the CIR scheme with backward error compensation and redistancing, $CFL = 3$, 200×200 ($\Delta x = 0.5$).

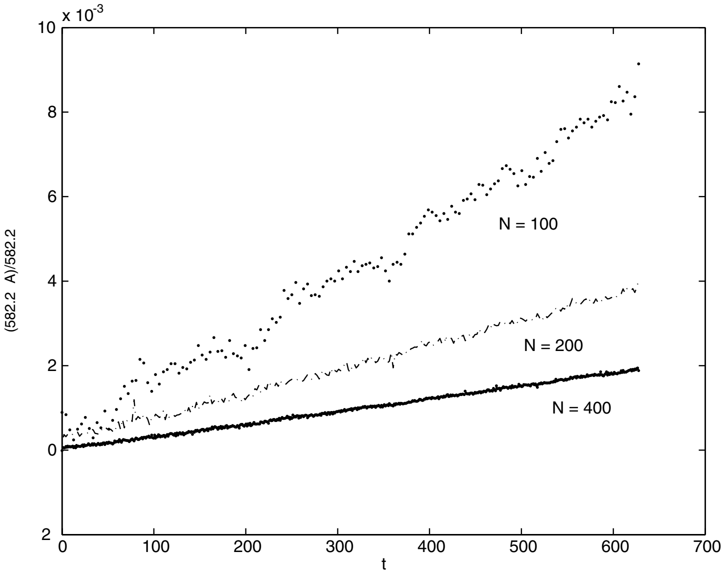

FIGURE 5.3. Zalesak's Problem. Relative area loss of the slotted disk as a function of time. The level set equation is computed using the CIR scheme with backward error compensation and redistancing, $CFL = 3$.

field is not continuous at these corner points. Simply applying the backward error compensation to compute the level set equation (2.1) may generate artifacts where the velocity field is not smooth. We propose two simple techniques to address the problem.

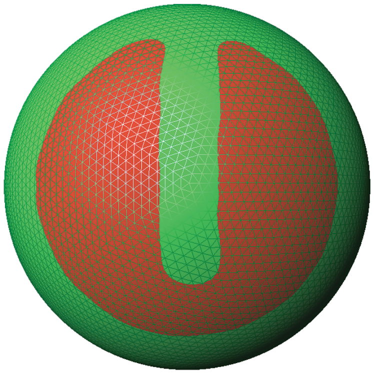

FIGURE 5.4. Zalesak's disk on a sphere. Reprinted from [13].

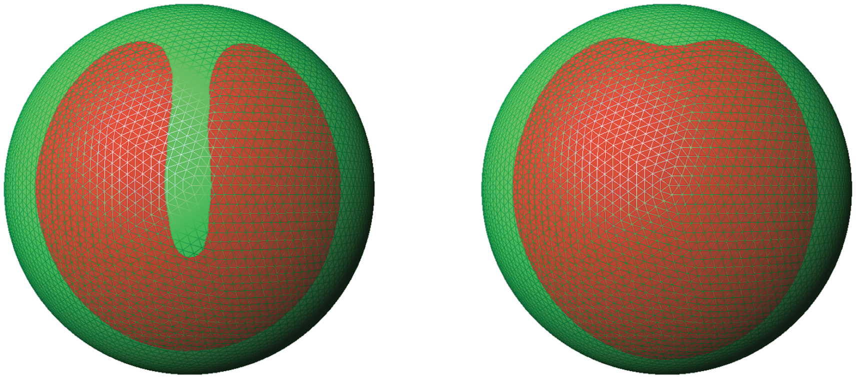

FIGURE 5.5. Zalesak's disk after one (left) and two (right) revolutions, computed by the CIR scheme. Reprinted from [13].

5.2.1. *Local turn-off of the error compensation.* The simplest way to overcome this problem is to set the backward compensation term (in the backward error compensation algorithm) to be zero wherever the nonsmoothness in the velocity field is detected. In the following examples we use the following detector. For a velocity field $\mathbf{V} = (u, v)$ in 2D defined on a uniform mesh, if at the grid point (x_i, y_j)

$$\begin{aligned} &||\mathbf{V}_{i+1,j} - 2\mathbf{V}_{i,j} + \mathbf{V}_{i-1,j}|| \leq \min(||\mathbf{V}_{i+1,j} - \mathbf{V}_{i,j}||, ||\mathbf{V}_{i,j} - \mathbf{V}_{i-1,j}||) \text{ and} \\ &||\mathbf{V}_{i,j+1} - 2\mathbf{V}_{i,j} + \mathbf{V}_{i,j-1}|| \leq \min(||\mathbf{V}_{i,j+1} - \mathbf{V}_{i,j}||, ||\mathbf{V}_{i,j} - \mathbf{V}_{i,j-1}||), \end{aligned} \tag{5.3}$$

we use the backward error compensation; otherwise we set the backward compensation term to zero. The advantage of this technique is its simplicity. Since the CIR scheme has very small diffusion compared to other first order schemes, it is good at maintaining the sharp corners of the interface without generating artifacts. However, this technique is sensitive to the choice of the nonsmoothness detector. When the nonsmoothness of the velocity field stays for a long time, this technique usually introduces too much diffusion into the solution.

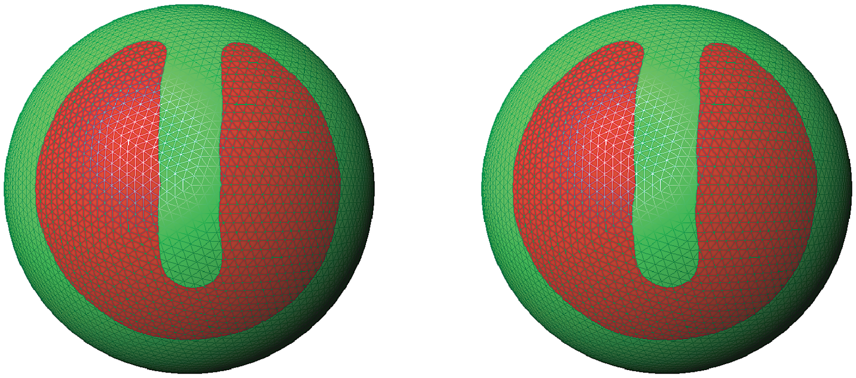

FIGURE 5.6. Zalesak's disk after one (left) and two (right) revolutions, computed by the CIR scheme with backward error compensation. Reprinted from [13].

Remark. A lower cost version of condition (5.3) can be formulated as

$$\begin{aligned} &|u_{i+1,j} - 2u_{i,j} + u_{i-1,j}| \leq \min(|u_{i+1,j} - u_{i,j}|, |u_{i,j} - u_{i-1,j}|) \ \text{ and} \\ &|v_{i,j+1} - 2v_{i,j} + v_{i,j-1}| \leq \min(|v_{i,j+1} - v_{i,j}|, |v_{i,j} - v_{i,j-1}|). \end{aligned} \tag{5.4}$$

When used with the local constant velocity technique introduced in Section 5.2.3, which is almost as accurate as the full backward error compensation algorithm, the difference between (5.3) and (5.4) is small in our numerical experiments. Note that one could also replace the u and v in (5.4) by their absolute values, which will also detect the stationary points of the velocity field.

5.2.2. *Improved interpolation technique for the CIR scheme.* The first order CIR scheme can generate some grid effects when the local velocity is almost zero. Consider equation (2.1) in 1D with the velocity $v(0) = 0$, $v(x) > 0$ for $x < 0$ and $v(x) < 0$ for $x > 0$. Initially the level set function is $\phi(x, 0) = -|x| + 0.5$. This corresponds to the case that two interfaces at $x = 0.5$ and -0.5 are about to merge. If computed with the first order CIR scheme, $\phi(0, t)$ will be 0.5 for all $t > 0$, *i.e.*, the two interfaces will never merge. See Figure 5.9. This phenomenon reminds us of the entropy-violating solution in the case of a sonic rarefaction wave when using Roe's approximate Riemann solver [21] for solving the Euler equation. There are many methods to fix the problem, such as Harten and Hyman [9], Osher [18], and Tadmor [31]. We are going to use a "velocity splitting" strategy, or a Lax–Friedrich framework analogous to the one used in the finite difference ENO scheme (Shu and Osher [23]). Recalling the CIR scheme (2.2), the following modified scheme can be used in every step of the backward error compensation algorithm (including the local constant velocity method in Section 5.2.3), only at places where the nonsmoothness of the velocity field is detected

$$\begin{aligned} \tilde{\Phi}(\mathbf{x_i}, t_{n+1}) &= \Phi(\hat{\mathbf{x}}_\mathbf{i} + \delta\mathbf{e}, t_n), \\ \tilde{\tilde{\Phi}}(\mathbf{x_i}, t_{n+1}) &= \Phi(\hat{\mathbf{x}}_\mathbf{i} - \delta\mathbf{e}, t_n), \\ \Phi(\mathbf{x_i}, t_{n+1}) &= \tfrac{1}{2}\{\tilde{\Phi}(\mathbf{x_i}, t_{n+1}) + \tilde{\tilde{\Phi}}(\mathbf{x_i}, t_{n+1})\}, \end{aligned} \tag{5.5}$$

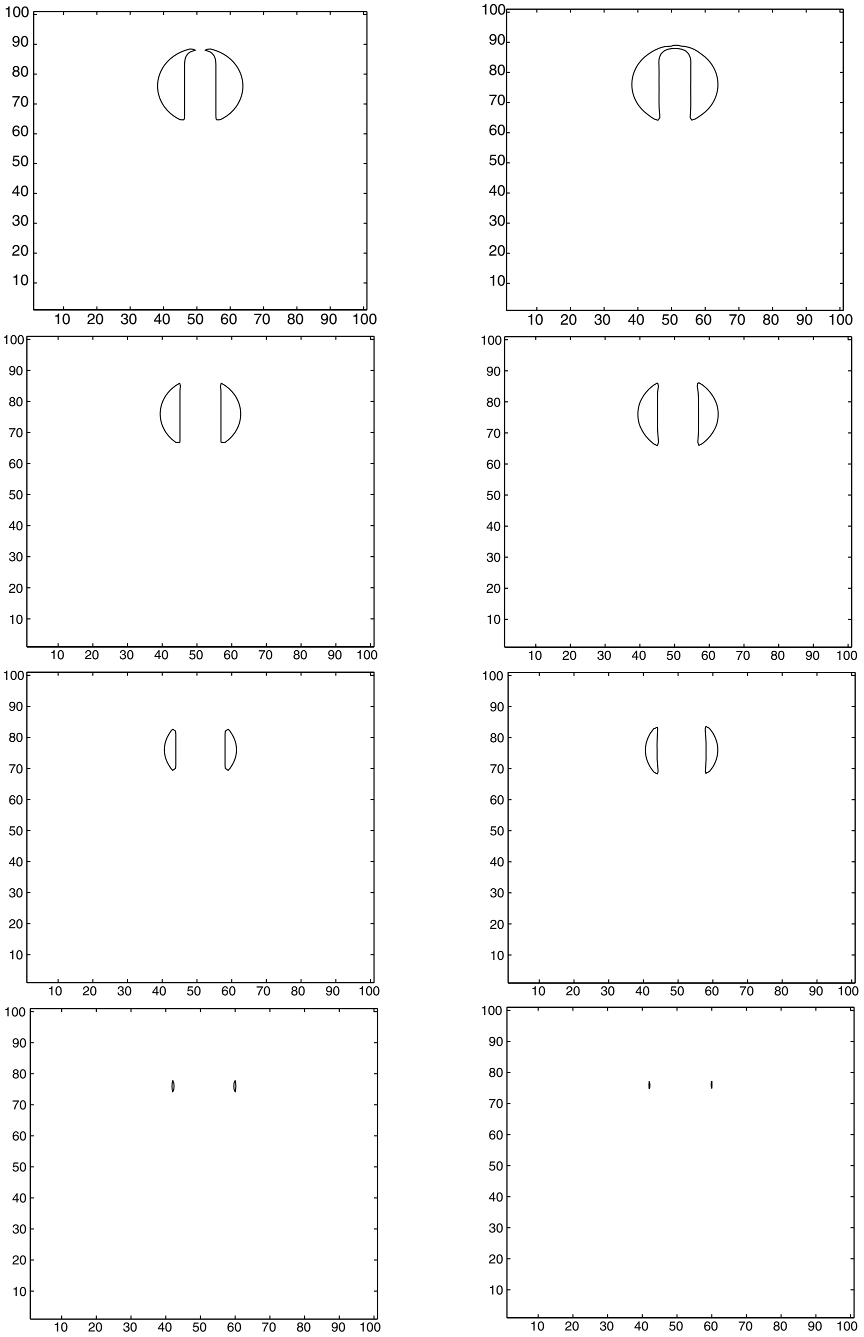

FIGURE 5.7. Shrinking Zalesak's Slotted Disk. Normal velocity 0.2, $\Delta x = \Delta y = 1$, $\Delta t = 0.4\Delta x$. Left: local turn-off of backward error comp. as in Section 5.2.1, $T = 11, 17, 23, 29$. Right: local constant velocity method as in Section 5.2.3 at times $T = 11, 17, 23, 31$.

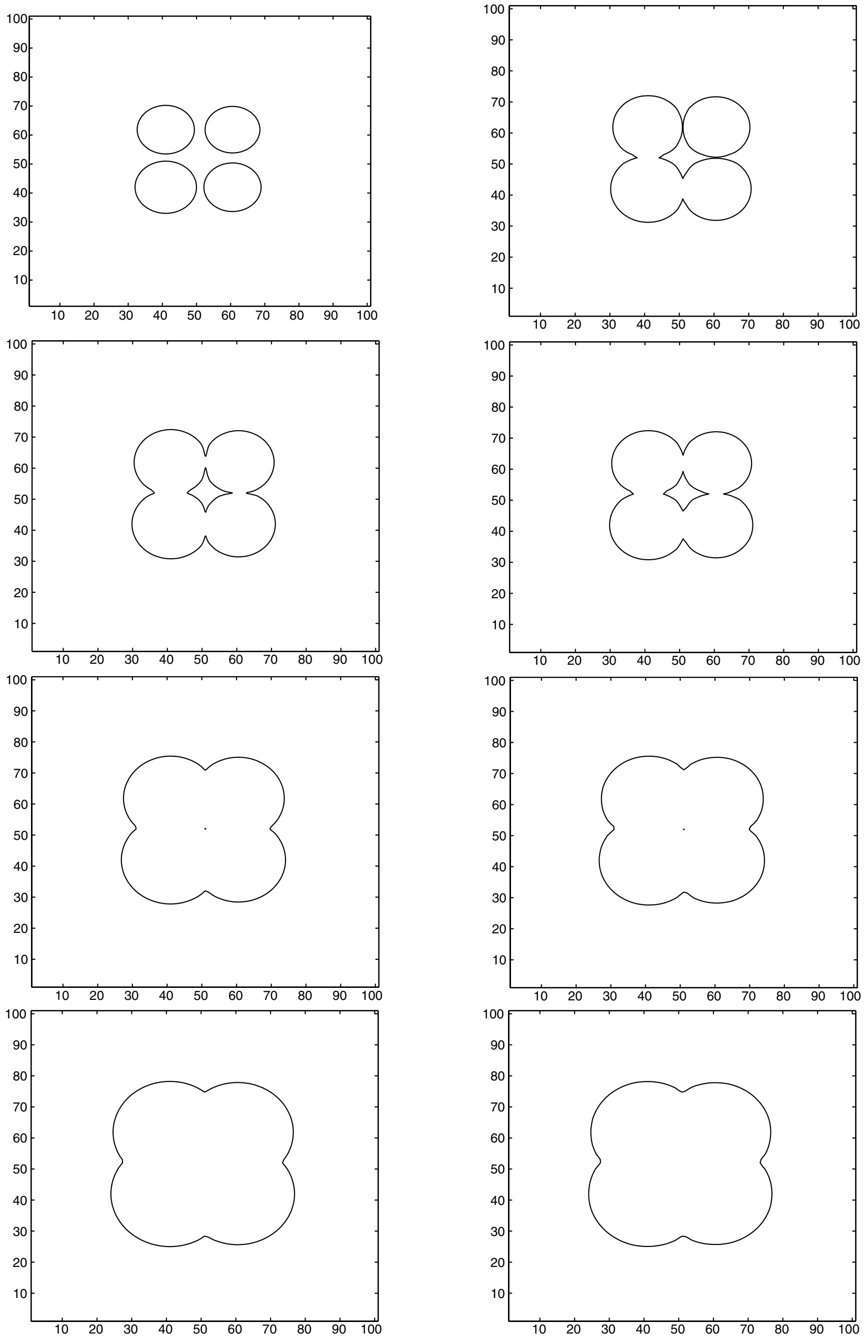

FIGURE 5.8. 4 expanding circles of slightly different radii. Normal velocity 0.2, $\Delta x = \Delta y = 1$, $\Delta t = 0.4\Delta x$. Left: local turn-off of backward error comp. as in Section 5.2.1, $T = 0, 11, 26, 40$. Right: local constant velocity method as in Section 5.2.3, $T = 9, 11, 27, 40$.

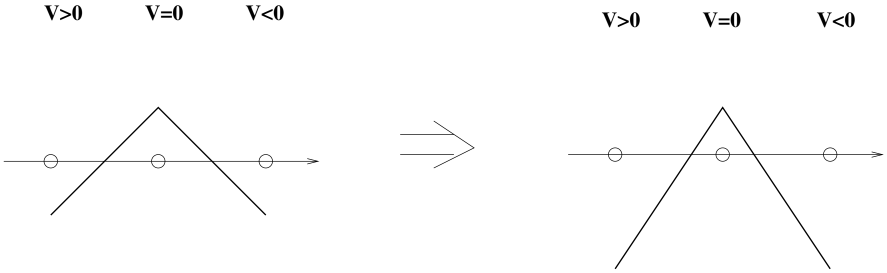

FIGURE 5.9. A nonmerging situation caused by the first order CIR scheme.

where $\hat{\mathbf{x}}_{\mathbf{i}} = \mathbf{x}_{\mathbf{i}} - \mathbf{v}(\mathbf{x}_{\mathbf{i}}, t_n)\Delta t_n$, $\Phi(\mathbf{y}, t_n)$ is the local linear spatial interpolation of $\{\Phi(\mathbf{x}_{\mathbf{j}}, t_n)\}$ at point $\mathbf{y}$, $\mathbf{e}$ is a unit vector not aligned with the grid, and $\delta \in (0, \Delta x)$ is a small perturbation factor. We choose $\delta\mathbf{e} = (0.2\Delta x, 0.2\Delta x)$ in 2D and $\delta\mathbf{e} = (0.2\Delta x, 0.2\Delta x, 0.2\Delta x)$ in 3D throughout the computation.

When the backward error compensation algorithm is applied, this technique reduces artifacts near interface corners and places where there are topological changes of the interface.

5.2.3. *Local constant velocity technique in the computation of the error compensation term.* We can locally freeze the velocity field in the first two steps of the backward error compensation algorithm, motivated by the less diffusive results for Zalesak's Problem in which the velocity field is smooth. For every grid point $\mathbf{x}_{\mathbf{i}}$ where the nonsmoothness of the velocity field is detected, do the following two steps:

Step 1. Solve the equation $\frac{\partial \phi}{\partial t} + \bar{\mathbf{v}}_{\mathbf{i}} \cdot \bigtriangledown\phi = 0$ forward in time to obtain $\tilde{\Phi}^{n+1}$ by the CIR scheme (2.2), with Φ^n being the initial value at time t_n and $\bar{\mathbf{v}}_{\mathbf{i}}(\mathbf{x}) = \mathbf{v}(\mathbf{x}_{\mathbf{i}}, t_n)$ for any $\mathbf{x}$.

Step 2. Solve the same equation backward in time by the same method to obtain $\breve{\Phi}^n_{\mathbf{i}}$. This is equivalent to solving the time reversed equation $\frac{\partial \phi}{\partial t} - \bar{\mathbf{v}}_{\mathbf{i}} \cdot \bigtriangledown\phi = 0$ forward in time by (2.2) for a time step Δt_n, with $\tilde{\Phi}^{n+1}$ being the initial value.

Note that $\tilde{\Phi}^{n+1}$ at only a few grid points needs to be calculated in Step 1 in order to obtain $\breve{\Phi}^n_{\mathbf{i}}$ in Step 2. For all other grid points $\breve{\Phi}^n_{\mathbf{i}}$ can be obtained in batch by the first two steps of the backward error compensation algorithm described in

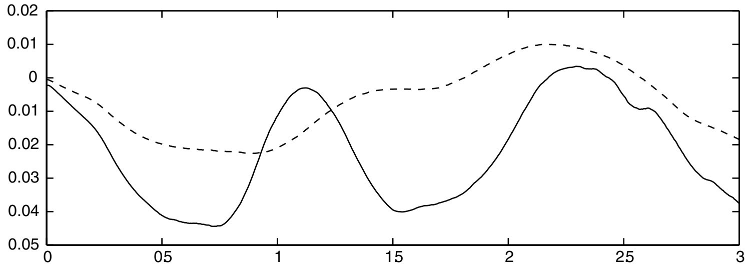

FIGURE 5.10. Relative volume error $(V(t) - \frac{4}{3}\pi 0.15^3)/(\frac{4}{3}\pi 0.15^3)$ against time. Solid line: 100^3 mesh; dash line: 200^3 mesh.

Section 2. After $\breve{\Phi}_{\mathbf{i}}^n$ at every grid point $\mathbf{x_i}$ is computed, we resume the original nonhomogeneous velocity field $\mathbf{v}(\mathbf{x}, t)$ and perform the usual Steps 3 and 4 of the backward error compensation algorithm.

Remarks. 1. Note that near singular points of the velocity field, essentially all numerical schemes degenerate to no more than first order. Being able to approximate the exact solution without generating excessive diffusion near singular points is important, and it is usually nontrivial to do. This technique does not seem to be sensitive to the choice of the nonsmoothness detector for the velocity field in the sense that one could apply it to a larger set of grid points containing those where there are singularities in the velocity field, without introducing excessive diffusion. In fact we have even applied this local constant velocity algorithm to all grid points in the computation of Zalesak's Problem and generated similar (slightly worse) results as in Figure 5.1. On the other hand, for this problem, if we set the backward compensation term to zero for all grid points (*i.e.*, apply the method of Section 5.2.1 everywhere), the slotted disk will shrink to almost nothing (with $\Delta x = 1$) after one revolution.

2. We have also tried to use $\bar{\mathbf{v}}_{\mathbf{i}}(\mathbf{x}) = \mathbf{v}(\mathbf{x_i}, t_{n+1})$ for any $\mathbf{x}$ in Step 2 of the above algorithm, for many of our test examples and the results are similar.

3. The backward error compensation algorithm can also be used as an interpolation technique (3rd order in smooth region) for Lipschitz continuous data. For example, given the Φ values at grid points $\{\mathbf{x_i}\}$, we can obtain an approximate Φ value at a point $\mathbf{y}$ as follows. First set $\mathbf{v} \equiv (\mathbf{x_j} - \mathbf{y})/\Delta x$, where $\mathbf{x_j}$ is a grid point closest to $\mathbf{y}$. Then solve equation (2.1) by the CIR scheme with backward error compensation for one step with the time step size Δx. The new Φ value at $\mathbf{x_j}$ will be the interpolated Φ value at $\mathbf{y}$. This is similar to the idea of extending a quantity defined on the interface to a neighborhood of the interface with a PDE-based method; see [34, 1]. Since it only requires local linear interpolation in space, it could be easily adapted to a triangular mesh or other irregular meshes. This will be further explored in the future.

5.3. Examples.

We present an example based on the slotted disk shown in Figure 5.1 in which the disk is shrinking with a normal speed 0.2. (The velocity field is $\mathbf{v} = -0.2 \bigtriangledown \phi / | \bigtriangledown \phi|$.) The disks at the different times are plotted in Figure 5.7. At the time $T = 11$ we can see that the upper corners of the slot become rounded and the lower corners stay sharp, which coincide with the entropy solution. Note that the exact annihilation time is $T = 31.25$. When computed by the method in Section 5.2.1, the disk vanishes between the times $T = 29$ and 30. Computed by the method in Sections 5.2.2 and 5.2.3, the disk vanishes between the times $T = 31$ and 32, which shows that the method has smaller diffusion near singularities.

Next we compute the merging of 4 circles expanding with constant normal velocity 0.2. The velocity field is given as $\mathbf{v} = 0.2 \bigtriangledown \phi / | \bigtriangledown \phi|$. The level set function is initially set to be negative inside the circles. The results are shown in Figure 5.8. Clearly all methods are able to maintain the sharp corners after merging without generating artifacts. The exact annihilation time of the inner part is estimated to be between $T = 27$ and 27.7. At the time $T = 27$ we can see from the graph that the inner part computed by the method in Sections 5.2.2 and 5.2.3 still exist (it disappears at $T = 28$), while the one computed by the method in Section 5.2.1 has vanished between $T = 26$ and 27.

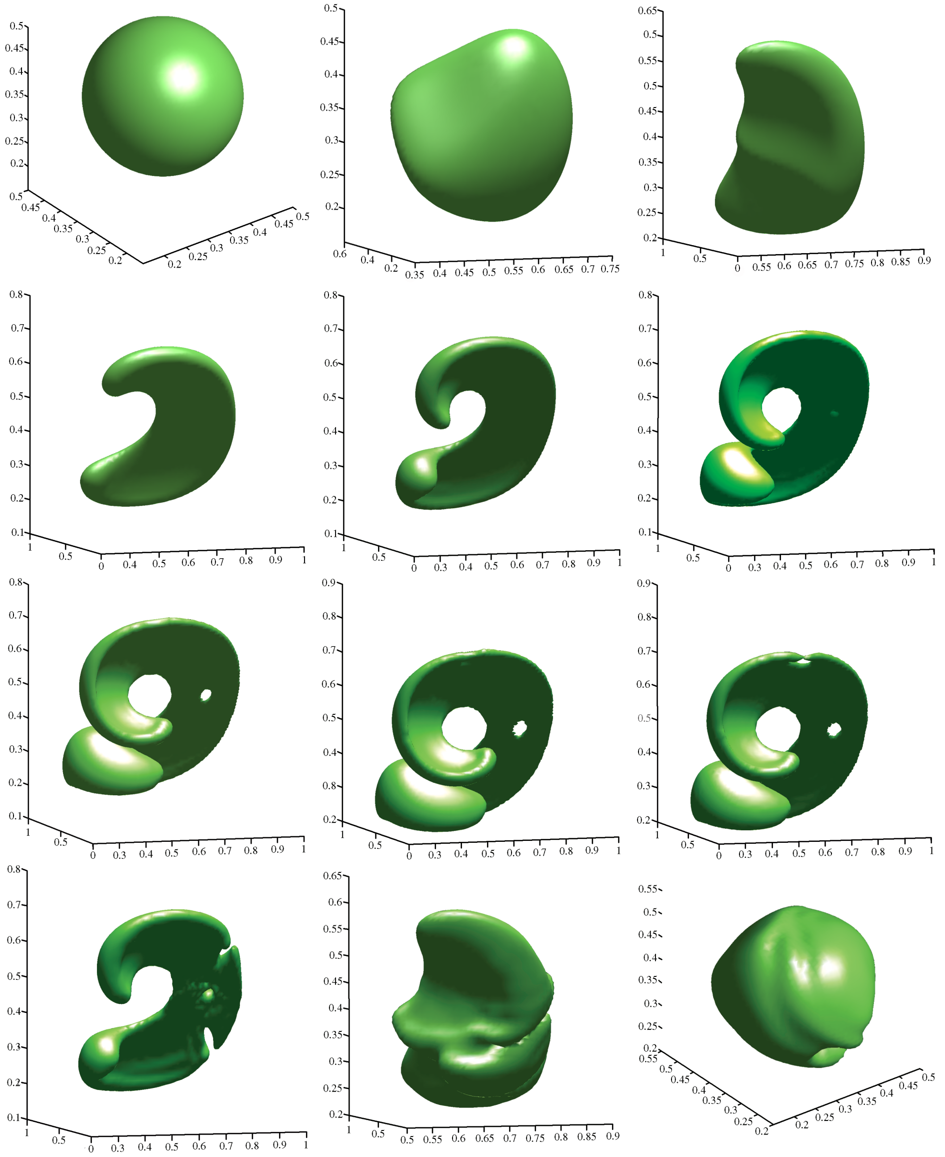

FIGURE 5.11. Deformation of a ball on a 100^3 mesh at times 0, $\frac{1}{5}$, $\frac{2}{5}$, $\frac{3}{5}$, $\frac{4}{5}$, $\frac{5}{5}$, $\frac{6}{5}$, $\frac{7}{5}$, $\frac{9}{5}$, $\frac{11}{5}$, $\frac{13}{5}$, 3.

Finally we conduct a test on the Enright problem [4]. It is a 3D sphere deformed by an incompressible flow field proposed by Leveque [15]. The computational domain is $[0,1]\times[0,1]\times[0,1]$ with $100\times 100\times 100$ uniform rectangular cells. Initially a sphere of radius 0.15 is centered at $(0.35, 0.35, 0.35)$. The velocity field is given by

$$\begin{aligned} u(x,y,z,t) &= 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)g(t), \\ v(x,y,z,t) &= -\sin(2\pi x)\sin^2(\pi y)\sin(2\pi z)g(t), \\ w(x,y,z,t) &= -\sin(2\pi x)\sin(2\pi y)\sin^2(\pi z)g(t), \end{aligned}$$

where $g(t) = \cos(\pi t/T)$ and $T = 3$. From the velocity field, one can see that the deforming process is time-reversed at $t = T/2$ and will restore the original sphere at $t = T$. At $t = T/2$, the 3D sphere has been deformed to be like a pancake, where some parts are as thin as the size of a grid cell. We compute the problem with the methods in Sections 5.2.2 and 5.2.3. The time step size is chosen with CFL factor 0.2 and no larger than $0.2\Delta x$. The computational results are shown at different times in Figure 5.11. On a 1.8 GHz processor (AMD Opteron 244), the computation takes 2 hours and 28 seconds. From the graphs we can see that the deformed sphere starts to break down near its thinnest part at $T = 1.2$. Since the recovered sphere at $t = T$ depends sensitively on the thickness of the deformed one at $t = T/2$, when the level set representation of the interface is at its limit, we observe large errors at later stages. The particle level set method [4] uses Lagrangian particles to correct the level set solution and is very accurate in recovering the sphere. However, compared to a standard high order implementation of the level set method (80% volume error at $t = T$ in a comparison test in [4]), the percentage volume error is small (under 4.5% during the computational time interval; see Figure 5.10). The computational results on a $200 \times 200 \times 200$ mesh can be found in Figure 5.12. The volume error is within 2.3%; see Figure 5.10.

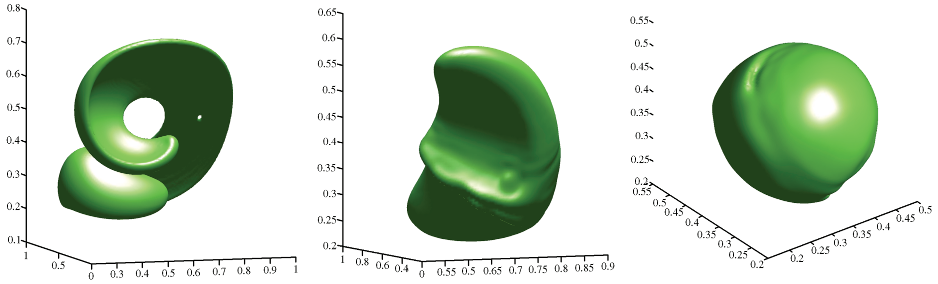

FIGURE 5.12. Deformation of a ball on a 200^3 mesh at times $\frac{7}{5}$, $\frac{13}{5}$, 3.

REFERENCES

[1] S. CHEN, B. MERRIMAN, S. OSHER, AND P. SMEREKA, *A simple level set method for solving Stefan problems*, J. Comput. Phys., 135 (1997), pp. 8–29. MR1461705 (98c:80002)

[2] R. COURANT, E. ISAACSON, AND M. REES, *On the solution of nonlinear hyperbolic differential equations by finite differences*, Comm. Pure Appl. Math., 5 (1952), pp. 243–255. MR0053336 (14:756e)

[3] T. F. DUPONT AND Y.-J. LIU, *Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function*, J. Comput. Phys., 190 (2003), pp. 311–324. MR2046766 (2004m:65162)

[4] D. ENRIGHT, R. FEDKIW, J. FERZIGER, AND I. MITCHELL, *A hybrid particle level set method for improved interface capturing*, J. Comput. Phys., 183 (2002), pp. 83–116. MR1944529 (2003j:76084)

[5] D. ENRIGHT, F. LOSASSO, AND R. FEDKIW, *A fast and accurate semi-Lagrangian particle level set method*, Computers and Structures, 83 (2005), pp. 479–490. MR2143508 (2005k:76104)

[6] M. FALCONE AND R. FERRETTI, *Convergence analysis for a class of high-order semi-Lagrangian advection schemes*, SIAM J. Numer. Anal., 35 (1998), pp. 909–940. MR1619910 (99c:65164)

[7] J. Glimm, J. W. Grove, X.-L. Li, K.-M. Shyue, Q. Zhang, and Y. Zeng, *Three dimensional front tracking*, SIAM J. Sci. Comput., 19 (1998), pp. 703–727. MR1616658 (99c:35153)

[8] J. Glimm, D. Marchesin, and O. McBryan, *Subgrid resolution of fluid discontinuities, II*, J. Comput. Phys., 37 (1980), pp. 336–354. MR0588257 (81k:76041)

[9] A. Harten and J. M. Hyman, *Self-adjusting grid methods for one-dimensional hyperbolic conservation laws*, J. Comput. Phys., 50 (1983), pp. 235–269. MR0707200 (85g:65111)

[10] A. Harten, S. Osher, B. Engquist, and S. Chakravarthy, *Uniformly high order accurate essentially non-oscillatory schemes, III*, J. Comput. Phys., 71 (1987), pp. 231–303. MR0897244 (90a:65199)

[11] G.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys., 126 (1996), pp. 202–228. MR1391627 (97e:65081)

[12] B.-M. Kim, Y.-J. Liu, I. Llamas, and J. Rossignac, *FlowFixer: Using BFECC for fluid simulation*, Eurographics Workshop on Natural Phenomena (2005).

[13] ———, *Advections with significantly reduced dissipation and diffusion*, IEEE Transactions on Visualization and Computer Graphics (in press).

[14] P. D. Lax, *On the stability of difference approximations to solutions of hyperbolic equations with variable coefficients*, Comm. Pure Appl. Math., 14 (1961), pp. 497–520. MR0145686 (26:3215)

[15] R. LeVeque, *High-resolution conservative algorithms for advection in incompressible flow*, SIAM J. Numer. Anal., 33 (1996), pp. 627–665. MR1388492 (98b:76049)

[16] X. D. Liu, S. Osher, and T. Chan, *Weighted essentially non-oscillatory schemes*, J. Comput. Phys., 115 (1994), pp. 200–212. MR1300340

[17] R. W. MacCormack, AIAA Paper 69-354 (1969).

[18] S. Osher, *Riemann solvers, the entropy condition, and difference approximations*, SIAM J. Numer. Anal., 21 (1984), pp. 217–235. MR0736327 (86d:65119)

[19] S. Osher and J. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi equations*, J. Comput. Phys, 79 (1988), pp. 12–49. MR0965860 (89h:80012)

[20] W. Rider and D. Kothe, *A marker particle method for interface tracking*, Proceedings of the Sixth International Symposium on Computational Fluid Dynamics, 163 (1995), p. 976.

[21] P. L. Roe, *Approximate Riemann solvers, parameter vectors, and difference schemes*, J. Comput. Phys, 43 (1981), pp. 357–372. MR0640362 (82k:65055)

[22] G. Russo and P. Smereka, *A remark on computing distance functions*, J. Comput. Phys, 163 (2000), pp. 51–67. MR1777721 (2001d:65139)

[23] C.-W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, J. Comput. Phys, 77 (1988), pp. 439–471. MR0954915 (89g:65113)

[24] ———, *Efficient implementation of essentially non-oscillatory shock-capturing schemes, II*, J. Comput. Phys, 83 (1989), pp. 32–78. MR1010162 (90i:65167)

[25] J. Strain, *Semi-Lagrangian methods for level set equations*, J. Comput. Phys, 151 (1999), pp. 498–533. MR1686375 (2000a:76133)

[26] ———, *A fast modular semi-Lagrangian method for moving interfaces*, J. Comput. Phys, 161 (2000), pp. 512–536. MR1764248

[27] ———, *A fast semi-Lagrangian contouring method for moving interfaces*, J. Comput. Phys, 170 (2001), pp. 373–394. MR1843614 (2002c:65035)

[28] M. Sussman and E. Fatemi, *An efficient, interface preserving level set re-distancing algorithm and its application to interfacial incompressible fluid flow*, SIAM J. Sci. Comput., 20 (1999), pp. 1165–1191. MR1675468 (2000f:65084)

[29] M. Sussman and E. G. Puckett, *A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows.*, J. Comput. Phys, 162 (2000), pp. 301–337. MR1774261 (2001c:76099)

[30] M. Sussman, P. Smereka, and S. Osher, *A level set method for computing solutions to imcompressible two-phase flow*, J. Comput. Phys, 119 (1994), pp. 146–159.

[31] E. Tadmor, *Numerical viscosity and the entropy condition for conservative difference schemes*, Math. Comp., 43 (1984), pp. 369–381. MR0758189 (86g:65163)

[32] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan, *A front-tracking method for the computations of multiphase flow*, J. Comput. Phys, 169 (2001), pp. 708–759.

[33] S. T. Zalesak, *Fully multidimensional flux-corrected transport*, J. Comput. Phys., 31 (1979), pp. 335–362. MR0534786 (80f:76048)

[34] H.-K. Zhao, T. Chan, B. Merriman, and S. Osher, *A variational level set approach to multiphase motion*, J. Comput. Phys., 127 (1996), pp. 179–195. MR1408069 (97g:80013)

Department of Computer Science, University of Chicago, Chicago, Illinois 60637
E-mail address: dupont@cs.uchicago.edu

School of Mathematics, Georgia Institute of Technology, Atlanta, Georgia 30332
E-mail address: yingjie@math.gatech.edu