Fourth- and Higher-order Interface Tracking Via Mapping and Adjusting Regular Semianalytic sets Represented by Cubic Splines

Article *in* SIAM Journal on Scientific Computing · November 2018

DOI: 10.1137/17M1149328

1 author:

Qinghai Zhang
Zhejiang University
30 PUBLICATIONS **341** CITATIONS

FOURTH- AND HIGHER-ORDER INTERFACE TRACKING VIA MAPPING AND ADJUSTING REGULAR SEMIANALYTIC SETS REPRESENTED BY CUBIC SPLINES

QINGHAI ZHANG*

Abstract. This work is a further development and the culmination along our research line of interface tracking in two dimensions [Zhang and Liu, J. Comput. Phys. 27 (2008): 4063–4088] [Zhang, SIAM J. Numer. Anal. 51 (2013): 2822–2850] [Zhang and Fogelson, SIAM J. Sci. Comput. 36 (2014): A2369–A2400] [Zhang and Fogelson, SIAM J. Numer. Anal. 54 (2016): 530–560]. Previously, we have proposed an analytic framework for explicit interface tracking and a fourth-order interface tracking method. In this paper, we continue our approach to propose a new method, called the cubic MARS method, that features (1) a solid theoretical foundation consisting of well developed concepts from distinct branches of mathematics, (2) a representation of the interface with cubic splines, (3) a novel algorithm for Boolean operations on linear/curvilinear polygons, and (4) fourth-, sixth-, and eighth-order accuracy via a refined control of the relation between the Eulerian grid size h and a Lagrangian length scale h_L of interface markers. Volume conservation of the cubic MARS method also has convergence rates at 4, 6, and 8. A simple analysis and numerical results from several benchmark problems show that the cubic MARS method is superior to existing interface tracking methods in terms of accuracy and efficiency. In particular, for the single-vortex test and the deformation test, errors of an eighth-order cubic MARS method are close to machine precision on a 128-by-128 Eulerian grid! Currently the new method does not tackle topological changes of the interface, but its theoretical foundation provides a solid footing for future treatments of such phenomena.

Key words. interface tracking, cubic splines, MARS, curvilinear polygon clipping, Yin sets, vortex-shear test, deformation test.

AMS subject classifications. 76T30, 65D07, 34A26

DOI. 10.1137/17M1149328 (Published in **SIAM J. Sci. Comput.** 2018)

Funding: This work was supported in part by the grant with approval number 11871429 from the National Natural Science Foundation of China.

1. Introduction.

Interface tracking (IT) is a fundamental problem in multiphase flows since in many cases the IT accuracy largely affects the fidelity of numerical simulation. Over the past decades, numerous methods have been developed, among which the three most popular methods are the front-tracking method [45], the level-set method [32], and the volume-of-fluid (VOF) method [21].

Level-set methods *implicitly* represent the interface as the zero isocontour of a signed distance function while front-tracking methods *explicitly* represent the interface as a set of connected markers. In VOF methods, an interface is implicitly described by volume fractions of the tracked material inside fixed control volumes, but this *static* implicit description must also be aided by an auxiliary explicit representation (such as a piecewise linear function) in order to *evolve* the interface. Within each time step, a VOF method consists of two substeps: in the first reconstruction substep the explicit representation of material regions is determined from volume fractions with additional assumptions on the piecewise function form; in the second advection substep the new volume fractions are calculated from the explicit representation and the velocity field. Coupled, adaptive, variant versions of these three families of methods have also been proposed; see, e.g., [40, 18, 42, 16, 15, 27, 38, 41, 28, 12, 1, 57, 13, 2, 9, 47, 26, 50] and references therein. As a notable example, the moment-of-fluid (MOF) method

* School of Mathematical Sciences, Zhejiang University, 38 Zheda Road, Hangzhou, Zhejiang Province, 310027 China (qinghai@zju.edu.cn)

[12, 1, 13, 2] reconstructs interfaces more accurately than VOF methods, and its reconstruction is somewhat independent from the underlying grid structure.

As the science of multiphase flows moves toward more and more complex phenomena, it demands the resolving of a wider and wider spectrum of time scales and length scales, which, in turn, requires that IT methods be more and more accurate and efficient. In spite of their tremendous successes, the aforementioned methods have a number of disadvantages that hinder their further improvements. For example, in both level-set methods and front-tracking methods there is a lack of intrinsic mechanisms to conserve volume for incompressible flows, and in VOF methods volume conservation sometimes comes at the price of losing one order of accuracy [50]. When the evolution of an interface exhibits topological changes, front-tracking methods often run into difficulties and have to resort to "surgical" operations that are short of theoretical justification. Both VOF methods and level set methods are criticized for failures of shape preservation at C^1 discontinuities: their excessive numerical diffusion tends to smooth an acute angle to be a circular arc and consequently deteriorates the IT accuracy. This accuracy deterioration is even worse in the cases of T-junctions or X-junctions when one deals with three or more phases. Based on IT results of the aforementioned methods, it is also difficult to achieve high accuracy in estimating derived quantities such as curvature and normal vectors of the interface. Indeed, it is shown in [54] that at least half of the accurate digits in the IT results will be lost if curvature is estimated by a second-order method. Coquerelle and Glockner [8] also discussed this issue and showed that the estimation of the curvature of an interface tracked by a low-order method diverges with mesh refinement, even though a convergent curvature estimation method is used.

In a physical domain $\Omega \subset \mathbb{R}^{\mathrm{D}}$, consider tracking a material M passively advected by a velocity field $\mathbf{u}(\mathbf{x}, t)$ that is continuous in time and Lipschitz continuous in space. One common approach to IT is to define a *color function* as

$$f(\mathbf{x}, t) := \begin{cases} 1 & \text{if there is } \mathsf{M} \text{ at } (\mathbf{x},\mathrm{t}), \\ 0 & \text{otherwise} \end{cases} \tag{1.1}$$

and represent the material region occupied by the tracked phase at time t as a moving point set

$$\mathcal{M}(t) := \{\mathbf{x} : f(\mathbf{x}, t) = 1\}. \tag{1.2}$$

Then the advection equation

$$\frac{\partial f}{\partial t} + \mathbf{u} \cdot \nabla f = 0 \tag{1.3}$$

is often considered as the "governing equation" of IT, to which one applies the techniques developed for hyperbolic conservation laws (HCLs). However, a direct application of numerical methods developed for HCLs to IT based on (1.3) is not appropriate for at least two reasons. Theoretically, discontinuities can emerge over time even for smooth initial data in the case of HCLs; in comparison the value of the color function is *always* 0 or 1, and hence strictly speaking (1.3) does not even hold! To numerically treat the interface as linearly degenerate discontinuities of the color function, even first-order accuracy would be extremely challenging to achieve [4]! More discussions on this issue can be found in [50, p. 2825]. In our opinion, a rigorous framework is needed to direct algorithmic design and to analyze the IT accuracy of front-tracking and VOF methods and other hybrid/variant methods.

To answer this need, we proposed such a generic framework called MARS [56] from the viewpoint of Mapping and Adjusting Regular Semianalytic sets. Using MARS, we formally proved the second-order accuracy of VOF methods, clarified many subtle issues such as the accuracy deterioration caused by local C^1 discontinuities, and analyzed other explicit IT methods such as the MOF methods and front-tracking methods. In particular, we have been emphasizing the purely geometric nature of the IT problem, as opposed to converting IT to solving a partial differential equation: *we tackle the geometric problem of IT with tools in geometry and topology.*

In the hope of uniting explicit IT methods, we also proposed a generic IT method called the MARS method [56], which is essentially a function composition of three unitary operations on the space of regular semianalytic sets; see Section 3 for more explanations. It can be shown that VOF methods, MOF methods, front-tracking methods, and many other hybrid methods may all be considered as special cases of the MARS method. In particular, the insights provided by the analytic framework lead to a fourth-order MARS method called the iPAM method [55]. In this method, material regions near the interface are modeled as linear polygons and tracked via characteristic tracing and polygon clipping. As such, it combines the advantageous features of VOF methods and front-tracking methods. As shown by theoretical analysis [56] and numerical experiments [55], the iPAM method has a number of distinguishing features including (i) fourth-order accuracy independent of dynamic C^1 discontinuities, (ii) rigorous volume conservation, (iii) direct applicability to both structured and unstructured grids, (iv) adaptive resolution of the interface, and (v) flexibility of decoupling the resolution of interface from that of the bulk flow.

In the iPAM method, we enforce a relation $h_L = r_h h^\alpha$ between the Eulerian grid size h of the control volumes (cells) and the Lagrangian length scale h_L, which is defined as the maximum distance between adjacent interface markers. The user-defined input parameters r_h and α provide a simple mechanism to deal with the largely varying ratio of h_L to h across different applications. An appropriate choice of r_h and α depends on the problem at hand and the fact that the interface is a set of co-dimension one. When the interface does not play an important role in the physical process, one may choose $\alpha = 1$ and $r_h \approx 1$ so that the IT accuracy is of the second order and the computational cost of IT is much lower than that of resolving the main flow. On the other hand, when the geometric information of an interface is crucial in determining the physics of the multiphase flow, one can choose $\alpha = 2$ and/or $r_h \ll 1$ so that the IT accuracy is of the fourth order and the computational cost of IT is of the same complexity as that of the main flow solver, thus balancing the costs of the two modules within the entire numerical model. Note that a choice of $\alpha > 2$ should be used with caution because even in two dimensions the cost of IT would asymptotically dominate that of the main-flow solver, which may not be appropriate.

Apart from the upper bound h_L, we also enforce a lower bound $r_{\mathsf{tiny}} h_L$ on distances of adjacent markers, for three reasons. First, this lower bound leads to an upper bound on the number of interface markers n_{mks}, which ensures the numerical stability of IT as then n_{mks} cannot grow indefinitely. Second, a number of robustness problems [24] crop up when two adjacent markers get too close to each other; enforcing the lower bound precludes these robustness problems. Last, by setting the user-defined input parameter $r_{\mathsf{tiny}} \approx 0.1$, we force the distances between adjacent interface markers to be roughly uniform, thus promoting the efficiency of marker usages.

Depending on which length scale out of h_L and h is used, the linear iPAM method with $h_L = O(h^2)$ has different convergence rates: it is fourth-order accurate in terms of

h, but only second-order accurate in terms of h_L. Hereafter, we adopt the convention to express convergence rates of an IT method in terms of h, unless explicitly stated otherwise. As the main reason, in reality IT is rarely a standalone problem, but almost always interacts with the main flow solver. Eventually, an IT method will be incorporated into the entire multiphase solver, for which h is a better choice than h_L for characterizing the overall accuracy.

The iPAM method has many advantages due to its solid theoretical ground and the key design that distances between adjacent interface markers be within the interval $[r_{\mathsf{tiny}} h_L, h_L]$. But a number of improvements on the iPAM method are still desirable.

(Q-1) In the iPAM method, interface markers are connected by linear segments and the fourth-order accuracy is obtained by choosing $\alpha = 2$. For a better efficiency, can we choose $\alpha = 1$ and connect interface markers by cubic splines to achieve the same accuracy?

(Q-2) For IT problems without dynamic kinks on the interface, it is already shown in [56] that fourth-, sixth-, and eighth-order accuracy can be achieved by respectively setting $\alpha = 1, \frac{3}{2}, 2$ in the iPAM method and by switching the interface representation from linear segments to cubic splines. However, results of other benchmark problems [56, section 6.2] demonstrate that even a single dynamic kink might deteriorate the convergence rate to no better than 4 for cubic-spline representation with $\alpha = 2$. So can we achieve sixth- and eighth-order accuracy with cubic splines even at the presence of dynamic C^1 discontinuities?

(Q-3) Rigorous volume conservation in the iPAM method is achieved by adjusting polygons via ear removal. However, in tracking multiple phases, a volume increase of one phase unavoidably yields a volume decrease of the other phase across the interface. Consequently, when three or more phases are to be tracked, this volume-adjusting approach may not achieve exact volume conservation for all phases. So can we resolve this issue without sacrificing the IT accuracy?

(Q-4) In the main-flow solver, spatial discretizations near the interface require that material regions of the tracked material be provided inside cells cut by the interface. This entails an algorithm of Boolean operations on *splinegons*, i.e. curvilinear polygon with splines as its boundary. Although a general-purpose splinegon-clipping algorithm is still a research frontier in computational geometry [10, 14], can we design a simple and efficient algorithm of splinegon clipping in the context of IT to facilitate the coupling of high-order IT methods to high-order main flow solvers?

In this paper, we give positive answers to all of the above questions.

Interface representation via splines is not a new idea: it is already adopted both in VOF methods [16, 28, 9, 50] and front-tracking methods [27]. However, the IT accuracy reported therein suggests that the power of splines was not fully unleashed. One possible accuracy deterioration comes from the fact that, when two knots get very close to each other, the linear system for the spline coefficients could be extremely ill-conditioned. Our answer to (Q-1) partially lies in enforcing the lower bound $r_{\mathsf{tiny}} h_L$ on distances of interface markers, as discussed on the previous page.

By Sard's theorem [37, 31], the C^1 discontinuities are a set of measure zero, and hence they should not affect the IT accuracy as the definition of IT errors and convergence rates are based on the 1-norm; see (3.3) in Section 3. The global accuracy deterioration of the tests in [56, section 6.2] is probably caused by the choice of $r_{\mathsf{tiny}} \approx 0.1$, which forces the spline knots to be roughly even distributed. We also believe that this accuracy deterioration would disappear as the grid size approaches

zero. Our answer to (Q-2) is thus extremely simple: we reduce the value of r_{tiny} from 0.1 to 0.01 and observe that the choices of $\alpha = 1, \frac{3}{2}, 2$ lead to convergence rates of 4, 6, and 8, respectively, or at least so before the IT accuracy reaches machine precision for the previous troublesome tests in [56, section 6.2]. Although a theoretical analysis is currently unavailable to prove these high convergence rates, we provide strong heuristic arguments in Section 5.2 to support our answer. Note that the use of the relation $h_L = r_h h^\alpha$ is partially, but not exclusively, the reason for achieving the high rates of convergence; see Sections 5.2.2 and 5.2.3 for more discussions.

Volume conservation has always been a major concern in the IT community, and there exist many VOF methods with excellent volume-conservation properties; see, e.g., [48, 7]. This emphasis is reasonable for applications with very long simulations; otherwise accumulated volume loss in time could empty the domain. When exact volume conservation is not available, one should ensure that the temporal accumulation of the volume conservation error does not hurt the fidelity of the simulation.

Our answers to (Q-3) may be a surprise to the reader: we completely remove the algorithmic steps that are exclusively devoted to volume conservation. As the first reason for this design, volume-conservation error is never greater than the IT error, at least in the 1-norm sense. Hence if the IT of the proposed method is very accurate, so is its volume conservation. Second, although we have given up *exact* volume conservation, we still have *asymptotic* volume conservation; furthermore, the high convergence rates of the proposed method implies fast convergence of volume conservation, as demonstrated by Table 5.3 in Section 5.2.2. Third, the removal of volume-conservation steps, especially the step of intersecting cell preimage to ambient fluid, greatly simplifies the algorithm. Fourth, the removal of polygon clipping from the core algorithm contributes to a better conditioning of the whole algorithm; see Section 5.2.3. Last, the proposed cubic MARS method can be applied without any algorithmic modifications both to incompressible flows and to compressible flows.

Since control volumes are usually simple polygons, we answer (Q-4) by proposing an algorithm in the appendix to clip cubic splinegons with linear polygons. This algorithm is tailor-made for explicit IT so that its algorithmic complexity is optimal.

The iPAM method may be considered as a hybrid method of the VOF method and the front-tracking method. However, the cubic MARS method proposed in this work intrinsically differs from both VOF methods and front-tracking methods. In VOF methods, volume fractions are the major representation of the interface; in our cubic MARS method, however, volume fractions are not needed as they can be derived completely from the explicit material regions. In VOF methods and level set methods, the geometric problem of IT is converted to a PDE problem whereas in the cubic MARS method we tackle the geometric IT problem with tools in geometry and topology. In front-tracking methods, a fluid phase is tracked via its boundary in a way such that it is difficult to retrieve information on the phase topology. In our cubic MARS method, however, we track the regular region occupied by the entire fluid phase, thus maintaining not only geometric but also topological information of the tracked phase. This provides a solid foundation for rigorous treatment of topological changes in the future. Due to the above reasons, we consider the cubic method as neither a VOF method nor a front-tracking method, but rather a new method in its own right. The name "cubic MARS method" refers to the generic MARS framework and emphasizes our philosophy to treat geometric and topological problems with geometry and topology.

The rest of this paper is organized as follows. In Section 2, we propose a metric

space called the Yin space as a mathematical model of physically meaningful material regions. Then in Section 3, we formulate the IT problem and the MARS method based on the actions of flow maps on the Yin space. In Section 4, we explain the algorithm of the cubic MARS method after a brief introduction to the iPAM method, with the key component of splinegon clipping detailed in the appendix. In Section 5, we perform benchmark tests and derive efficiency-comparing formulas to demonstrate that the cubic MARS method is superior to existing IT methods in terms of both accuracy and efficiency. Even for two of the most difficult IT tests, the cubic MARS method achieves accuracy close to machine precision on a 128-by-128 Eulerian grid! Finally in Section 6 we conclude this paper with a number of research prospects.

2. Modeling Physically Meaningful Material Regions. In this section, we collect concepts and results from various branches of mathematics to propose a model of physically meaningful material regions. This modeling space is appropriate in that
(a) each element is describable by a *finite* sequence of symbol structures,
(b) the space is closed under relevant procedures,
(c) the space is amenable to numerical analysis of IT,
(d) the space furnishes a solid theoretical foundation for future treatments of topological changes.

Condition (a) helps the design of data structures. For example, each Jordan curve is described by a finite number of polynomial functions and each physically meaningful material region is described by a finite number of oriented Jordan curves. In this work, the relevant procedures in condition (b) are Boolean operations and the regularized topological operations $^{\perp}$, $\cap$, and $\cup^{\perp\perp}$, as defined in Section 2.2. The word "amenable" in Condition (c) is a very loaded word. To make a long story short, this amenability condition requires that, in the context of IT, the modeling sets of physically meaningful material regions be *open*. For more details, the reader is referred to the theory of donating regions [51, 52] and its application to the analysis of VOF methods [50].

Hereafter we arrange elementary definitions and well-known theorems in a way so that they are self-contained and afford a solid footing for subsequent sections.

2.1. Boolean algebra. An *algebra* is a set $\mathcal{A}$ equipped with a number of *operations*, i.e. functions $\mathcal{A}^n \to \mathcal{A}$ where n is a finite integer. An operation with $n = 1, 2$ is *unitary* and *binary*, respectively. The set $\mathcal{A}$ is called the *universe* of the algebra; an algebra is *finite* if its universe is a finite set.

DEFINITION 2.1 (Boolean algebra). *A* Boolean algebra *is an algebra with universe* $\mathbb{B}$*, two binary operations generically called* join $\vee$ *and* meet $\wedge$*, a unary operation* $'$ *generically called* complementation*, and two distinguished members* $\hat{0}$ *and* $\hat{1}$ *in* $\mathbb{B}$*, such that for all* $x, y, z \in \mathbb{B}$*,*
(BA-1) $x \wedge \hat{1} = x$ *and* $x \vee \hat{0} = x$*,*
(BA-2) $x \wedge x' = \hat{0}$ *and* $x \vee x' = \hat{1}$*,*
(BA-3) $x \vee y = y \vee x$ *and* $x \wedge y = y \wedge x$*,*
(BA-4) $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ *and* $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$*.*
We denote such a Boolean algebra by

$$\mathbf{B} := (\mathbb{B},\ \vee,\ \wedge,\ ',\ \hat{0},\ \hat{1}). \tag{2.1}$$

Other definitions of Boolean algebra contain more axioms such as the DeMorgan's laws; see [5, p. 116] and [17, p. 10] for two examples. Huntington [22] showed that the

axiomatic laws in Definition 2.1 form a minimal set of axioms satisfied by a Boolean algebra.

The *symmetric difference* or *exclusive disjunction* "$\oplus$" is defined as

$$\forall x, y \in \mathbb{B}, \qquad x \oplus y := (x \wedge y') \vee (x' \wedge y). \tag{2.2}$$

2.2. The topological space of regular sets. The standard notation "$\cup$," "$\cap$," and "$\setminus$" are used to denote the conventional set-theoretic binary operations of union, intersection, and difference, respectively.

A *topological space* is a set $\mathcal{X}$ together with $\mathfrak{T}$, a collection of subsets of $\mathcal{X}$ called *open set*s such that $\emptyset$ and $\mathcal{X}$ are open and such that arbitrary unions and finite intersections of open sets are open. Then $\mathfrak{T}$ is a *topology* on $\mathcal{X}$. As a common example, the standard topology on the Euclidean space $\mathbb{R}^{\mathrm{D}}$ is determined as follows: a point set $\mathcal{P}$ is *open* if for each point $\mathbf{x} \in \mathcal{P}$ there exists an open ball centered at $\mathbf{x}$ such that this open ball is entirely contained in $\mathcal{P}$.

In a topological space $\mathcal{X}$, the *complement* of a subset $\mathcal{P} \subseteq \mathcal{X}$, written $\mathcal{P}'$, is the set $\mathcal{X} \setminus \mathcal{P}$. A subset of $\mathcal{X}$ is *closed* if its complement is open. The *closure* of a set $\mathcal{P} \subseteq \mathcal{X}$, written $\mathcal{P}^-$, is the intersection of all closed supersets of $\mathcal{P}$. The *interior* of $\mathcal{P}$, written $\mathcal{P}^\circ$, is the union of all open subsets of $\mathcal{P}$. The *exterior* of $\mathcal{P}$, written $\mathcal{P}^\perp := \mathcal{P}'^\circ := (\mathcal{P}')^\circ$, is the interior of its complement. By the identity $\mathcal{P}^- = \mathcal{P}'^{\circ\prime}$ [17, p. 58], we have $\mathcal{P}^\perp = \mathcal{P}^{-\prime}$. A point $\mathbf{x} \in \mathcal{X}$ is a *boundary point* of $\mathcal{P}$ if $\mathbf{x} \notin \mathcal{P}^\circ$ and $\mathbf{x} \notin \mathcal{P}^\perp$. The *boundary* of $\mathcal{P}$, written $\partial\mathcal{P}$, is the set of all boundary points of $\mathcal{P}$. It can be shown that $\mathcal{P}^\circ = \mathcal{P} \setminus \partial\mathcal{P}$ and $\mathcal{P}^- = \mathcal{P} \cup \partial\mathcal{P}$.

An open set $\mathcal{P} \subseteq \mathcal{X}$ is *regular* if it coincides with the interior of its own closure, i.e. if $\mathcal{P} = \mathcal{P}^{-\circ}$. A closed set $\mathcal{P} \subseteq \mathcal{X}$ is *regular* if it coincides with the closure of its own interior, i.e. if $\mathcal{P} = \mathcal{P}^{\circ-}$. By the duality of the interior and closure operators, $\mathcal{P}^\circ = \mathcal{P}'^{-\prime}$, and hence $\mathcal{P}$ is a *regular open set* if and only if $\mathcal{P} = \mathcal{P}^{\perp\perp} := \left(\mathcal{P}^\perp\right)^\perp$. For any subset $\mathcal{Q} \subseteq \mathcal{X}$, $\mathcal{Q}^{\perp\perp}$ is a regular open set[1] and $\mathcal{Q}^{\circ-}$ is a regular closed set[2].

THEOREM 2.2 (MacNeille [30] and Tarski [43]). *Let $\mathbb{B}$ denote the class of all regular open sets of a topological space $\mathcal{X}$ and define $\mathcal{P} \cup^{\perp\perp} \mathcal{Q} := (\mathcal{P} \cup \mathcal{Q})^{\perp\perp}$ for all $\mathcal{P}, \mathcal{Q} \subseteq \mathcal{X}$. Then $\mathbf{B}_o := (\mathbb{B},\ \cup^{\perp\perp},\ \cap,\ ^\perp,\ \emptyset,\ \mathcal{X})$ is a Boolean algebra.*

Proof. See [17, section 10]. □

Similarly, it can be shown that, with appropriately defined operations, regular closed sets of a topological space $\mathcal{X}$ also form a Boolean algebra [25, p. 39].

Regular sets, open or closed, capture the salient feature that physically meaningful material regions are free of lower-dimensional elements such as isolated points and curves in two dimensions and dangling faces in three dimensions. However, they are not yet perfect for representing physically meaningful regions: some of them cannot be described by a finite number of symbol structures. For example, some sets have nowhere differentiable boundaries, which, in their parametric forms, are usually infinite series of continuous functions [36]. Another pathological case is more subtle: intersecting two regular sets with piecewise smooth boundaries may yield an infinite

[1] Since closure preserves inclusions and complementation reverses them [25], $\mathcal{P} \subseteq \mathcal{Q} \Rightarrow \mathcal{Q}^\perp \subseteq \mathcal{P}^\perp$. If $\mathcal{P}$ is open, $\mathcal{P} \subseteq \mathcal{P}^- \Rightarrow \mathcal{P}^\perp \subseteq \mathcal{P}' \Rightarrow \mathcal{P}^{\perp-} \subseteq \mathcal{P}'^- = \mathcal{P}' \Rightarrow \mathcal{P} \subseteq \mathcal{P}^{\perp\perp}$. Replacing $\mathcal{P}$ with $\mathcal{P}^\perp$ yields $\mathcal{P}^\perp \subseteq \mathcal{P}^{\perp\perp\perp}$; on the other hand, setting $\mathcal{Q} = \mathcal{P}^{\perp\perp}$ yields $\mathcal{P}^{\perp\perp\perp} \subseteq \mathcal{P}^\perp$. Therefore $\mathcal{P}^\perp = \mathcal{P}^{\perp\perp\perp}$ if $\mathcal{P}$ is open. For any $\mathcal{P}$, $\mathcal{P}^\perp$ is open, and hence $\mathcal{P}^{\perp\perp} = \left(\mathcal{P}^{\perp\perp}\right)^{\perp\perp}$ always holds. The above argument justifies $(\cdot)^{\perp\perp}$ as the regularization operator that always yields a regular open set.

[2] $\mathcal{P}^\circ \subseteq \mathcal{P} \Rightarrow \mathcal{P}^{\circ-} \subseteq \mathcal{P}^-$. Setting $\mathcal{P} = \mathcal{Q}^{\circ-}$ yields $\mathcal{Q}^{\circ-\circ-} \subseteq \mathcal{Q}^{\circ--} = \mathcal{Q}^{\circ-}$. As for the other direction, $\mathcal{Q}^\circ \subseteq \mathcal{Q}^{\circ-}$ implies $\mathcal{Q}^\circ \subseteq \mathcal{Q}^{\circ-\circ}$ since $\mathcal{Q}^\circ$ is an open subset of $\mathcal{Q}^{\circ-}$ and $\mathcal{Q}^{\circ-\circ}$ is the largest open subset of $\mathcal{Q}^{\circ-}$. Then $\mathcal{Q}^{\circ-} \subseteq \mathcal{Q}^{\circ-\circ-}$.

number of disjoint regular sets. For example, consider

$$\begin{cases} \mathcal{A}_p := \{(x,y) \in \mathbb{R}^2 : -2 < y < \sin\frac{1}{x},\ 0 < x < 1\}, \\ \mathcal{A}_s := \{(x,y) \in \mathbb{R}^2 : 0 < y < 1,\ -1 < x < 1\}. \end{cases} \tag{2.3}$$

Although both $\mathcal{A}_p$ and $\mathcal{A}_s$ are described by two inequalities, their intersection is a disjoint union of an infinite number of regular sets; see [33, Figure 4-1, Figure 4-2]. This poses a fundamental problem that Boolean operations of two regular sets may yield results that are not representable by a finite number of entities.

Therefore, we need to find a proper subspace of regular sets, each element of which is finitely describable. This search eventually[3] leads to the consideration of semianalytic sets.

2.3. Semianalytic sets. Semianalytic and semialgebraic sets are fundamental concepts in algebraic geometry and they originated more than six decades ago in the context of decision methods for elementary algebra and geometry [44, 39].

DEFINITION 2.3. *A set $\mathcal{S} \subseteq \mathbb{R}^{\mathrm{D}}$ is* semianalytic *if there exist a finite number of analytic functions $g_i : \mathbb{R}^{\mathrm{D}} \to \mathbb{R}$ such that $\mathcal{S}$ is in the universe of a finite Boolean algebra formed from the sets*

$$\mathcal{X}_i = \left\{\mathbf{x} \in \mathbb{R}^{\mathrm{D}} : g_i(\mathbf{x}) \geq 0\right\}. \tag{2.4}$$

The g_i's are called the generating functions *of $\mathcal{S}$. In particular, a semianalytic set is* semialgebraic *if all of its generating functions are polynomials.*

Recall that a function is *analytic* if and only if its Taylor series at $\mathbf{x}_0$ converges to the function in some neighborhood for every $\mathbf{x}_0$ in its domain. In the example of (2.3), $\mathcal{A}_s$ is semianalytic while $\mathcal{A}_p$ is not, because the Taylor series of $g_2(x,y) := \sin\frac{1}{x} - y$ at the origin does not converge. In contrast, for a regular open set $\mathcal{S}$ that is also semianalytic, we can be sure that for each $\mathbf{x} \in \mathcal{S}$ and any sufficiently small $r > 0$, the local neighborhood $\mathcal{S} \cap \mathcal{B}_r(\mathbf{x})$ can be described by a finite set of "well-behaved" functions.

By definition, semianalytic sets are closed under set complementation, finite union, and finite intersection. Consequently, all algebraic and analytic varieties are semianalytic sets because

$$\left\{\mathbf{x} \in \mathbb{R}^{\mathrm{D}} : g(\mathbf{x}) = 0\right\} = \left\{\mathbf{x} \in \mathbb{R}^{\mathrm{D}} : g(\mathbf{x}) \geq 0\right\} \cap \left\{\mathbf{x} \in \mathbb{R}^{\mathrm{D}} : -g(\mathbf{x}) \geq 0\right\}. \tag{2.5}$$

Also, one can replace "$\geq$" by "$>$" in Definition 2.3 since

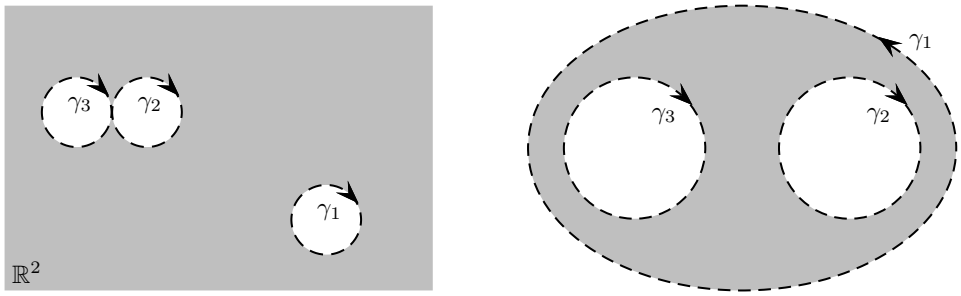
$$\left\{\mathbf{x} \in \mathbb{R}^{\mathrm{D}} : g(\mathbf{x}) > 0\right\} = \mathbb{R}^{\mathrm{D}} \setminus \left\{\mathbf{x} \in \mathbb{R}^{\mathrm{D}} : -g(\mathbf{x}) \geq 0\right\}. \tag{2.6}$$

Therefore, semianalytic sets are also closed under topological operations. More precisely, if $\mathcal{S} \subset \mathbb{R}^{\mathrm{D}}$ is semianalytic, so are $\mathcal{S}^\circ$ and $\mathcal{S}^-$.

2.4. $\mathbb{Y}$: Yin sets for modeling physically meaningful regions. In our previous work [56], physically meaningful regions are modeled by bounded regular open semianalytic sets with the Euclidean topology. We emphasize that these regular sets must be open so that they are amenable to numerical analysis based on donating regions [51, 50]. If the boundedness condition is imposed not on the set itself, but on its boundary, we obtain the concept of Yin sets, which will be used in this work.

DEFINITION 2.4. *A* Yin set *$\mathcal{Y} \subseteq \mathbb{R}^2$ is a regular open semianalytic set whose boundary is bounded. The class of all such Yin sets is called the* Yin space *$\mathbb{Y}$.*

[3] The exclusion of nonphysical regular sets involves the triangulation of a topological space into a simplicial complex and imposing certain conditions on the embedding of the resulting topological polyhedron in three or higher dimensions; see [33, section 4.2] for an accessible exposition.



(a) an unbounded Yin set (b) a bounded Yin set

Fig. 2.1: Jordan curve representation of a connected Yin set $\mathcal{M} \subset \mathbb{R}^2$. A shaded region represents $\mathcal{M}$ and the dashed curves its boundary $\partial\mathcal{M}$ that consists of a number of Jordan curves. The representation only entails choosing orientations of these Jordan curves so that in both (a) and (b) we have $\mathcal{M} = \text{int}(\gamma_1) \cap \text{int}(\gamma_2) \cap \text{int}(\gamma_3)$. In particular, $\partial\mathcal{M}$ in subplot (a) is not a manifold because γ_2 and γ_3 have an improper intersection. Note that an isolated intersection point q of two curves is a *proper intersection* if at a sufficiently small local neighborhood of q one curve lies at both sides of the other curve; otherwise it is an *improper intersection* [51, Definition 4.1].

By Theorem 2.2 and Definition 2.3, regular open semianalytic/semialgebraic sets form a Boolean algebra since they are the intersection of the universes of two Boolean algebras. Furthermore, Boolean operations on Yin sets preserve the boundedness of a bounded boundary, hence we have the following.

THEOREM 2.5. *The algebra* $\mathbf{Y} := (\mathbb{Y},\ \cup^{\perp\perp},\ \cap,\ ^{\perp},\ \emptyset,\ \mathbb{R}^2)$ *is a Boolean algebra.*

Recall from Section 2.2 that the exterior of $\mathcal{P}$, $\mathcal{P}^{\perp} := (\mathcal{P}')^{\circ}$, is the interior of its complement and we define $\mathcal{P} \cup^{\perp\perp} \mathcal{Q} := (\mathcal{P} \cup \mathcal{Q})^{\perp\perp}$.

We employ Yin sets instead of bounded regular open semianalytic sets for two reasons. First, Yin sets are slightly more general. Second, by Definition 2.1, the two distinguished members $\hat{0}$ and $\hat{1}$ of the Boolean algebra of Yin sets are simply $\emptyset$ and $\mathbb{R}^2$, respectively. This is not true if we restrict the regular sets to be bounded.

The *interior* of a Jordan curve γ_i, denoted by $\text{int}(\gamma_i)$, is the complement of γ_i that always lies at the left of an observer who traverses γ_i according to its orientation. It can be shown that a connected Yin set $\mathcal{M}$ can *always* be expressed as

$$\mathcal{M} = \text{int}(\gamma_1) \cap \text{int}(\gamma_2) \cap \cdots \cap \text{int}(\gamma_\ell), \tag{2.7}$$

where $\{\gamma_i : i = 1, \ldots, \ell\}$ is a set of oriented Jordan curves that are *pairwise almost disjoint*, i.e. they do not have proper intersections and the number of their improper intersections is finite; see Figure 2.1 for two examples. Consequently, *any* Yin set $\mathcal{M}$ has the representation

$$\mathcal{M} = \cup_j \cap_i \text{int}\left(\gamma_{j,i}\right), \tag{2.8}$$

where j is the index of connected components of $\mathcal{M}$. The significance of (2.7) and (2.8) is that a Yin set can be tracked by tracking its boundary, which justifies IT.

We define the *volume of a Yin set* $\mathcal{S} \in \mathbb{Y}$ as

$$\|\mathcal{S}\| := \int_{\mathcal{S}} \mathrm{d}\mathbf{x}, \tag{2.9}$$

where the integral is to be interpreted in the sense of Riemann integrals. If $\mathcal{S}$ is bounded, the integral in (2.9) clearly exists due to $\mathcal{S}$ being regular; then $\|\mathcal{S}\|$ is a well-defined nonnegative number. If $\mathcal{S}$ is unbounded, we define $\|\mathcal{S}\| = +\infty$.

A metric on the Yin space $d : \mathbb{Y} \times \mathbb{Y} \to \mathbb{R}$ can be defined as

$$d(\mathcal{S}, \mathcal{P}) := \|\mathcal{S} \oplus \mathcal{P}\|, \tag{2.10}$$

which satisfies nonnegativity $d(\mathcal{S}, \mathcal{P}) \geq 0$, identity of indiscernibles $d(\mathcal{S}, \mathcal{S}) = 0$, symmetry $d(\mathcal{S}, \mathcal{P}) = d(\mathcal{P}, \mathcal{S})$, and triangle inequality $d(\mathcal{S}, \mathcal{P}) \leq d(\mathcal{S}, \mathcal{T}) + d(\mathcal{T}, \mathcal{P})$.

3. Formulating the IT problem and the MARS method. In the IT problem, we are usually given *a priori* a velocity field $\mathbf{u}(\mathbf{x}, t)$, by which each fluid phase is passively advected. The ordinary differential equation

$$\frac{\mathrm{d}\,\mathbf{x}}{\mathrm{d}\,t} = \mathbf{u}(\mathbf{x}, t) \tag{3.1}$$

admits a unique solution for any given initial time t_0 and initial position $p_0 \in \mathbb{R}^{\mathrm{D}}$ if the time-dependent velocity field $\mathbf{u}(\mathbf{x}, t)$ is continuous in time and Lipschitz continuous in space. This uniqueness gives rise to a flow map $\phi : \mathbb{R}^{\mathrm{D}} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}^{\mathrm{D}}$ that takes the initial position p_0 of a Lagrangian particle p, the initial time t_0, and the time increment $\pm k$ and returns $p(t_0 \pm k)$, the position of p at the final time $t_0 \pm k$:

$$\begin{cases} \phi_{t_0}^{+k}(p) := p(t_0 + k) = p(t_0) + \int_{t_0}^{t_0+k} \mathbf{u}(p(t), t)\,\mathrm{d}\,t, \\ \phi_{t_0}^{-k}(p) := p(t_0 - k) = p(t_0) + \int_{t_0}^{t_0-k} \mathbf{u}(p(t), t)\,\mathrm{d}\,t. \end{cases} \tag{3.2}$$

The flow map also generalizes to arbitrary point sets in a straightforward way,

$$\phi_{t_0}^{\pm k}(\mathcal{M}) = \{\phi_{t_0}^{\pm k}(p) : p \in \mathcal{M}\}.$$

We will also use the shorthand notation $\overleftarrow{M} := \phi_{t_0+k}^{-k}(M)$ and $\overrightarrow{M} := \phi_{t_0}^{+k}(M)$ when t_0 and k are clear from the context. If we further restrict the above point set as a Yin set, then the flow map ϕ for given t_0 and k can be considered as a unitary operation $\phi_{t_0}^{k} : \mathbb{Y} \to \mathbb{Y}$. This viewpoint yields the formulation of IT as follows.

Definition 3.1 (The IT problem). *Model the region occupied by the tracked material M at time t by a Yin set $\mathcal{M}(t) \in \mathbb{Y}$. The* IT problem *is the determination of $\mathcal{M}(T)$ from the initial condition $\mathcal{M}(t_0)$ and the given velocity field $\mathbf{u}(\mathbf{x}, t)$, $t \in [t_0, T]$.*

To measure the IT accuracy, we utilize the metric (2.10) to define [12, 2, 50]

$$E_1(t_n) := \|\mathcal{M}(t_n) \oplus \mathcal{M}^n\| = \sum_{\mathcal{C} \subset \Omega} \|\mathcal{M}_{\mathcal{C}}(t_n) \oplus \mathcal{M}_{\mathcal{C}}^n\|\,, \tag{3.3}$$

where $\mathcal{M}(t_n)$ is the exact material region at t_n, $\mathcal{M}^n$ is an approximation of $\mathcal{M}(t_n)$, and the control volumes $\mathcal{C}$'s constitute a partition of the computational domain Ω.

Definition 3.2 (Accuracy of an IT method). *In numerically solving the IT problem posed in Definition 3.1, an IT method $\mathcal{L}_{\mathrm{IT}}$,*

$$\mathcal{M}^{n+1} = \mathcal{L}_{\mathrm{IT}}\big(\mathcal{M}(t_n), \mathbf{u}\big) \approx \mathcal{M}(t_{n+1}), \tag{3.4}$$

is of the βth-order accuracy (in the 1-norm) if $E_1(T) = O(k^\beta)$ as the time step size k approaches zero. $\mathcal{L}_{\mathrm{IT}}$ is said to be consistent *if $\beta > 0$.*

As a generic explicit IT method, the MARS method [56] is simply a concatenation of three unitary operations on the Yin space $\mathbb{Y}$.

DEFINITION 3.3. *A* MARS method *is an IT method of the form*

$$\mathcal{M}^{n+1} = \mathfrak{L}^n_{\text{Mars}}\mathcal{M}^n := \left(\chi_{n+1} \circ \varphi^k_{t_n} \circ \psi_n\right) \mathcal{M}^n, \tag{3.5}$$

where $\mathcal{M}(t_n) \in \mathbb{Y}$ *is approximated by* $\mathcal{M}^n \in \mathbb{Y}$, $\varphi : \mathbb{Y} \to \mathbb{Y}$ *is a* mapping operation *that approximates the exact flow map* ϕ *both in time and in space by sending one semialgebraic set at* t_n *to another at* $t_n + k$, $\psi_n : \mathbb{Y} \to \mathbb{Y}$ *is an* augmentation operation *at* t_n *to prepare* $\mathcal{M}^n$ *for the mapping, and* $\chi_{n+1} : \mathbb{Y} \to \mathbb{Y}$ *is an* adjustment operation *after the mapping to fulfill some representation invariants of* $\mathcal{M}^{n+1}$.

It is shown in [56, section 4.4] that the IT error of a MARS method is bounded as

$$E_1(t_n) \le E^{\text{ODE}} + E^{\text{MAP}} + E^{\text{REP}} + E^{\text{AUG}} + E^{\text{ADJ}}, \tag{3.6}$$

where E^{ODE} and E^{MAP} are the errors, respectively, associated with the temporal and spatial approximation of the exact flow map ϕ, E^{REP} is the error caused by representing the material region with a semialgebraic set at the initial time, E^{AUG} is the accumulated error of augmenting the semialgebraic sets, and E^{ADJ} is the accumulated error of adjusting the mapped semialgebraic sets.

The theory and formulation in Sections 2 and 3 hold for both two and three dimensions, but hereafter we focus only on two dimensions.

4. Algorithms. We first review the iPAM method and then proceed to the explanation of the cubic MARS method; it is informative to compare these two methods.

4.1. The iPAM method. Partition the domain Ω into a collection of fixed control volumes (cells). For ease of exposition we assume that the cells are obtained from structured grids with uniform grid size h. At a time instant t, a cell is called a *pure cell* if $\mathcal{M}(t) \cap \mathcal{C}$ is either the empty set $\emptyset$ or $\mathcal{C}$; otherwise it is an *interface cell*. In both the PAM method and the iPAM method, the type of a cell is labeled by an integer while the material region of an interface cell is further represented by a set of polygons; see [57, Figure 1] for an illustration.

DEFINITION 4.1 (The iPAM method [55]). *Let* $[r_{\mathsf{tiny}} h_L, h_L]$ *be given as the range of allowed distances between adjacent interface markers. Within each time step* $[t_n, t_n + k]$, the iPAM method *advances* $\mathcal{M}^n$ *to* $\mathcal{M}^{n+1}$ *by applying six substeps to each* interface candidate *cell that might become an interface cell at* $t_n + k$*:*

(iPAM-1) Calculate the cell preimage $\varphi^{-k}_{t_n+k}(\mathcal{C})$*, where* $\varphi^{-k}_{t_n+k}$ *is a numerical approximation of the exact flow map* $\phi^{-k}_{t_n+k}$.

(iPAM-2) Compute the intersection $\varphi^{-k}_{t_n+k}(\mathcal{C}) \cap \mathcal{M}^n$.

(iPAM-3) Trace the intersection forward to obtain its image $\mathcal{M}^{n+1} \cap \mathcal{C}$.

(iPAM-4) If $\|p_i - p_{i+1}\|_2$*, i.e. the length of an interface edge* $\overline{p_i p_{i+1}}$*, is greater than* h_L*, divide* $\varphi^{-k}_{t_n+k}(\overline{p_i p_{i+1}})$ *into* $\left\lceil \frac{\|p_i - p_{i+1}\|_2}{h_L} \right\rceil$ *equilength subedges by inserting new markers and add images of new markers in between* p_i *and* p_{i+1}*. Repeat until no interface edge is longer than* h_L.

(iPAM-5) If an interface edge $\overline{p_j p_{j+1}}$ *has its length smaller than* $r_{\mathsf{tiny}} h_L$*, replace this edge with its midpoint* $\frac{p_j + p_{j+1}}{2}$.

(iPAM-6) (Optional.) If $\mathbf{u}$ *is divergence-free, attempt to enforce volume conservation by adjusting the volume of* $\mathcal{M}^{n+1} \cap \mathcal{C}$ *to that of its preimage in (iPAM-2), using the VAPER algorithm described in [55, section 4.2].*

The steps (iPAM-1,2,3) and (iPAM-4,5) are illustrated in [57, Figure 4(a)–(c)] and [55, Figure 6], respectively. The substep (iPAM-4) enforces an upper bound of

the distance between two adjacent markers by subdividing long interface edges with newly added markers. In comparison, (iPAM-5) enforces a lower bound by replacing edges of negligible lengths with their midpoints, which is equivalent to imposing an upper bound on the maximum number of interface markers.

In Definition 4.1, (iPAM-1,2,4) can be considered as an augmentation operation for inserting additional vertices into the interface at the beginning of the time step. In (iPAM-3), the discrete flow map acts on the material regions. (iPAM-5) and the VAPER algorithm in (iPAM-6) can be regarded as an adjustment operation. Therefore, the iPAM method is a MARS method.

PROPOSITION 4.2 (Convergence of the iPAM method [56]). *The iPAM method is (2α)th-order accurate in the 1-norm (3.3) if*

(a) its discrete flow map φ is at least (2α)th-order accurate in temporal integration;
(b) the time step size satisfies $k = O(h)$;
(c) $h_L = r_h h^\alpha$ is fulfilled with the constants $r_h = O(1)$ and $\alpha \geq 1$.

Proof. The conclusion follows from (3.6) and a numerical analysis on each of the individual IT errors; see [56, section 4.4] for the details. □

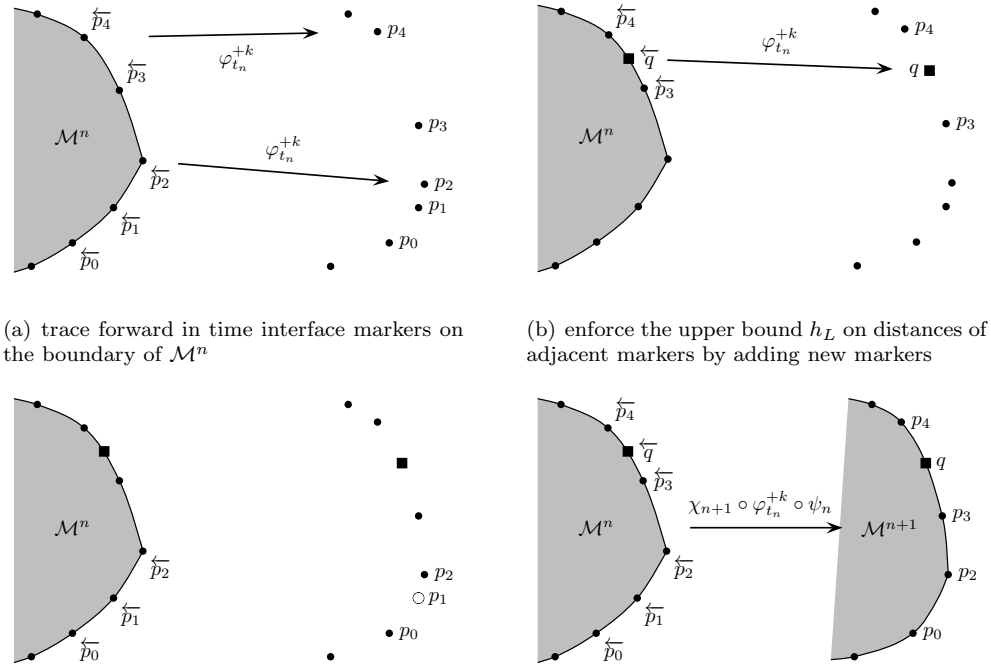
4.2. The cubic MARS method. To further improve the accuracy and efficiency of the iPAM method, we approximate the interface by cubic splines. Conceptually this is a straightforward idea, but the practical implementation is very subtle. For example, local volume conservation in step (iPAM-6) becomes difficult as the VAPER algorithm does not easily generalize to curvilinear polygons.

Fortunately, the metric (2.10) implies that the error of volume conservation is bounded by that of IT. Hence if the interface can be tracked with very high accuracy, we do not have to devote any algorithmic steps exclusively to volume conservation.

In the iPAM method, volume conservation comes at the cost of tracing back the cell boundary in (iPAM-1), computing polygon intersection in (iPAM-2), and tracing forward the intersection in (iPAM-3), as detailed in Definition 4.1. These steps were indispensable because otherwise one would not know which targeting volume the material region should be adjusted to. However, if volume conservation does not need to be explicitly enforced, these three steps (iPAM-1,2,3) can be replaced by a single step, in which interface markers are simply advected forward in time.

DEFINITION 4.3 (The cubic MARS method for a single phase). *Let $[r_{\mathsf{tiny}} h_L,\, h_L]$ be given as the range of allowed distances between adjacent interface markers. Within each time step $[t_n, t_n + k]$,* the cubic MARS method *takes as its input a set of cubic splines representing $\partial\mathcal{M}(t_n)$, advances it to $\partial\mathcal{M}^{n+1}$, and computes $\mathcal{M}^{n+1} \cap \mathcal{C}$ for each interface cell $\mathcal{C}$:*

(CubiMARS-1) Trace forward in time knots of $\partial\mathcal{M}^n$ to a sequence of points $\{p_j\}$ at time $t_n + k$.
(CubiMARS-2) If any chordal length $\|p_j - p_{j+1}\|_2$ is greater than h_L,
  (a) locate $\overleftarrow{p_j} = (x(s_j), y(s_j))$ and $\overleftarrow{p_{j+1}} = (x(s_{j+1}), y(s_{j+1}))$ on $\partial\mathcal{M}^n(s)$ as the preimages of p_j and p_{j+1},
  (b) divide the interval $[s_j, s_{j+1}]$ into $\left\lceil \frac{\|p_j - p_{j+1}\|_2}{h_L} \right\rceil$ equilength subintervals, compute the corresponding new markers on $\partial\mathcal{M}^n(s)$,
  (c) advect the new markers by the flow map,
  (d) insert the advected new markers in between p_j and p_{j+1}.
  Repeat the above substeps until no chordal length is greater than h_L.
(CubiMARS-3) For any chordal length $\|p_j - p_{j+1}\|_2$ smaller than $r_{\mathsf{tiny}} h_L$, remove p_j or p_{j+1} from the point sequence. If the preimage of p_j (or p_{j+1}) is



(a) trace forward in time interface markers on the boundary of $\mathcal{M}^n$

(b) enforce the upper bound h_L on distances of adjacent markers by adding new markers

(c) enforce the lower bound $r_{\mathsf{tiny}} h_L$ on distances of adjacent markers by removing markers

(d) fit a new spline as the boundary of $\mathcal{M}^{n+1}$

Fig. 4.1: The first four steps of the cubic MARS method. In subplot (a), the interface markers on $\partial\mathcal{M}^n$ are mapped by the discrete flow map $\varphi_{t_0}^{+k}$ to their images. In subplot (b), the distance between p_3 and p_4 is found to be larger than the upper bound h_L, and a new marker (the solid square) is added so that distances between both p_3, q and q, p_4 are smaller than h_L. In subplot (c), the distance between p_1 and p_2 is found to be smaller than the lower bound $r_{\mathsf{tiny}} h_L$, and the marker p_1 (the hollow circle) is removed because the preimage of p_2 is a C^1 discontinuity on $\partial\mathcal{M}^n$. In subplot (d), we obtain $\partial\mathcal{M}^{n+1}$ by fitting a new spline through the new chain of markers "$\cdots \to p_0 \to p_2 \to p_3 \to q \to p_4 \to \cdots$" with p_2 as a knot with only C^0 continuity. According to Definition 3.3, these four steps of the cubic MARS method constitute a MARS method: ψ_n is the operation that adds new markers on $\partial\mathcal{M}^n$ so that the upper bound is enforced, and χ_{n+1} the operation that remove marker images on $\partial\mathcal{M}^{n+1}$ so that the lower bound is enforced.

a C^0-knot of $\partial\mathcal{M}^n$, remove p_{j+1} (or p_j); otherwise remove the point that forms a smaller triangle with its two adjacent markers.

(CubiMARS-4) *Construct a new cubic spline from the updated marker sequence $\{p_j\}$ as the representation of $\partial\mathcal{M}^{n+1}$.*

(CubiMARS-5) *(Optional.) Compute $\mathcal{M}^{n+1} \cap \mathcal{C}$ for each interface cell $\mathcal{C}$ by the spline-gon clipping algorithm explained in the appendix.*

As illustrated in Figure 4.1, the first four steps of the proposed method constitute a MARS method, hence justifying the name of the new method. In comparison to the iPAM method, step (CubiMARS-1) replaces steps (iPAM-1,2,3) in the

iPAM method. Step (CubiMARS-2) is essentially the same as step (iPAM-4), except that here new markers have to be located and inserted on the *spline* instead of a linear edge. Step (CubiMARS-3) precludes ill-conditioning of spline fitting and prevents a number of robustness problems in computational geometry [24]. Similar to (iPAM-5), step (CubiMARS-3) enforces a lower bound of distances between adjacent markers. However, in (iPAM-5) a short edge is always replaced by its midpoint while in (CubiMARS-3) one of the endpoints is removed, leaving the other one intact. This change stems from properly handling C^1 discontinuities on the interface: if a C^1 discontinuity is known *a priori*, only C^0 continuity should be enforced at this knot when constructing the spline. As far as spline construction is concerned, the averaging procedure in (iPAM-5) may spuriously change the location of a C^1 discontinuity, resulting in an *unnecessary* loss of accuracy. Hence (CubiMARS-3) is more appropriate than (iPAM-5) in representing the interface with splines; see Section 5.1.

The last step does not change results of the cubic MARS method; it simply generates a *local* representation of the solution from the *global* solution; see the appendix for details of the algorithm. There are two main reasons for having the local solution. First, it makes the cubic MARS method amenable to be coupled to high-order finite-volume methods as material regions inside control volumes must be available in discretizing spatial operators. Second, this design facilitates accurate and efficient treatments of topological changes since such an algorithm should be based on the *local* material regions instead of the *global* chain of spline knots.

It is also informative to note that, during the process of a topological change, there must exist a crucial time instant where the boundary of the phase is not a manifold; see Figure 2.1 (a) for an example. This is another reason why we propose Yin sets as a model of physically meaningful material regions: a manifold with boundary is not general enough to handle topological changes. In a future paper we will report our algorithms of dealing with topological changes via Yin sets.

5. Tests. In this section, we numerically solve several standard benchmark tests by the proposed cubic MARS method, compare its results to those of the iPAM methods and other representative IT methods, and demonstrate that the cubic MARS method can be much more accurate and/or efficient than the (linear and cubic) iPAM methods and other existing methods.

For fourth-, sixth-, and eighth-order pathline tracing, we use the classic fourth-order Runge–Kutta method, the explicit method by Verner [46], and that by Dormand and Prince [11], respectively. These methods are chosen solely based on ease of implementation, and it would not affect the performance of the cubic MARS method if one uses another explicit time integrator with the same accuracy.

As discussed in Section 3, the analysis of IT errors and convergence rates are based on E_1 as in (3.3). However, a straightforward calculation of E_1 tends to be ill-conditioned. For numerical tests, we instead use an alternative error criteria,

$$E_g(t_n) := \sum_{\mathcal{C}\subset\Omega} \big| \|\mathcal{M}_{\mathcal{C}}(t_n)\| - \|\mathcal{M}^n_{\mathcal{C}}\| \big|, \tag{5.1}$$

where $\mathcal{M}_{\mathcal{C}} = \mathcal{M} \cap \mathcal{C}$. It is easy to see that for any Yin set $\mathcal{M}$ we have $E_g \to E_1$ as the Eulerian grid size h approaches zero.

5.1. Rotation of the Zalesak disk. This test was first introduced by Zalesak [49] and later used by Rudman [35] and many other researchers. A slotted circle is placed in a purely rotational velocity field and returns to its initial location after a full revolution of 2π rotation.

Parameters	Values
computational domain	$[0,1]\times[0,1]$
simulation time	$t\in[0,T]$
shape parameters	$C=(0.5,0.75)$, $R=0.15$
velocity periods	$T=2,\ 8$
Courant number	$\mathrm{Cr}=1$
Eulerian grid sizes	$h=\frac{1}{16},\ \frac{1}{32},\ \frac{1}{64},\ \frac{1}{128}$
Lagrangian length scales	$h_L=O(h),\ O(h^{\frac{3}{2}}),\ O(h^2)$, $r_{\mathsf{tiny}}=0.01$

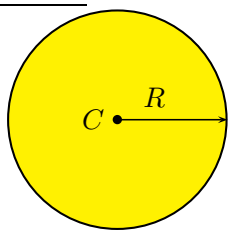


Fig. 5.1: Initial setup and other parameters of the reversed single vortex test

Since the rotation is an isometry that preserves the distance between any pair of interface markers, the steps (cubiMARS-2,3) are never invoked during the test. In addition, there is no need in the iPAM method to adjust volume because the total volume is conserved by the isometry for each time step. Therefore, the cubic MARS method is the same as the *cubic iPAM method* that follows (iPAM-1,2,3,4,5) but represents the interface with cubic splines; this is confirmed in our numerical experiments. Thus we do not reproduce the numerical results here, but simply refer the reader to those of the cubic iPAM method in Figures 4 and 5 and Tables 1 and 2 in [56].

There the fourth-, sixth-, and eighth-order of accuracy of the cubic MARS method with respectively $\alpha=1,\frac{3}{2},2$ is clearly demonstrated. We emphasize that the four kinks at the corners of the slotted disk are *always* specified as knots of the cubic spline. In addition, only C^0 continuity is enforced for these four corners when fitting the spline.

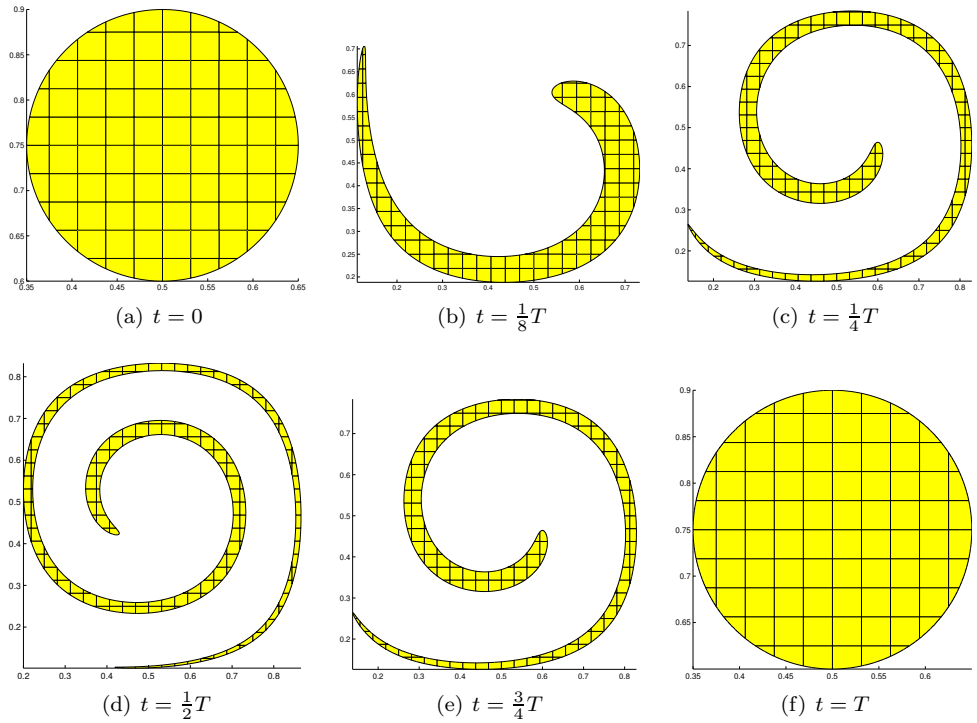
5.2. Vortex shear of a circular disk. In this test [34], a nonuniform, transient velocity field given by the stream function

$$\psi(x,y)=-\frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y)\cos\left(\frac{\pi t}{T}\right) \tag{5.2}$$

is imposed on a circular disk. The initial setup and other test parameters are shown in Figure 5.1. At time $t=\frac{T}{2}$, the velocity field is reversed by the cosinusoidal temporal factor so that the exact solution at $t=T$ is the same as the initial condition.

In Figure 5.2, we illustrate results of the cubic MARS method for tracking a single phase. During the first half period of simulation, the velocity field stretches the material region into a thin filament that spirals toward the vortex center, potentially tearing it apart. As shown in Figure 5.2 and [55, Figure 12], both the cubic MARS method and the iPAM method preserve very well the thin tail as one piece for all time instants. Hence both methods fulfill the analytic invariant that there be no topological changes of the material region under the action of a homeomorphic flow map. As for the test case of $T=2$, the stretching of the circular disk is much less severe; see [56, Figure 6.4(b)] for a plot of the material regions at $t=\frac{T}{2}=1$.

5.2.1. The evolution of marker distribution. Following [55], we define a ratio of Lagrangian length scales to indicate how the averaged distance between adjacent



(a) $t = 0$ (b) $t = \frac{1}{8}T$ (c) $t = \frac{1}{4}T$

(d) $t = \frac{1}{2}T$ (e) $t = \frac{3}{4}T$ (f) $t = T$

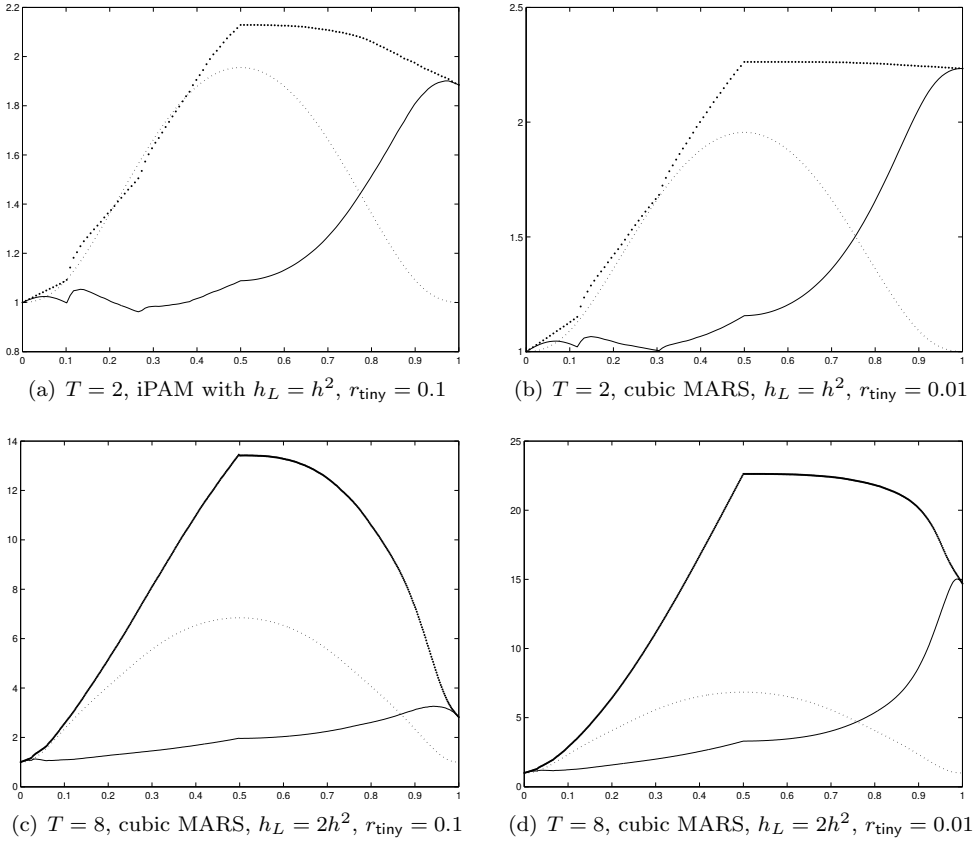
Fig. 5.2: Results of the cubic MARS method for the vortex shear test with $T = 8$, $h = \frac{1}{32}$, $h_L = 2h^2$, and $r_{\mathsf{tiny}} = 0.01$. The number of markers at $t = 0, \frac{T}{2}, T$ is 1003, 21878, 14177, respectively; see Figure 5.3(d) for the complete evolution of the number of interface markers. Based on Richardson extrapolation, the IT error at $t = \frac{T}{2}$ is 1.84×10^{-10}; see Table 5.4. Based on the exact solution, the IT error at $t = T$ is 9.85×10^{-11}; see Table 5.2.

markers changes with respect to the initial setup,

$$r_{\mathsf{mks}}(t) := \frac{\mathsf{L}_{\partial\mathcal{M}}(t)\, n_{\mathsf{mks}}(t_0)}{\mathsf{L}_{\partial\mathcal{M}}(t_0)\, n_{\mathsf{mks}}(t)}, \tag{5.3}$$

where the initial interface $\partial\mathcal{M}(t_0)$ has been divided into arcs with the same chordal length $\frac{h_L}{2}$, $\mathsf{L}_{\partial\mathcal{M}}(t)$ is the cumulative chordal length of $\partial\mathcal{M}(t)$, and $n_{\mathsf{mks}}(t)$ is the number of interface markers on $\partial\mathcal{M}(t)$. By this initial setup and the definition of $r_{\mathsf{tiny}} h_L$ being the minimum distance between adjacent markers, we have $r_{\mathsf{mks}} \in [2r_{\mathsf{tiny}}, 2]$.

In subplots (a) and (b) of Figure 5.3, values of $\frac{n_{\mathsf{mks}}(t)}{n_{\mathsf{mks}}(t_0)}$, $\frac{\mathsf{L}_{\partial\mathcal{M}}(t)}{\mathsf{L}_{\partial\mathcal{M}}(t_0)}$, and $\frac{1}{r_{\mathsf{mks}}(t)}$ for tests $T = 2$ are plotted with respect to $\frac{t-t_0}{T}$ for both the iPAM method with $r_{\mathsf{tiny}} = 0.1$ and the cubic MARS method with $r_{\mathsf{tiny}} = 0.01$. During the first half period, as the circular disk is stretched, the total length of the interface is monotonically increasing. This is confirmed by results of both methods as $\frac{\mathsf{L}_{\partial\mathcal{M}}(t)}{\mathsf{L}_{\partial\mathcal{M}}(t_0)}$ increases monotonically. Meanwhile, values of $\frac{n_{\mathsf{mks}}(t)}{n_{\mathsf{mks}}(t_0)}$ for both methods also increase monotonically and proportionally, since new interface markers are inserted to enforce h_L as the maximum distance between adjacent markers. In contrast, the indicator $r_{\mathsf{mks}}(t)$ decreases mildly with



(a) $T = 2$, iPAM with $h_L = h^2$, $r_{\mathsf{tiny}} = 0.1$

(b) $T = 2$, cubic MARS, $h_L = h^2$, $r_{\mathsf{tiny}} = 0.01$

(c) $T = 8$, cubic MARS, $h_L = 2h^2$, $r_{\mathsf{tiny}} = 0.1$

(d) $T = 8$, cubic MARS, $h_L = 2h^2$, $r_{\mathsf{tiny}} = 0.01$

Fig. 5.3: The efficiency evolution of interface markers for single vortex tests on the 64^2 grid. The abscissa is $\frac{t-t_0}{T}$ while the ordinate is a ratio. The dark dots represent $\frac{n_{\mathsf{mks}}(t)}{n_{\mathsf{mks}}(t_0)}$, the light dots $\frac{\mathsf{L}_{\partial\mathcal{M}}(t)}{\mathsf{L}_{\partial\mathcal{M}}(t_0)}$, and the line $\frac{1}{r_{\mathsf{mks}}(t)}$, i.e. the ratio of averaged markers per unit length at time t to that at the initial time t_0.

noticeable fluctuations, which is not a surprise because the averaged distance between adjacent markers is $h_L/2$ at the initial time. At the end of the first half period, we do not expect a large reduction of $r_{\mathsf{mks}}(t)$ since the change of markers during this period is dominated by the insertion of new markers.

As the circular disk returns to its initial shape in the second half period, results of both methods show a decreasing length of the interface and a decreasing number of markers, meanwhile the indicator $r_{\mathsf{mks}}(t)$ decreases with no discernible fluctuations. Despite these qualitatively common features, the two methods do differ quantitatively: the number of the remaining markers at the end of the simulation of the iPAM method is about 25% less than that of the cubic MARS method. This is caused by their different values of r_{tiny}; a smaller value of r_{tiny} implies a higher number of markers per unit length and thus more markers can coexist within the same distance.

For the tests with $T = 2$, the reduction of r_{tiny} from 0.1 to 0.01 only incurred slight increases of $\frac{n_{\mathsf{mks}}(T/2)}{n_{\mathsf{mks}}(t_0)}$ and $\frac{n_{\mathsf{mks}}(T)}{n_{\mathsf{mks}}(t_0)}$. In comparison, results of tests $T = 8$ shown

Table 5.1: Errors and convergence rates based on E_g for the vortex shear test with $T = 2$. The value of r_{tiny} is 0.1 and 0.01 for the linear iPAM method and the cubic MARS method, respectively.

Methods	16^2	rate	32^2	rate	64^2	rate	128^2
linear iPAM, $h_L = 0.1h$	6.63e-04	2.99	8.34e-05	2.99	1.05e-05	2.94	1.36e-06
linear iPAM, $h_L = h^{\frac{3}{2}}$	2.79e-04	3.03	3.42e-05	3.02	4.23e-06	3.00	5.40e-07
linear iPAM, $h_L = h^2$	1.14e-05	4.50	5.04e-07	4.44	2.32e-08	4.06	1.39e-09
cubic MARS, $h_L = 0.1h$	3.53e-06	4.98	1.12e-07	4.99	3.52e-09	5.00	1.10e-10
cubic MARS, $h_L = h^{\frac{3}{2}}$	3.50e-08	6.51	3.84e-10	6.56	4.08e-12	6.86	3.51e-14
cubic MARS, $h_L = h^2$	4.93e-11	7.87	2.10e-13	9.89	2.22e-16	–	2.22e-16

in subplots (c) and (d) of Figure 5.3 are more prominent: the same reduction of r_{tiny} leads to increases of $\frac{n_{\mathsf{mks}}(T/2)}{n_{\mathsf{mks}}(t_0)}$ and $\frac{n_{\mathsf{mks}}(T)}{n_{\mathsf{mks}}(t_0)}$ by factors of 2 and 5, respectively. This implies that, when the interface length $\mathsf{L}_{\partial\mathcal{M}}(t)$ undergoes large variations, a smaller value of r_{tiny} tends to leave more markers on the final interface.

Naturally it appears that the decrease of r_{tiny} from 0.1 to 0.01 has had a negative impact on the efficiency: both the memory storage and the CPU time increase. However, as shown in the next subsection, the reduction of r_{tiny} from 0.1 to 0.01 often leads to an accuracy improvement that outweighs the increase of computational cost.

5.2.2. Two key factors of accuracy improvements: the spline representation and the reduction of r_{tiny}. Errors and convergence rates of the cubic MARS method for $T = 2$, together with those of the iPAM method, are listed in Table 5.1. The convergence rates 4, 6, 8 of the cubic MARS methods for $\alpha = 1, \frac{3}{2}, 2$ are clearly demonstrated. In addition, the cubic MARS method is much more accurate than the iPAM method by many orders of magnitudes. This higher accuracy also implies a better efficiency as a much larger h_L can be adopted in the cubic MARS method to achieve the same accuracy as that of the linear iPAM method.

The accuracy improvement of the cubic MARS method over the linear iPAM method comes from two main factors: the employment of cubic splines to represent the interface and the reduction of r_{tiny}. By Definition 4.3, the number of interface markers $n_{\mathsf{mks}}(t)$ in a given test is determined by the choice of h_L and the value of r_{tiny}. Hence $n_{\mathsf{mks}}(t)$ is mostly independent of the representation of interface, be it a linear or cubic spline.

To single out the effect of r_{tiny} on efficiency and accuracy, we compare results of the cubic iPAM method and the cubic MARS method for the longer test of $T = 8$ in the first two lines of Table 5.2. For the fixed choice of $h_L = 0.2h$, the ratios of accuracy improvement on the four successively refined grids are 1.7, 16.1, 101.5, and 2963.6, while timing results indicate that the corresponding CPU-time ratios of the cubic MARS method over the cubic iPAM method are, respectively, 3.0, 4.7, 6.7, and 8.5. On the finest grid, the ninefold increase in CPU time is compensated with an accuracy improvement by a factor of almost three thousand! Clearly, the reduction of r_{tiny} is a very effective (and also very efficient) way for accuracy improvement. In addition, the improvements on convergence rates are very satisfactory.

In [56], it is shown analytically that the cubic iPAM method with $\alpha = 2$ can be eighth-order accurate if no C^1 discontinuities are generated dynamically. Numerical experiments also suggest that such a local C^1 discontinuity may result in a deteriora-

Table 5.2: A comparison of various IT methods on their errors and convergence rates for the vortex shear test with $T = 8$. As far as the accuracy of this particular test is concerned, the cubic iPAM method only differs from the cubic MARS method in that the former has $r_{\mathsf{tiny}} = 0.1$ and the latter has $r_{\mathsf{tiny}} = 0.01$. Hence a comparison of these two methods reveals the effect of r_{tiny}. We compute all results of the cubic iPAM method, with those on the first three grids validated by [56, Table 3]. We also list results of three representative VOF methods, the moment-of-fluid method with adaptive mesh refinement, a hybrid VOF–front-tracking method, and two level set methods. The reader is cautioned that the complexity of cubic MARS methods with $h_L = O(h^{\frac{3}{2}})$ and $h_L = O(h^2)$ is asymptotically higher than VOF and front-tracking methods; see Table 5.5 and Proposition 5.1.

Methods	32^2	rate	64^2	rate	128^2	rate	256^2
cubic iPAM ($h_L = 0.2h$)	1.22e-06	1.71	3.72e-07	2.35	7.28e-08	-0.04	7.32e-08
cubic MARS ($h_L = 0.2h$)	7.22e-07	4.97	2.30e-08	5.00	7.17e-10	4.86	2.47e-11
cubic MARS ($h_L = 0.5h^{\frac{3}{2}}$)	3.67e-09	8.01	1.43e-11	6.91	1.19e-13	6.92	9.83e-16
cubic MARS ($h_L = 2h^2$)	9.85e-11	8.50	2.72e-13	10.26	2.22e-16	–	2.22e-16
Rider/Kothe/Puckett [34]	4.78e-02	2.78	6.96e-03	2.27	1.44e-03	–	–
Stream/Youngs [18]	3.61e-02	1.85	1.00e-02	2.22	2.16e-03	–	–
Improved PLIC-VOF [29]	5.78e-03	1.71	1.77e-03	2.42	3.30e-04	1.93	8.69e-05
AMR-MOF [2]	2.33e-02	2.89	3.15e-03	2.64	5.04e-04	–	–
Hybrid markers [3]	2.52e-03	2.96	3.23e-04	2.99	4.06e-05	2.99	5.11e-06
LS-WENO-5 [19]	–	–	–	–	2.32e-03	2.7	3.63e-03
RKDG-CLS-4 [23]	–	–	3.43e-03	2.3	7.06e-04	2.4	1.35e-04

Table 5.3: Volume-conservation errors and convergence rates of the cubic MARS method with $r_{\mathsf{tiny}} = 0.01$ for vortex shear tests.

Cases	32^2	rate	64^2	rate	128^2	rate	256^2
$T = 2$, $h_L = 0.1h$	2.36e-06	4.98	7.48e-08	4.99	2.35e-09	5.00	7.34e-11
$T = 2$, $h_L = h^{\frac{3}{2}}$	1.54e-08	7.50	8.52e-11	6.83	7.49e-13	6.68	7.31e-15
$T = 2$, $h_L = h^2$	3.02e-11	7.99	1.19e-13	9.07	2.22e-16	–	2.22e-16
$T = 8$, $h_L = 0.2h$	2.77e-07	4.97	8.82e-09	5.01	2.74e-10	4.66	1.08e-11
$T = 8$, $h_L = 0.5h^{\frac{3}{2}}$	1.52e-09	9.07	2.82e-12	8.14	9.98e-15	5.49	2.22e-16
$T = 8$, $h_L = 2h^2$	3.83e-11	7.33	2.39e-13	8.40	7.08e-16	–	2.22e-16

Table 5.4: Errors and convergence rates of the cubic MARS method with $r_{\mathsf{tiny}} = 0.01$ for the vortex shear test $T = 8$ with the final simulation time as $T/2$. Since no analytic solution is known, convergence rates are computed with Richardson extrapolation with errors defined as the difference of solutions on a coarse grid and a fine grid.

Methods	16^2–32^2	rate	32^2–64^2	rate	64^2–128^2
$h_L = 0.2h$	2.61e-05	3.84	1.83e-06	3.94	1.19e-07
$h_L = 0.5h^{\frac{3}{2}}$	1.26e-07	5.50	2.79e-09	6.95	2.25e-11
$h_L = 2h^2$	7.29e-08	8.63	1.84e-10	8.40	5.43e-13

tion of global accuracy when the local errors at the kink dominate those elsewhere. If locations of these kinks are known, we can customize spline fitting in a manner such that only C^0 continuity is enforced there; cf. Section 5.1. However, it is difficult to estimate locations of dynamic kinks with sufficient accuracy. One alternative way to alleviate the accuracy deterioration is to reduce the chordal lengths at the vicinity of these kinks. More detailed investigations on results of the cubic MARS method show that, when a kink is generated dynamically, the nearby chordal lengths of spline fitting tend to be close to $r_{\mathsf{tiny}} h_L$, i.e. the minimum distance between adjacent markers, probably due to the stretching of the interface which triggers frequent addition of new markers in (CubiMARS-2). Consequently, when r_{tiny} is reduced from 0.1 to 0.01, the error caused by the dynamic kink is also reduced. By the argument in the proof of [56, Lemma 3.9], a reduction of r_{tiny} by a factor of 10 yields a *local* error reduction by a factor of 100. We therefore speculate that, for the first two lines of Table 5.2, the accuracy improvement on the 128^2 grid is mainly executed by the reduction of local errors near the kink while the improvement on the 256^2 grid is executed by both the local error reduction and a better global resolution of the interface; c.f. the fourth-order accuracy of cubic splines.

In Table 5.2 we also compare results of the cubic MARS method to those of other representative IT methods. In particular, we reproduce results in the literature for three PLIC VOF methods, one MOF method with adaptive mesh refinement (AMR), one hybrid VOF–front-tracking method, and two level set methods that employ either the weighted essentially nonoscillatory scheme (WENO) or the discontinuous Galerkin (DG) formulation. It is clear that on the same grid the cubic MARS method is much more accurate than each of the other methods by many orders of magnitudes.

As discussed before, *asymptotic* volume conservation rather than *exact* volume conservation is intended in the cubic MARS method. In Table 5.3, we demonstrate that the asymptotic volume conservation has indeed been achieved with the same high convergence rates as those of IT. It is also confirmed that each volume conservation error is no greater than the corresponding geometric error of IT.

In summary, convergence rates of the cubic MARS method with $\alpha = 1, \frac{3}{2}, 2$ are demonstrated to be 4, 6, 8, respectively. We also believe that these convergence rates of the cubic MARS method for a given test would continue in the asymptotic range of $h \to 0$ so long as r_{tiny} is chosen to be sufficiently small, because the IT error (3.3) is based on the 1-norm and Sard's theorem dictates that the C^1 discontinuities on an interface form a set of measure zero. In our numerical tests, this is confirmed up to machine precision.

The careful reader may argue that the reversed velocity field has a temporal symmetry with respect to $t = T/2$, and this might contribute to the improvement of convergence rates. To exclude this factor, we end this subsection by performing the tests $T = 8$ again using the cubic MARS method, the only difference being that the simulation stops at $t = T/2$. This test is obviously more stringent in the sense that the final solution contains a kink and there is no temporal symmetry in the velocity field. Errors and convergence rates based on Richardson extrapolation are listed in Table 5.4. Although the errors are larger than those in Table 5.2, the convergence rates 4, 6, and 8 are clearly maintained.

5.2.3. Length-scale flexibility and excellent conditioning of the cubic MARS method. In many multiphase fluids the length scale of the interface is much smaller than that of the main flow. Accordingly, the cubic MARS method decouples h_L, the length scale of IT, from h, the resolution of the main flow. The free parameters

r_h and α in the relation $h_L = r_h h^\alpha$ furnish a simple, efficient, and flexible mechanism to achieve subgrid accuracy for IT without increasing the resolution of the main flow. Essentially we supplement the Eulerian grids of the main-flow solver with another semi-Lagrangian grid for interface tracking. The very loose coupling of these two grids partly accounts for the excellent conditioning of the cubic MARS method.

For certain family of unsplit VOF advection algorithms, the change of volume fraction of a cell is calculated by constructing a donating region for each cell face, then intersecting the donating region to ambient fluid phase [50], and finally summing up the flux through cell faces. As an advantage to interface capturing methods that directly apply HCL schemes, the accuracy of these unsplit VOF methods can be of the second order. Unfortunately, computing the intersection of two polygons can be arbitrarily ill-conditioned, which is a road barrier to accuracy close to machine precision both in unsplit VOF advection algorithms and the iPAM method.

From the iPAM method to the cubic MARS method, the principle of decoupling length scales of the interface and the main flow has been pushed one step further: the intersection of cell image of ambient fluid has been removed. Hence the potential accuracy deterioration from the ill-conditioning of polygon clipping is also avoided. Consequently, the IT accuracy of the cubic MARS method only depends on the accuracy of the time integrator applied to *individual* interface markers and the accuracy of spline fitting through these markers. When coupled to a main flow solver, the potential ill-conditioning of the splinegon clipping algorithm can also be avoided by carefully selecting those control volumes (or unions of control volumes) where no cell faces form small angles with the interface and where the spatial operators in the governing equation of the main flow can be discretized.

The spline fitting in (CubiMARS-4) is very well conditioned because of the narrow variation of distances of adjacent interface markers enforced by the interval $[r_{\text{tiny}} h_L, h_L]$ in Definition 4.3. In addition, errors of spline fitting do not accumulate in time, thanks to removing the steps for exact volume conservation.

Now is a good time to discuss the reasons for the machine precision achieved by the cubic MARS method in Tables 5.2 and 5.3. First, the above discussion implies that the cubic MARS method *should* have excellent conditioning. Second, the IT error E_g as defined in (5.1) is not a *relative* error but an *absolute* error. Hence, a machine precision $\epsilon_{\text{machine}}$ in E_g does not imply prefect conditioning. Instead, we can only deduce that the conditioning of the cubic MARS method for this particular test is no greater than $1/\|\mathcal{M}(t_0)\| \approx 14$, where the initial volume of the tracked phase for this test is $\|\mathcal{M}(t_0)\| = \pi R^2 \approx 0.07$. Third, it is well known [6, page 86] that, as the time step size k decreases, the truncation error of a time integrator decreases while the rounding error may increase if k is smaller than some threshold value. Hence the grid size $h = \frac{1}{128}$ is close to the optimal grid size by which an eighth-order explicit Runge–Kutta (ERK) method produces machine precision. Third, the steps of an ERK method only consist of addition, subtraction, multiplication, division, and function evaluation of the velocity field through analytic expressions. Fourth, for the adopted ERK methods catastrophic loss of accuracy in finite-precision computation, such as those caused by subtracting two close numbers and adding two numbers with largely varying magnitudes, are unlikely to happen [20] in the IT context. Last, for this particular test, the temporal symmetry of the velocity field tends to counteract the accumulation of the rounding errors.

5.2.4. An efficiency comparison of the cubic MARS method to optimal second-order IT methods. The efficiency comparison starts from a crucial fact

Table 5.5: Timing results of the cubic MARS method with $r_{\mathsf{tiny}} = 0.01$ for vortex shear tests. Numbers within a grid column are CPU times in seconds on the author's laptop with a single Intel Core$^{\mathrm{TM}}$ i7-5500U CPU at 2.40GHz. Numbers within a "rate" column are the base-2 logarithm of the ratio of adjacent timing results.

Cases	32^2	rate	64^2	rate	128^2	rate	256^2
$T = 2$, $h_L = 0.1h$	12	1.00	24	2.17	108	2.45	592
$T = 2$, $h_L = h^{\frac{3}{2}}$	6	1.50	17	2.47	94	2.72	619
$T = 2$, $h_L = h^2$	6	2.50	34	2.77	232	2.95	1790
$T = 8$, $h_L = 0.2h$	332	2.62	2034	2.52	11674	2.43	62700
$T = 8$, $h_L = 0.5h^{\frac{3}{2}}$	416	2.89	3079	2.90	22952	2.91	172510
$T = 8$, $h_L = 2h^2$	477	2.99	3786	3.04	31224	3.09	265873

that the interface is a set of codimension one. Consequently, the number of interface cells, to which a VOF method needs to be applied, is $O(\frac{1}{h})$. However, to advance the initial interface to the final time, the VOF method has to be repeated for $O(\frac{1}{h})$ time steps. Hence the optimal complexity of a VOF method is $O(\frac{1}{h^2})$ in two dimensions. By similar arguments, the complexity of a cubic MARS method with $h_L = O(h^\alpha)$ is $O(\frac{1}{h^{1+\alpha}})$. This is confirmed by Table 5.5, where timing results of the eighth-order cubic MARS method for the vortex shear tests show that the choice of $\alpha = 2$ yields an increase rate 3 of the CPU time.

In two dimensions the main flow solver also has an optimal complexity of $O(\frac{1}{h^3})$, and hence when coupled together the cubic MARS method and the main flow solver are asymptotically well balanced in cost. Furthermore, if the CPU time spent on IT is still considered too labor intensive, one can choose a larger value of r_h in the relation $h_L = r_h h^2$ so that the time spent on IT is only a small fraction of that spent on solving for the main flow. Usually the optimal complexity of the main flow solver is achieved by multigrid methods, implying that the constant inside $O(\frac{1}{h^3})$ for the main flow solver is much larger than one; see, e.g., [53] and references therein. Therefore, by simply changing the value of r_h, a user can tweak, to her satisfaction, the *constant* CPU-time ratio of the eighth-order cubic MARS to the main flow solver. This is another illustration of the flexibility of the proposed cubic MARS method.

The following is the main analytic result on the comparison of our eighth-order cubic MARS method to a second-order IT method with optimal complexity.

PROPOSITION 5.1. *Let h and h_L denote the length scale of fixed control volumes and that of interface markers, respectively. Consider in two dimensions a second-order IT method with the optimal complexity $O(\frac{1}{h^2})$ and an eighth-order cubic MARS method with $h_L = r_h h^2$ and hence the complexity $O(\frac{1}{h^3})$. Let $T_2(\epsilon)$ and $T_8(\epsilon)$ denote functions of the CPU time of these two methods for achieving a predefined accuracy ϵ. Then the speedup of the cubic MARS method over the second-order method for a given test problem, defined as $S_{8|2}(\epsilon) := \frac{T_2(\epsilon)}{T_8(\epsilon)}$, can be expressed as*

$$S_{8|2}(\epsilon) = \frac{1}{n_T} E_2 E_8^{-\frac{3}{8}} \epsilon^{-\frac{5}{8}}, \tag{5.4}$$

where n_T denote the corresponding ratio of CPU time of the cubic MARS method to that of the optimal second-order method on a fixed reference grid for the same test problem, and E_2 and E_8 denote the corresponding IT errors of the two methods.

Proof. To achieve the same accuracy ϵ, the grid sizes of a second-order method and an eight-order method are, respectively,

$$h_2 = \theta(\epsilon^{\frac{1}{2}}), \qquad h_8 = \theta(\epsilon^{\frac{1}{8}}), \tag{5.5}$$

and hence the CPU times of these methods are, respectively,

$$T_2 = C_2 \frac{1}{h_2^2}, \qquad T_8 = C_8 \frac{1}{h_8^3}, \tag{5.6}$$

where C_2, C_8 depends only on the test problem and the internal details of the second and eighth-order methods. The power coefficients of h_2 and h_8 being, respectively, 2 and 3 comes from the given conditions that the complexities of the second-order IT method and the eighth-order cubic MARS method are, respectively, $O(\frac{1}{h^2})$ and $O(\frac{1}{h^3})$. Substitute (5.5) into (5.6) and we have

$$S_{8|2}(\epsilon) = C_{8|2}\epsilon^{-\frac{5}{8}}, \tag{5.7}$$

where the coefficient $C_{8|2}$ is a constant for any fixed test problem. To determine $C_{8|2}$, we run the two methods on some reference grid and record the values of n_T, E_2, and E_8 as results of the numerical experiments. Then in order for the second-order method to achieve the same accuracy of the cubic MARS method, the number of times of grid refinement is

$$n_{2\to 8} = \log_4 \frac{E_2}{E_8}, \tag{5.8}$$

where the base 4 comes from the fact that every halving of the grid size reduces the IT error by a factor of 4; c.f. the definition of a second-order method. Then (5.7) yields

$$S_{8|2}(E_8) = C_{8|2}E_8^{-\frac{5}{8}} = \frac{(2^2)^{n_{2\to 8}}}{n_T},$$

where 2^2 in the numerator refers to the fact that every halving of the grid size increases the CPU time by a factor of 4. Therefore we have

$$C_{8|2} = \frac{1}{n_T} E_2 E_8^{-\frac{3}{8}}. \tag{5.9}$$

Then the proof is completed by the substitution of (5.9) into (5.7). ∎

We emphasize that Proposition 5.1 does not depend on the convention set forth in Section 1 that convergence rates are expressed not in terms of h_L but in terms of h. Although the cubic MARS method with $h_L = O(h^2)$ is only fourth-order accurate in terms of h_L, the expression of h_8 in (5.5) and that of T_8 in (5.6) remain the same after applying the relation $h_L = O(h^2)$. As another example, a second-order VOF method can be made fourth-order accurate by adaptively refining control volumes near the interface so that their sizes are $h_L = O(h^2)$. Then the CPU time of this fourth-order VOF method would be $O(\frac{1}{h_L}) = O(\frac{1}{h^2})$ per time step and this method would still have to take $O(\frac{1}{h_L}) = O(\frac{1}{h^2})$ time steps, and hence its total complexity amounts to $O(\frac{1}{h_L^2}) = O(\frac{1}{h^4})$. In the proof of Proposition 5.1, the replacement of h_2 in (5.5) and T_2 in (5.6) with $h_4 = \theta(\epsilon^{\frac{1}{4}})$ and $T_4 = C_4\frac{1}{h_4^4}$ yields the same conclusion.

As the main point of this discussion, the convergence rate of a method is merely a *label*; what really matters is the *performance* only partially indicated by the label. To sum up, the *ultimate* reason for the strong result in Proposition 5.1 is the remarkable *cost-effectiveness* of the cubic MARS method, not its high convergence rates in terms of h.

To exemplify the utility of Proposition 5.1, we implemented the improved PLIC-VOF method [29], ran it on the 32^2 grid for the vortex-shear test with $T = 8$, and found that the total running time is about 32 seconds on the author's laptop. Since in Table 5.5 the CPU time of the cubic MARS method for the same test on the same laptop is 477, we have $n_T = \frac{477}{32} \approx 15$. From Table 5.2, we have $E_2 = 5.78 \times 10^{-3}$ and $E_8 = 9.85 \times 10^{-11}$. Hence to achieve a predefined accuracy ϵ, the speedup of the proposed eighth-order cubic MARS method over the improved PLIC VOF method [29] for the vortex-shear test with $T = 8$ is

$$S_{\text{MARS|iPLIC}}(\epsilon) \approx \frac{1}{15} \times 5.78 \times 10^{-3} \times (9.85 \times 10^{-11})^{-\frac{3}{8}} \epsilon^{-\frac{5}{8}} \approx 2.19\epsilon^{-\frac{5}{8}}. \tag{5.10}$$

For $\epsilon = 10^{-4}, 10^{-6}, 10^{-8}$, we have $S_{\text{MARS|iPLIC}}(\epsilon) \approx 693, 12332, 219290$, respectively. In other words, to achieve four, six, and eight decimal digits of IT accuracy in the vortex-shear test, the cubic MARS method is more efficient than the improved PLIC VOF method by, respectively, hundreds, tens of thousand, and hundreds of thousand of times! As the predefined accuracy ϵ is further reduced, the speedup increases as a power function of ϵ: every time one additional decimal digit of accuracy is required, the speedup increases by a factor of about four.

The above argument can be repeated for any IT method in Table 5.2 other than the cubic MARS method. This demonstrates the superior efficiency of the cubic MARS method. More generally, Proposition 5.1 and its proof show that, when high accuracy is needed, high convergence rates in IT tend to yield high efficiency.

It is true that for many applications the accuracy of $\epsilon = 10^{-4}, 10^{-6}, 10^{-8}$ might be unnecessarily high. However, if surface tension is not negligible and the curvature has to be estimated from the IT results, then these accuracy are not high at all. As shown in [54], at least half of the accuracy digits in IT are lost in a second-order method when the IT results are used for curvature estimation. Therefore, the best accuracy of curvature estimation that a second-order method *can* achieve out of the IT accuracy $\epsilon = 10^{-4}, 10^{-6}, 10^{-8}$ is less than $10^{-2}, 10^{-3}, 10^{-4}$, respectively. In summary, computational efficiency and the ubiquitousness of curvature estimation in multiphase flows advocate for the practical significance of high-order IT methods with high cost-effectiveness.

5.3. Deformation of a circular disk. This test has long been regarded as one of the most difficult interface tracking problems. The velocity field is given by the stream function

$$\psi(x, y) = \frac{1}{n_{\mathrm{v}}\pi} \sin(n_{\mathrm{v}}\pi(x + 0.5)) \cos(n_{\mathrm{v}}\pi(y + 0.5)) \cos\left(\frac{\pi t}{T}\right), \tag{5.11}$$

where n_{v} is the number of vortices in the computational domain and the temporal factor reverses the flow at $t = \frac{T}{2}$. The initial setup and other test parameters are shown in Figure 5.4.

Results of the cubic MARS method for tracking a single phase on the coarsest grid are shown in Figure 5.5, where the material regions are plotted on the actual grids. Although the disk is stretched to an extremely thin width at $x = 0.5$, the

Parameters	Values
computational domain	$[0,1]\times[0,1]$
simulation time	$t\in[0,T]$
shape parameters	$C=(0.5,0.5)$, $R=0.15$
velocity parameters	$T=2$, $n_{\mathsf{v}}=8$
Courant number	$\mathrm{Cr}=1$
Eulerian grid sizes	$h=\frac{1}{32},\ \frac{1}{64},\ \frac{1}{128}$
Lagrangian length scales	$h_L=O(h),\ O(h^{\frac{3}{2}}),\ O(h^2)$, $r_{\mathsf{tiny}}=0.01$

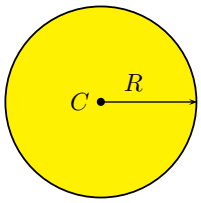


Fig. 5.4: Initial setup of the deformation field test

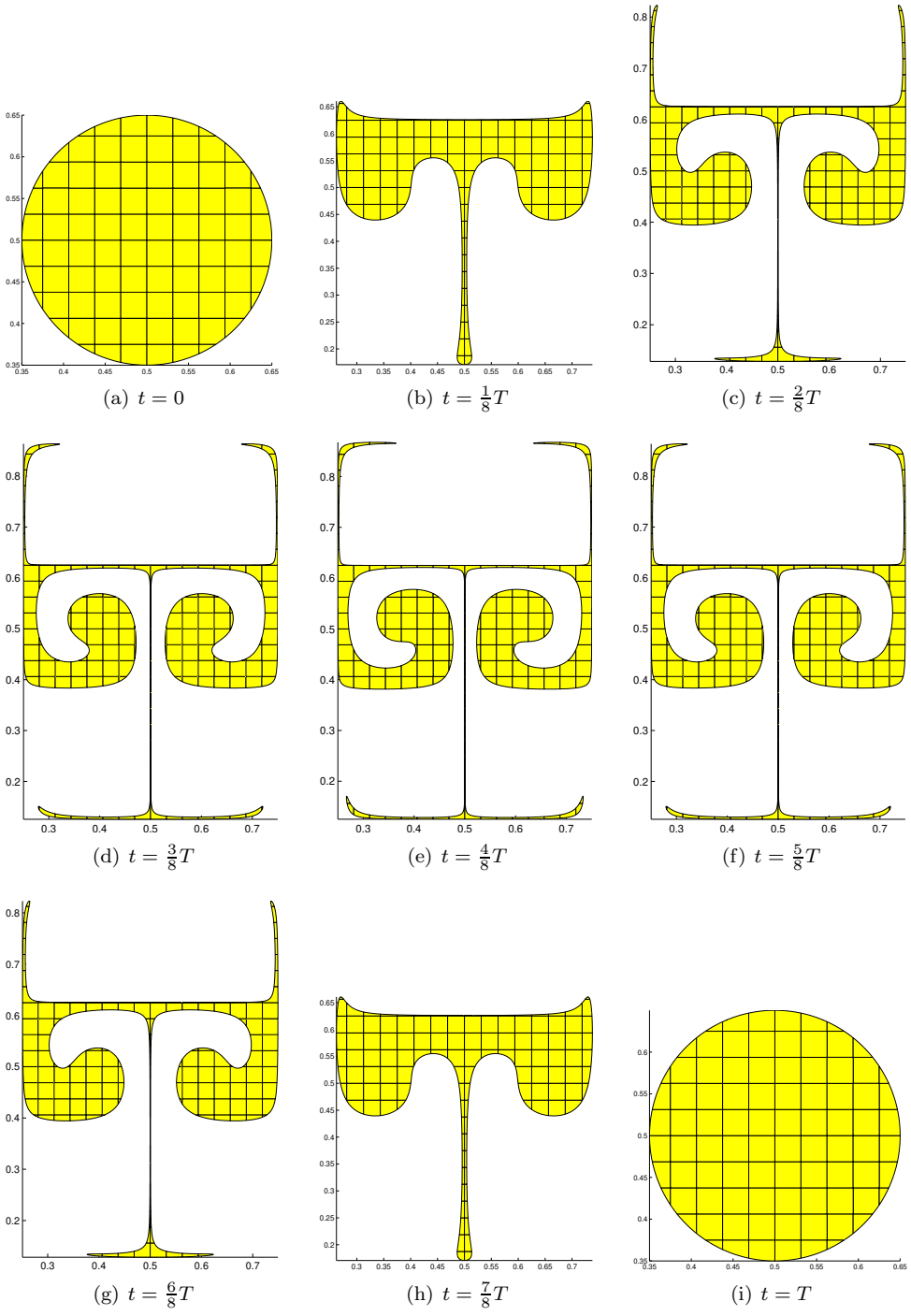
Table 5.6: Errors and convergence rates for the deformation test in Figure 5.4. Those of the iPAM method are taken from [55].

Methods	32^2	rate	64^2	rate	128^2
linear iPAM, $h_L=h^{\frac{3}{2}}$, $r_{\mathsf{tiny}}=0.1$	3.27e-04	2.92	4.31e-05	2.96	5.55e-06
linear iPAM, $h_L=h^2$, $r_{\mathsf{tiny}}=0.1$	2.20e-05	5.35	5.39e-07	5.04	1.64e-08
cubic MARS, $h_L=0.1h$, $r_{\mathsf{tiny}}=0.01$	6.99e-06	4.98	2.22e-07	5.00	6.96e-09
cubic MARS, $h_L=h^{\frac{3}{2}}$, $r_{\mathsf{tiny}}=0.01$	2.15e-07	7.54	1.16e-09	6.85	1.01e-11
cubic MARS, $h_L=h^2$, $r_{\mathsf{tiny}}=0.01$	2.95e-11	7.72	1.40e-13	9.30	2.22e-16

cubic MARS method is able to preserve the topology of the disk for all time instants, fulfilling the topological invariant that there be neither breaking up nor merging of material regions. In addition, the subgrid resolution shown in Figure 5.5 clearly illustrates the flexibility of the cubic MARS method in decoupling the Lagrangian length scale h_L from the Eulerian grid size h.

Evolutions of the indicator $r_{\mathsf{mks}}(t)$ and the number of interface markers are shown in Figure 5.6, where the time history of the number of interface markers n_{mks} has three distinctive stages. Corresponding to subplots (a) to (e) in Figure 5.5, the first stage is featured by the fast growth of the interface length $\mathsf{L}_{\partial\mathcal{M}}(t)$, which frequently invokes (CubiMARS-2) in Definition 4.3. The fact of $\frac{n_{\mathsf{mks}}(t)}{n_{\mathsf{mks}}(t_0)}$ being very close to $\frac{\mathsf{L}_{\partial\mathcal{M}}(t)}{\mathsf{L}_{\partial\mathcal{M}}(t_0)}$ indicates an even distribution of the markers on the interface. Consequently, the indicator $r_{\mathsf{mks}}(t)$ is very close to 1 in the first stage. The second stage roughly corresponds to subplots (e) to (g) in Figure 5.5, during which the interface length is being reduced, but n_{mks} stays roughly the same due to the small value of r_{tiny}. In other words, the evenly spaced interface markers are pushed closer to each other, but the distances of most adjacent markers are still greater than $r_{\mathsf{tiny}}h_L$. The third stage roughly corresponds to subplots (g) to (i) in Figure 5.5. As the interface length quickly diminishes, interface markers move closer and closer to each other, which frequently invokes (CubiMARS-3) in Definition 4.3. Finally, the number of interface markers at the end of the simulation is about 5.3 times greater than that at the initial instant, leading to the final value of the indicator at about 0.18. These features in the above discussion are qualitatively the same for the other two finer grids.

Errors and convergence rates of the cubic MARS method, together with those of the iPAM method, are listed in Table 5.6. Once again, the convergence rates are over



(a) $t = 0$ (b) $t = \frac{1}{8}T$ (c) $t = \frac{2}{8}T$

(d) $t = \frac{3}{8}T$ (e) $t = \frac{4}{8}T$ (f) $t = \frac{5}{8}T$

(g) $t = \frac{6}{8}T$ (h) $t = \frac{7}{8}T$ (i) $t = T$

Fig. 5.5: Results of the cubic MARS method for the deformation test on the 32^2 grid with $h_L = h^2$ and $r_{\mathsf{tiny}} = 0.01$. The maximum number of interface markers is 4258 and the final interface tracking error is 2.95×10^{-11}.

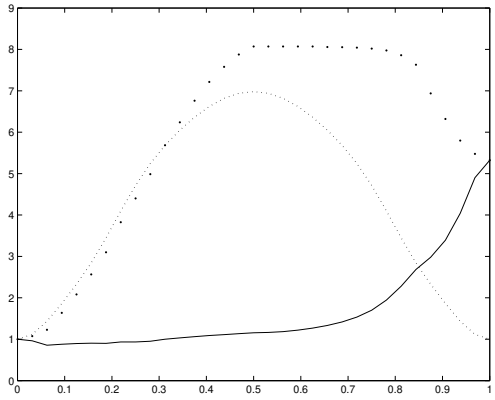


Fig. 5.6: Time history of the indicator $r_{\mathsf{mks}}(t)$ and the number of interface markers for the deformation test shown in Figure 5.5. The abscissa is $\frac{t-t_0}{T}$ while the ordinate is a ratio. The dark dots represent $\frac{n_{\mathsf{mks}}(t)}{n_{\mathsf{mks}}(t_0)}$, the light dots $\frac{\mathsf{L}_{\partial\mathcal{M}}(t)}{\mathsf{L}_{\partial\mathcal{M}}(t_0)}$, and the line $\frac{1}{r_{\mathsf{mks}}(t)}$. The number of markers at t_0 is 531.

6 and 8 for the two choices of $h_L = h^{\frac{3}{2}}$ and $h_L = h^2$, respectively. In particular, the cubic MARS method with $h_L = h^2$, $r_{\mathsf{tiny}} = 0.01$, and $h = \frac{1}{128}$ resolves the final solution close to machine precision. Even on the coarsest grid of $h = \frac{1}{32}$, it achieves more than ten decimal digits of accuracy with a maximum number of interface markers at 4258. Similar to vortex-shear tests, the cubic MARS method is much more accurate than the iPAM method by many orders of magnitudes. Comparing the second and the third rows in Table 5.6, we observe that the cubic MARS method with $h_L = 0.1h$ is better than the iPAM method with $h_L = h^2$ not only in terms of accuracy but also in terms of efficiency.

6. Conclusion. Based on MARS, an analytic framework for IT, we have proposed a cubic MARS method for high-order interface tracking in two dimensions. For several standard benchmark tests considered as among the most difficult, errors of the proposed method are close to machine precision. This high accuracy is realized via a spline representation of the interface, a novel algorithm for splinegon-polygon clipping, and a refined control of the relation between a Lagrangian length scale of interface markers and the Eulerian length scale of control volumes. Theoretical analysis and numerical experiments demonstrate that the proposed method is superior to existing IT methods in terms of accuracy and efficiency.

Prospects for future research follow. First, the proposed method also applies to compressible flows. For curvature-driven flows, the cubic MARS method can be coupled to the author's HFES algorithm [54] to achieve fourth-, sixth-, or eighth-order accuracy. Second, the proposed metric space for modeling physically meaningful material regions is amenable to accurate and efficient treatments of topological changes, and the development of such algorithms is currently a work in progress. Finally, the augmentation of the proposed method to three dimensions is feasible via a curvilinear polyhedra clipping algorithm. Utilizing bivariate cubic splines and the choice of $h_L = O(h^{3/2})$, one can achieve sixth-order accuracy in three dimensions, with the computational costs well balanced between interface tracking and the main flow solver.

Appendix. Splinegon clipping: the intersection of two Yin sets.

In Section 2 we have proposed the Yin sets as a representation of physically meaningful material regions. In the cubic MARS method (see Definition 4.3), sometimes the intersection of two Yin sets needs to be determined to obtain a local representation of the solution. In this appendix, we formulate the core algorithm of our splinegon clipping method in a way so that Section 2 and Algorithm 1 are self-contained. By doing this we hope Algorithm 1 could be adopted in a wide range of applications, where Boolean algebra on physically meaningful material regions are needed.

Based on the solid footing of Section 2, our new algorithm handles all all degenerate cases and is of optimal complexity. Without loss of generality, we assume that $\partial\mathcal{C}$ is positively oriented and that each of two Yin sets $\mathcal{M}$ and $\mathcal{C}$ has a single Jordan curve as its boundary. Denoting by $\mathring{\gamma}$ the oriented boundary of $\mathcal{M}$, we enumerate all possibilities on the relative position of such $\mathcal{M} = \mathrm{int}(\mathring{\gamma})$ and $\mathcal{C}$ as follows.

(Case-1) $\partial\mathcal{C} = \mathring{\gamma}$.
(Case-2) $\partial\mathcal{C} \cap \mathring{\gamma}$ contains at most one point.
(Case-3) $\partial\mathcal{C} \cap \mathring{\gamma}$ only contains isolated points and its cardinality is at least two.
(Case-4) $\partial\mathcal{C} \cap \mathring{\gamma}$ contains curve segments.

Note that the intersection points as above can be either proper or improper; see Figure 2.1. As a useful result in computational geometry, a point p is in the bounded complement of a Jordan curve if and only if any ray starting from p to infinity has an odd number of proper intersections with the Jordan curve.

Out of the above four possible scenarios, (Case-1) is regarded to hold when the numerical value of $\|\mathcal{M} \oplus \mathcal{C}\|$ is less than a user-defined small positive real number. For (Case-2), the single point must be an improper intersection. Consequently, we must have one of the following: (i) $\mathcal{C} \subset \mathcal{M}$, (ii) $\mathcal{M} \subset \mathcal{C}$, (iii) $\mathcal{C} \cap \mathcal{M} = \emptyset$, and (iv) $\mathring{\gamma}$ is negatively oriented and is a subset of $\mathcal{C}$. In other words, (iv) means that the boundary of $\mathcal{C} \cap \mathcal{M}$ consists of both of the two Jordan curves $\mathring{\gamma}$ and $\partial\mathcal{C}$. Clearly, these scenarios can be easily detected by checking whether a point on $\mathring{\gamma}$ belongs to $\mathcal{C}$.

In the rest of this appendix, we focus on the nontrivial scenarios (Case-3) and (Case-4). As shown in Algorithm 1, $\mathcal{C}$ is intended to be a control volume and $\mathcal{M}$ the material region. Besides $\mathcal{C}$ and $\mathcal{M}$, a finite point set $P_\cap$ is also designed as another input parameter to characterize $\mathring{\gamma} \cap \partial\mathcal{C}$.

- For (Case-3), $P_\cap$ contains the isolated intersection points, proper or improper.
- For (Case-4), $P_\cap$ contains the endpoints of the curve segments in $\mathring{\gamma} \cap \partial\mathcal{C}$, in addition to the isolated intersection points.

Since $\mathcal{C}$ is a linear polygon and the Jordan curve $\mathring{\gamma}$ is represented by cubic (or lower-order) splines, these isolated intersection points can be easily calculated by explicit algebraic formulas. In particular, the formulas are simple for a rectangle as it consists of four line segments of the form $x = x_0, x_1,\ y \in [y_0, y_1]$ or $y = y_0, y_1,\ x \in [x_0, x_1]$. Also because of this simplicity, any one-dimensional intersection of a spline to $\mathcal{C}$ is easily detected. Thus the design of taking $P_\cap$ as an input parameter neither incurs any loss of generality nor avoids any difficulties. The preconditions listed in Algorithm 1 follow directly from the above discussions.

At line 1 of Algorithm 1, we divide $\mathring{\gamma}$ into a number of spline segments. Then line 2 is implemented by checking whether or not the midpoint of γ_i is in the interior of $\partial\mathcal{C}$. The lines 3-7 deal with the special case that all γ_i's are outside of $\mathcal{C}$, which also implies that all points in $P_\cap$ are isolated improper intersections. Due to the "return" statements at line 4 and line 6, the points p_i^s, p_i^e in line 8 do exists.

Algorithm 1: Clipping a splinegon with a simple linear polygon

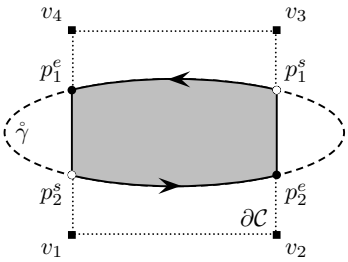
Input: An oriented Jordan curve $\mathring{\gamma}$,
a simple linear polygon $\mathcal{C}$ with positively oriented boundary $\partial\mathcal{C}$,
$P_\cap$, a set of isolated intersection points characterizing $\mathring{\gamma} \cap \partial\mathcal{C}$.

Precondition: (i) $\mathring{\gamma} \neq \partial\mathcal{C}$,
(ii) the cardinality of $P_\cap$ is at least two.

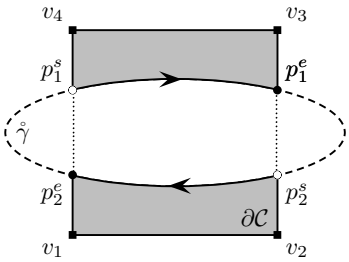
Output: $\left\{\mathring{\Gamma}_i\right\}$, a set of simple closed splines with positive orientation.

Postconditions: (i) $\mathcal{C} \cap \text{int}\,(\mathring{\gamma}) = \cup_i^{\perp\perp} \text{int}\left(\mathring{\Gamma}_i\right)$;
(ii) $\mathring{\Gamma}_i \neq \mathring{\Gamma}_j$ implies that they are almost disjoint;

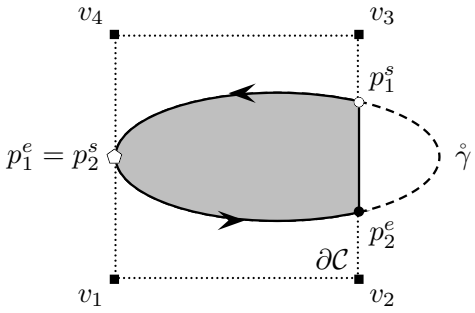
break $\mathring{\gamma}$ into curve segments separated by isolated intersection points
$\{\gamma_i\} \leftarrow$ the set of curve segments inside $\mathcal{C}$
if $\{\gamma_i\}$ *is an empty set* **and** $\exists p \in \partial\mathcal{C}$ *such that* $p \in \text{int}(\mathring{\gamma})$ **then**
  return $\left\{\mathring{\Gamma}_i\right\} = \{\partial\mathcal{C}\}$
else if $\{\gamma_i\}$ *is an empty set* **then**
  return $\left\{\mathring{\Gamma}_i\right\} = \emptyset$
end
$p_i^s, p_i^e \leftarrow$ the starting and ending points of γ_i
mark all starting points in $\{p_i^s\}$ as "undiscovered"
$c \leftarrow 0$ `// index of closed splines in the output`
while *there exists an undiscovered starting point, say,* p_r^s, **do**
  `anchor` $\leftarrow p_r^s$; `front` $\leftarrow p_r^e$; $c \leftarrow c+1$
  initialize $\mathring{\Gamma}_c$ with γ_r, the curve segment with `front` as an endpoint
  `closed` $\leftarrow$ false;
  while *`closed` = false* **do**
    γ_r cuts $\mathcal{C}$ into two splinegons, and the one at the left of γ_r is $\mathcal{C}_r$
    if *`front`*$=p_j^s$ *for some undiscovered* p_j^s **and** $\gamma_j \subset \mathcal{C}_r$ **then**
      $p_{\min} \leftarrow p_j^s$
    else
      start from `front` to traverse $\partial\mathcal{C}$ counterclockwise; set $p_{\min}$ as the next undiscovered $p_i^s \neq$`front` or the next vertex of $\mathcal{C}$; if they are equally close to `front`, choose p_i^s
    end
    if $p_{\min} \neq$ *`front`* **then**
      append into $\mathring{\Gamma}_c$ the line segment from `front` to $p_{\min}$
    end
    `front` $\leftarrow p_{\min}$
    if *`front`=`anchor`* **then**
      `closed` $\leftarrow$ true
    else if *`front`*$= p_j^s$ *for some undiscovered* p_j^s **then**
      append γ_j into $\mathring{\Gamma}_c$ and mark p_j^s as "discovered"
      `front` $\leftarrow p_j^e$
    end
  end
  mark `anchor` as "discovered"
end



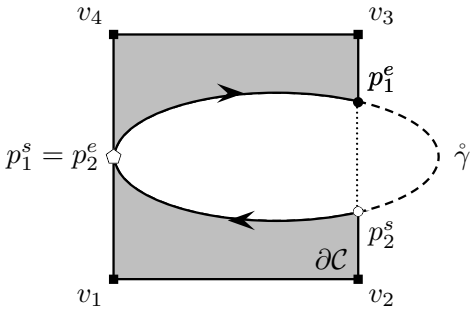
(a) positively oriented $\mathring{\gamma}$ with no improper intersections



(b) negatively oriented $\mathring{\gamma}$ with no improper intersectons



(c) positively oriented $\mathring{\gamma}$ with an improper intersection



(d) negatively oriented $\mathring{\gamma}$ with an improper intersection

Fig. A.1: Intersecting a simple closed splinegon with a simple linear polygon. Referring to Algorithm 1, the hollow and filled circles are the starting and ending points of the curve segments, respectively. They are also proper intersections. In contrast, a hollow pentagon represents an overlapping starting point and ending point, and is thus an improper intersection. The solid squares represent vertices of the linear polygon. The arrows indicate the orientation of the simple closed spline $\mathring{\gamma}$. The shaded regions represent the intersection.

We illustrate the rest of Algorithm 1 by four typical cases in Figure A.1. For subplot (a), suppose p_1^s is picked as the first undiscovered point p_r^s on line 11 of Algorithm 1. Then p_1^s is always the `anchor` during the construction of $\mathring{\Gamma}_1$. After the execution of lines 12–13, $\mathring{\Gamma}_1$ only contains the spline segment from p_1^s to p_1^e. The condition "`front`$= p_j^s$ for some undiscovered p_j^s" at line 17 clearly does not hold and the execution of line 20 yields $p_{\min} = p_2^s$. Since $p_2^s \neq$`front`, the linear segment $\overrightarrow{p_1^e p_2^s}$ is appended into $\mathring{\Gamma}_1$ at line 23. After setting $p_{\min} = p_2^s$ as the new `front` at line 25, the condition at line 28 holds. Then at line 29 we append γ_2 into $\mathring{\Gamma}_1$ and mark p_2^s as "discovered." At the end of the first inner while loop, p_2^e is set as the new `front`, with $\mathring{\Gamma}_1$ containing the spline segment from p_1^s to p_1^e, the line segment $\overrightarrow{p_1^e p_2^s}$, and the spline segment from p_2^s to p_2^e. During the second execution of the inner while loop, $p_{\min}$ is set to p_1^s at line 20, and hence the two conditions at lines 22 and 26 hold whereas that at line 28 does not. Now that $\mathring{\Gamma}_1$ is closed and all "undiscovered" starting points have been discovered, the algorithm terminates within a single outer loop and returns $\{\mathring{\Gamma}_1\}$ as the output.

The example illustrated in subplot (b) of Figure A.1 is complementary to that in subplot (a). Suppose p_2^s is picked as the first undiscovered point p_r^s on line 11 of Algorithm 1. Then p_2^s is the `anchor`. At the end of the first and second execution of the inner while loop both `front` is set to $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively. After another execution of the inner loop, p_2^s is discovered and $\mathring{\Gamma}_1$ is closed. Thus $\mathring{\Gamma}_1$ is constructed after three consecutive executions of the inner while loop. However, p_1^s remains as an undiscovered starting point, and the outer while loop is invoked again to construct $\mathring{\Gamma}_2$. After another three analogous executions of the inner while loop, the algorithm terminates and the set $\{\mathring{\Gamma}_1, \mathring{\Gamma}_2\}$ is returned.

The subplots (c) and (d) in Figure A.1 illustrate lines 16–18 in Algorithm 1 for handling improper intersections. In the case of subplot (c), cut $\mathcal{C}$ with γ_1 and we have two splinegons, the one on the left of γ_1 is $\mathcal{C}_r = \mathcal{C}_1$, the splinegon with its edges as the spline segment γ_1 and the directed linear segments $\overrightarrow{p_2^s v_1}$, $\overrightarrow{v_1 v_2}$, $\overrightarrow{v_2 p_1^s}$. Since $\gamma_2 \subset \mathcal{C}_1$, the condition at line 17 holds and $p_{\min}$ is set to p_2^s. The condition at line 28 holds while none of the conditions at lines 22 and 26 hold. Hence at the end of the first inner while loop, p_2^e is set as the new `front`, with $\mathring{\Gamma}_1$ containing the spline segment from p_1^s to p_1^e and the spline segment from p_2^s to p_2^e. In contrast, in subplot (d), the splinegon at the left of γ_2 is $\mathcal{C}_r = \mathcal{C}_2$, the splinegon with its edges as the spline segment γ_2 and the directed linear segments $\overrightarrow{p_1^s v_1}$, $\overrightarrow{v_1 v_2}$, $\overrightarrow{v_2 p_2^s}$. Since $\gamma_1 \not\subset \mathcal{C}_2$, the condition at line 17 does not hold and $p_{\min}$ is set to v_1. Then the condition at line 22 holds while none of the conditions at lines 26 and 28 holds. Hence at the end of the first inner while loop, v_1 is set as the new `front`, with $\mathring{\Gamma}_1$ containing the spline segment from p_1^s to p_1^e and the linear segment $\overrightarrow{p_1^s v_1}$.

THEOREM A.1. *Consider a simple linear polygon $\mathcal{C}$ with positively oriented boundary $\partial\mathcal{C}$ and an oriented Jordan curve $\mathring{\gamma} \neq \partial\mathcal{C}$. If $\mathring{\gamma} \cap \partial\mathcal{C}$ consists of two or more points, Algorithm 1 generates a set of positively oriented and pairwise almost disjoint Jordan curves $\left\{\mathring{\Gamma}_i\right\}$ such that*

$$\mathcal{C} \cap int\,(\mathring{\gamma}) = \cup_i^{\perp\perp} int\left(\mathring{\Gamma}_i\right). \tag{A.1}$$

Proof. By Theorem 2.5, $\mathcal{Y} := \mathcal{C} \cap \text{int}\,(\mathring{\gamma})$ must be a Yin set because both $\mathcal{C}$ and $\text{int}(\mathring{\gamma})$ are Yin sets. By (2.7) and (2.8), the boundary of each connected component of $\mathcal{Y}$ is uniquely represented by Jordan curves that are pairwise almost disjoint. By enumerating all possibilities on the relative position of $\mathring{\gamma}$ and $\partial\mathcal{C}$, one can show that the conditions $\mathring{\gamma} \neq \partial\mathcal{C}$ and $\#\{\mathring{\gamma} \cap \partial\mathcal{C}\} \geq 2$ imply that $\mathcal{C} \cap \text{int}\,(\mathring{\gamma})$ are the regularized union of a finite set of pairwise disjoint Yin sets, each of which is homeomorphic to the open ball. In other words, the form in (A.1) is a simplification of (2.8), due to the fact that each of $\mathcal{C}$ and $\text{int}(\mathring{\gamma})$ has their boundary as a single Jordan curve.

The boundary of $\mathcal{Y}$ must be a subset of $\partial\mathcal{C} \cup \mathring{\gamma}$. In addition, the isolated points given in $P_\cap$ divide the input Jordan curve $\mathring{\gamma}$ into a chain of directed curve segments; this also holds for $\partial\mathcal{C}$. Each Jordan curve $\mathring{\Gamma}_i$ in the output is a concatenation of some of these directed curve segments. In addition, the segments of $\mathring{\gamma}$ that are outside $\mathcal{C}$ cannot be part of $\partial\mathcal{Y}$ whereas those inside $\mathcal{C}$ must be. Similarly, for segments of $\partial\mathcal{C}$, those outside $\text{int}\,(\mathring{\gamma})$ cannot be part of $\partial\mathcal{Y}$ whereas those inside $\text{int}\,(\mathring{\gamma})$ must be.

Because both $\partial\mathcal{C}$ and $\mathring{\gamma}$ are simple curves, there are at most four incident segments at each of their intersection points, two from $\mathring{\gamma}$ and the other two from $\partial\mathcal{C}$. Hence the key of Algorithm 1 is to connect the directed segments of $\partial\mathcal{C}$ inside $\text{int}\,(\mathring{\gamma})$ and those of $\mathring{\gamma}$ inside $\mathcal{C}$ to form oriented Jordan curves in the output. The correctness of this

core aspect of Algorithm 1 can be shown by an induction on the number of segments appended into $\mathring{\Gamma}_c$. As the induction basis, the first segment $\gamma_r \subset \mathring{\gamma}$ clearly exists and is inside $\mathcal{C}$. In the inductive step, the choice of the next segment to be appended consists of several cases as follows.

(NS-1) `front` is not an intersection point, and hence it must be a vertex of $\mathcal{C}$. This case is determined by line 20: the next segment must be a line segment on $\partial\mathcal{C}$.

(NS-2) `front`$= p_r^e$ is a proper intersection. Hence the other incident segment in $\mathring{\gamma}$ cannot belong to the boundary of $\mathcal{C} \cap \mathrm{int}\,(\mathring{\gamma})$; this justifies lines 19-21 and lines 22-24 of Algorithm 1.

(NS-3) p_r^e is an improper intersection. Then by definition the other incident segment in $\mathring{\gamma}$ might also belong to the boundary of $\mathcal{C} \cap \mathrm{int}\,(\mathring{\gamma})$. However, if this is the case, both incident segments must be in the same splinegon $\mathcal{C}_r$ as defined at line 16, because the inner while loop in lines 15-32 is for assembling a single Jordan loop. This justifies lines 16-18 and lines 28-31 of Algorithm 1.

To sum up, the inner while loop corresponding to the inductive step maintains an invariant that each segment appended to $\mathring{\Gamma}_i$ be a subset of $\partial\mathcal{Y}$. The loop terminates when `anchor` is discovered again, which will eventually happen because the propagation of the `front` on $\partial\mathcal{C}$ is among a closed sequence of a finite number of points.

The postcondition (ii) of Algorithm 1 holds because of the following.

- Intersection points of the output Jordan curves form a subset of those of the input Jordan curves.
- A proper intersection of the input Jordan curves is never an intersection of the output Jordan curves because two of the four incident edges at this intersection, one from $\mathring{\gamma}$ and one from $\partial\mathcal{C}$, are not even in the output Jordan curves any more.
- By the discussions in (NS-3), an improper intersection of the input Jordan curves must either be an improper intersection of the output Jordan curves or a nonintersection.

Finally, (A.1) holds because each Jordan curve $\mathring{\Gamma}_i$ has positive orientation and their boundary are pairwise almost disjoint. □

Acknowledgment. The author would like to thank two anonymous referees, whose comments lead to an improvement of the exposition.

REFERENCES

[1] H. T. Ahn and M. Shashkov, *Multi-material interface reconstruction on generalized polyhedral meshes*, J. Comput. Phys., 226 (2007), pp. 2096–2132.

[2] ———, *Adaptive moment-of-fluid method*, J. Comput. Phys., 228 (2009), pp. 2792–2821.

[3] E. Aulisa, S. Manservisi, and R. Scardovelli, *A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking*, J. Comput. Phys., 197 (2004), pp. 555–584.

[4] J. W. Banks, T. Aslam, and W. J. Rider, *On sub-linear convergence for linearly degenerate waves in capturing schemes*, J. Comput. Phys., 227 (2008), pp. 6985 – 7002.

[5] S. Burris and H. P. Sankappanavar, *A Course in Universal Algebra*, Springer, the millennium ed., 2012. ISBN:978-0-9880552-0-9.

[6] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons, Ltd, 3rd ed., 2016. ISBN:9781119121503.

[7] R. Comminal, J. Spangenberg, and J. H. Hattel, *Cellwise conservative unsplit advection for the volume of fluid method*, J. Comput. Phys., 283 (2015), pp. 582–608.

[8] M. Coquerelle and S. Glockner, *A fourth-order accurate curvature computation in a level set framework for two-phase flows subjected to surface tension forces*, J. Comput. Phys., 305 (2016), pp. 838–876.

[9] S. V. Diwakar, Sarit K. Das, and T. Sundararajan, *A quadratic spline based interface (QUASI) reconstruction algorithm for accurate tracking of two-phase flows*, J. Comput. Phys., 228 (2009), pp. 9107–9130.
[10] D. P. Dobkin, D. L. Souvaine, and C. J. Van Wyk, *Decomposition and intersection of simple splinegons*, Algorithmica, 3 (1988), pp. 473–485.
[11] J. R. Dormand and P. J. Prince, *High order embedded Runge-Kutta formulae*, J. Comput. Appl. Math., 7 (1981), pp. 67–75.
[12] V. Dyadechko and M. Shashkov, *Moment-of-fluid interface reconstruction*, Tech. Report LA-UR-05-7571, Los Alamos National Laboratory, 2005. `http://cnls.lanl.gov/~shashkov/`.
[13] ———, *Reconstruction of multi-material interfaces from moment data*, J. Comput. Phys., 227 (2008), pp. 5361–5384.
[14] A. Eigenwillig, L. Kettner, E. Schomer, and N. Wolpert, *Exact, efficient, and complete arrangement computation for cubic curves*, Computational Geometry, 35 (2006), pp. 36–73. Special Issue on the 20th ACM Symposium on Computational Geometry.
[15] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell, *A hybrid particle level set method for improved interface capturing*, J. Comput. Phys., 183 (2002), pp. 83–116.
[16] I. Ginzburg and G. Wittum, *Two-phase flows on interface refined grids modeled with VOF, staggered finite volumes, and spline interpolants*, J. Comput. Phys., 166 (2001), pp. 302–335.
[17] S. Givant and P. Halmos, *Introduction to Boolean Algebras*, Springer, 2009. ISBN:978-0-387-40293-2.
[18] D. J. E. Harvie and D. F. Fletcher, *A new volume of fluid advection algorithm: The stream scheme*, J. Comput. Phys., 162 (2000), pp. 1–32.
[19] M. Herrmann, *A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids*, J. Comput. Phys., 227 (2008), pp. 2674–2706.
[20] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, 2nd ed., 2002. ISBN: 0-89871-521-0.
[21] C. W. Hirt and B. D. Nichols, *Volume of fluid (VOF) method for the dynamics of free boundaries*, J. Comput. Phys., 39 (1981), pp. 201–225.
[22] E. V. Huntington, *Sets of independent postulates for the algebra of logic*, Transactions of the American Mathematical Society, 5 (1904), pp. 288–309.
[23] Z. Jibben and M. Herrmann, *An arbitrary order Runge–Kutta discontinuous Galerkin approach to reinitialization for banded conservative level sets*, J. Comput. Phys., 349 (2017), pp. 453–473.
[24] L. Kettner, K. Mehlhorn, S. Pion, S. Schirra, and C. Yap, *Classroom examples of robustness problems in geometric computations*, Comput. Geom., 40 (2008), pp. 61–78.
[25] K. Kuratowski and A. Mostowski, *Set Theory, with an Introduction to Descriptive Set Theory*, vol. 86 of Studies in Logic and the Foundations of Mathematics, North-Holland Publishing Co., Amsterdam, 1976. ISBN: 978-0720404708.
[26] V. Le Chenadec and H. Pitsch, *A 3D unsplit forward/backward volume-of-fluid approach and coupling to the level set method*, J. Comput. Phys., 233 (2013), pp. 10–33.
[27] C.-Y. Li, S. V. Garimella, and J. E. Simpson, *Fixed-grid front–tracking algorithm for solidification prblems, part I: method and validation*, Numerical Heat Transfer, Part B: Fundamentals, 43 (2003), pp. 117–141.
[28] J. López, J. Hernández, P. Gómez, and F. Faura, *A volume of fluid method based on multidimensional advection and spline interface reconstruction*, J. Comput. Phys., 195 (2004), pp. 718–742.
[29] ———, *An improved PLIC-VOF method for tracking thin fluid structures in incompressible two-phase flows*, J. Comput. Phys., 208 (2005), pp. 51–74.
[30] H. MacNeille, *Partially ordered sets*, Trans. Amer. Math. Soc., 42 (1937), pp. 416–460.
[31] J. W. Milnor, *Topology from the Differentiable Viewpoint*, Princeton Landmarks in Mathematics and Physics, Princeton University Press, 1997. ISBN-13: 978-0691048338.
[32] S. Osher and J. A. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.
[33] A. A. G. Requicha, *Mathematical models of rigid solid objects*, Technical Memorandum 28, The University of Rochester, Rochester NY, November 1977.
[34] W. J. Rider and D. B. Kothe, *Reconstructing volume tracking*, J. Comput. Phys., 141 (1998), pp. 112–152.
[35] M. Rudman, *Volume-tracking methods for interfacial flow calculations*, Inter. J. Numer. Meth. Fluids, 24 (1997), pp. 671–691.
[36] H. Sagan, *An elementary proof that Schoenberg's space-filling curve is nowhere differentiable*, Mathematics Magazine, 65 (1992), pp. 125–128.

[37] A. Sard, *The measure of the critical values of differentiable maps*, Bull. Amer. Math. Soc., 48 (1942), pp. 883–890.
[38] R. Scardovelli and S. Zaleski, *Interface reconstruction with least-square fit and split Eulerian-Lagrangian advection*, Inter. J. Numer. Meth. Fluids, 41 (2003), pp. 251–274.
[39] A. Seidenberg, *A new decision method for elementry algebra*, Annals of Mathematics, 60 (1954), pp. 365–374.
[40] M. Sussman, A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. L. Welcome, *An adaptive level set approach for incompressible two-phase flows*, J. Comput. Phys., 148 (1999), pp. 81–124.
[41] M. Sussman and M. Y. Hussaini, *A discontinuous spectral element method for the level set equation*, J. Sci. Comput., 19 (2003), pp. 479–500.
[42] M. Sussman and E. Puckett, *A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows*, J. Comput. Phys., 162 (2000), pp. 301–337.
[43] A. Tarski, *über additive and multiplikative Mengenkörper und Mengenfunktionen*, Sprawozdania z Posiedzeń Towarzystwa Naukowego Warszawskiego, Wydzial III Nauk Matematyczno-fizycznych, 30 (1937), pp. 151–181.
[44] ———, *A decision method for elementary algebra and geometry*, U.S. Air Force Project Rand, R-109. Prepared for publication by J. C. C. McKinsey, The Rand Corporation, Santa Monica, California, 2nd ed. ed., 1948.
[45] G. Tryggvason, B. Bunner, D. Juric, W. Tauber, S. Nas, J. Han, N. Al-Rawahi., and Y.-J. Jan, *A front–tracking method for the computations of multiphase flow*, J. Comput. Phys., 169 (2001), pp. 708–759.
[46] J. H. Verner, *Explicit Runge–Kutta methods with estimates of the local truncation error*, SIAM J. Numer. Anal., 15 (1978), pp. 772–790.
[47] Y. Wang, S. Simakhina, and M. Sussman, *A hybrid level set-volume constraint method for incompressible two-phase flow*, J. Comput. Phys., 231 (2012), pp. 6348–6471.
[48] G. D. Weymouth and D. K.-P. Yue, *Conservative volume-of-fluid method for free-surface simulations on Cartesian-grids*, J. Comput. Phys., 229 (2010), pp. 2853–2865.
[49] S. T. Zalesak, *Fully multi-dimensional flux corrected transport algorithms for fluid flow*, J. Comput. Phys., 31 (1979), pp. 335–362.
[50] Q. Zhang, *On a family of unsplit advection algorithms for volume-of-fluid methods*, SIAM J. Numer. Anal., 51 (2013), pp. 2822–2850.
[51] ———, *On donating regions: Lagrangian flux through a fixed curve*, SIAM Review, 55 (2013), pp. 443–461.
[52] ———, *On generalized donating regions: Classifying Lagrangian fluxing particles through a fixed curve in the plane*, J. Math. Anal. Appl., 424 (2015), pp. 861–877.
[53] ———, *GePUP: Generic projection and unconstrained PPE for fourth-order solutions of the incompressible Navier-Stokes equations with no-slip boundary conditions*, J. Sci. Comput., 67 (2016), pp. 1134–1180.
[54] ———, *HFES: a height function method with explicit input and signed output for high-order estimations of curvature and unit vectors of planar curves*, SIAM J. Numer. Anal., 55 (2017), pp. 1024–1056.
[55] Q. Zhang and A. Fogelson, *Fourth-order interface tracking in two dimensions via an improved polygonal area mapping method*, SIAM J. Sci. Comput., 36 (2014), pp. A2369–A2400.
[56] ———, *MARS: An analytic framework of interface tracking via mapping and adjusting regular semi-algebraic sets*, SIAM J. Numer. Anal., 54 (2016), pp. 530–560.
[57] Q. Zhang and P. L.-F. Liu, *A new interface tracking method: The polygonal area mapping method*, J. Comput. Phys., 227 (2008), pp. 4063–4088.