Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Short Note

On reinitializing level set functions

Chohong Min

Mathematics Department, KyungHee University, Seoul 130-701, Republic of Korea

ARTICLE INFO

Article history:
Received 30 June 2009
Received in revised form 18 November 2009
Accepted 27 December 2009
Available online 4 January 2010

Keywords:
Level set method
Reinitialization
Subcell fix
ENO
Gauss–Seidel

ABSTRACT

In this paper, we consider reinitializing level functions through equation $\phi_t + sgn(\phi^0)(\|\nabla\phi\| - 1) = 0$ [16]. The method of Russo and Smereka [11] is taken in the spatial discretization of the equation. The spatial discretization is, simply speaking, the second order ENO finite difference with subcell resolution near the interface. Our main interest is on the temporal discretization of the equation. We compare the three temporal discretizations: the second order Runge–Kutta method, the forward Euler method, and a Gauss–Seidel iteration of the forward Euler method. The fact that the time in the equation is fictitious makes a hypothesis that all the temporal discretizations result in the same result in their stationary states. The fact that the absolute stability region of the forward Euler method is not wide enough to include all the eigenvalues of the linearized semi-discrete system of the second order ENO spatial discretization makes another hypothesis that the forward Euler temporal discretization should invoke numerical instability. Our results in this paper contradict both the hypotheses. The Runge–Kutta and Gauss–Seidel methods obtain the second order accuracy, and the forward Euler method converges with order between one and two. Examining all their properties, we conclude that the Gauss–Seidel method is the best among the three. Compared to the Runge–Kutta, it is twice faster and requires memory two times less with the same accuracy.

1. Introduction

The level set method [10] represents an interface $\Gamma \subset \mathbb{R}^d$ as the zero level set of a continuous function $\phi : \mathbb{R}^d \rightarrow \mathbb{R}$, and tracks the movement of the interface through a convection equation $\phi_t + V \cdot \nabla\phi = 0$, where V is the velocity of the movement. In this way, the level set method converts the geometric problem into a partial differential equation, and enables the well established technologies of partial differential equations to work in solving the geometric problem.

The evolution of $\phi_t + V \cdot \nabla\phi = 0$ often distorts the level function in a sense that its slope is too flat or too steep around the interface. In such cases, a small pertubation of the level function may result in a big change of interface location, and the level function may lose enough regularity near the interface. It is therefore desired to replace the level function with a better behaved one, the signed distance function to the interface. The signed distance function has many advantages: it is uniquely determined as the viscosity solution of the Eikonal equation [3], and the magnitude of its gradient is uniform. Given a level function $\phi^0 : \mathbb{R}^d \rightarrow \mathbb{R}$, the reinitialization process finds the signed distance function to the interface $\Gamma^0 = \{x|\phi^0(x) = 0\}$. The reintialization can be thought of calculating two distance functions to Γ^0. One distance function is calculated in the region $\{x|\phi^0(x) > 0\}$ with the positive sign and the other one is in the region $\{x|\phi^0(x) < 0\}$ with the negative sign. The signed distance function is the viscosity solution of the following Eikonal equation

$$\begin{cases} \|\nabla\phi\| = 1 \\ sgn(\phi) = sgn(\phi^0) \end{cases}. \tag{1}$$

E-mail address: chohong@khu.ac.kr

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.12.032

Here *sgn* denotes the sign value, taking either 1, -1, or 0. In [3], Crandall et al proved the convergence of monotone finite difference methods to the viscosity solution of the Eikonal equation. One of the most efficient methods is combine the monotone Godunov Hamiltonian [1] with the ENO finite differences [13,7]. The method discretized at every grid node constitutes a large system. There are mainly two approches for solving the system of Eq. (1). In a single pass, fast marching method [12] solves the system by visiting grid nodes in the order of causality. The visiting order of causality is implemented by the Heap sorting algorithm. Though it takes more than one passes, fast sweeping method [8] does not have to visit grid nodes in the order of causility, but visit grid nodes in the simple raster-scan orderings.

In the case of solving the evolutionary equation $\phi_t + V \cdot \nabla\phi = 0$, there is a more suitable equation for the reinitialization than the stationary Eq. (1). If the level function was reinitialized at the previous time step, the level function at the current is already very similar to the signed distance function. For such cases, the following time dependent Eikonal equation, introduced in [16], works very efficiently.

$$\begin{cases} \phi_t + sgn(\phi^0)(\|\nabla\phi\| - 1) = 0 \\ \phi(x,0) = \phi^0(x) \end{cases} \tag{2}$$

If a level function ϕ^0 is similar to the signed distance function, the time evolution of Eq. (2) with a tiny time span will obtain the signed distance function. The solution $\phi(x,t)$ converges to the signed distance function as $t \to \infty$, which can be easily verified by solving the characteristic ordinary differential equation of the above equation. Without the signum term, Eq. (2) is a Hamilton–Jacobi equation that has been successfully discretized by the Runge–Kutta method in time and the ENO finite differences in space. The signum plays an important role to fix the interface during the reinitialization, but its discontinuity invokes a lot of difficulties in the RK2–ENO coupling. In [16], the signum term was smeared out in a narrow band of the interface, and Eq. (2) was treated as a standard Hamilton–Jacobi equation with smooth Hamiltonian, but the artificial smearing moves the interface and would decrease the volume inside the interface in considerable amount. In [14], volume conservation was imposed in reinitialization to prevent such artificial volume shrinking. A simple but very efficient treatment of the signum term was achieved in [11,2] by adopting a subcell resolution technique [6]. The treatment basically approximates the interface location in the stencil of ENO finite differences and separately solves Eq. (2) in region $\{\phi^0 > 0\}$and $\{\phi^0 < 0\}$. Through the separation, the equation is treated as a Hamilton–Jacobi equation with smooth Hamiltonian, and the signum term need not be smoothed.

Throughout this paper, we consider reinitializing level functions through Eq. (2). The method of [9], which is a slight improvement of Russo and Smereka [11], is taken as the spatial discretization of the equation. The spatial discretization is, simply speaking, the second order ENO finite difference with subcell resolution near the interface. Our main interest is on the temporal discretization of the equation. We compare the three temporal discretizations; the second order Runge–Kutta method, the forward Euler method, and a Gauss–Seidel iteration of the forward Euler method. The solution of the equation reaches a stationary state, and any temporal derivative vanishes in the stationary state. The observation that the temporal derivative does not matter in the stationary state suggests that all the three temporal discretizations give the same results eventually. On the other hand, note that the absolute stability region of the forward Euler method is not wide enough to include all the linearized eigenvalues of the second order ENO spatial discretization. That observation suggests that the forward Euler temporal discretization should invoke numerical instability. Our results in this paper contradict both the suggestions. The forward Euler and the Gauss–Seidel methods are not numerically unstable. The solution of the Runge–Kutta method is different from that of the forward Euler, but is nearly the same as that of the Gauss–Seidel.

2. Spatial discretization

We take the spatial discretization of Eq. (2) by the method in [9,4], which is a slight improvement of Russo and Smereka [11]. We briefly review the discretization method in this section. Since the discretization is carried out in a dimension-by-dimension manner, we consider only the two dimensional case.

2.1. Standard discretization of $\|\nabla\phi\|$

The Hamiltonian $\|\nabla\phi\|$is discretized by putting the second order ENO finite differences in the arguments of the Godunov numerical Hamiltonian [10].

$$\|\nabla\phi\|_{ij} \simeq H_G\left(D_x^+\phi_{ij}, D_x^-\phi_{ij}, D_y^+\phi_{ij}, D_y^-\phi_{ij}\right)$$

Here $D_x^\pm$ and $D_y^\pm$ denote the one-sided ENO finite differences at x- and y-directions. Since it is a dimension by dimension approach, we state only the x-direction

$$D_x^+\phi_{ij} = \frac{\phi_{i+1,j} - \phi_{ij}}{\Delta x} - \frac{\Delta x}{2}\mathrm{minmod}\left(D_{xx}\phi_{ij}, D_{xx}\phi_{i+1,j}\right),$$
$$D_x^-\phi_{ij} = \frac{\phi_{i,j} - \phi_{i-1,j}}{\Delta x} + \frac{\Delta x}{2}\mathrm{minmod}\left(D_{xx}\phi_{ij}, D_{xx}\phi_{i-1,j}\right).$$

Here $D_{xx}\phi_{ij} = (\phi_{i-1,j} - 2\phi_{ij} + \phi_{i+1,j})/(\Delta x^2)$ is the central approximation of the derivative ϕ_{xx} at (x_i, y_j). The minmod function is zero when the two arguments have different signs, and takes the argument with smaller absolute value when the two have the same sign. The Godunov Hamiltonian H_G is given as

$$H_G(a,b,c,d) = \begin{cases} \sqrt{\max((a^-)^2,(b^+)^2) + \max((c^-)^2,(d^+)^2)} & \text{when } sgn(\phi^0) \geqslant 0, \\ \sqrt{\max((a^+)^2,(b^-)^2) + \max((c^+)^2,(d^-)^2)} & \text{when } sgn(\phi^0) < 0. \end{cases}$$

2.2. Subcell fix near interface

Near the interface, the finite differences $D_x^{\pm}\phi_{ij}$ and $D_y^{\pm}\phi_{ij}$ in the previous section are modified to impose the condition that $\phi = 0$ whenever $\phi^0 = 0$. Let us first consider modifying $D_x^+\phi_{ij}$ in the case $\phi^0_{ij} \cdot \phi^0_{i+1,j} < 0$. On interval $[x_i, x_{i+1}] \times \{y_j\}$, there exists an interface point (x_Γ, y_j) on which ϕ^0 is zero. Using the values $\phi^0_{i-1,j}, \phi^0_{i,j}, \phi^0_{i+1,j}$, and $\phi^0_{i+2,j}$, we construct the quadratic ENO polynomial of ϕ^0, and the root of the polynomial approximates the interface location x_Γ. An elementary algebra solves $\Delta x^+ = x_\Gamma - x_i$ in the procedure as

$$\Delta x^+ = \begin{cases} \Delta x \cdot \left(\frac{1}{2} + \frac{\phi^0_{ij} - \phi^0_{i+1,j} - sgn\left(\phi^0_{ij} - \phi^0_{i+1,j}\right)\sqrt{D}}{\phi^0_{xx}} \right) & \text{if} \quad |\phi^0_{xx}| > \epsilon, \\ \Delta x \cdot \frac{\phi^0_{ij}}{\phi^0_{ij} - \phi^0_{i+1,j}} & \text{else.} \end{cases}$$

Here $\phi^0_{xx} = \text{ minmod}(\phi^0_{i-1j} - 2\phi^0_{ij} + \phi^0_{i+1,j}, \phi^0_{ij} - 2\phi^0_{i+1,j} + \phi^0_{i+2,j})$ is the minmod value of the two undivided differences of $\phi^0, D = (\phi^0_{xx}/2 - \phi^0_{ij} - \phi^0_{i+1,j})^2 - 4\phi^0_{ij}\phi^0_{i+1,j}$ is the discriminant of the quadratic polynomial, and *sgn* is either 1 or -1. The condition $|\phi^0_{xx}| \leqslant \epsilon$ indicates that the quadratic polynomial is nearly linear, and the interface location should be linearly interpolated. In all tried examples, we took $\epsilon = 10^{-10}$. Now we impose $\phi = 0$ on the calculated interface point (x_Γ, y_j), and the finite difference $D_x^+\phi_{ij}$ is modified as

$$D_x^+\phi_{ij} = \frac{0 - \phi_{ij}}{\Delta x^+} - \frac{\Delta x^+}{2} \text{ minmod}(D_{xx}\phi_{ij}, D_{xx}\phi_{i+1,j}).$$

Since the modification works in a dimension-by-dimension manner, and we complete this section with the formula of $D_x^-\phi_{ij}$ in the case of $\phi^0_{ij} \cdot \phi^0_{i-1,j} < 0$

$$\Delta x^- = \begin{cases} \Delta x \cdot \left(\frac{1}{2} + \frac{\phi^0_{ij} - \phi^0_{i-1,j} - sgn\left(\phi^0_{ij} - \phi^0_{i-1,j}\right)\sqrt{D}}{\phi^0_{xx}} \right) & \text{if} |\phi^0_{xx}| > \epsilon, \\ \Delta x \cdot \frac{\phi^0_{ij}}{\phi^0_{ij} - \phi^0_{i-1,j}} & \text{else.} \end{cases}$$

Here $\phi^0_{xx} = \text{ minmod}(\phi^0_{i-1j} - 2\phi^0_{ij} + \phi^0_{i+1,j}, \phi^0_{ij} - 2\phi^0_{i-1,j} + \phi^0_{i-2,j})$ and $D = (\phi^0_{xx}/2 - \phi^0_{ij} - \phi^0_{i-1,j})^2 - 4\phi^0_{ij}\phi^0_{i-1,j}$.

$$D_x^-\phi_{ij} = \frac{\phi_{ij} - 0}{\Delta x^-} + \frac{\Delta x^-}{2} \text{ minmod}(D_{xx}\phi_{ij}, D_{xx}\phi_{i-1,j}).$$

3. Temporal discretization

The spatial discretization in the previous section is, simply speaking, the second order ENO finite difference with subcell resolution near the interface. As the second order accuracy in space is matched up with the same accuracy in time, we take the following second order Runge–Kutta method (RK2) as our first choice of the temporal discretization. Among the several versions of the second order Runge–Kutta method, we take the TVD Runge–Kutta method

$$\begin{aligned} \tilde{\phi}^{n+1}_{ij} &= \phi^n_{ij} - \Delta t_{ij} \cdot sgn\left(\phi^0_{ij}\right) \cdot \left[H_G\left(D_x^+\phi^n_{ij}, D_x^-\phi^n_{ij}, D_y^+\phi^n_{ij}, D_y^-\phi^n_{ij}\right) - 1\right], \\ \tilde{\phi}^{n+2}_{ij} &= \tilde{\phi}^{n+1}_{ij} - \Delta t_{ij} \cdot sgn(\phi^0_{ij}) \cdot \left[H_G\left(D_x^+\tilde{\phi}^{n+1}_{ij}, D_x^-\tilde{\phi}^{n+1}_{ij}, D_y^+\tilde{\phi}^{n+1}_{ij}, D_y^-\tilde{\phi}^{n+1}_{ij}\right) - 1\right], \\ \phi^{n+1} &= \frac{\phi^n + \tilde{\phi}^{n+2}}{2}. \end{aligned} \tag{3}$$

The time is fictitious in the reinitialization equation; it just plays a role to lead the solution into a stationary state. When the solution reaches the stationary state, any consistent temporal discretization should vanish. This observation suggests that all the convergent temporal discretizations should result in the same accuracy. From this reason, our second choice is the forward Euler method (FE) which requires only half of the computational cost of RK2

$$\phi^{n+1}_{ij} = \phi^n_{ij} - \Delta t_{ij} \cdot sgn(\phi^0_{ij}) \cdot \left[H_G\left(D_x^+\phi^n_{ij}, D_x^-\phi^n_{ij}, D_y^+\phi^n_{ij}, D_y^-\phi^n_{ij}\right) - 1\right] \tag{4}$$

Since the time is fictitious, our third choice totally ignores the time and takes the following Gauss–Seidel iteration (GS); we do not advance the level function, but just update it.

$$\phi_{ij} = \phi_{ij} - \Delta t_{ij} \cdot sgn\left(\phi_{ij}^0\right) \cdot \left[H_G\left(D_x^+\phi_{ij}, D_x^-\phi_{ij}, D_y^+\phi_{ij}, D_y^-\phi_{ij}\right) - 1\right] \tag{5}$$

The visiting order of grid nodes in RK2 or in FE does not matter, but it does in GS. The causality visiting of fast marching method [12] is the best one in efficiency, but the simple raster-scan visiting of fast sweeping method [17], which is our choice, is efficient enough in practice. In a grid $Nx \times Ny$, the following four raster-scan visitings are alternatively taken.

```
for $i$ = 1:Nx
for $j$ = 1:Ny
update$\phi_{ij}$
```

```
for $i$ = 1:Nx
for $j$ = Ny:1
update$\phi_{ij}$
```

```
for $i$ = Nx:1
for $j$ = 1:Ny
update$\phi_{ij}$
```

```
for $i$ = Nx:1
for $j$ = Ny:1
update$\phi_{ij}$
```

In three-dimensions, there are eight raster-scan visitings, see the details in [17]. In all the three temporal discretizations, the term $sgn(\phi_{ij}^0)$ strictly takes a value among 1, 0, and -1. The time step Δt_{ij} is taken as

$$\Delta t_{ij} = \text{cfl} \cdot \min(\Delta x^+, \Delta x^-, \Delta y^+, \Delta y^-).$$

Without the presence of interface points, $\Delta x^{\pm}$ and $\Delta y^{\pm}$ are set to be Δx and Δy, respectively. With interface points present, i.e. $\phi_{ij}^0 \cdot \phi_{i\pm1,j}^0 < 0$ or $\phi_{ij}^0 \cdot \phi_{i,j\pm1}^0 < 0$, they denote the distances from the grid node (x_i, y_j) to the interface points, and are calculated as described in the previous section. The cfl number is taken to be .45 in two-dimensions and .3 in three-dimensions. Note that the time steps are not uniform; it becomes smaller at a grid node next to the interface.

4. Numerical experiments

We compare RK2, FE, and GS temporal discretizations in two and three dimensions. One iteration convects the distance function in the amount of $\text{cfl} \cdot \min(\Delta x, \Delta y, \Delta z)$. The cfl numbers are .45(2D) and .3(3D). To guarantee its full convergence, each method in a grid $Nx \times Ny \times Nz$ is iterated $3 \cdot \max(Nx, Ny, Nz)$ times. Each method in a grid $Nx \times Ny$ is iterated $2 \cdot \max(Nx, Ny)$ times.

4.1. 2D smooth interface

We test the reinitialization problem proposed in [15]. The initial level set function is defined in a computational domain $[-2, 2]^2$ as

$$\phi^0(x, y) = ((x-1)^2 + (y-1)^2 + 0.1)\left(\sqrt{x^2 + y^2} - 1\right).$$

It defines the interface as a circle with center at the origin and radius 1. The function is not a signed distance function and its gradients vary widely. Table 1 shows that all the three methods are third order accurate near the interface. In the whole do-

Table 1
Accuracy of RK2, FE, and GS methods for the example of 2D smooth interface: the error near the interface is measured at nodes (i,j) with condition $|\phi_{ij}| < 1.2 \cdot \Delta x$, and the error in the whole domain with condition $\phi_{ij} > -.8$. Note that the kink point (0,0) is excluded with the latter condition.

	Error in the whole domain				Error near the interface			
Grid	L^1 error	Rate	L^∞ error	Rate	L^1 error	Rate	L^∞ error	Rate
RK2								
64^2	2.75×10^{-4}		4.14×10^{-3}		3.67×10^{-5}		1.84×10^{-4}	
128^2	7.61×10^{-5}	1.86	1.53×10^{-3}	1.44	4.42×10^{-6}	3.06	2.15×10^{-5}	3.10
256^2	1.99×10^{-5}	1.94	5.56×10^{-4}	1.46	5.71×10^{-7}	2.95	2.74×10^{-6}	2.97
512^2	5.33×10^{-6}	1.90	1.60×10^{-4}	1.80	7.13×10^{-8}	3.00	3.43×10^{-7}	3.00
FE								
64^2	9.10×10^{-4}		4.88×10^{-3}		3.51×10^{-5}		1.84×10^{-4}	
128^2	3.45×10^{-4}	1.40	1.98×10^{-3}	1.30	4.43×10^{-6}	2.98	2.29×10^{-5}	3.00
256^2	1.32×10^{-4}	1.38	7.78×10^{-4}	1.35	6.13×10^{-7}	2.86	3.54×10^{-6}	2.69
512^2	5.01×10^{-5}	1.40	3.45×10^{-4}	1.17	7.28×10^{-7}	3.07	4.40×10^{-7}	3.01
GS								
64^2	2.73×10^{-4}		4.15×10^{-3}		3.68×10^{-5}		1.84×10^{-4}	
128^2	7.44×10^{-5}	1.88	1.52×10^{-3}	1.44	4.38×10^{-6}	3.07	2.15×10^{-5}	3.09
256^2	1.93×10^{-5}	1.94	4.24×10^{-4}	1.85	5.77×10^{-7}	2.92	2.77×10^{-6}	2.96
512^2	4.90×10^{-6}	1.95	1.13×10^{-4}	1.90	7.13×10^{-8}	3.02	3.43×10^{-7}	3.02

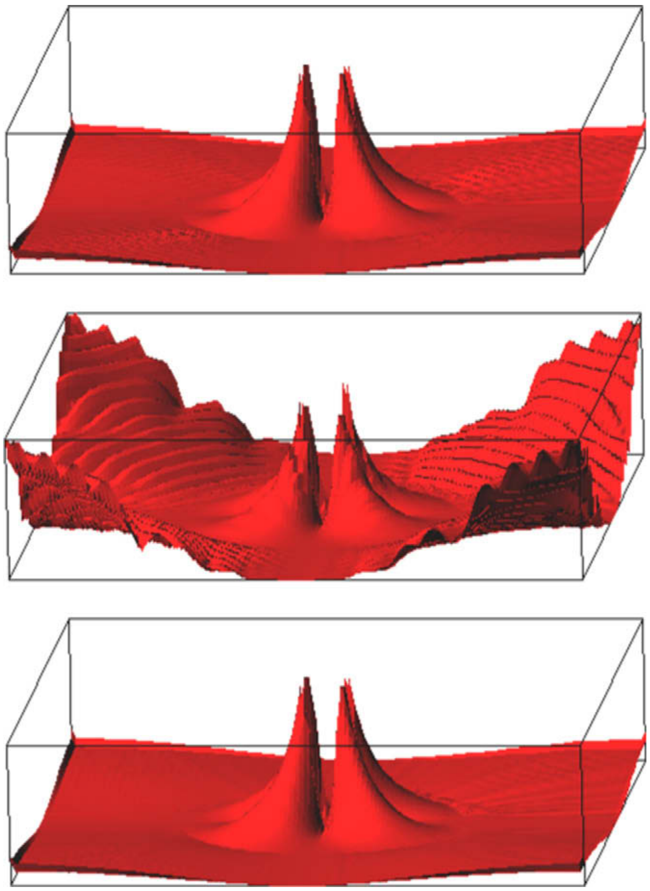

Fig. 1. Error distribution of RK2(top), FE(middle), and GS(bottom) methods in a 128^2 grid for the example of 2D smooth interface. For the visualization, the errors are normalized so that their maximums are all one. Observe that only the error distribution of FE method is oscillatory.

main RK2 and GS methods are second order accurate, however the accuracy of FE method drops to between one and two. Fig. 1 depicts the error distributions, and Fig. 2 compares the speed of error decays of the three methods.

4.2. 2D interface with kinks

Two circles of radius r are placed at $(\pm a, 0)$ on the plane. Let $0 < a < r$, so that the two circles intersect each other. The interface is taken to be the boundary of the union of the two circles. The signed distance function for the interface is

$$\phi(x,y) = \begin{cases} min\left(\sqrt{x^2 + \left(y \pm \sqrt{r^2 - a^2}\right)^2}\right) & \text{if } \frac{a-x}{\sqrt{(a-x)^2+y^2}} \geqslant \frac{a}{r} \text{and} \frac{x+a}{\sqrt{(x+a)^2+y^2}} \geqslant \frac{a}{r}, \\ min\left(\sqrt{(x \pm a)^2 + y^2} - r\right) & \text{else}. \end{cases}$$

We take $r = 1$ and $a = .7$, and the computational domain is $[-2, 2]^2$. The initial level function is defined as $\phi^0(x,y) = \phi(x,y) \cdot ((x-1)^2 + (y-1)^2 + .1)$. Fig. 3 illustrates the reinitialization that transforms the initial level function into the signed distance function. Fig. 4 compares the speed of error decays of the methods.

Table 2 shows that all the methods give nearly the same result near the interface. In the whole domain, RK2 and GS methods produce nearly the same result, which is more accurate than the result of FE method. Only RK2 and GS methods in the whole domain converge clearly with the first order rate, but the other convergence rates are fluctuating between one and three. The signed distance function has kinks on the whole y-axis and a line segment $[-a, a]$ on the x-axis. Two kink points $(0, \pm\sqrt{r^2 - a^2})$ corrupts the accuracy in the quadrilateral of vertices $(0, \pm\sqrt{r^2 - a^2})$ and $(\pm a, 0)$. Excluding the kinks and the region influenced by the kinks, the table shows the clear second order accuracy of RK2 and GS methods.

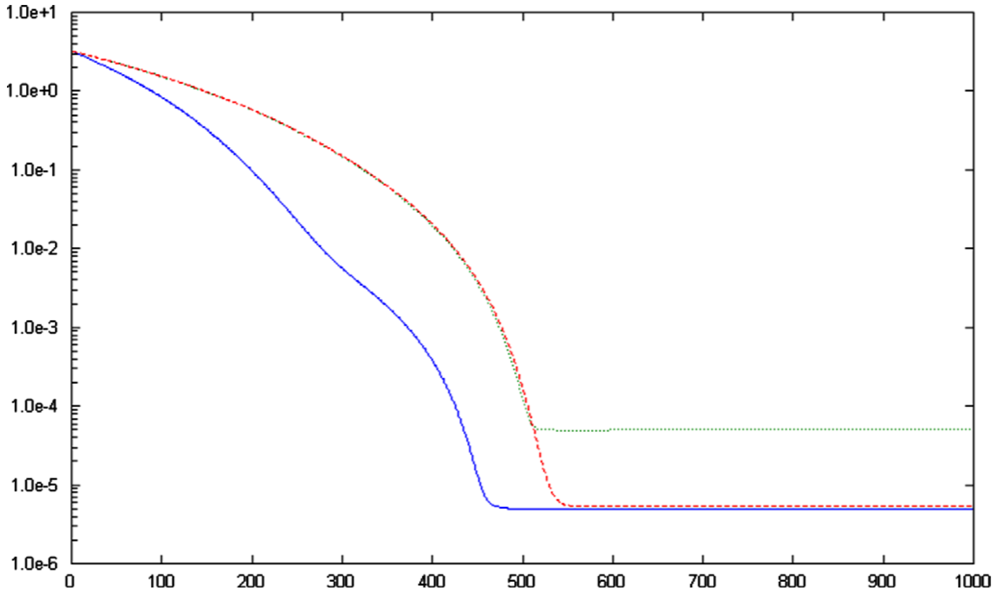

Fig. 2. Convergence of RK2(dashed), FE(dotted) and GS(solid) methods in a 512^2 grid for the example of 2D smooth interface. The curves are the graphs of their L^1 errors in the whole domain with respect to iteration number.

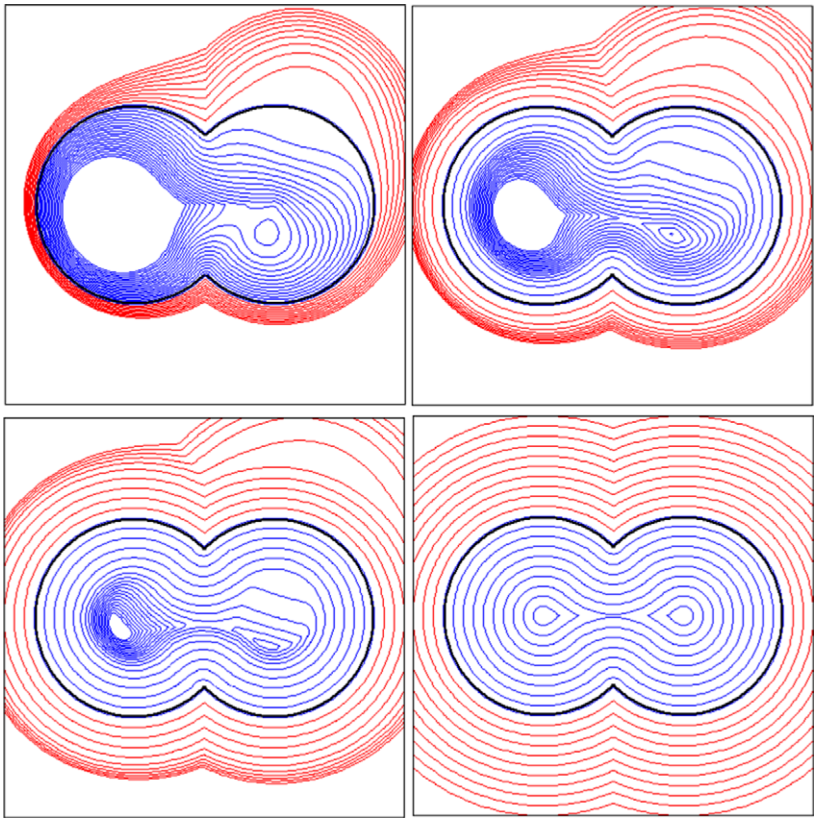

Fig. 3. Contours of the level function in the example of 2D interface with kinks. RK2 method was used to reinitialize the level function in a 128^2 grid. The figures are taken when the iterations are applied 0(top-left), 20(top-right), 40(bottom-left), and 80(bottom-right) times. Drawn are contours of levels from -2 to 1 with step size .1. The zero level set is drawn with thick solid line.

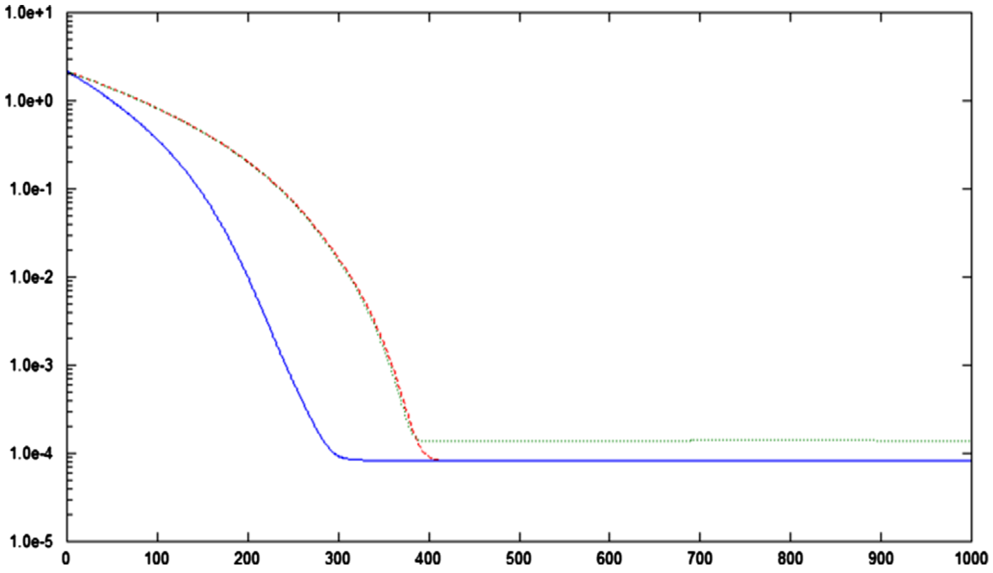

Fig. 4. Convergence of RK2(dashed), FE(dotted) and GS(solid) methods in a 512^2 grid for the example of 2D interface with kinks. The curves are the graphs of their L^1 errors in the whole domain with respect to iteration number.

Table 2
Accuracy of RK2, FE, and GS methods for the example of 2D interface with kinks. The error near the interface is measured at each node (i,j) with condition $|\phi_{ij}| < 1.2 \cdot \Delta x$, the error in the whole domain is measured at all the nodes, and the error on the region without kinks is with condition $|x_i| \geqslant .1$ and $\frac{|x_i|}{a+.2} + \frac{|y_j|}{\sqrt{r^2-a^2}+.2} \geqslant 1$.

	Error in the whole domain				Error near the interface			
Grid	L^1 error	Rate	L^∞ error	Rate	L^1 error	Rate	L^∞ error	Rate
RK2								
128^2	3.38×10^{-4}		6.10×10^{-3}		1.00×10^{-5}		6.44×10^{-4}	
256^2	1.63×10^{-4}	1.05	3.33×10^{-3}	0.87	2.74×10^{-6}	1.87	5.66×10^{-4}	0.19
512^2	8.26×10^{-5}	0.98	1.61×10^{-3}	1.05	1.01×10^{-6}	1.44	3.98×10^{-4}	0.51
1024^2	3.46×10^{-5}	1.26	7.19×10^{-4}	1.16	9.15×10^{-8}	3.46	9.72×10^{-5}	2.03
FE								
128^2	4.42×10^{-4}		6.85×10^{-3}		9.87×10^{-6}		6.44×10^{-4}	
256^2	2.33×10^{-4}	0.92	5.00×10^{-3}	0.45	2.75×10^{-6}	1.84	5.66×10^{-4}	0.19
512^2	1.39×10^{-4}	0.75	3.62×10^{-3}	0.47	1.01×10^{-6}	1.45	3.98×10^{-4}	0.51
1024^2	7.87×10^{-5}	0.82	2.24×10^{-3}	0.69	9.17×10^{-8}	3.46	9.72×10^{-5}	2.03
GS								
128^2	3.39×10^{-4}		6.10×10^{-3}		1.00×10^{-5}		6.44×10^{-4}	
256^2	1.62×10^{-4}	1.07	3.33×10^{-3}	0.87	2.74×10^{-6}	1.87	5.66×10^{-4}	0.19
512^2	8.19×10^{-5}	0.98	1.61×10^{-3}	1.05	1.01×10^{-6}	1.44	3.98×10^{-4}	0.51
1024^2	3.43×10^{-5}	1.26	7.17×10^{-4}	1.17	9.15×10^{-8}	3.46	9.72×10^{-5}	2.03

	Error in the region without kinks					
Iterations grid	RK2		FE		GS	
	L^∞ error	Rate	L^∞ error	Rate	L^∞ error	Rate
128^2	2.11×10^{-3}		2.32×10^{-3}		2.11×10^{-3}	
256^2	8.28×10^{-4}	1.35	1.10×10^{-3}	1.08	6.18×10^{-4}	1.77
512^2	2.40×10^{-4}	1.79	4.73×10^{-4}	1.22	1.68×10^{-4}	1.88
1024^2	6.53×10^{-5}	1.88	1.96×10^{-4}	1.27	4.44×10^{-5}	1.92

4.3. 3D smooth interface

We extend the example of 2D smooth interface to three-dimensions. The initial level function is defined in a computational domain $[-2,2]^3$ as

$$\phi^0(x,y,z) = ((x-1)^2 + (y-1)^2 + (z-1)^2 + 0.1)\left(\sqrt{x^2+y^2+z^2} - 1\right).$$

Table 3
Accuracy of RK2, FE, and GS methods for the example of 3D smooth interface. The error near the interface is measured at nodes (i,j,k) with condition $|\phi_{ijk}| < 1.2 \cdot \Delta x$, and the error in the whole domain is with condition $\phi_{ijk} > -.8$. Note that the kink point (0,0,0) is excluded with the latter condition.

	Error in the whole domain				Error near the interface			
Grid	L^1 error	Rate	L^∞ error	Rate	L^1 error	Rate	L^∞ error	Rate
RK2								
32^3	1.96×10^{-3}		2.00×10^{-2}		2.27×10^{-4}		9.32×10^{-4}	
64^3	4.74×10^{-4}	2.05	6.93×10^{-3}	1.53	3.06×10^{-5}	2.89	1.18×10^{-4}	2.98
128^3	1.17×10^{-4}	2.02	2.20×10^{-3}	1.66	4.04×10^{-6}	2.92	1.67×10^{-5}	2.82
256^3	2.91×10^{-5}	2.01	5.96×10^{-4}	1.88	5.15×10^{-7}	2.97	2.10×10^{-6}	2.99
FE								
32^3	3.22×10^{-3}		2.00×10^{-2}		2.45×10^{-4}		9.53×10^{-4}	
64^3	1.17×10^{-3}	1.46	6.92×10^{-3}	1.53	3.08×10^{-5}	2.99	1.30×10^{-4}	2.87
128^3	4.10×10^{-4}	1.52	2.20×10^{-3}	1.65	4.10×10^{-6}	2.91	2.36×10^{-5}	2.46
256^3	1.21×10^{-4}	1.76	8.07×10^{-4}	1.45	5.26×10^{-7}	2.96	2.94×10^{-6}	3.00
GS								
32^3	1.91×10^{-3}		2.00×10^{-2}		2.19×10^{-4}		1.02×10^{-3}	
64^3	4.67×10^{-4}	2.04	6.93×10^{-3}	1.53	3.00×10^{-5}	2.87	1.25×10^{-4}	3.03
128^3	1.15×10^{-4}	2.01	2.19×10^{-3}	1.67	3.97×10^{-6}	2.92	1.73×10^{-5}	2.86
256^3	2.87×10^{-5}	2.01	5.95×10^{-4}	1.88	5.10×10^{-7}	2.96	2.17×10^{-6}	3.00

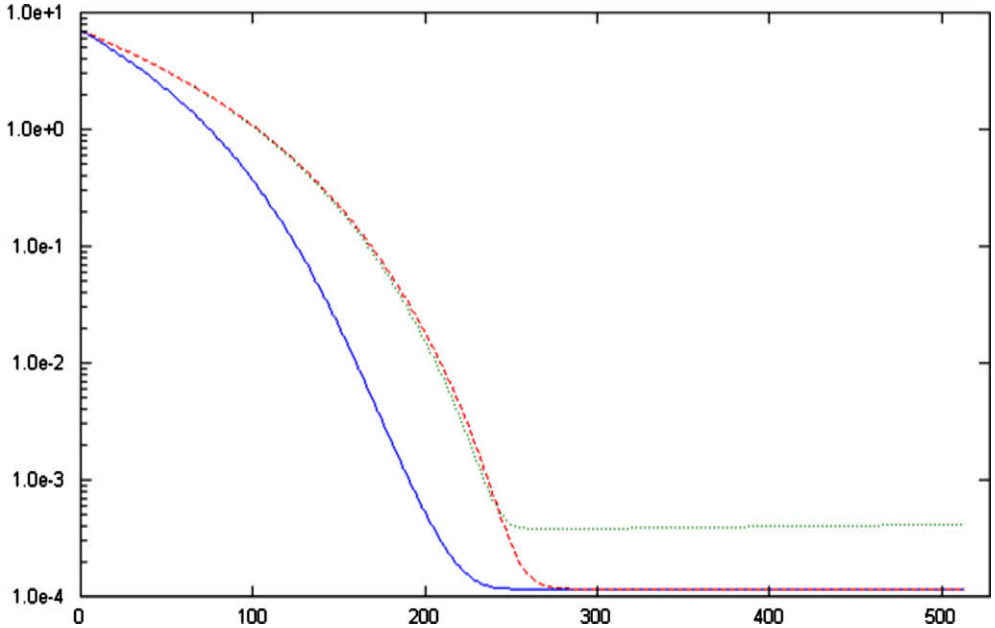

Fig. 5. Convergence of RK2(dashed), FE(dotted) and GS(solid) methods in a 128^3 grid for the example of 3D smooth interface. The curves are the graphs of their L^1 errors in the whole domain with respect to iteration number.

Table 3 shows that all the methods are third order accurate near the interface. In the whole domain RK2 and GS methods are second order accurate, however the accuracy of FE method drops to between one and two. Fig. 5 compares the error decays of the three methods.

5. Conclusion

We have considered the three temporal discretizations : the second order Runge–Kutta method (RK2), the forward Euler method (FE), and the Gauss–Seidel update of the forward Euler method (GS). Each of the temporal discretizations was combined with the second order ENO finite difference in [11,9]. We tried various examples to verify the two hypotheses in the introduction: one is that all the temporal discretizations give the same result eventually, and the other one is that FE method invokes numerical instability.

All the results indicate the following two facts. The three methods are all third order accurate near the interface, when measured in smooth region. RK2 and GS methods are second order accurate in the whole domain, but the accuracy of FE method drops to between one and two.

FE method did not invoke any numerical instability, however its oscillatory nature can be seen in the magnified view of Fig. 1. The absolute stability region of FE method does not include all the eigenvalues of the linearized system of the second order ENO finite difference. When FE method in the combination of the ENO is applied to convection problems, it would invoke numerical instability typically in the form of oscillation, but the effect was very weak in its application to reinitialization; the oscillation can be observed only when the level functions are magnified about a thousand times. Though small, the oscillation should hinder the solution of FE method from reaching a stationary state. This explains why FE and RK2 methods produce different results.

The Gauss–Seidel method for linear system is well-known to play a role of relaxation, or smoothing, if the linear system is diagonally dominant [5]. The system discretized by the ENO finite difference with subcell resolution is non-linear, and its stability analysis is far beyond the scope of this article, but it seems almost certain that the Gauss–Seidel method also plays the role of relaxation in the non-linear system. One empirical evidence is that there is no oscillation in the error of GS method in Fig. 1. The relaxation kills the oscillation of FE method, and the solution of GS method reaches its stationary state. This explains why GS and RK2 methods produced almost identical results in all the tried examples.

Examining these facts, we conclude that GS method is the best among the three temporal discretizations. Compared to RK2 method, it is twice faster and requires memory two times less with the same accuracy.

Acknowledgments

This research was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2008-331-C00045).

References

[1] M. Bardi, S. Osher, The nonconvex multidimensional riemann problem for Hamilton–Jacobi equations, SIAM J. Math. Anal. 22 (1991) 344–351.
[2] L.-T. Cheng, Y.-H. Tsai, Redistancing by flow of time dependent Eikonal equation, J. Comput. Phys. 227 (2008) 4002–4017.
[3] M.G. Crandall, P.-L. Lions, Two approximations of solutions of Hamilton–Jacobi equations, Math. Comput. 43 (1984) 1–19.
[4] A. du Chéné, C. Min, F. Gibou, Second-order accurate computation of curvatures in a level set framework, J. Sci. Comput. 35 (2008) 114–131.
[5] G. Golub, C. Loan, Matrix Computations, The John Hopkins University Press, 1989.
[6] A. Harten, Eno schemes with subcell resolution, J. Comput. Phys. 83 (1989) 148–184.
[7] A. Harten, B. Enquist, S. Osher, S. Chakravarthy, Uniformly high-order accurate essentially non-oscillatory schemes III, J. Comput. Phys. 71 (1987) 231–303.
[8] C. Kao, S. Osher, J. Qian, Lax–Friedrichs sweeping scheme for static Hamilton–Jacobi equations, J. Comput. Phys. 196 (2004) 367–391.
[9] C. Min, F. Gibou, A second order accurate level set method on non-graded adaptive Cartesian grids, J. Comput. Phys. 225 (2007) 300–321.
[10] S. Osher, J. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.
[11] G. Russo, P. Smereka, A remark on computing distance functions, J. Comput. Phys. 163 (2000) 51–67.
[12] J. Sethian, A fast marching level set method for monotonically advancing fronts, Proc. Natl. Acad. Sci. 93 (1996) 1591–1595.
[13] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schemes, J. Comput. Phys. 77 (1988) 439–471.
[14] M. Sussman, E. Fatemi, An efficient interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow, SIAM J. Sci. Comput. 20 (1999) 1165–1191.
[15] M. Sussman, E. Fatemi, P. Smereka, S. Osher, An improved level set method for incompressible two-phase flows, Comput. Fluids 27 (1998) 663–680.
[16] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[17] Y. Tsai, L. Cheng, S. Osher, H. Zhao, Fast sweeping algorithms for a class of Hamilton–Jacobi equations, SIAM J. Numer. Anal. 41 (2003) 673–694.