Journal of Computational Physics 228 (2009) 4273–4290

Contents lists available at ScienceDirect

Journal of Computational Physics

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

High order numerical methods to two dimensional delta function integrals in level set methods ☆

Xin Wen

LSEC, Institute of Computational Mathematics, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, P.O. Box 2719, Beijing 100190, China

ARTICLE INFO

Article history:
Received 4 May 2008
Received in revised form 9 January 2009
Accepted 8 March 2009
Available online 17 March 2009

AMS subject classification:
65Jxx

Keywords:
Delta function integral
High order numerical method
Error estimate
Level set method

ABSTRACT

In this paper we design and analyze a class of high order numerical methods to delta function integrals appearing in level set methods in two dimensional case. The methods comprise approximating the mesh cell restrictions of the delta function integral. In each mesh cell the two dimensional delta function integral can be rewritten as a one dimensional ordinary integral with the smooth integrand being a one dimensional delta function integral, and thus is approximated by applying standard one dimensional high order numerical quadratures and high order numerical methods to one dimensional delta function integrals proposed in [X. Wen, High order numerical methods to a type of delta function integrals, J. Comput. Phys. 226 (2007) 1952–1967]. We establish error estimates for the method which show that the method can achieve any desired accuracy by assigning the corresponding accuracy to the sub-algorithms and has better accuracy under an assumption on the zero level set of the level set function which holds generally. Numerical examples are presented showing that the second to fourth order methods implemented in this paper achieve or exceed the expected accuracy and demonstrating the advantage of using our high order numerical methods.

1. Introduction

We study in this paper a class of high order numerical methods to the following type of delta function integrals

$$\int_{\mathbb{R}^n} f(\mathbf{x})\|\nabla u(\mathbf{x})\|\delta(u(\mathbf{x}))d\mathbf{x}, \tag{1.1}$$

where $f(\mathbf{x})$ is a weight function, $u(\mathbf{x})$ is a level set function whose zero points define a manifold Γ of codimension one. In this paper we consider the two dimensional case $n = 2$. In this case Γ is a one dimensional curve in the two dimensional space. The functions $f(\mathbf{x}), u(\mathbf{x})$ are assumed to have sufficient smoothness and their values are only provided at grid points of a regular mesh. Numerical evaluations of delta function integrals (1.1) in two and three dimensions in the above context appear in many applications of level set methods, see for example [2,11,26].

One approach widely studied in the literature to approximate (1.1) is the numerical quadrature approach. Assume the values of $f(\mathbf{x}), u(\mathbf{x})$ are given at grid points of the following uniform mesh on $\mathbb{R}^2$

☆ Research supported in part by the Knowledge Innovation Project of the Chinese Academy of Sciences Grants K5501312S1, K5502212F1, K7290312G7 and K7502712F7, and NSFC Grant 10601062.

E-mail address: wenxin@amss.ac.cn

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.03.004

$$\{\mathbf{x_j}\}_{\mathbf{j}\in\mathbb{Z}^2}, \quad \mathbf{x_j} = \left(x_{j_1}^{(1)}, x_{j_2}^{(2)}\right),$$
$$x_{j_k}^{(k)} = x_0^{(k)} + j_k h, \quad j_k \in \mathbb{Z}, k = 1, 2. \tag{1.2}$$

Then the numerical quadrature methods to delta function integrals (1.1) in two dimension can be generally written in the form

$$h^2 \sum_{\mathbf{j}\in\mathbb{Z}^2} f(\mathbf{x_j}) \|\nabla_D u(\mathbf{x_j})\| \tilde{\delta}(\mathbf{x_j}; u), \tag{1.3}$$

where $\nabla_D u(\mathbf{x_j})$ is the difference approximation to $\nabla u(\mathbf{x_j})$, $\tilde{\delta}(\mathbf{x_j}; u)$ is an approximate delta function which can be defined by grid point values of u. Utilizing approximate delta functions is related to the regularization technique of the Dirac delta function which have various applications [4–7,10,12–14,16–18,21,22]. The efficiency of the numerical quadratures (1.3) depends on the choice of appropriate approximate delta functions. In the work [18] Tornberg and Engquist showed that a common technique to construct the approximate delta function in (1.3) suffers from $O(1)$ errors. After that different approximate delta functions have been designed in the literature which yield first or second order numerical quadratures (1.3). In [3] Engquist et al. proposed a first order accurate approximate delta function based on one dimensional discrete delta functions and a variable support size formula. They also developed in [3] a second order accurate approximate delta function based on approximations to product formula for multidimensional delta functions which is more complex to apply. The product formula method following Peskin [12,13] has the advantage that it can achieve any desired accuracy by using one dimensional discrete delta functions with corresponding discrete moment conditions, as proved in [18]. However the high order version of the product formula method has not been implemented when the curve Γ is represented by a level set function. In [15] Smereka proposed both a first and second order accurate approximate delta function by using a technique for solving elliptic equations with discontinuous source terms. The proof of accuracy of this approximate delta function is presented in [1]. In [19] Towers proposed both a first and second order accurate approximate delta function by using difference approximations to derivatives of the smoothed heaviside function or those of the integral of the heaviside function. The analysis of accuracy of these methods are considered in [20]. However approximate delta functions higher than second order accuracy for the numerical quadratures (1.3) have not been designed and implemented yet. Therefore to design numerical quadratures (1.3) higher than second order accuracy to the delta function integrals(1.1) remains to be studied.

In this paper we design and analyze a class of high order numerical methods to the delta function integrals (1.1) in two dimension. The strategy of the methods in this paper is different from the numerical quadrature approach. These methods are constructed by considering the approximation of the restriction of the two dimensional delta function integral in each mesh cell. This is also a natural strategy. By using this strategy, Min and Gibou designed in [8,9] a second order geometric integration method for computing (1.1) via decomposing the zero level set Γ into simplices. Such a strategy has also been applied to another type of delta function integrals

$$\int_{\mathbb{R}^n} \alpha(\mathbf{x}) \prod_{i=1}^{n} \delta(\beta_i(\mathbf{x})) d\mathbf{x}, \quad n = 1, 2, 3, \tag{1.4}$$

where the common zero points of the level set functions $\beta_i(\mathbf{x})$ are essentially finite number of points in the space. Second to fourth order numerical methods to (1.4) in one to three dimensions have been designed in [23]. In order to obtain suitable approximation to the restriction of the delta function integral (1.4) in a mesh cell, the methods in [23] involve checking the existence of common zero points of level set functions and applying the technique of changing interpolation space.

Naturally the methods in this paper also involve checking the existence of zero points of the level set function in the delta function integral (1.1). Namely we need to check the intersection between a mesh cell and the zero level set Γ of the level set function. Our strategy to approximate the restriction of the delta function integral (1.1) in a mesh cell intersecting with Γ is based on the fact that the two dimensional delta function integral in the mesh cell can be rewritten as a one dimensional ordinary integral with the smooth integrand being a one dimensional delta function integral. The transformed one dimensional integral takes one of two forms according to the comparison of the two components of gradient of u in the cell which can be checked from the mesh point values of u. Therefore high order numerical methods to approximate the mesh cell restriction of the two dimensional delta function integral (1.1) in principle can be constructed by applying high order numerical quadratures to one dimensional ordinary integrals and high order numerical methods to one dimensional delta function integrals. The high order numerical quadratures to ordinary integrals are standard. The high order numerical methods to one dimensional delta function integrals have already been studied [23,24]. In this paper we apply the high order numerical method designed in [23] to a type of delta function integrals including the one dimensional case. The algorithm so designed to approximate the mesh cell restrictions of the two dimensional delta function integral (1.1) comprises the numerical method proposed in this paper. The method contains several sub-algorithms including Newton iteration to solve one dimensional interpolation polynomials, Numerical quadrature to one dimensional ordinary integrals, difference approximation formula, interpolation formula and numerical method to approximate one dimensional delta function integrals. We carry out error analysis for the method proposed in this paper and prove that the method can achieve any desired accuracy to the two dimensional delta function integrals (1.1) provided the sub-algorithms in the method attain the corresponding accuracy which is straightforward to fulfill. We also prove the better accuracy of the method in this paper under an assumption on

the zero level set Γ of the level set function u. The assumption is satisfied commonly by a smooth closed curve Γ. The key point of the error estimates is that the approximation of the two dimensional delta function integral (1.1) in any given mesh cell may not be accurate enough, while for such a mesh cell there exist several neighboring cells so that the approximate two dimensional delta function integral restricted to the union of these mesh cells has sufficient accuracy. We implement second to fourth order numerical methods in this paper and the numerical examples show that these methods achieve or exceed the expected accuracy indicated by the error analysis. We also present numerical test to demonstrate the advantage of our high order numerical methods with regard to computational complexity. In this paper we consider the two dimensional case of the delta function integrals (1.1). Recently we have also extended the idea in this paper to design high order numerical methods to the delta function integrals (1.1) in three dimensional case [25].

This paper is organized as follows. In Section 2 we discuss the main strategy, numerical implementation and algorithm description of the method proposed in this paper. In Section 3 we establish the error estimates for our method which show that the method can achieve any desired accuracy by choosing corresponding accuracy in the sub-algorithms in the method and has better accuracy under an assumption on the zero level set of the level set function. In Section 4 we present numerical examples in which second to fourth order methods are shown to achieve or exceed the expected accuracy and efficiency of high order numerical methods is demonstrated. We conclude the paper in Section 5.

2. High order numerical methods

2.1. Main strategy

Consider the delta function integral (1.1) in two dimensional case

$$\int_{\mathbb{R}^2} f(x,y)\sqrt{(u_x)^2+(u_y)^2}\,\delta(u(x,y))\,dx\,dy. \tag{2.1}$$

Assume $\mathbb{R}^2$ is covered by a uniform mesh $(x_i,y_j),(i,j)\in\mathbb{Z}^2$ with the mesh size h. Denote

$$I_{i,j}=\int_{y_j}^{y_{j+1}}\int_{x_i}^{x_{i+1}} f(x,y)\sqrt{(u_x)^2+(u_y)^2}\,\delta(u(x,y))\,dx\,dy. \tag{2.2}$$

Then

$$\int_{\mathbb{R}^2} f(x,y)\sqrt{(u_x)^2+(u_y)^2}\,\delta(u(x,y))\,dx\,dy=\sum_{(i,j)\in\mathbb{Z}^2} I_{i,j}. \tag{2.3}$$

Therefore the delta function integral (2.1) can be computed by approximating each $I_{i,j}$. Let $\widehat{I}_{i,j}$ be the approximation to $I_{i,j}$. Then our goal is to provide algorithm to yield $\widehat{I}_{i,j}$.

If a cell $C_{i,j}=[x_i,x_{i+1}]\times[y_j,y_{j+1}]$ is away from the zero level set, then we can just set $\widehat{I}_{i,j}=0$. Therefore we only need to consider the computation of $I_{i,j}$ in the case that the cell $C_{i,j}$ nontrivially intersects with the zero level set. We will discuss how to check the intersection between a cell and the zero level set in the next subsection. The computation in the nontrivial cases relies on the observation that the two dimensional delta function integral (2.2) can be formally rewritten as the following two forms

$$I_{i,j}=\int_{x_i}^{x_{i+1}}\left(\int_{y_j}^{y_{j+1}} f(x,y)\sqrt{(u_x)^2+(u_y)^2}\,\delta(u(x,y))\,dy\right)dx, \tag{2.4}$$

$$I_{i,j}=\int_{y_j}^{y_{j+1}}\left(\int_{x_i}^{x_{i+1}} f(x,y)\sqrt{(u_x)^2+(u_y)^2}\,\delta(u(x,y))\,dx\right)dy. \tag{2.5}$$

Namely the two dimensional delta function integral $I_{i,j}$ can be regarded as a one dimensional ordinary integral with the integrand being a one dimensional delta function integral. Here ordinary integral means the integrand is bounded rather than delta function. In fact for the nontrivial cases that the cell $C_{i,j}$ intersecting with the zero level set, at least one of the above two forms is valid. This will be discussed in detail in the next subsection.

Based on this observation, our idea to approximate $I_{i,j}$ is to transform the two dimensional delta function integral into the one dimensional ordinary integral illustrated by (2.4) and (2.5) with the integrand being the one dimensional delta function integral. Therefore high order numerical method for computing $I_{i,j}$ can be expected by applying high order numerical quadratures to one dimensional ordinary integral which are standard, and high order numerical methods to one dimensional delta function integral which have already been developed [23,24].

For the convenience of description, we discuss the case that the form (2.4) is valid. Denote

$$F(x)=\int_{y_j}^{y_{j+1}} f(x,y)\sqrt{(u_x)^2+(u_y)^2}\,\delta(u(x,y))\,dy,\quad x_i\leqslant x\leqslant x_{i+1}. \tag{2.6}$$

Then (2.4) is written to be

$$I_{i,j} = \int_{x_i}^{x_{i+1}} F(x)dx. \tag{2.7}$$

In order to get high order numerical results by applying numerical quadratures to the one dimensional integral (2.7), the function $F(x)$ needs to have sufficient smoothness. However, this can be untrue. For example we consider the situation shown in the left part of Fig. 2.1, where the zero level set Γ cuts through the cell $C_{i,j}$ and intersects with one horizontal side and one vertical side. Let $(\tilde{x}, y_{j+1}), x_i < \tilde{x} < x_{i+1}$ be the point on Γ. Then it is seen that in this situation $F(x)$ is zero for $x \in [x_i, \tilde{x})$ and is nonzero for $x \in (\tilde{x}, x_{i+1}]$. Namely $F(x)$ is discontinuous at $x = \tilde{x}$. Moreover, it is even possible that $F(x)$ has two discontinuity points on the interval $[x_i, x_{i+1}]$. We consider the case shown in the right part of Fig. 2.1, where the zero level set Γ is nearly parallel to x-axis near the cell $C_{i,j}$ and intersects with one horizontal side and two vertical sides. Let $(x', y_{j+1}), (x'', y_{j+1}), x_i < x' < x'' < x_{i+1}$ be the points on Γ. It is seen that $F(x)$ is zero for $x \in (x', x'')$ and is nonzero for $x \in [x_i, x') \cup (x'', x_{i+1}]$. In this case $F(x)$ is discontinuous at $x = x', x''$.

Since $F(x)$ can be discontinuous on the interval $[x_i, x_{i+1}]$ with one or more discontinuity points, it is improper to apply numerical quadrature to (2.7) in order to obtain high order numerical results. To deal with this issue, we can transform (2.7) into the one dimensional integral with smooth integrand as follows. Since we are discussing that the form (2.4) is valid, one can see from the next subsection that this implies u_y is $O(1)$ away from zero near the cell $C_{i,j}$. Thus we can introduce the definitions

$$Y_{i,j}(x) \text{ satisfying that } (x, Y_{i,j}(x)), x \in [x_i, x_{i+1}] \text{ are points on } \Gamma, \tag{2.8}$$

$$Y_{i,j}^m = \min_{x \in [x_i, x_{i+1}]} Y_{i,j}(x), \quad Y_{i,j}^M = \max_{x \in [x_i, x_{i+1}]} Y_{i,j}(x), \tag{2.9}$$

$$F_m(x) = \int_{Y_{i,j}^m - h}^{Y_{i,j}^M + h} f(x,y)\sqrt{(u_x)^2 + (u_y)^2}\,\delta(u(x,y))dy, \quad x_i \leqslant x \leqslant x_{i+1}, \tag{2.10}$$

$$\Lambda_{i,j} = \{x \in [x_i, x_{i+1}] | Y_{i,j}(x) \in [y_j, y_{j+1}]\}. \tag{2.11}$$

Notice that $F_m(x)$ is smooth on $[x_i, x_{i+1}]$ and thus on $\Lambda_{i,j}$. We have the following equality

$$I_{i,j} = \int_{x_i}^{x_{i+1}} F(x)dx = \int_{\Lambda_{i,j}} F(x)dx = \int_{\Lambda_{i,j}} F_m(x)dx. \tag{2.12}$$

Namely by restricting the integral domain, we can represent $I_{i,j}$ to be the one dimensional integral with the smooth integrand $F_m(x)$. In fact since the two dimensional delta function integral $I_{i,j}$ represents curve integral, (2.12) implies we transform the curve integral into the equivalent form of the integral in x-variable. Thus our strategy to approximate $I_{i,j}$ in the case of the form (2.4) being valid is to apply high order numerical quadratures to the last form in (2.12).

One additional issue needs to be treated by using the last form of (2.12) instead of (2.7) is to determine the integral domain $\Lambda_{i,j}$. We will use an interval $[x_{i,j}^L, x_{i,j}^R]$ contained in $[x_i, x_{i+1}]$ to approximate $\Lambda_{i,j}$. The definition of the interval boundary points will be discussed in detail in the next subsection. Thus our strategy is to set $\widehat{I}_{i,j}$ to be high order numerical quadrature to the integral $\int_{x_{i,j}^L}^{x_{i,j}^R} F_m(x)dx$. We mention that it is possible that a single $\widehat{I}_{i,j}$ yielded by our algorithm is not accurate enough to approximate $I_{i,j}$. For the case shown in the left part of Fig. 2.1, our algorithm gives $[x_{i,j}^L, x_{i,j}^R] = [\tilde{x}, x_{i+1}]$ which is exactly the domain $\Lambda_{i,j}$. So in this case $\widehat{I}_{i,j}$ can be the high order approximation to $I_{i,j}$. However for the case shown in the right part of Fig. 2.1, our algorithm gives $[x_{i,j}^L, x_{i,j}^R] = [x_i, x_{i+1}]$ while in this case $\Lambda_{i,j} = [x_i, x'] \cup [x'', x_{i+1}]$. In this case $\int_{x_{i,j}^L}^{x_{i,j}^R} F_m(x)dx$ includes the integral on the small curve segment outside the cell $C_{i,j}$. Thus $\widehat{I}_{i,j}$ cannot be the high order approximation to $I_{i,j}$. Instead it can be the high order approximation to $I_{i,j} + I_{i,j+1}$. From the algorithm description in the next subsections we will see that in this case our algorithm yields $\widehat{I}_{i,j+1} = 0$. Therefore although $\widehat{I}_{i,j}$ and $\widehat{I}_{i,j+1}$ are not accurate approximation to $I_{i,j}, I_{i,j+1}$ respectively in this case,

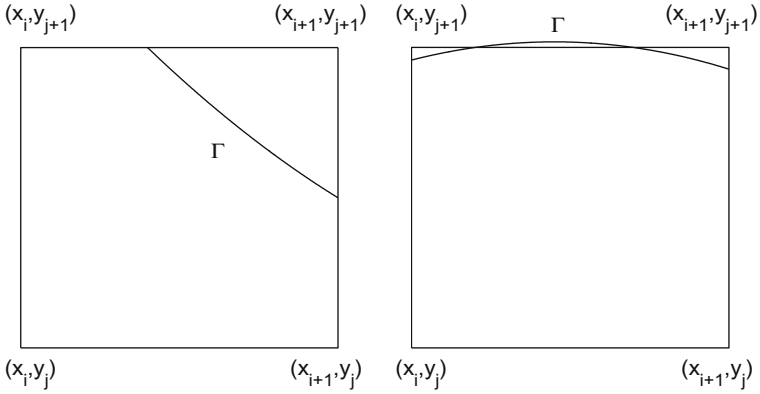

Fig. 2.1. Situations in which $F(x)$ is nonsmooth on $[x_i, x_{i+1}]$.

their summation can be high order accurate. By this reason the summation $\sum_{(i,j)\in\mathbb{Z}^2}\widehat{I}_{i,j}$ still can be high order approximation to (2.3). This is also the key idea used in the proof of the error estimates for our method in Section 3. Namely although the approximation to the two dimensional delta function integral in a single cell may not be accurate enough, for such a cell there exist certain neighboring cells so that the approximation to the delta function integral in the union of these cells has high order accuracy.

The computations of the interval boundary points $x^L_{i,j}, x^R_{i,j}$ will be discussed in the next subsection. Assume their computed values to be $\widehat{x}^L_{i,j}, \widehat{x}^R_{i,j}$. Then the numerical quadrature to the integral $\int_{\widehat{x}^L_{i,j}}^{\widehat{x}^R_{i,j}} F_m(x)dx$ which yields the value of $\widehat{I}_{i,j}$ is written to be

$$(\hat{x}^R_{i,j}-\hat{x}^L_{i,j})\sum_{k=1}^{K} w_k F_m(\hat{x}^L_{i,j}+n_k(\hat{x}^R_{i,j}-\hat{x}^L_{i,j})), \tag{2.13}$$

where w_k and $n_k\in[0,1]$ are weights and nodal points of the quadrature rule respectively. For example the fourth order Simpson rule satisfies $K=3$ and $w_1=w_3=\frac{1}{6}, w_2=\frac{4}{6}, n_1=0, n_2=\frac{1}{2}, n_3=1$.

Therefore for computing $\widehat{I}_{i,j}$ we need to compute the values of $F_m(x)$ at several quadrature points which belong to $[x_i, x_{i+1}]$. According to the definition (2.10), the computation of $F_m(x)$ at each quadrature point corresponds to numerical approximation to a one dimensional delta function integral. From the definition (2.10) one sees that the delta function in the integrand is concentrated at $y=Y_{i,j}(x)$ which is the y-component of the zero point of u. The exact value of $F_m(x)$ is related to the value of the integrand in (2.10) at this zero point. Since this zero point may not be a grid point and we only have the grid point values of f and u, we can use interpolation together with difference approximation to approximate the value of $F_m(x)$. We will discuss the computation of $F_m(x)$ in the next subsection in detail.

In summary our procedure to yield $\widehat{I}_{i,j}$ is as follows. We only need to treat the nontrivial case that the cell $C_{i,j}$ intersects with the zero level set, otherwise we set $\widehat{I}_{i,j}=0$. In the nontrivial cases, we check which one of the two forms (2.4) and (2.5) is valid. In the case that the form (2.4) is valid, we firstly obtain the computed values $\hat{x}^L_{i,j}, \hat{x}^R_{i,j}$ of the boundary points of the interval $[x^L_{i,j}, x^R_{i,j}]$ which approximates the integral domain $\Lambda_{i,j}$ in (2.12). Then we set $\widehat{I}_{i,j}$ to be the numerical quadrature to $\int_{\widehat{x}^L_{i,j}}^{\widehat{x}^R_{i,j}} F_m(x)dx$ given by (2.13), with the evaluation of $F_m(x)$ at nodal points corresponding to numerical approximations to one dimensional delta function integrals. Similar strategy can be adopted in the case that the form (2.5) is valid.

2.2. Numerical implementation

After introducing the main strategy of our methods, we now discuss the issue of numerical implementation.

Firstly we discuss the check of the intersection between a cell $C_{i,j}$ and the zero level set. Denote $u_{k,l}=u(x_k,y_l), \forall(k,l)\in\mathbb{Z}^2$. According to the signs of four vertex values $u_{i,j}, u_{ij+1}, u_{i+1j}, u_{i+1j+1}$ in the cell, there are following four possible cases for fine enough mesh

Case CL.

(1) $u_{i,j}, u_{ij+1}, u_{i+1j}, u_{i+1j+1}$ have the same sign.
(2) One of $u_{i,j}, u_{ij+1}, u_{i+1j}, u_{i+1j+1}$ is zero, the other three have the same sign.
(3) Two of $u_{i,j}, u_{ij+1}, u_{i+1j}, u_{i+1j+1}$ are zero, the other two have the same sign.
(4) At least two of $u_{i,j}, u_{ij+1}, u_{i+1j}, u_{i+1j+1}$ have opposite signs.

The case that three or four of $u_{i,j}, u_{ij+1}, u_{i+1j}, u_{i+1j+1}$ are zero is impossible for fine enough mesh and a smooth zero level set Γ. Since the norm of the gradient of the level set function is bounded away from zero near the zero level set. This will also be used to determine the validity of the forms (2.4) and (2.5) in the following.

Cases (1) and (2) correspond to that the curve Γ has no or slight intersection with the cell $C_{i,j}$, therefore in these cases naturally we set $\widehat{I}_{i,j}=0$.

Case (4) represents the situation that the curve Γ nontrivially cuts through the cell $C_{i,j}$ and in this situation we use the strategy introduced in the last subsection to calculate $\widehat{I}_{i,j}$.

Case (3) is a special case. Without loss of generality we consider the case that $u_{i,j}=u_{i+1,j}=0$ and $u_{ij+1}, u_{i+1,j+1}$ have the same sign. In this case Γ is nearly parallel to x-axis and $|u_x|<|u_y|$ near this cell for reasonably fine mesh. From the definition of $x^L_{i,j}, x^R_{i,j}$ which will be given in the following we obtain $x^L_{i,j}=x_i, x^R_{i,j}=x_{i+1}$ in this case. If we apply the strategy in the last subsection then the resulting $\widehat{I}_{i,j}$ is the approximation to $\int_{x_i}^{x_{i+1}} F_m(x)dx$ which is $I_{i,j}+I_{ij-1}$ in this case. With the same procedure applying to the cell C_{ij-1} one sees that the resulting $\widehat{I}_{ij-1}$ is the same as $\widehat{I}_{i,j}$. Thus $\widehat{I}_{i,j}+\widehat{I}_{ij-1}$ is the approximation to $2(I_{i,j}+I_{ij-1})$. However we need $\widehat{I}_{i,j}+\widehat{I}_{ij-1}$ to be the approximation to $I_{i,j}+I_{ij-1}$. Therefore it is seen that case (3) is a special case. In this special case, we need to set $\widehat{I}_{i,j}$ to be half of the value yielded by the strategy introduced in the last subsection. By adopting this strategy, in the above case the sum of resulting $\widehat{I}_{i,j}$ and $\widehat{I}_{ij-1}$ is the approximation to $I_{i,j}+I_{ij-1}$, which is the expected result.

Next we discuss the validity of the two forms (2.4) and (2.5) of $I_{i,j}$. This means the validity of the one dimensional delta function integrals in the two forms. Clearly the form (2.4) is valid provided $u_y(x,y)$ at zero points of $u(x,y)$ near the cell $C_{i,j}$ are nonzero. Similarly the validity of the form (2.5) depends on the nonzero of $u_x(x,y)$ at zero points of $u(x,y)$ near the cell $C_{i,j}$.

Since $\sqrt{(u_x)^2+(u_y)^2}$ is the normal derivative of $u(x,y)$ at its zero level set, which should have $O(1)$ positive lower bound at the zero level set for a well-defined level set function u. Therefore at the zero points of $u(x,y)$ at least one of u_x and u_y should be $O(1)$ away from zero. This implies at least one of the two forms (2.4) and (2.5) is valid. To check which of (2.4) and (2.5) is valid, we can compare u_x and u_y at the center point $\left(\frac{x_i+x_{i+1}}{2},\frac{y_j+y_{j+1}}{2}\right)$ of the cell $C_{i,j}$. If at this point $|u_x| \geqslant |u_y|$, then the form (2.5) is valid. Otherwise the form (2.4) is valid. In practical computation, since we only have the grid point values of u, we can compare the difference approximations to $|u_x|$ and $|u_y|$ at center point of the cell. Namely we compare the quantities

$$\tilde{u}^x_{i,j} = |u_{i+1,j+1} + u_{i+1,j} - u_{i,j+1} - u_{i,j}|, \tag{2.14}$$

$$\tilde{u}^y_{i,j} = |u_{i+1,j+1} + u_{i,j+1} - u_{i+1,j} - u_{i,j}|. \tag{2.15}$$

If $\tilde{u}^x_{i,j} \geqslant \tilde{u}^y_{i,j}$, then the form (2.5) is valid. Otherwise the form (2.4) is valid.

In the following we discuss the case that the form (2.4) is checked to be valid in consistent with the discussion in the last subsection. The case that the form (2.5) is valid can be similarly treated.

We then discuss the determination of the interval $[x^L_{i,j}, x^R_{i,j}]$ to approximate the integral domain $\Lambda_{i,j}$ in (2.12). $\Lambda_{i,j}$ is the projection to x-axis of the zero level set Γ restricted to the cell $C_{i,j}$. Define

$$\Upsilon_{i,j} = \{(x,y)|(x,y) \text{ being the intersection point between the zero level set } \Gamma \text{ and the edges of the cell } C_{i,j}\}. \tag{2.16}$$

The elements of $\Upsilon_{i,j}$ are at the boundary of the zero level set Γ restricted to the cell $C_{i,j}$. Thus the x-component of the elements of $\Upsilon_{i,j}$ are at the boundary of the set $\Lambda_{i,j}$. Therefore we can construct the interval $[x^L_{i,j}, x^R_{i,j}]$ by defining

$$x^L_{i,j} = \min_{(x,y)\in\Upsilon_{i,j}} x, \quad x^R_{i,j} = \max_{(x,y)\in\Upsilon_{i,j}} x. \tag{2.17}$$

From the above definition, we have the following strategy to obtain $x^L_{i,j}, x^R_{i,j}$.

Strategy 1. Give $x^L_{i,j}, x^R_{i,j}$.

$x^L_{i,j} = x_{i+1}$
- if $u_{i,j}u_{i,j+1} \leqslant 0$, then $x^L_{i,j} = x_i$
- else
  - ☐ if $u_{i,j+1}u_{i+1,j+1} < 0$
    Let $(\tilde{x}, y_{j+1}), x_i < \tilde{x} < x_{i+1}$ be one zero point of u, set $x^L_{i,j} = \min(x^L_{i,j}, \tilde{x})$
  - ☐ if $u_{i,j}u_{i+1,j} < 0$
    Let $(\tilde{x}, y_j), x_i < \tilde{x} < x_{i+1}$ be one zero point of u, set $x^L_{i,j} = \min(x^L_{i,j}, \tilde{x})$
- end

$x^R_{i,j} = x_i$
- if $u_{i+1,j}u_{i+1,j+1} \leqslant 0$, then $x^R_{i,j} = x_{i+1}$
- else
  - ☐ if $u_{i,j+1}u_{i+1,j+1} < 0$
    Let $(\tilde{x}, y_{j+1}), x_i < \tilde{x} < x_{i+1}$ be one zero point of u, set $x^R_{i,j} = \max(x^R_{i,j}, \tilde{x})$
  - ☐ if $u_{i,j}u_{i+1,j} < 0$
    Let $(\tilde{x}, y_j), x_i < \tilde{x} < x_{i+1}$ be one zero point of u, set $x^R_{i,j} = \max(x^R_{i,j}, \tilde{x})$
- end

In Strategy 1, when one checks that $u_{i,j+1}u_{i+1,j+1} < 0$ or $u_{i,j}u_{i+1,j} < 0$, then one needs to determine $\tilde{x}$ so that $(\tilde{x}, y_{j+1})$ or $(\tilde{x}, y_j)$ is the zero point of u. Since we only have the grid point values of u, $\tilde{x}$ needs to be computed approximately. Naturally one can choose an interpolation polynomial with same values as u at some grid points, then use Newton iteration to solve the zero point of the polynomial as the approximation to $\tilde{x}$. Generally the computed $\tilde{x}$ has the error $O(h^{R+1})$ if an Rth order polynomial is used and the derivative of the polynomial is $O(1)$ away from zero. For example let $k = j$ or $j+1$, the first to third order interpolation polynomials can be chosen to have values $u_{i,k}, u_{i+1,k}$ at x_i, x_{i+1}, $u_{i,k}, u_{i+1,k}, u_{i+2,k}$ at x_i, x_{i+1}, x_{i+2} and $u_{i-1,k}, u_{i,k}, u_{i+1,k}, u_{i+2,k}$ at $x_{i-1}, x_i, x_{i+1}, x_{i+2}$ respectively. These polynomials are expressed as follows

$$P_1(x) = u_{i,k} + (u_{i+1,k} - u_{i,k})\frac{x - x_i}{h}, \tag{2.18}$$

$$P_2(x) = u_{i,k} + \left(-\frac{3}{2}u_{i,k} + 2u_{i+1,k} - \frac{1}{2}u_{i+2,k}\right)\frac{x-x_i}{h} + \left(\frac{1}{2}u_{i,k} - u_{i+1,k} + \frac{1}{2}u_{i+2,k}\right)\left(\frac{x-x_i}{h}\right)^2, \tag{2.19}$$

$$\begin{aligned} P_3(x) = {} & u_{i,k} + \left(-\frac{1}{3}u_{i-1,k} - \frac{1}{2}u_{i,k} + u_{i+1,k} - \frac{1}{6}u_{i+2,k}\right)\frac{x-x_i}{h} + \left(\frac{1}{2}u_{i-1,k} - u_{i,k} + \frac{1}{2}u_{i+1,k}\right)\left(\frac{x-x_i}{h}\right)^2 \\ & + \left(-\frac{1}{6}u_{i-1,k} + \frac{1}{2}u_{i,k} - \frac{1}{2}u_{i+1,k} + \frac{1}{6}u_{i+2,k}\right)\left(\frac{x-x_i}{h}\right)^3 \end{aligned} \tag{2.20}$$

for $k = j$ or $j+1$.

We can use Newton iteration to solve the zero point of the polynomial higher than first order. The initial guess can be taken to be the zero point of the first order polynomial, namely

$$x_{ini} = x_i + h\left(\frac{-u_{i,k}}{u_{i+1,k} - u_{i,k}}\right). \tag{2.21}$$

If the derivative of the function $u(x, y_k)$ is $O(1)$ away from zero on $[x_i, x_{i+1}]$, then the interpolation polynomial is monotone near the interval $[x_i, x_{i+1}]$, and the Newton iteration can correctly converge to the expected zero point. If the derivative of the function $u(x, y_k)$ is near zero on $[x_i, x_{i+1}]$, it is possible that the interpolation polynomial is non-monotone and there is another zero point of the polynomial outside $[x_i, x_{i+1}]$ which the Newton iteration can converge to. Clearly the correct zero point we need is inside the interval $[x_i, x_{i+1}]$. Therefore in this case the Newton iteration does not give approximation to the expected zero point. Fortunately we will show in Section 3 in the error estimates for our methods that in the case of the derivative of $u(x, y_k)$ being small on $[x_i, x_{i+1}]$, it is not necessary to obtain accurate approximation to $\tilde{x}$. Therefore in the case that the iterated values of $\tilde{x}$ come outside the interval $[x_i, x_{i+1}]$ during the Newton iteration which implies the derivative of $u(x, y_k)$ is small on the interval, we can just quit the Newton iteration and set the computed $\tilde{x}$ to be the initial guess x_{ini}. If the iterated values of $\tilde{x}$ do not come outside the interval $[x_i, x_{i+1}]$ during the Newton iteration, then the iteration should correctly converge to the expected zero point. From the above consideration, we can give the following algorithm to compute $\tilde{x}$ if $u_{i,k}u_{i+1,k} < 0, k = j$ or $j+1$. In the algorithm we let $\tilde{x}_c$ denote the computed $\tilde{x}$.

Algorithm I. Give computed $\tilde{x}$

Choose an Rth order interpolation polynomial to $u(x, y_k)$ near the interval $[x_i, x_{i+1}]$. For $R = 1, 2, 3$ we can choose the polynomials in (2.18)–(2.20).

Let the initial guess to be (2.21) and use Newton iteration to compute the zero point of the polynomial.

If the iterated values of the zero point during the Newton iteration come outside the interval $[x_i, x_{i+1}]$, then quit the Newton iteration and set $\tilde{x}_c = x_{ini}$.

Otherwise set $\tilde{x}_c = x_{New}$, where x_{New} denotes the computed zero point by the Newton iteration with given tolerance E_T. Namely the absolute value of the polynomial at x_{New} is less than E_T.

With Strategy 1 and Algorithm I, we then can compute $x_{i,j}^L, x_{i,j}^R$ and obtain their approximate values $\hat{x}_{i,j}^L, \hat{x}_{i,j}^R$. The next issue is to evaluate the values of F_m at nodal points in the quadrature rule (2.13) which yields the value of $\widehat{I}_{i,j}$. According to the definition (2.10) this corresponds to numerical approximations to one dimensional delta function integrals. High order numerical methods to approximate one dimensional delta function integrals have already been studied [23,24]. In this paper we use the numerical method developed in [23] to a type of delta function integrals which include the one dimensional case.

We are considering that the form (2.4) is valid. Recall the definition of $Y_{i,j}(x), x \in [x_i, x_{i+1}]$ in (2.8) for this situation. The exact value of the one dimensional delta function integral (2.10) is given by

$$F_m(x) = \frac{\alpha(x, Y_{i,j}(x))}{|u_y(x, Y_{i,j}(x))|}, \quad \text{for } x \in [x_i, x_{i+1}], \tag{2.22}$$

where we denote

$$\alpha(x, y) = f(x, y)\sqrt{(u_x)^2 + (u_y)^2}. \tag{2.23}$$

The numerical method proposed in [23] is to perform interpolation in the level set function variable space to approximate (2.22). Since u_y is $O(1)$ away from zero near the cell $C_{i,j}$, we define the inverse function $\widehat{Y}_{i,j}(x, z)$ satisfying

$$u(x, \widehat{Y}_{i,j}(x, z)) = z, \quad \text{for } x \in [x_i, x_{i+1}],\ z \text{ near zero}. \tag{2.24}$$

Observe that

$$Y_{i,j}(x) = \widehat{Y}_{i,j}(x, 0), \quad x \in [x_i, x_{i+1}]. \tag{2.25}$$

Introduce the function

$$\widehat{F}_x(z) = \frac{\alpha(x, \widehat{Y}_{i,j}(x, z))}{|u_y(x, \widehat{Y}_{i,j}(x, z))|}, \quad \text{for } x \in [x_i, x_{i+1}]. \tag{2.26}$$

Then

$$\widehat{F}_x(0) = F_m(x), \tag{2.27}$$

$$\widehat{F}_x(u(x, y_k)) = \frac{\alpha(x, y_k)}{|u_y(x, y_k)|}, \quad \text{for } k \text{ near } j \tag{2.28}$$

using the fact that $\widehat{Y}_{i,j}(x, u(x, y_k)) = y_k$.

The values $\frac{\alpha(x,y_k)}{|u_y(x,y_k)|}, \forall x \in [x_i, x_{i+1}]$ can be approximated from grid point values of α and u. To interpolate $\alpha(x,y_k)$ we first obtain the approximate grid point values of α by difference approximation. From the definition (2.23) we need the approximate grid point values of u_x, u_y, which can be obtained from grid point values of u by difference approximation. Suppose we use a Tth order difference approximation formula. For example for $T = 4$ we can use the formula

$$u_x(x_l,y_k) \approx \frac{1}{h}\left(-\frac{1}{12}u_{l+2,k} + \frac{2}{3}u_{l+1,k} - \frac{2}{3}u_{l-1,k} + \frac{1}{12}u_{l-2,k}\right), \tag{2.29}$$

similarly for u_y. Denote $\alpha_{l,k}$ to be the obtained approximate values of $\alpha(x_l,y_k)$. Then $\alpha(x,y_k)$ is interpolated from these approximate grid point values $\alpha_{l,k}$. Suppose we use a Uth order interpolation formula. For example for $U = 2,3,4$ we can use the following formulas

$$\alpha(x,y_k) \approx \alpha_{i,k} + (\alpha_{i+1,k} - \alpha_{i,k})\epsilon, \quad \text{for } U = 2, \tag{2.30}$$

$$\alpha(x,y_k) \approx \alpha_{i,k} + \left(-\frac{3}{2}\alpha_{i,k} + 2\alpha_{i+1,k} - \frac{1}{2}\alpha_{i+2,k}\right)\epsilon + \left(\frac{1}{2}\alpha_{i,k} - \alpha_{i+1,k} + \frac{1}{2}\alpha_{i+2,k}\right)\epsilon^2, \quad \text{for } U = 3, \tag{2.31}$$

$$\begin{aligned}\alpha(x,y_k) \approx \alpha_{i,k} &+ \left(-\frac{1}{3}\alpha_{i-1,k} - \frac{1}{2}\alpha_{i,k} + \alpha_{i+1,k} - \frac{1}{6}\alpha_{i+2,k}\right)\epsilon + \left(\frac{1}{2}\alpha_{i-1,k} - \alpha_{i,k} + \frac{1}{2}\alpha_{i+1,k}\right)\epsilon^2 \\ &+ \left(-\frac{1}{6}\alpha_{i-1,k} + \frac{1}{2}\alpha_{i,k} - \frac{1}{2}\alpha_{i+1,k} + \frac{1}{6}\alpha_{i+2,k}\right)\epsilon^3, \quad \text{for } U = 4,\end{aligned} \tag{2.32}$$

where

$$\epsilon = \frac{x - x_i}{h}. \tag{2.33}$$

We already have the approximate grid point values of u_y. Then similarly we can interpolate $u_y(x,y_k)$ from approximate grid point values of u_y by a Uth order interpolation formula. We also interpolate $u(x,y_k)$ from grid point values of u by a Uth order interpolation formula. Denote the interpolated values of $\alpha(x,y_k), u_y(x,y_k)$ and $u(x,y_k)$ to be $\alpha^I(x,y_k), u_y^I(x,y_k)$ and $u^I(x,y_k)$ respectively. From (2.28) one has

$$\widehat{F}_x(u^I(x,y_k)) \approx \frac{\alpha^I(x,y_k)}{|u_y^I(x,y_k)|} \equiv F^I_{x,k} \quad \text{for } k \text{ near } j. \tag{2.34}$$

Remind that our goal is to compute $F_m(x)$ at nodal points in the quadrature rule (2.13). From (2.27) and (2.34) it is seen that we have the (approximate) values of $\widehat{F}_x$ at points $u^I(x,y_k)$ for k near j which compose an irregular mesh near the point zero, and we need to approximate the value of $\widehat{F}_x$ at zero. Thus the evaluation of the one dimensional delta function integral (2.10) is equivalent to interpolating $\widehat{F}_x(0)$ from the (approximate) irregular grid point values of $\widehat{F}_x(z)$ near zero, where the variable z is essentially the level set function variable. This shows the design principle of the method developed in [23]. The interpolation can be performed by using Newton interpolation. Suppose we use a Vth order interpolation formula. For $V = 2,3,4$ the formula take the following forms

$$F_m(x) = \widehat{F}_x(0) \approx F^I_{x,j} + (F^I_{x,j+1} - F^I_{x,j})P_{jj+1}, \quad \text{for } V = 2, \tag{2.35}$$

$$F_m(x) = \widehat{F}_x(0) \approx F^I_{x,j} + (F^I_{x,j+1} - F^I_{x,j})P_{jj+1} + [(F^I_{x,j+2} - F^I_{x,j})P_{jj+2} - (F^I_{x,j+1} - F^I_{x,j})P_{jj+1}]P_{j+1j+2}, \quad \text{for } V = 3, \tag{2.36}$$

$$\begin{aligned}F_m(x) = \widehat{F}_x(0) \approx F^I_{x,j-1} &+ (F^I_{x,j} - F^I_{x,j-1})P_{j-1j} + [(F^I_{x,j+1} - F^I_{x,j-1})P_{j-1j+1} - (F^I_{x,j} - F^I_{x,j-1})P_{j-1j}]P_{jj+1} \\ &+ \left\{\left(F^I_{x,j+2} - F^I_{x,j-1}\right)P_{j-1j+2}P_{jj+2} - (F^I_{x,j} - F^I_{x,j-1})P_{j-1j}P_{jj+2}\right. \\ &- \left.\left[(F^I_{x,j+1} - F^I_{x,j-1})P_{j-1j+1} - (F^I_{x,j} - F^I_{x,j-1})P_{j-1j}\right]P_{jj+1}\right\}P_{j+1j+2}, \quad \text{for } V = 4,\end{aligned} \tag{2.37}$$

where we denote

$$P_{ll'} = \frac{-u^I(x,y_l)}{u^I(x,y_{l'}) - u^I(x,y_l)}. \tag{2.38}$$

Summarizing the above discussion, the algorithm to compute $F_m(x)$ for $x \in [x_i, x_{i+1}]$ is given as follows:

Algorithm II. Give computed $F_m(x)$.

Use a Tth order difference approximation formula to approximate grid point values of u_x, u_y from grid point values of u.

Use a Uth order formula to interpolate $\alpha(x,y_k), u_y(x,y_k), u(x,y_k)$ from approximate or exact grid point values of α, u_y, u for some k near j. Denote the interpolated values to be $\alpha^I(x,y_k), u_y^I(x,y_k), u^I(x,y_k)$, and define $F^I_{x,k}$ as in (2.34).

Use a Vth order Newton interpolation formula to approximate $F_m(x) = \widehat{F}_x(0)$ from approximate irregular grid point values of $\widehat{F}_x$. For $V = 2,3,4$ the formula take the form (2.35)–(2.37).

2.3. Algorithm description

With the discussions of strategy and numerical implementation of our method in the above subsections, we now can present the algorithm of our method. The algorithm is given by

Algorithm I*. For each cell $C_{i,j}$ compute $\widehat{I}_{i,j}$ which is the approximation to $I_{i,j}$.
Sum up $\widehat{I}_{i,j}$ for all mesh cells to give the numerical approximation to the delta function integral (2.1).

In the above algorithm the key algorithm to compute $\widehat{I}_{i,j}$ for each cell is given as follows:

Algorithm II*. Give $\widehat{I}_{i,j}$
According to signs of cell vertex values of u one has four possible cases shown in Case CL in Section 2.1.
For cases (1), (2) set $\widehat{I}_{i,j} = 0$.
For case (4) first compare the quantities $\tilde{u}^x_{i,j}, \tilde{u}^y_{i,j}$ defined in (2.14) and (2.15).

If $\tilde{u}^x_{i,j} < \tilde{u}^y_{i,j}$
Set $\widehat{I}_{i,j}$ to be the quadrature formula (2.13), with $\hat{x}^L_{i,j}, \hat{x}^R_{i,j}$ given by Strategy 1 in Section 2.1 and Algorithm I in Section 2.2, and values of $F_m(x)$ at nodal points computed by Algorithm II in Section 2.2.

If $\tilde{u}^x_{i,j} \geqslant \tilde{u}^y_{i,j}$
$\widehat{I}_{i,j}$ can be computed in similar principle based on the form (2.5).
For case (3), set $\widehat{I}_{i,j}$ to be half of the value given by algorithm for case (4).

Algorithm II* include the following parameters: R, E_T, S, T, U, V, where R and E_T are the order of interpolation polynomial and tolerance in Newton iteration in Algorithm I respectively, S denotes the order of the quadrature rule (2.13), T, U, V are the order of difference approximation, the order of interpolation formula and the order of Newton interpolation in Algorithm II respectively. In the next section we will give error estimates for our method given by Algorithms I* and II*. We show that our method can achieve any desired convergence order by selecting corresponding parameters.

3. Error estimates

In this section we give error estimates for our method given by Algorithms I* and II* which show that our method can achieve any desired convergence order. We will give the error estimates for our method in general cases, and also prove the better accuracy of our method when the following assumption on the zero level set Γ of u holds

Assumption 3.1. There are only finite points denoted by $(X_l, Y_l), l = 1, 2, \ldots, L$ on the zero level set Γ at which the tangential vectors are in the direction $(\pm 1, \pm 1)$. Moreover, $\forall M > 0$, $\exists N > 0$ s.t. for any small $h > 0$, the points on Γ at which absolute value of the slope of the tangential vector belonging to $(1 - Mh, 1 + Mh)$ are contained in the area $\bigcup_{l=1}^{L}(X_l - Nh, X_l + Nh) \times (Y_l - Nh, Y_l + Nh)$.

Assumption 3.1 is satisfied generally for a smooth closed curve Γ in two dimensional space. For example the assumption is satisfied when Γ is a closed curve with positive mean curvature.
The main theorem we will prove in this section is as follows:

Theorem 3.1. *Let $m_1 = \min(R, S, T, U, V), m_2 = \min(R + 1, S, T, U, V)$, where R, S, T, U, V are parameters in Algorithm II*. Assume parameter E_T in Algorithm I satisfies $E_T = O(h^{R+1})$. Then the method given by Algorithms I* and II* is m_1th order accurate, namely*

$$\left| \sum_{(i,j)\in\mathbb{Z}^2} \widehat{I}_{i,j} - \int_{\mathbb{R}^2} f(x,y)\sqrt{(u_x)^2 + (u_y)^2}\,\delta(u(x,y))\,dx\,dy \right| = O(h^{m_1}). \tag{3.1}$$

*Moreover, if Assumption 3.1 on the zero level set Γ holds, then the method is m_2*th order accurate, namely

$$\left| \sum_{(i,j)\in\mathbb{Z}^2} \widehat{I}_{i,j} - \int_{\mathbb{R}^2} f(x,y)\sqrt{(u_x)^2 + (u_y)^2}\,\delta(u(x,y))\,dx\,dy \right| = O(h^{m_2}). \tag{3.2}$$

In order to prove Theorem 3.1 we will use the following Lemmas 3.1–3.7. We firstly introduce some definitions.

Consider a cell $C_{i,j}$ having intersection with the zero level set Γ. Recall the definition of $Y_{i,j}(x), x \in [x_i, x_{i+1}]$ in (2.8), which is defined if the quantities $\tilde{u}^x_{i,j}, \tilde{u}^y_{i,j}$ given in (2.14) and (2.15) satisfy $\tilde{u}^x_{i,j} < \tilde{u}^y_{i,j}$. Similarly if $\tilde{u}^x_{i,j} \geqslant \tilde{u}^y_{i,j}$, we can define the smooth function $X_{i,j}(y), y \in [y_j, y_{j+1}]$ as follows:

$$X_{i,j}(y) \text{ satisfying that } (X_{i,j}(y), y), y \in [y_j, y_{j+1}] \text{ are points on } \Gamma. \tag{3.3}$$

Introduce the quantity

$$d_{i,j} = \begin{cases} 0 & \text{if } \tilde{u}^x_{i,j} < \tilde{u}^y_{i,j}, \\ 1 & \text{if } \tilde{u}^x_{i,j} \geqslant \tilde{u}^y_{i,j}. \end{cases} \tag{3.4}$$

Define the set

$$D_{i,j} = \begin{cases} \{(i,l)|\exists x \in [x_i, x_{i+1}], \text{s.t.}(x, Y_{i,j}(x)) \in C_{i,l}\} & \text{if } d_{i,j} = 0, \\ \{(k,j)|\exists y \in [y_j, y_{j+1}], \text{s.t.}(X_{i,j}(y), y) \in C_{k,j}\} & \text{if } d_{i,j} = 1. \end{cases} \tag{3.5}$$

Let $N_{i,j}$ denote the number of elements in the set $D_{i,j}$. Define the subset of $D_{i,j}$

$$E_{i,j} = \begin{cases} D_{i,j} & \text{if } d_{k,l} = d_{i,j}, \forall (k,l) \in D_{i,j}, \\ \{(i,j)\} & \text{if } \exists (k,l) \in D_{i,j} \text{ s.t. } d_{k,l} \neq d_{i,j}. \end{cases} \tag{3.6}$$

Namely if the elements of $D_{i,j}$ yield the same relation between the two components of the gradient of u, then $E_{i,j}$ is the same as $D_{i,j}$. Otherwise $E_{i,j}$ contains only the element (i,j). Denote

$$M_I = \inf_{(x,y)\in\Gamma} \sqrt{(u_x(x,y))^2 + (u_y(x,y))^2}, \tag{3.7}$$

which is a positive quantity.

Lemma 3.1. $N_{i,j} \leqslant 3$ for fine enough mesh such that both $\left|\frac{\tilde{u}^x_{i,j}}{2h} - |u_x(x,y)|\right|$ and $\left|\frac{\tilde{u}^y_{i,j}}{2h} - |u_y(x,y)|\right|$ are less than $\frac{M_I}{2\sqrt{5}}$, $\forall (x,y) \in C_{k,l}, -2 \leqslant k-i, l-j \leqslant 2$, where M_I is defined in (3.7).

Proof. Without loss of generality we discuss the case $d_{i,j} = 0$. If Γ does not intersect with the horizontal sides of the cell $C_{i,j}$, then $N_{i,j} = 1$. Otherwise without loss of generality assume (x', y_{j+1}) is the zero point of u satisfying $x' \in [x_i, \frac{x_i + x_{i+1}}{2}]$. Due to the mesh condition stated in the lemma, one has $|u_x(x,y)| < |u_y(x,y)| + \frac{M_I}{\sqrt{5}}, \forall (x,y) \in C_{i,l}, j-2 \leqslant l \leqslant j+2$. Combining this with the definition (3.7) one obtains $|\frac{u_x(x, Y_{i,j}(x))}{u_y(x, Y_{i,j}(x))}| < 2, \forall (x, Y_{i,j}(x)) \in C_{i,l}, j-2 \leqslant l \leqslant j+2$. Therefore $Y_{i,j}(x) \in (y_j, y_{j+2})$ for $x \in [x_i, x']$. If $\exists x'' \in [x', x_{i+1}]$ s.t. $Y_{i,j}(x'') \leqslant y_j$, then $Y_{i,j}(x) \in (y_{j-1}, y_{j+2})$ for $x \in [x', x_{i+1}]$. On the other hand, if $\exists x'' \in [x', x_{i+1}]$ s.t. $Y_{i,j}(x'') \geqslant y_{j+2}$, then $Y_{i,j}(x) \in (y_j, y_{j+3})$ for $x \in [x', x_{i+1}]$. From above discussions, one sees that $N_{i,j}$ is at most 3. □

Lemma 3.2. *The set $E_{i,j}$ satisfies the following properties*

(i) $(i,j) \in E_{i,j}$.
(ii) $E_{k,l} = E_{i,j}, \forall (k,l) \in E_{i,j}$.
(iii) $E_{k,l} \cap E_{i,j} = \emptyset$, if $(k,l) \notin E_{i,j}$.

Proof. (i) and (ii) can be checked from the definition of $E_{i,j}$ in (3.6). To see (iii), assume (iii) is untrue, namely $\exists (k', l') \in E_{k,l} \cap E_{i,j}$ when $(k,l) \notin E_{i,j}$. From $(k,l) \notin E_{i,j}$ and (i) one has $E_{k,l} \neq E_{i,j}$. However from $(k', l') \in E_{k,l} \cap E_{i,j}$ and (ii) one has $E_{k,l} = E_{k',l'} = E_{i,j}$. This is contradiction. Therefore (iii) is true. □

Lemma 3.3. *The computed $F_m(x), x \in [x_i, x_{i+1}]$ given by Algorithm II has the error $O(h^{m_3})$ with $m_3 = \min(T, U, V)$, where T, U, V are parameters in Algorithm II.*

This lemma can be checked from the description of Algorithm II and the proof is omitted.

Lemma 3.4. *If the cell $C_{i,j}$ belongs to case (4) of Case CL and $d_{i,j} = 0$, then $\widehat{I}_{i,j}$ given by Algorithm II* satisfies*

$$\widehat{I}_{i,j} - \int_{\hat{x}^L_{i,j}}^{\hat{x}^R_{i,j}} F_m(x)dx = O(h^{m_4+1}) \tag{3.8}$$

with

$$m_4 = \min(S, T, U, V), \tag{3.9}$$

where S, T, U, V are parameters in Algorithm II.*

This lemma can be obtained by applying Lemma 3.3 and the proof is omitted.

Lemma 3.5. *Assume the cell $C_{i,j}$ belongs to case (4) of Case CL and $d_{i,j} = 0$. If*

$$|u_x| > M_0 \tag{3.10}$$

near the cell, where M_0 is a positive quantity independent of h, and parameters E_T, R in Algorithm I satisfy $E_T = O(h^{R+1})$, then the error between $x^L_{i,j}, x^R_{i,j}$ defined by Strategy 1 and their computed values $\hat{x}^L_{i,j}, \hat{x}^R_{i,j}$ given by Strategy 1 and Algorithm I are

$$\hat{x}^L_{i,j} = x^L_{i,j} + O(h^{R+1}), \quad \hat{x}^R_{i,j} = x^R_{i,j} + O(h^{R+1}). \tag{3.11}$$

The proof for this lemma is standard and is omitted.

Lemma 3.6. *For a cell $C_{i,j}$ satisfying $d_{i,j} = 0$, it holds*

$$\int_{x_i}^{x_{i+1}} F_m(x)dx = \sum_{(k,l)\in D_{i,j}} I_{k,l}. \tag{3.12}$$

This lemma can be checked from definitions (2.2), (2.10) and (3.5).

Lemma 3.7. *Assume parameters E_T, R in Algorithm I satisfy $E_T = O(h^{R+1})$. Then $\widehat{I}_{k,l}$ for (k,l) adjacent to (i,j) yielded by Algorithm II* have the following error estimates*

$$\sum_{(k,l)\in E_{i,j}} (\widehat{I}_{k,l} - I_{k,l}) = \begin{cases} O(h^{m_4+1}) & \text{if } E_{i,j} = D_{i,j}, \\ O(h^{m_4+1}) + O(h^{R+1}) & \text{if } E_{i,j} \neq D_{i,j}, \end{cases} \tag{3.13}$$

where m_4 is defined in (3.9).

The proof of this lemma is relatively long and is presented in the Appendix. In the case $E_{i,j} = D_{i,j}$ and the case $d_{i,j} = 0$ for example, the strategy is to prove that $\sum_{(k,l)\in E_{i,j}} \widehat{I}_{k,l}$ is the high order approximation to $\int_{x_i}^{x_{i+1}} F_m(x)dx$ which according to Lemma 3.6 is just $\sum_{(k,l)\in E_{i,j}} I_{k,l}$. In the case $E_{i,j} \neq D_{i,j}$ one needs to additionally consider the error between the interval boundary points $x_{i,j}^L, x_{i,j}^R$ and their computed values $\hat{x}_{i,j}^L, \hat{x}_{i,j}^R$.

Remark 3.1. From the definition (3.6), the case $E_{i,j} = D_{i,j}$ implies $E_{i,j}$ can contain the index (i,j) as well as its neighboring ones. Therefore Lemma 3.7 ensures that the summation $\sum_{(k,l)\in E_{i,j}} \widehat{I}_{k,l}$ is the high order accurate approximation to $\sum_{(k,l)\in E_{i,j}} I_{k,l}$, while each $\widehat{I}_{k,l}$ is not necessarily accurate enough to approximate $I_{k,l}$. This is the key point in the proof of error estimates for our method, as mentioned in Sections 1 and 2.1. Compared with the first case in the error estimate (3.13), the extra term $O(h^{R+1})$ in the second case arises from the approximation to the intersection points $x_{k,l}^L, x_{k,l}^R$ between the zero level set Γ and the edges of the cells. In the first case, the errors of computations of $x_{k,l}^L, x_{k,l}^R$ are canceled among the cells $C_{k,l}$ for $(k,l) \in E_{i,j}$ since these cells have the same choice among the two forms (2.4) and (2.5). By using the error estimate (3.13) we can obtain in Theorem 3.1 that our method has better accuracy under Assumption 3.1 than general cases. Since we can show that under Assumption 3.1 the number of cells $C_{i,j}$ such that $E_{i,j} \neq D_{i,j}$ is finite. Namely there are only finite cells having different choice among the two forms (2.4), (2.5) compared with neighboring cells. Therefore the error term $O(h^{R+1})$ directly passes to the error estimate of the method without multiplying by $O(\frac{1}{h})$ which is the number of cells essentially intersecting with the zero level set Γ.

With the above preparation we now give the proof of the main theorem on the error estimates for our method.

Proof of Theorem 3.1. Denote

$$S_U = \{(k,l) | \text{ the cell } C_{k,l} \text{ contains points on } \Gamma\}. \tag{3.14}$$

According to definition of $I_{i,j}$ and Algorithm II*, for any $(i,j) \notin S_U$ one has $I_{i,j} = \widehat{I}_{i,j} = 0$. Therefore

$$\int_{\mathbb{R}^2} f(x,y)\sqrt{(u_x)^2 + (u_y)^2}\,\delta(u(x,y))dxdy = \sum_{(i,j)\in S_U} I_{i,j}, \tag{3.15}$$

$$\sum_{(i,j)\in\mathbb{Z}^2} \widehat{I}_{i,j} = \sum_{(i,j)\in S_U} \widehat{I}_{i,j}. \tag{3.16}$$

Next we prove

$$\sum_{(i,j)\in S_U} I_{i,j} = \sum_{(i,j)\in S_U} \frac{1}{N_{i,j}} \left(\sum_{(k,l)\in E_{i,j}} I_{k,l} \right), \tag{3.17}$$

$$\sum_{(i,j)\in S_U} \widehat{I}_{i,j} = \sum_{(i,j)\in S_U} \frac{1}{N_{i,j}} \left(\sum_{(k,l)\in E_{i,j}} \widehat{I}_{k,l} \right). \tag{3.18}$$

According to definition of $E_{i,j}$, any element in $E_{i,j}$ belongs to S_U. Therefore the right hand side of (3.17) can be written to be

$$\sum_{(i,j)\in S_U} \gamma_{i,j} I_{i,j}. \tag{3.19}$$

To prove (3.17) we need to check that $\gamma_{i,j} = 1$ in (3.19). $\forall (i',j') \in S_U$, according to (iii) in Lemma 3.2 one has that $\sum_{(i,j)\notin E_{i',j'}} \frac{1}{N_{i,j}} (\sum_{(k,l)\in E_{i,j}} I_{k,l})$ does not contain the term $I_{i',j'}$. Therefore the coefficient before the term $I_{i',j'}$ in the expression

$$\sum_{(i,j)\in E_{i',j'}} \frac{1}{N_{i,j}} \left(\sum_{(k,l)\in E_{i,j}} I_{k,l} \right) \tag{3.20}$$

is also $\gamma_{i',j'}$. According to (ii) in Lemma 3.2, (3.20) is just

$$\sum_{(k,l)\in E_{i',j'}} I_{k,l}. \tag{3.21}$$

Therefore $\gamma_{i',j'} = 1, \forall (i',j') \in S_U$. Thus (3.17) is proved. Similarly (3.18) can be proved.

Now utilizing (3.15)–(3.18) and applying Lemma 3.7 one obtains

$$\left| \sum_{(i,j)\in\mathbb{Z}^2} \widehat{I}_{i,j} - \int_{\mathbb{R}^2} f(x,y)\sqrt{(u_x)^2 + (u_y)^2}\,\delta(u(x,y))dxdy \right| = \left| \sum_{(i,j)\in S_U} \frac{1}{N_{i,j}} \left[\sum_{(k,l)\in E_{i,j}} (\widehat{I}_{k,l} - I_{k,l}) \right] \right| = N_V O(h^{R+1}) + N_U O(h^{m_4+1}), \tag{3.22}$$

where N_U denotes the number of elements in S_U, and N_V denotes the number of elements in the set

$$S_V = \{(k,l) \in S_U | E_{k,l} \neq D_{k,l}\}. \tag{3.23}$$

One has $N_U = O(\frac{1}{h})$ and $N_V \leqslant N_U$. Therefore

$$N_V O(h^{R+1}) + N_U O(h^{m_4+1}) = O(h^R) + O(h^{m_4}). \tag{3.24}$$

Combining (3.22) and (3.24) one obtains the estimates (3.1).

Next we prove the estimates (3.2) under Assumption 3.1 on the zero level set Γ. We need to estimate N_V in (3.22). $\forall (i,j) \in S_V$, we consider the case $d_{i,j} = 0$, namely $\tilde{u}^x_{i,j} < \tilde{u}^y_{i,j}$. Then $\exists (i,l) \in D_{ij}$ s.t. $\tilde{u}^x_{i,l} \geqslant \tilde{u}^y_{i,l}$. Since $\tilde{u}^x_{i,k}, \tilde{u}^y_{i,k}$ are second order approximation to $|u_x(x_{i+\frac{1}{2}}, y_{k+\frac{1}{2}})|, |u_y(x_{i+\frac{1}{2}}, y_{k+\frac{1}{2}})|$, where $x_{i+\frac{1}{2}} = \frac{x_i + x_{i+1}}{2}, y_{k+\frac{1}{2}} = \frac{y_k + y_{k+1}}{2}$, at least one of the following two cases occurs

(I) $|u_x(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})| - |u_y(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})|$ and $|u_x(x_{i+\frac{1}{2}}, y_{l+\frac{1}{2}})| - |u_y(x_{i+\frac{1}{2}}, y_{l+\frac{1}{2}})|$ have opposite signs.

(II) One of $|u_x(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})| - |u_y(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})|$ and $|u_x(x_{i+\frac{1}{2}}, y_{l+\frac{1}{2}})| - |u_y(x_{i+\frac{1}{2}}, y_{l+\frac{1}{2}})|$ is $O(h^2)$.

Denote $Y_{i+\frac{1}{2}j} = Y_{ij}(x_{i+\frac{1}{2}})$. In both cases, we have

$$\left| \left| u_x\left(x_{i+\frac{1}{2}}, Y_{i+\frac{1}{2}j}\right) \right| - \left| u_y\left(x_{i+\frac{1}{2}}, Y_{i+\frac{1}{2}j}\right) \right| \right| \leqslant M^y_S N_{ij} h + O(h^2) \leqslant 6 M^y_S h \tag{3.25}$$

for fine enough mesh such that $O(h^2) \leqslant 3M^y_S h$, where M^y_S denotes

$$M^y_S = \sup_{(x,y)\in C_S} |u_{xy}(x,y)| + \sup_{(x,y)\in C_S} |u_{yy}(x,y)| \tag{3.26}$$

with

$$C_S = \bigcup_{(i,j)\in S_U} C_{i,j}. \tag{3.27}$$

Similarly to the deduction of (A.19) in the Appendix we have

$$\left| u_y\left(x_{i+\frac{1}{2}}, Y_{i+\frac{1}{2}j}\right) \right| \geqslant \frac{M_I}{\sqrt{5}} \tag{3.28}$$

for reasonably fine mesh.

Combining (3.25) and (3.28) one obtains

$$\left| \left| \frac{u_x(x_{i+\frac{1}{2}}, Y_{i+\frac{1}{2}j})}{u_y(x_{i+\frac{1}{2}}, Y_{i+\frac{1}{2}j})} \right| - 1 \right| \leqslant 6\sqrt{5} \frac{M^y_S}{M_I} h. \tag{3.29}$$

Namely the absolute value of the slope of the tangential vector of Γ at $(x_{i+\frac{1}{2}}, Y_{i+\frac{1}{2}j})$ belongs to $(1 - 6\sqrt{5}\frac{M^y_S}{M_I}h, 1 + 6\sqrt{5}\frac{M^y_S}{M_I}h)$. Applying Assumption 3.1, $\exists N_1 > 0$ s.t. $(x_{i+\frac{1}{2}}, Y_{i+\frac{1}{2}j}) \in \bigcup_{l=1}^{L} (X_l - N_1 h, X_l + N_1 h) \times (Y_l - N_1 h, Y_l + N_1 h)$. Similarly to the proof of Lemma 3.1 one has that $y_{j-1} < Y_{i+\frac{1}{2}j} < y_{j+2}$ for reasonably fine mesh since $\tilde{u}^x_{i,j} < \tilde{u}^y_{i,j}$. Therefore

$$C_{i,j} \subset \bigcup_{l=1}^{L} (X_l - (N_1+2)h, X_l + (N_1+2)h) \times (Y_l - (N_1+2)h, Y_l + (N_1+2)h), \quad \forall (i,j) \in S_V, d_{i,j} = 0. \tag{3.30}$$

Similarly for the case $d_{i,j} = 1$, one can deduce

$$C_{i,j} \subset \bigcup_{l=1}^{L} (X_l - (N_2+2)h, X_l + (N_2+2)h) \times (Y_l - (N_2+2)h, Y_l + (N_2+2)h), \quad \forall (i,j) \in S_V, d_{i,j} = 1, \tag{3.31}$$

where N_2 is the value of N in Assumption 3.1 corresponding to $M = 6\sqrt{5}\frac{M^x_S}{M_I}$, with M^x_S denoting

$$M^x_S = \sup_{(x,y)\in C_S} |u_{xx}(x,y)| + \sup_{(x,y)\in C_S} |u_{xy}(x,y)| \tag{3.32}$$

and C_S defined in (3.27).

Denote $N_3 = \max(N_1, N_2)$. Then one has

$$C_{i,j} \subset \bigcup_{l=1}^{L} (X_l - (N_3+2)h, X_l + (N_3+2)h) \times (Y_l - (N_3+2)h, Y_l + (N_3+2)h), \quad \forall (i,j) \in S_V. \tag{3.33}$$

Therefore N_V as the number of elements in S_V should be no more than $4(N_3+2)^2 L$ which is finite independent of h. Thus

$$N_V O(h^{R+1}) + N_U O(h^{m_4+1}) = O(h^{R+1}) + O(h^{m_4}). \tag{3.34}$$

Combining (3.22) and (3.34) one obtains the estimates (3.2) under Assumption 3.1. □

4. Numerical examples

In this section we present numerical examples to show the accuracy of our methods. In the following examples we use Simpson rule in the numerical quadrature (2.13), namely we choose $S = 4$ in Algorithm II*. We choose $T = 4$ in Algorithm II* by using the fourth order difference approximation formula such as (2.29). For other parameters in Algorithm II* we test the following three set of choices:

Method A: $R = 1, E_T = h^2, U = V = 2$,
Method B: $R = 2, E_T = h^3, U = V = 3$,
Method C: $R = 3, E_T = h^4, U = V = 4$.

In the first example the level set function does not satisfy Assumption 3.1 and has low regularity. Thus numerical methods cannot achieve higher than second order accuracy in this example. In the other examples the level set functions are smooth and satisfy Assumption 3.1. According to Theorem 3.1, methods A, B, C have second to fourth order accuracy respectively for these examples. We will see that these methods achieve or are better than the expected accuracy in these examples. We will also demonstrate the advantage of using high order numerical methods for computing (2.1) in the last example.

Example 4.1. This is an example tested in [3]. Let $d(\Gamma, x, y)$ be the signed distance function to the capsule shaped curve appearing in Fig. 1 of [3] with $a = 0.1\sqrt{2}, L = 1.4$ and $u(x, y) = d(\Gamma, x, y)(\sin(4\pi x) + 2)(\cos(2\pi y) + 1.6), f(x, y) = \nabla\tilde{f} \cdot \mathbf{n}_\Gamma$, where $\tilde{f}(x, y) = \cos(x)\sin(y)$, and $\mathbf{n}_\Gamma$ is the normal vector to Γ. The exact value of (2.1) is zero. For a given mesh size we randomly shift the uniform mesh in the x and y directions for 50 times. Table 4.1 presents the averaged absolute errors of the three methods over the 50 trials. The last column in the table presents the estimated convergence rates. The level set function in this example does not satisfy Assumption 3.1. In this case if the level set function is smooth, then methods A, B, C have first to third order accuracy according to Theorem 3.1. However the level set function in this example has low regularity. Therefore numerical methods should not achieve higher than second order accuracy in this example. One sees that the estimated convergence rates shown in Table 4.1 are first, more than second and third order respectively. Moreover from the results on fine meshes one sees that method C is just more than second order accuracy. This matches with the fact that the low regularity of the level set function in this example prevents numerical methods to achieve higher than second order accuracy.

Example 4.2. This is an example tested in [15]. Let $u(x, y) = x^2 + y^2 - 1, f(x, y) = 3x^2 - y^2$. The exact value of (2.1) is 2π. Table 4.2 presents the averaged relative errors of the three methods over 50 trials in which the uniform computational mesh is randomly shifted in the x and y directions. One sees that methods A, C achieve the expected second and fourth order accuracy, while method B is better than the expected third order accuracy. The better performance of method B is related to the fact that the level set function $u(x, y)$ in this test is a second order polynomial. Therefore second or higher order interpolation

Table 4.1
Example 4.1, averaged absolute errors of methods A, B, C.

Mesh size	0.05	0.025	0.0125	0.00625	0.003125	0.0015625	R_e
Method A	1.05E−2	5.13E−3	2.19E−3	1.10E−3	5.37E−4	2.53E−4	1.08
Method B	4.74E−3	5.86E−4	6.63E−5	1.48E−5	2.93E−6	7.03E−7	2.53
Method C	5.19E−2	2.09E−4	3.19E−5	8.73E−6	1.93E−6	3.59E−7	3.08

Table 4.2
Example 4.2, averaged relative errors of methods A, B, C, $u(x, y) = x^2 + y^2 - 1$.

Mesh size	0.1	0.05	0.025	0.0125	0.00625	0.003125	R_e
Method A	7.59E−4	1.75E−4	4.12E−5	9.90E−6	2.62E−6	6.46E−7	2.04
method B	1.11E−4	8.63E−6	4.26E−7	2.70E−8	1.38E−9	7.33E−11	4.13
method C	3.93E−5	2.28E−6	1.43E−7	8.78E−9	5.50E−10	3.44E−11	4.02

Table 4.3
Example 4.2, averaged relative errors of methods A, B, C, $u(x, y) = e^{x^2+y^2-1} - 1$.

Mesh size	0.1	0.05	0.025	0.0125	0.00625	0.003125	R_e
Method A	1.31E−3	3.56E−4	9.03E−5	2.29E−5	5.77E−6	1.43E−6	1.97
method B	1.79E−4	2.34E−5	2.54E−6	1.42E−7	3.32E−8	4.57E−9	3.11
method C	5.58E−5	3.00E−6	1.85E−7	1.15E−8	7.23E−10	4.62E−11	4.03

Table 4.4
Example 4.3, averaged relative errors of methods A, B, C, $u(x,y)=\frac{x^2}{(1.5)^2}+\frac{y^2}{(0.75)^2}-1$.

Mesh size	0.1	0.05	0.025	0.0125	0.00625	0.003125	R_e
Method A	1.45E−3	3.31E−4	7.74E−5	2.18E−5	5.07E−6	1.35E−6	2.01
method B	5.61E−4	2.78E−5	1.66E−6	8.81E−8	6.11E−9	4.30E−10	4.06
method C	5.26E−4	2.73E−5	1.58E−6	1.00E−7	6.37E−9	4.13E−10	4.05

polynomial as well as third or higher order interpolation formula to this function actually yield exact results. As a comparison, Table 4.3 presents the same type of errors of the three methods when choosing the level set function $u(x,y)=e^{x^2+y^2-1}-1$ and the same weight function. In this test one sees that the three methods give the expected convergence rates.

Example 4.3. This is an example tested in [15]. Let $u(x,y)=\frac{x^2}{(1.5)^2}+\frac{y^2}{(0.75)^2}-1, f(x,y)=1$. The exact value of (2.1) is ≈ 7.26633616541076. Table 4.4 presents the averaged relative errors of the three methods over 50 trials in which the uniform computational mesh is randomly shifted in the x and y directions and rotated. As in Example 4.2, methods A, C achieve the expected accuracy while method B is better than the expected accuracy since the level set function $u(x,y)$ is a second order polynomial. We then choose the level set function $u(x,y)=e^{\frac{x^2}{(1.5)^2}+\frac{y^2}{(0.75)^2}-1}-1$ and the same weight function, and present the same type of errors of the three methods in Table 4.5. In this test methods A, C achieve second and fourth order accuracy, and method B is less than fourth order accuracy. By observing the results on fine meshes one sees that method B is actually third order accuracy, matching with the expected result given by Theorem 3.1.

Example 4.4. This is an example tested in [19]. Let $u(x,y)=\frac{x^2}{9}+\frac{y^2}{4}-1, f(x,y)=\nabla\cdot(\nabla u(x,y)/\|\nabla u(x,y)\|)$. The exact value of (2.1) is 2π. The grid point values of $f(x,y)$ are evaluated using the fourth order difference approximation such as (2.29) to simulate practical applications in which the analytical expression of $u(x,y)$ is unknown. Table 4.6 presents the averaged relative errors of the three numerical methods over 50 trials in which the uniform computational mesh is rotated by 45° and

Table 4.5
Example 4.3, averaged relative errors of methods A, B, C, $u(x,y)=e^{\frac{x^2}{(1.5)^2}+\frac{y^2}{(0.75)^2}-1}-1$.

Mesh size	0.1	0.05	0.025	0.0125	0.00625	0.003125	R_e
Method A	1.47E−3	3.18E−4	9.21E−5	1.93E−5	4.96E−6	1.14E−6	2.06
method B	5.70E−4	3.14E−5	2.17E−6	1.90E−7	2.48E−8	3.04E−9	3.49
method C	8.28E−4	4.09E−5	2.58E−6	1.60E−7	9.52E−9	6.02E−10	4.06

Table 4.6
Example 4.4, averaged relative errors of methods A, B, C, $u(x,y)=\frac{x^2}{9}+\frac{y^2}{4}-1$.

Mesh size	0.2	0.1	0.05	0.025	0.0125	0.00625	R_e
Method A	3.14E−3	7.78E−4	1.94E−4	4.87E−5	1.21E−5	3.01E−6	2
method B	1.20E−3	2.68E−4	6.54E−5	2.65E−6	1.18E−7	8.76E−9	3.53
method C	6.87E−4	2.13E−5	1.07E−6	5.62E−8	3.47E−9	2.17E−10	4.28

Table 4.7
Example 4.4, averaged relative errors of methods A, B, C, $u(x,y)=e^{\frac{x^2}{9}+\frac{y^2}{4}-1}-1$.

Mesh size	0.2	0.1	0.05	0.025	0.0125	0.00625	R_e
Method A	3.95E−3	1.01E−3	2.50E−4	6.19E−5	1.54E−5	3.94E−6	2
method B	2.12E−3	4.64E−4	2.65E−5	1.54E−6	5.74E−8	5.91E−9	3.87
method C	8.44E−4	4.04E−5	2.38E−6	1.46E−7	9.14E−9	5.72E−10	4.08

Table 4.8
Example 4.5, averaged relative errors of methods A, B, C.

Mesh size	0.1	0.05	0.025	0.0125	0.00625	0.003125	R_e
Method A	4.24E−3	1.44E−3	2.49E−4	7.71E−5	1.87E−5	4.65E−6	1.99
method B	3.22E−3	2.49E−4	3.31E−5	3.48E−6	4.43E−7	6.99E−8	3.09
method C	1.30E−3	3.24E−5	1.30E−6	7.31E−8	3.08E−9	2.11E−10	4.49

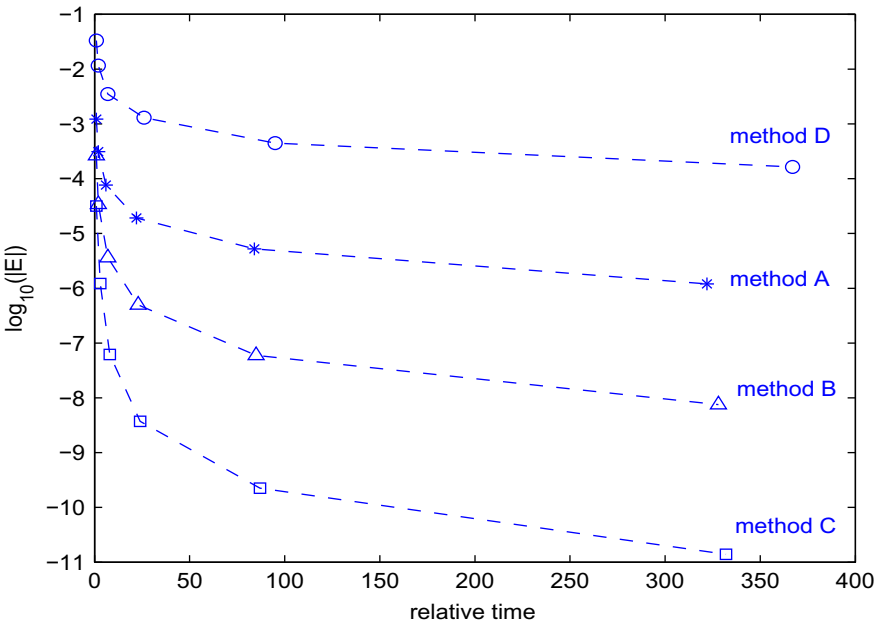

Fig. 4.1. Relation between $\log_{10}(|E|)$ and the relative time for different methods.

randomly shifted in the x and y directions. Similar phenomenon to previous examples are observed. Methods A, C achieve the expected accuracy while method B is better than the expected third order accuracy. We then test choosing the level set function $u(x,y) = e^{\frac{x^2}{9}+\frac{y^2}{4}-1} - 1$ and the same weight function. Table 4.7 presents the same type of errors of the three methods. In this test methods A, C achieve the expected second and fourth order accuracy and method B is still better than third order accuracy. In comparison, method B is less accurate than method C.

Example 4.5. Let $u(x,y) = e^{\frac{x^2}{(1.5)^2}+\frac{y^2}{(0.75)^2}-1} - 1, f(x,y) = 3x + y^2$. The exact value of (2.1) is ≈ 2.37032870455484. Table 4.8 presents the averaged relative errors of the three methods over 50 trials in which the uniform computational mesh is randomly shifted in the x and y directions and rotated. The methods A, B, C are observed to be second to fourth order accurate respectively as ensured by Theorem 3.1.

We also test the computational complexity of our methods in this example. As a comparison we also present the results for the approximate delta function with a variable support size formula proposed in [3] using the linear hat discrete delta function. We call this method as method D. As shown in [3] this method is slightly better than first order accurate. We test the mesh sizes from 0.05 to 0.0015625. We denote $\log_{10}(|E|)$ to be the $\log_{10}$ of the averaged relative error and define the relative time to be the actual computational time divided by the computational time used by method D on the most course mesh. Fig. 4.1 shows the relation between $\log_{10}(|E|)$ and the relative time for different methods. It is obvious that method with higher order accuracy has higher efficiency. Namely the method with higher order accuracy achieves much better accuracy using the same computational time. It is even observed from the results on fine meshes that the fourth order method C uses less computational time than method D when using the same mesh size. Therefore the advantage of using high order numerical methods for computing the delta function integrals (2.1) is demonstrated.

5. Conclusion

In this paper we studied a class of high order numerical methods to delta function integrals appearing in level set methods in two dimensional case. The methods were constructed by considering the approximation of the delta function integral restricted to mesh cells. In each mesh cell the two dimensional delta function integral can be rewritten as a one dimensional ordinary integral with the smooth integrand being a one dimensional delta function integral. The form of the one dimensional integral takes one of two choices according to the comparison of the two components of gradient of the level set function which can be checked from the mesh point values of the level set function. Consequently the mesh cell restriction of the two dimensional delta function integral were approximated by applying standard one dimensional high order numerical quadratures and high order numerical methods to delta function integrals designed in [23] for the one dimensional case. The algorithm designed under such principle to approximate the mesh cell restrictions of the two dimensional delta function integral (2.1) comprises the numerical method proposed in this paper. The method contains several sub-algorithms including Newton iteration to solve one dimensional interpolation polynomials, Numerical quadrature to one dimensional ordinary

integrals, difference approximation formula, interpolation formula and numerical method to approximate one dimensional delta function integrals. We established error estimates for the proposed method which show that the method can achieve any desired accuracy by choosing the corresponding accuracy in the sub-algorithms of the method. We also proved that the method has better accuracy under an assumption on the zero level set of the level set function which is satisfied commonly by a smooth closed zero level set. We presented numerical examples in which second to fourth order methods were implemented and shown to achieve or exceed the expected accuracy indicated by the error analysis. The advantage of using high order numerical methods for computing the two dimensional delta function integrals was also demonstrated by the numerical test. We considered the two dimensional case of the delta function integrals (1.1) in this paper. Recently the idea in this paper has also been extended to develop high order numerical methods to the delta function integrals (1.1) in three dimensional case [25].

Appendix A

In this appendix we give the proof of Lemma 3.7.

Proof. We prove (3.13) for the case that $d_{i,j} = 0$. The case $d_{i,j} = 1$ can be similarly analyzed.

We now prove the first part in (3.13) in the case of $E_{i,j} = D_{i,j}$. The discussions are classified according to value of $N_{i,j}$ which is no more than 3 by Lemma 3.1.

If $N_{i,j} = 1$, namely $D_{i,j} = \{(i,j)\}$, then $y_j < Y_{i,j}(x) < y_{j+1}$, for $x \in [x_i, x_{i+1}]$. In this case according to Strategy 1 one has $\hat{x}^L_{i,j} = x^L_{i,j} = x_i, \hat{x}^R_{i,j} = x^R_{i,j} = x_{i+1}$. According to Lemmas 3.4 and 3.6 one has

$$\widehat{I}_{i,j} = \int_{\hat{x}^L_{i,j}}^{\hat{x}^R_{i,j}} F_m(x)dx + O(h^{m_4+1}) = \int_{x_i}^{x_{i+1}} F_m(x)dx + O(h^{m_4+1}) = I_{i,j} + O(h^{m_4+1}). \tag{A.1}$$

If $N_{i,j} = 2$, then the two cells corresponding to the elements in $D_{i,j}$ are adjacent since Γ is smooth. Without loss of generality we assume the two elements in $D_{i,j}$ are (i,j) and $(i,j+1)$. Denote

$$\widehat{Y}_i = Y_{i,j}(x_i), \quad \widehat{Y}_{i+1} = Y_{i,j}(x_{i+1}). \tag{A.2}$$

If $(\widehat{Y}_i - y_{j+1})(\widehat{Y}_{i+1} - y_{j+1}) < 0$, without loss of generality we consider the case $\widehat{Y}_i > y_{j+1}, \widehat{Y}_{i+1} < y_{j+1}$. According to Strategy 1 and Algorithm I one has $\hat{x}^L_{i,j+1} = x_i, x_i < \hat{x}^R_{i,j+1} = \hat{x}^L_{i,j} < x_{i+1}, \hat{x}^R_{i,j} = x_{i+1}$. Utilizing Lemmas 3.4 and 3.6 one has

$$\sum_{(k,l)\in D_{i,j}} \widehat{I}_{k,l} = \int_{\hat{x}^L_{i,j+1}}^{\hat{x}^R_{i,j+1}} F_m(x)dx + \int_{\hat{x}^L_{i,j}}^{\hat{x}^R_{i,j}} F_m(x)dx + O(h^{m_4+1}) = \int_{x_i}^{x_{i+1}} F_m(x)dx + O(h^{m_4+1}) = \sum_{(k,l)\in D_{i,j}} I_{k,l} + O(h^{m_4+1}). \tag{A.3}$$

If $(\widehat{Y}_i - y_{j+1})(\widehat{Y}_{i+1} - y_{j+1}) \geqslant 0$, we classify two cases.

(I) $\widehat{Y}_i = \widehat{Y}_{i+1} = y_{j+1}$.
(II) $\widehat{Y}_i \neq y_{j+1}$ or $\widehat{Y}_{i+1} \neq y_{j+1}$.

For case (I), Strategy 1 yields $\hat{x}^L_{i,j} = \hat{x}^L_{i,j+1} = x_i, \hat{x}^R_{i,j} = \hat{x}^R_{i,j+1} = x_{i+1}$. In this case the cells $C_{i,j}, C_{i,j+1}$ belong to case (3) of Case CL according to signs of cell vertex values of u. Therefore from Algorithm II* and Lemma 3.4 one has

$$\widehat{I}_{i,j} = \widehat{I}_{i,j+1} = \frac{1}{2}\int_{x_i}^{x_{i+1}} F_m(x)dx + O(h^{m_4+1}), \tag{A.4}$$

which yields

$$\sum_{(k,l)\in D_{i,j}} \widehat{I}_{k,l} = \int_{x_i}^{x_{i+1}} F_m(x)dx + O(h^{m_4+1}) = \sum_{(k,l)\in D_{i,j}} I_{k,l} + O(h^{m_4+1}). \tag{A.5}$$

For case (II), without loss of generality we consider $\widehat{Y}_i < y_{j+1}, \widehat{Y}_{i+1} \leqslant y_{j+1}$. Then the cell $C_{i,j+1}$ belongs to case (1) or (2) of Case CL, therefore

$$\widehat{I}_{i,j+1} = 0. \tag{A.6}$$

The cell $C_{i,j}$ belongs to case (4) of Case CL. Strategy 1 yields $\hat{x}^L_{i,j} = x_i, \hat{x}^R_{i,j} = x_{i+1}$. From Algorithm II* and Lemma 3.4 one has

$$\widehat{I}_{i,j} = \int_{x_i}^{x_{i+1}} F_m(x)dx + O(h^{m_4+1}). \tag{A.7}$$

From (A.6) and (A.7) one also obtains (A.5) for case (II).

If $N_{i,j} = 3$, without loss of generality we assume the three adjacent elements in $D_{i,j}$ are $(i,j), (i,j+1), (i,j+2)$. Since the curve $(x, Y_{i,j}(x)), x \in [x_i, x_{i+1}]$ occupies three cells, one has

$$\exists x' \in [x_i, x_{i+1}],\ \text{s.t.}\ \left|\frac{dY_{i,j}(x')}{dx}\right| \geqslant 1. \tag{A.8}$$

Combining this with (3.7) gives $|u_x(x', Y_{i,j}(x'))| \geqslant \frac{M_I}{\sqrt{2}}$. For fine enough mesh such that

$$|u_x(x,y)| \geqslant \frac{M_I}{2\sqrt{2}}, \quad \text{for } (x,y) \in \bigcup_{(i,l)\in D_{i,j}} C_{i,l}, \tag{A.9}$$

then $Y_{i,j}(x)$ is monotone on $[x_i, x_{i+1}]$. Thus one of $\widehat{Y}_i, \widehat{Y}_{i+1}$ is no less than y_{j+2} and another one is no more than y_{j+1}. Without loss of generality we consider $\widehat{Y}_i \geqslant y_{j+2}, \widehat{Y}_{i+1} \leqslant y_{j+1}$. If $\widehat{Y}_i = y_{j+2}, \widehat{Y}_{i+1} = y_{j+1}$, then (A.5) can be deduced following discussions for the case $N_{i,j} = 1$. If $\widehat{Y}_i = y_{j+2}, \widehat{Y}_{i+1} < y_{j+1}$ or $\widehat{Y}_i > y_{j+2}, \widehat{Y}_{i+1} = y_{j+1}$, then (A.5) can be deduced following discussions for the case $N_{i,j} = 2$. If $\widehat{Y}_i > y_{j+2}, \widehat{Y}_{i+1} < y_{j+1}$, then Strategy 1 and Algorithm I yields

$$\hat{x}^L_{i,j+2} = x_i, \quad \hat{x}^R_{i,j} = x_{i+1}, \tag{A.10}$$

$$x_i < \hat{x}^R_{i,j+2}, \quad \hat{x}^L_{i,j} < x_{i+1}, \tag{A.11}$$

$$\hat{x}^L_{i,j+1} = \min\left(\hat{x}^R_{i,j+2}, \hat{x}^L_{i,j}\right), \quad \hat{x}^R_{i,j+1} = \max\left(\hat{x}^R_{i,j+2}, \hat{x}^L_{i,j}\right). \tag{A.12}$$

Moreover we have

$$\hat{x}^R_{i,j+2} < \hat{x}^L_{i,j} \qquad \text{for fine enough mesh stated below.} \tag{A.13}$$

This can be seen as follows. According to Strategy 1, $x^R_{i,j+2}, x^L_{i,j}$ satisfy

$$Y_{i,j}(x^R_{i,j+2}) = y_{j+2}, \quad Y_{i,j}(x^L_{i,j}) = y_{j+1}. \tag{A.14}$$

Assume the mesh is fine enough to satisfy the condition stated in Lemma 3.1 for $(x,y) \in \bigcup_{(i,l)\in D_{i,j}} C_{i,l}$. Following the same deduction as in the proof of Lemma 3.1 one has $|\frac{dY_{i,j}(x)}{dx}| < 2$ for $x \in [x_i, x_{i+1}]$. Therefore

$$x^L_{i,j} > x^R_{i,j+2} + \frac{h}{2}. \tag{A.15}$$

Using the condition (A.9) and applying Lemma 3.5 to cells $C_{i,j}$ and $C_{i,j+2}$ one has

$$\hat{x}^R_{i,j+2} = x^R_{i,j+2} + O(h^{R+1}),\ \hat{x}^L_{i,j} = x^L_{i,j} + O(h^{R+1}) \quad \text{with } R \geqslant 1. \tag{A.16}$$

Combining (A.15) and (A.16) one obtains (A.13).

From (A.11), (A.12) and (A.13) one has

$$x_i < \hat{x}^R_{i,j+2} = \hat{x}^L_{i,j+1} < \hat{x}^R_{i,j+1} = \hat{x}^L_{i,j} < x_{i+1}. \tag{A.17}$$

Utilizing (A.10) and (A.17), according to Lemmas 3.4 and 3.6 one has

$$\begin{aligned}\sum_{(k,l)\in D_{i,j}} \widehat{I}_{k,l} &= \int_{\hat{x}^L_{i,j+2}}^{\hat{x}^R_{i,j+2}} F_m(x)dx + \int_{\hat{x}^L_{i,j+1}}^{\hat{x}^R_{i,j+1}} F_m(x)dx + \int_{\hat{x}^L_{i,j}}^{\hat{x}^R_{i,j}} F_m(x)dx + O(h^{m_4+1}) = \int_{x_i}^{x_{i+1}} F_m(x)dx + O(h^{m_4+1}) \\ &= \sum_{(k,l)\in D_{i,j}} I_{k,l} + O(h^{m_4+1}).\end{aligned} \tag{A.18}$$

From the above discussions, we have proved that for all possible cases $N_{i,j} = 1, 2$ or 3, (A.5) can always be deduced, which proves the first part in (3.13).

We then prove the second part in (3.13). If $E_{i,j} \neq D_{i,j}$, according to definition (3.6) we have $E_{i,j} = \{(i,j)\}$ and $\exists (i,l) \in D_{i,j}$ s.t. $d_{i,l} = 1 \neq d_{ij} = 0$. Assume the mesh is fine enough such that both $|\frac{\tilde{u}^x_{i,l}}{2h} - |u_x(x,y)||$ and $|\frac{\tilde{u}^y_{i,l}}{2h} - |u_y(x,y)||$ are less than $\frac{M_I}{2\sqrt{5}}$ for $(x,y) \in \bigcup_{(i,k)\in D_{i,j}} C_{i,k}$. From $\tilde{u}^x_{i,l} \geqslant \tilde{u}^y_{i,l}$ and using similar deduction in the proof of Lemma 3.1 one gets

$$|u_x(x, Y_{i,j}(x))| \geqslant \frac{M_I}{\sqrt{5}}, \quad \forall (x, Y_{i,j}(x)) \in \bigcup_{(i,k)\in D_{i,j}} C_{i,k}. \tag{A.19}$$

Therefore $Y_{i,j}(x)$ is monotone on $x \in [x_i, x_{i+1}]$ which implies $\Lambda_{i,j} = [x^L_{i,j}, x^R_{i,j}]$ and (2.12) gives

$$I_{i,j} = \int_{x^L_{i,j}}^{x^R_{i,j}} F_m(x)dx. \tag{A.20}$$

Utilizing (A.19), according to Lemma 3.5 one has

$$\hat{x}^L_{i,j} = x^L_{i,j} + O(h^{R+1}), \quad \hat{x}^R_{i,j} = x^R_{i,j} + O(h^{R+1}). \tag{A.21}$$

Combining (A.20) and (A.21) gives

$$I_{i,j} = \int_{\tilde{x}_{i,j}^L}^{\tilde{x}_{i,j}^R} F_m(x)dx + O(h^{R+1}). \tag{A.22}$$

Together with (A.22) and Lemma 3.4 one obtains the second part in (3.13). □

References

[1] J.T. Beale, A proof that a discrete delta function is second-order accurate, J. Comput. Phys. 227 (2008) 2195–2197.
[2] L.-T. Cheng, Construction of shapes arising from the Minkowski problem using a level set approach, J. Sci. Comput. 19 (2003) 123–138.
[3] B. Engquist, A.K. Tornberg, R. Tsai, Discretization of dirac delta functions in level set methods, J. Comput. Phys. 207 (1) (2005) 28–51.
[4] S. Jin, H.L. Liu, S. Osher, R. Tsai, Computing multivalued physical observables for the semiclassical limit of the Schrödinger equation, J. Comput. Phys. 205 (2005) 222–241.
[5] S. Jin, H.L. Liu, S. Osher, R. Tsai, Computing multi-valued physical observables for high frequency limit of symmetric hyperbolic systems, J. Comput. Phys. 210 (2005) 497–518.
[6] S. Jin, X. Wen, Hamiltonian-preserving schemes for the Liouville equation with discontinuous potentials, Commun. Math. Sci. 3 (2005) 285–315.
[7] S. Jin, X. Wen, Hamiltonian-preserving schemes for the Liouville equation of geometrical optics with discontinuous local wave speeds, J. Comput. Phys. 214 (2006) 672–697.
[8] C. Min, F. Gibou, Geometric integration over irregular domains with application to level-set methods, J. Comput. Phys. 226 (2007) 1432–1443.
[9] C. Min, F. Gibou, Robust second-order accurate discretizations of the multi-dimensional Heaviside and Dirac delta functions, J. Comput. Phys. 227 (2008) 9686–9695.
[10] S. Osher, R.P. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Springer Verlag, 2002.
[11] D. Peng, B. Merriman, S. Osher, H. Zhao, M. Kang, A PDE-based fast local level set method, J. Comput. Phys. 155 (1999) 410–438.
[12] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[13] C.S. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–511.
[14] J.A. Sethian, Level Set Methods and Fast Marching Methods, Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Science, Cambridge University Press, 1999.
[15] P. Smereka, The numerical approximation of a delta function with application to level set methods, J. Comput. Phys. 211 (2006) 77–90.
[16] M. Sussman, P. Smereka, S. Osher, A level set method for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[17] A.-K. Tornberg, B. Engquist, Regularization techniques for numerical approximation of PDEs with singularities, J. Sci. Comput. 19 (2003) 527–552.
[18] A.-K. Tornberg, B. Engquist, Numerical approximations of singular source terms in differential equations, J. Comput. Phys. 200 (2004) 462–488.
[19] J.D. Towers, Two methods for discretizing a delta function supported on a level set, J. Comput. Phys. 220 (2) (2007) 915–931.
[20] J.D. Towers, A convergence rate theorem for finite difference approximations to delta functions, J. Comput. Phys. 227 (13) (2008) 6591–6597.
[21] S.O. Unverdi, G. Tryggvason, A front-tracking method for viscous, incompressible multi-fluid flows, J. Comput. Phys. 100 (1992) 25–37.
[22] G.W. Wei, Y.B. Zhao, Y. Xiang, Discrete singular convolution and its application to the analysis of plates with internal supports. Part 1: Theory and algorithm, Internat. J. Numer. Methods Engrg. 55 (2002) 913C946.
[23] X. Wen, High order numerical methods to a type of delta function integrals, J. Comput. Phys. 226 (2007) 1952–1967.
[24] X. Wen, High order numerical quadratures to one dimensional delta function integrals, SIAM J. Sci. Comput. 30 (4) (2008) 1825–1846.
[25] X. Wen, High order numerical methods to three dimensional delta function integrals in level set methods, preprint.
[26] H.-K. Zhao, T. Chan, B. Merriman, S. Osher, A variational level set approach to multiphase motion, J. Comput. Phys. 127 (1996) 179–195.