mathematics

Article

Numerical Methods for a Two-Species Competition-Diffusion Model with Free Boundaries

Shuang Liu † and Xinfeng Liu *,†

Department of Mathematics, University of South Carolina, Columbia, SC 29208, USA; shuang@math.sc.edu

* Correspondence: xfliu@math.sc.edu; Tel.: +1-803-576-5849; Fax: +1-803-777-3783

† These authors contributed equally to this work.

Received: 31 March 2018; Accepted: 28 April 2018; Published: 3 May 2018

Abstract: The systems of reaction-diffusion equations coupled with moving boundaries defined by Stefan condition have been widely used to describe the dynamics of spreading population and with competition of two species. To solve these systems numerically, new numerical challenges arise from the competition of two species due to the interaction of their free boundaries. On the one hand, extremely small time steps are usually needed due to the stiffness of the system. On the other hand, it is always difficult to efficiently and accurately handle the moving boundaries especially with competition of two species. To overcome these numerical difficulties, we introduce a front tracking method coupled with an implicit solver for the 1D model. For the general 2D model, we use a level set approach to handle the moving boundaries to efficiently treat complicated topological changes. Several numerical examples are examined to illustrate the efficiency, accuracy and consistency for different approaches.

Keywords: competition-diffusion model; stefan problems; level set method; front-tracking method; front-fixing method

1. Introduction

The reaction-diffusion equations over a changing domain to describe the dynamics of a two-species competition-diffusion model usually take the following form,

$$\frac{\partial U}{\partial t} - D_1 \Delta U = f_1(U,V) \ \text{ for } \ \mathbf{x} \in \Omega_1(t),\ t>0; \ \ U = 0 \ \text{ for } \mathbf{x} \in \partial\Omega_1(t),\ t>0. \tag{1}$$

$$\frac{\partial V}{\partial t} - D_2 \Delta V = f_2(U,V) \ \text{ for } \ \mathbf{x} \in \Omega_2(t),\ t>0; \ \ U = 0 \ \text{ for } \mathbf{x} \in \partial\Omega_2(t),\ t>0. \tag{2}$$

The nonlinear functions $f_1(U,V)$ and $f_2(U,V)$ are assumed to be C^1 functions satisfying $f_i(\mathbf{0}) = 0$, i = 1,2, and in the literature they are often taken to be two-species Lotka-Volterra type competition functions. In the rest of this paper, we will take two-species Lotka-Volterra type competition functions as an example to demonstrate the numerical methods.

The evolution of the moving domains $\Omega_i(t) \subset \mathbb{R}^N$, $i = 1,2$, or rather their boundaries $\partial\Omega_i(t)$, i = 1,2 is determined by the one phase Stefan condition which, in the case $\partial\Omega_i(t)$, $i = 1,2$ are C^1 manifolds in $\mathbb{R}^N$. For example, the evolution of species U can be described as follows:

> For any point $\mathbf{x} \in \partial\Omega_1(t)$, it moves with velocity $\mu_1 |\nabla_{\mathbf{x}} U(t,\mathbf{x})| \mathbf{n}(\mathbf{x})$, where $\mathbf{n}(\mathbf{x})$ is the unit outward normal of $\Omega_1(t)$ at $\mathbf{x}$, and μ is a given positive constant.

The moving boundaries $\partial\Omega_1(t)$, $i = 1,2$ is generally called the "free boundary", and it is well known that, in general, their smoothness is not guaranteed, even if the initial function $U(0,\mathbf{x})$, $V(0,\mathbf{x})$ and initial domain $\Omega_i(0)$, $i = 1,2$ are both smooth.

Mathematically, the two-species competition-diffusion model with two free boundaries has been intensively studied for 1D [1–4] and for high dimensions with radial symmetry [5,6] through profound mathematical analysis. For instance, the existence of the positive traveling wave solutions connecting different constant equilibria has been addressed in [7–9]. The asymptotic spreading speed associated with the Cauchy problem has been studied in [10–12]. Many other theoretical results for general models have been achieved in [13–17] and references cited therein.

In contrast, very few numerical methods have been developed to solve such free boundary problems. Most recently, some efficient numerical methods have been introduced to solve the single species model [18]. In general, extremely small time steps are required due to the stiffness of the system. On the other hand, it is always difficult to efficiently and accurately handle the moving boundaries especially for two species. To efficiently handle the moving boundaries, level set methods [19–24] and front tracking methods [25–28] are two popular numerical approaches. One distinct feature of front tracking [29–34] is using a pure Lagrangian approach to explicitly track locations of interfaces, but it is difficult to handle topological bifurcations in high dimensions with interaction of two free boundaries, while the level set method can efficiently overcome such difficulties. In this paper, we will introduce a front tracking framework to solve the system for the one-dimensional case, and a level set approach is employed for two dimensions.

The rest of paper is organized as follows. In Section 2, we introduce two approaches for one-dimensional two-species competition system , i.e., a front tracking approach and a front fixing approach. In Section 3, a level set method is discussed for a more general two-dimensional case. In Section 4, numerical examples are performed to show the efficiency, accuracy and consistency for these different approaches. Finally, a brief conclusion is drawn in Section 5.

2. Numerical Methods for 1D Two-Species Competition-Diffusion Model

A two-species competition-diffusion model with two free boundaries for the density of population of the competing species U and V depending on time t and spatial variable x states as follows:

$$U_t - D_1 U_{xx} = \gamma_1 U(1 - U - K_1 V), \qquad t > 0, \quad 0 < x < S_1(t), \tag{3}$$

$$V_t - D_2 V_{xx} = \gamma_2 V(1 - V - K_2 U), \qquad t > 0, \quad 0 < x < S_2(t), \tag{4}$$

$$U_x(t,0) = V_x(t,0) = 0, \qquad t \geq 0, \tag{5}$$

$$S_1'(t) = -\mu_1 U_x(t, S_1(t)), \quad S_2'(t) = -\mu_2 V_x(t, S_2(t)), \qquad t \geq 0, \tag{6}$$

$$U(x,t) = 0 \quad for\ x \geq S_1(t), \quad V(x,t) = 0 \quad for\ x \geq S_2(t), \qquad t \geq 0, \tag{7}$$

$$U(x,0) = U_0(x), \quad V(x,0) = V_0(x), \qquad for\ x \in [0,\infty), \tag{8}$$

$$S_1(0) = S_1^0 > 0, \quad S_2(0) = S_2^0 > 0. \tag{9}$$

where $U(x,t)$ and $V(x,t)$ represent the population densities of the two species at the position x and time t. D_1 and D_2 are the diffusion rates of species U and V. γ_1 and γ_2 are net birth rates of species U and V. K_1 and K_2 are the competition coefficients of species U and V. The parameters μ_1 and μ_2 measure the intention to spread into the new territories of u and v. Here, the two free boundaries $S_1(t)$ and $S_2(t)$ describe the spreading fronts of two competing species $U(t,x)$ and $V(t,x)$. We envision that the two species initially occupy the interval $[0, S_1(0)]$ and $[0, S_2(t)]$, respectively, at time t.

2.1. Method 1: Front-Tracking Method for 1D Two-Species Competition-Diffusion Model

The problem lies in solving the nonlinear parabolic partial differential Equations (3)–(9) in the unbounded fixed domain $(0,\infty) \times (0,L)$ for the variables (t,x). Let us consider the step size discretization $k = \triangle t$, $h = \triangle x = L/M$, and the mesh points (t^n, x_j), with

$t^n = kn$, $n \geq 0$, $x_j = jh$, $0 \leq j \leq M$ and M positive integer. Let us denote the approximate value of $U(t^n, x_j)$ and the approximate value of $V(t^n, x_j)$ at the mesh point (t^n, x_j) ,

$$u_j^n \approx U(t^n, x_j), \quad v_j^n \approx V(t^n, x_j)$$

Step1: Track the position of the moving front $S_1(t)$.

According to the Stefan condition

$$S_1'(t) = -\mu_1 U_x(t, S_1(t)), \quad t \geq 0, \tag{10}$$

we consider using the central approximation of the spatial derivatives to approximate $\frac{\partial U}{\partial x}(t, S_1(t))$, which can be divided into the following four cases.

1 When $x_i \leq S_1^n < x_{i+1}$, $i = 2, 3...M-1$ as depicted in Figure 1, denoting $d = \frac{S_1^n - x_i}{h}$. Let us first consider the symmetric point of x_{i-1} respect to the position S_1^n, which is denoted by $\tilde{x}_{i-1}$. In particular, when $S_1^n = x_i$, $\tilde{x}_{i-1} = x_{i+1}$. We use the Lagrange interpolation to construct polynomial P^L from the value of d, h, u_{i-2}^n, u_{i-1}^n, u_i^n and S_1^n, thus at $\tilde{x}_{i-1}$, we use the value of P^L at $\tilde{x}_{i-1}$ instead of $u(\tilde{x}_{i-1})$,

$$\frac{\partial U}{\partial x}(t^n, S_1^n) \approx \frac{P^L(\tilde{x}_{i-1}) - u_{i-1}^n}{2(1+d)h}, \quad i = 2, 3, \ldots, M-1.$$

$$\frac{\partial U}{\partial x}(t^n, S_1^n) \approx \frac{P^L(\tilde{x}_{i-1}) - u_{i-1}^n}{2(1+d)h}, \quad i = 2, 3, \ldots, M-1.$$

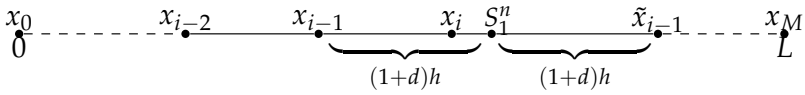

Figure 1. case 1: $x_i \leq S_1^n < x_{i+1}$.

2 When $0 = x_0 < S_1^n \leq x_1$, the central scheme approximation of the spatial derivatives to approximate $\frac{\partial U}{\partial x}(t, S_1(t))$ involves the fictitious value u_{-1}^n at the point $(t^n, -h)$. The value u_{-1}^n can be estimated from the second-order discretization of the boundary condition (10),

$$\frac{u_1^n - u_{-1}^n}{2h} = 0 \tag{11}$$

which implies that $u_{-1}^n = u_1^n = 0$. It is obvious that all the values of u_i^n on the grid points are equal to 0 except u_0^n. Numerically, we take $S_1'(t) = 0$, and this can be explained by the fact the species is only located inside one grid mesh. The simulation should stop here indicating that a more refined mesh is needed.

3 When $x_1 < S_1^n < x_2$ as shown in Figure 2, denoting $d = \frac{S_1^n - x_1}{h}$. Let us first consider the symmetric point of x_0 with respect to the position S_1^n, which is denoted by $\tilde{x}_0$. Then we consider the value of $u_{-1}^n = u_1^n$, and use the Lagrange interpolation to construct polynomial P^L from the value of h, d, u_{-1}^n, u_0^n, u_1^n and S_1^n. Then at $\tilde{x}_0$, we use the value of P^L at $\tilde{x}_0$ instead of $u(\tilde{x}_0)$.

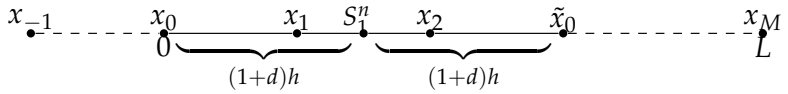

Figure 2. case 3: $x_1 < S_1^n < x_2$.

4 When $S_1^n = x_M$, it implies that the spreading of the populations already goes out of the computational domain $[0, L]$, and the simulation should stop here.

Step2: Track the position of the moving front of $S_2(t)$.

Repeat step1 to $S_2(t)$.

Step3: Update the value of $U(t^{n+1}, x_i)$ *and* $V(t^{n+1}, x_i)$.

1 When $x_i = S_1^{n+1}$ and $x_j = S_2^{n+1}$, then we know that $U(t^{n+1}, x_i) = 0$. Let $u_i^{n+1} = 0$, $u_l^{n+1} = 0$, for $l = i+1, i+2...M$ and $v_j^{n+1} = 0$, $v_m^{n+1} = 0$, for $m = j+1, j+2...M$. We consider the central approximation of the spatial derivatives U_{xx} at x_l, for $l = 0, 1, 2, ..., i-1$, and the central approximation of the spatial derivatives V_{xx} at x_m, for $m = 0, 1, 2, ..., j-1$, where U and V are updated by the backward Euler

$$\begin{cases} \dfrac{u_l^{n+l} - u_l^n}{k} = D_1 \dfrac{u_{l-1}^{n+1} - 2u_l^{n+1} + u_{l+1}^{n+1}}{h^2} + \gamma_1 u_l^{n+1}(1 - u_l^{n+1} - K_1 v_l^{n+1}),\ l = 0, 1, ...i-1. \\ \dfrac{v_m^{n+l} - v_m^n}{k} = D_2 \dfrac{v_{m-1}^{n+1} - 2v_m^{n+1} + v_{m+1}^{n+1}}{h^2} + \gamma_2 v_m^{n+1}(1 - v_m^{n+1} - K_2 u_m^{n+1}),\ m = 0, 1, ...j-1. \end{cases} \tag{12}$$

Then use the *Picard* Iteration (or Newton Iteration) to solve the nonlinear system (12).

2 When $x_i < S_1^{n+1} < x_{i+1}$ and $x_j < S_2^{n+1} < x_{j+1}$, denoting $R_1 = \frac{S_1^{n+1} - x_i}{h}$ and $R_2 = \frac{S_2^{n+1} - x_i}{h}$, we use the Lagrange interpolation to construct polynomial P_1^L from the value of h, R_1, u_{i-2}^{n+1}, u_{i-1}^{n+1}, u_i^{n+1} and S_1^{n+1} and polynomial P_2^L from the value of h, R_2, v_{j-2}^{n+1}, v_{j-1}^{n+1}, v_j^{n+1} and S_2^{n+1}. Then at x_{i+1} and x_{j+1}, we use the value of P_1^L at x_{i+1} instead of u_{i+1}^{n+1} and the value of P_2^L at x_{j+1} instead of v_{j+1}^{n+1}. For the solution u at x_l, for $l = 0, 1, 2, ..., i-1$, a standard central approximation in space with backward Euler in time will be employed. $u_l^{n+1} = 0$, for $l = i+1, ...M$. For the solution v at x_l, for $l = 0, 1, 2, ..., j-1$, a standard central approximation in space with backward Euler in time will be employed. $v_l^{n+1} = 0$, for $l = j+1, ...M$. U and V is updated by the backward Euler in time

$$\begin{cases} \dfrac{u_l^{n+l} - u_l^n}{k} = D_1 \dfrac{u_{l-1}^{n+1} - 2u_l^{n+1} + u_{l+1}^{n+1}}{h^2} + \gamma_1 u_l^{n+1}(1 - u_l^{n+1} - K_1 v_l^{n+1}),\ l = 0, 1, ...i-1. \\ \dfrac{u_i^{n+1} - u_i^n}{k} = D_1 \dfrac{u_{i-1}^{n+1} - 2u_i^{n+1} + P_1^L(x_{i+1})}{h^2} + \gamma_1 u_i^{n+1}(1 - u_i^{n+1} - K_1 v_i^{n+1}). \\ \dfrac{v_m^{n+l} - v_m^n}{k} = D_2 \dfrac{v_{m-1}^{n+1} - 2v_m^{n+1} + v_{m+1}^{n+1}}{h^2} + \gamma_2 v_m^{n+1}(1 - v_m^{n+1} - K_2 u_m^{n+1}),\ m = 0, 1, ...j-1. \\ \dfrac{v_j^{n+1} - v_j^n}{k} = D_2 \dfrac{v_{j-1}^{n+1} - 2v_j^{n+1} + P_2^L(x_{j+1})}{h^2} + \gamma_2 v_j^{n+1}(1 - v_j^{n+1} - K_2 u_j^{n+1}). \end{cases} \tag{13}$$

Picard Iteration (or Newton Iteration) will be applied to solve the nonlinear system (13).

3 When $x_i = S_1^{n+1}$ and $x_j < S_2^{n+1} < x_{j+1}$, then we know that $U(t^{n+1}, x_i) = 0$. Let $u_i^{n+1} = 0$, $u_l^{n+1} = 0$, for $l = i+1, ...M$. We consider the central approximation of the spatial derivatives U_{xx} at x_l, for $l = 0, 1, 2, ..., i-1$. Denoting $R_2 = \frac{S_2^{n+1} - x_i}{h}$, we use the Lagrange interpolation to construct polynomial P_2^L from the value of h, R_2, v_{j-2}^{n+1}, v_{j-1}^{n+1}, v_j^{n+1} and S_2^{n+1}. Then at x_{j+1}, we use the value of P_2^L at x_{j+1} instead of v_{j+1}^{n+1}, where U and V is updated by the backward Euler in time

$$\begin{cases} \dfrac{u_l^{n+l}-u_l^n}{k} = D_1\dfrac{u_{l-1}^{n+1}-2u_l^{n+1}+u_{l+1}^{n+1}}{h^2} + \gamma_1 u_l^{n+1}(1-u_l^{n+1}-K_1 v_l^{n+1}),\ l=0,1,...i-1. \\ \dfrac{v_m^{n+l}-v_m^n}{k} = D_2\dfrac{v_{m-1}^{n+1}-2v_m^{n+1}+v_{m+1}^{n+1}}{h^2} + \gamma_2 v_m^{n+1}(1-v_m^{n+1}-K_2 u_m^{n+1}),\ m=0,1,...j-1. \\ \dfrac{v_j^{n+1}-v_j^n}{k} = D_2\dfrac{v_{j-1}^{n+1}-2v_j^{n+1}+P_2^L(x_{j+1})}{h^2} + \gamma_2 v_j^{n+1}(1-v_j^{n+1}-K_2 u_j^{n+1}). \end{cases} \quad (14)$$

Then use the *Picard* Iteration (or Newton Iteration) to solve the nonlinear system (14).

4 When $x_i < S_1^{n+1} < x_{i+1}$ and $x_j = S_2^{n+1}$, denoting $R_1 = \frac{S_1^{n+1}-x_i}{h}$, we use the Lagrange interpolation to construct polynomial P_1^L from the value of h, R_1, u_{i-2}^{n+1}, u_{i-1}^{n+1}, u_i^{n+1} Then at x_{i+1}, we use the value of P_1^L at x_{i+1} instead of u_{i+1}^{n+1}. For the solution u at x_l, for $l = 0, 1, 2, ..., i-1$, a standard central approximation in space with backward Euler in time will be employed. $u_l^{n+1} = 0$, for $l = i+1, ...M$. For the solution v at x_l, for $l = 0, 1, 2, ..., j-1$, a standard central approximation in space with backward Euler in time will be employed. $v_l^{n+1} = 0$, for $l = j+1, ...M$, where U and V is updated by the backward Euler in time

$$\begin{cases} \dfrac{u_l^{n+l}-u_l^n}{k} = D_1\dfrac{u_{l-1}^{n+1}-2u_l^{n+1}+u_{l+1}^{n+1}}{h^2} + \gamma_1 u_l^{n+1}(1-u_l^{n+1}-K_1 v_l^{n+1}),\ l=0,1,...i-1. \\ \dfrac{u_i^{n+1}-u_i^n}{k} = D_1\dfrac{u_{i-1}^{n+1}-2u_i^{n+1}+P_1^L(x_{i+1})}{h^2} + \gamma_1 u_i^{n+1}(1-u_i^{n+1}-K_1 v_i^{n+1}). \\ \dfrac{v_m^{n+l}-v_m^n}{k} = D_2\dfrac{v_{m-1}^{n+1}-2v_m^{n+1}+v_{m+1}^{n+1}}{h^2} + \gamma_2 v_m^{n+1}(1-v_m^{n+1}-K_2 u_m^{n+1}),\ m=0,1,...j-1. \end{cases} \quad (15)$$

Picard Iteration (or Newton Iteration) will be applied to solve the nonlinear system (15).

2.2. Method 2: Front-Fixing Method for 1D Two-Species Competition-Diffusion Model

Here we consider transforming Equation (3) and Equation (4) into problems with a fixed domain $[0, 1]$ separately.

Step1. Update the front of $S_1(t)$ and the value of U by front fixing method.

Let us transform Equation (3) into a problem with a fixed domain $[0, 1]$ using the Landau transformation [35,36]

$$y(t,x) = \frac{x}{S_1(t)}, \quad M(t,y) = U(t,x), \quad WtoM(t,y) = V(t,x). \quad (16)$$

Then Equation (3) turns into the form:

$$H(t)\frac{\partial M}{\partial t} - H'(t)\frac{y}{2}\frac{\partial M}{\partial y} - D_1\frac{\partial^2 M}{\partial y^2} = H(t)\gamma_1 M(1-M-K_1 WtoM), \quad t>0, \quad 0<y<1, \quad (17)$$

where:

$$H(t) = S_1^2(t), \quad t \geq 0. \quad (18)$$

Boundary conditions (5) and Stefan condition (6) take the form:

$$\frac{\partial M}{\partial y}(t,0) = 0, \qquad M(t,1) = 0, \quad t>0, \quad (19)$$

and

$$H'(t) = -2\mu_1 \frac{\partial M}{\partial y}(t,1), \quad t > 0, \tag{20}$$

respectively, while the initial conditions (9) become:

$$H(0) = (S_1^0)^2, \quad M(0,y) = M_0(y) = U_0(yS_1^0), \quad 0 \le y \le 1. \tag{21}$$

Conditions (8) for the initial function $U_0(x)$ are translated to $W_0(z)$ as follows:

$$M_0(y) \in C^2([0,1]), \quad M_0'(0) = M_0(1) = 0, \quad M_0(y) > 0, \quad 0 \le y < 1. \tag{22}$$

After the transformation, the new problem has been changed to solve the nonlinear parabolic partial differential equations (17) in the unbounded fixed domain $(0,\infty) \times (0,1)$ for the variables (t,y). Let us consider the step size discretization $k = \triangle t$, $h = \triangle y = 1/M$, and the mesh points (t^n, y_j), with $t^n = kn$, $n \ge 0$, $y_j = jh$, $0 \le j \le M$ and M positive integer. Let us denote the approximate value of $M(t^n, y_j)$ at the mesh point (t^n, y_j),

$$m_j^n \approx M(t^n, y_j), \quad wtoM_j^n \approx WtoM(t^n, y_j). \tag{23}$$

and let H^n be the approximation of $H(t^n)$. Let us consider the forward approximation of the time derivatives,

$$\frac{m_j^{n+1} - m_j^n}{k} \approx \frac{\partial M}{\partial t}(t^n, y_j), \quad \frac{H^{n+1} - H^n}{k} \approx H'(t^n), \tag{24}$$

and the central approximation of the spatial derivatives,

$$\frac{m_{j+1}^n - m_{j-1}^n}{2h} \approx \frac{\partial M}{\partial y}(t^n, y_j), \quad \frac{m_{j-1}^n - 2m_j^n + m_{j+1}^n}{h^2} \approx \frac{\partial^2 M}{\partial y^2}(t^n, y_j). \tag{25}$$

From (24) and (25), Equation (17) is approximated by:

$$H^n \frac{m_j^{n+1} - m_j^n}{k} - \frac{y_j}{2} \frac{m_{j+1}^n - m_{j-1}^n}{2h} \left(\frac{H^{n+1} - H^n}{k}\right) - D_1 \frac{m_{j-1}^n - 2m_j^n + m_{j+1}^n}{h^2} \tag{26}$$

$$= H^n \gamma_1 m_j^n (1 - m_j^n - wtoM_j^n), \; n \ge 0, \; 0 \le j \le M-1.$$

As usual, we assume that the Equation (26) can be also approximated at $j = 0$. Equation (26) written for $j = 0$ involves the fictitious value m_{-1}^n at the point $(t^n, -h)$. The value m_{-1}^n is eliminated from the discretization of the boundary and initial condition (21) and (22),

$$\frac{m_1^n - m_{-1}^n}{2h} = 0, \quad m_M^n = 0, \quad n \ge 0. \tag{27}$$

Transformed Stefan condition (20) is discretized using first order forward approximation for $H'(t)$ and three points backward spatial approximation of $\frac{\partial M}{\partial y}(t,1)$:

$$\frac{H^{n+1} - H^n}{k} = -\frac{\mu_1}{h}(3m_M^n - 4m_{M-1}^n + m_{M-2}^n), \quad n \ge 0. \tag{28}$$

$$H^{n+1} = H^n - \frac{\mu_1 k}{h}(3m_M^n - 4m_{M-1}^n + m_{M-2}^n), \quad n \ge 0.$$

to preserve accuracy of order $O(k) + O(h^2)$.

Finally, we have

$$\begin{cases} m_0^{n+1} = (1 - \dfrac{2D_1k}{H^n h^2} + k\gamma_1(1 - m_0^n - K_1 w_0^n))m_0^n + 2\dfrac{D_1k}{H^n h^2} m_1^n. \\ m_j^{n+1} = a_j^n m_{j-1}^n + b_j^n m_j^n + c_j^n m_{j+1}^n, \quad n \geq 0, \quad 0 < j \leq M-1. \\ m_M^{n+1} = 0. \end{cases} \tag{29}$$

where the coefficients are given by:

$$\begin{aligned} a_j^n &= \frac{D_1k}{H^n h^2} - \frac{y_j \mu_1 k(4m_{M-1}^n - m_{M-2}^n)}{4h^2 H^n}, \\ b_j^n &= 1 - \frac{2D_1k}{H^n h^2} + k\gamma_1(1 - m_j^n - K_1 wtoM_j^n), \\ c_j^n &= \frac{D_1k}{H^n h^2} + \frac{y_j \mu_1 k(4m_{M-1}^n - m_{M-2}^n)}{4h^2 H^n}. \end{aligned}$$

Step2. Update the front $S_2(t)$ and the value of V by front fixing method.

Let us transform Equation (4) into a problem with a fixed domain $[0,1]$ using the Landau transformation [35,36]

$$z(t,x) = \frac{x}{S_2(t)}, \quad W(t,z) = V(t,x), \quad MtoW(t,z) = U(t,x). \tag{30}$$

Then Equation (4) turns into the form:

$$G(t)\frac{\partial W}{\partial t} - G'(t)\frac{z}{2}\frac{\partial W}{\partial z} - D_2\frac{\partial^2 W}{\partial z^2} = G(t)\gamma_2 W(1 - W - K_2 MtoW), \quad t > 0, \quad 0 < z < 1, \tag{31}$$

where:

$$G(t) = S_2^2(t), \quad t \geq 0. \tag{32}$$

Boundary conditions (5) and Stefan condition (6) take the form:

$$\frac{\partial W}{\partial z}(t,0) = 0, \qquad W(t,1) = 0, \quad t > 0, \tag{33}$$

and

$$G'(t) = -2\mu_2 \frac{\partial W}{\partial z}(t,1), \quad t > 0, \tag{34}$$

respectively, while the initial conditions (9) become:

$$G(0) = (S_2^0)^2, \quad W(0,z) = W_0(z) = V_0(zS_2^0), \quad 0 \leq z \leq 1. \tag{35}$$

Conditions (8) for the initial function $U_0(x)$ are translated to $W_0(z)$ as follows:

$$W_0(z) \in C^2([0,1]), \quad W_0'(0) = W_0(1) = 0, \quad W_0(z) > 0, \quad 0 \leq z < 1. \tag{36}$$

After the transformation, the new problem lies in solving the nonlinear parabolic partial differential equations (31) in the unbounded fixed domain $(0,\infty) \times (0,1)$ for the variables (t,z). Let us consider the step size discretization $k = \triangle t$, $h = \triangle z = 1/M$, and the mesh points (t^n, z_j), with $t^n = kn$, $n \geq 0$, $z_j = jh$, $0 \leq j \leq M$ and M positive integer. Let us denote the approximate value of $W(t^n, z_j)$ at the mesh point (t^n, z_j),

$$w_j^n \approx W(t^n, z_j), \quad mtoW_j^n \approx MtoW(t^n, z_j). \tag{37}$$

and let G^n be the approximation of $G(t^n)$. Let us consider the forward approximation of the time derivatives,

$$\frac{w_j^{n+1}-w_j^n}{k} \approx \frac{\partial W}{\partial t}(t^n, z_j), \quad \frac{G^{n+1}-G^n}{k} \approx G'(t^n), \tag{38}$$

and the central approximation of the spatial derivatives,

$$\frac{w_{j+1}^n - w_{j-1}^n}{2h} \approx \frac{\partial W}{\partial z}(t^n, z_j), \quad \frac{w_{j-1}^n - 2w_j^n + w_{j+1}^n}{h^2} \approx \frac{\partial^2 W}{\partial z^2}(t^n, z_j). \tag{39}$$

From (38) and (39), Equation (31) is approximated by:

$$G^n \frac{w_j^{n+1}-w_j^n}{k} - \frac{z_j}{2}\frac{w_{j+1}^n - w_{j-1}^n}{2h}\left(\frac{G^{n+1}-G^n}{k}\right) - D_2\frac{w_{j-1}^n - 2w_j^n + w_{j+1}^n}{h^2} \tag{40}$$

$$= G^n \gamma_2 w_j^n (1 - w_j^n - K_2 mtoW_j^n),\ n \geq 0,\ 0 \leq j \leq M-1.$$

As usual, we again assume that Equation (40) can be also approximated at $j = 0$. Equation (40) written for $j = 0$ involves the fictitious value w_{-1}^n at the point $(t^n, -h)$. The value w_{-1}^n is eliminated from the discretization of the boundary and initial condition (35) and (36),

$$\frac{w_1^n - w_{-1}^n}{2h} = 0, \quad w_M^n = 0, \quad n \geq 0. \tag{41}$$

Transformed Stefan condition (34) is discretized using first order forward approximation for $G'(t)$ and three points backward spatial approximation of $\frac{\partial W}{\partial z}(t, 1)$:

$$\frac{G^{n+1}-G^n}{k} = -\frac{\mu_2}{h}(3w_M^n - 4w_{M-1}^n + w_{M-2}^n), \quad n \geq 0. \tag{42}$$

$$G^{n+1} = G^n - \frac{\mu_2 k}{h}(3w_M^n - 4w_{M-1}^n + w_{M-2}^n), \quad n \geq 0.$$

to preserve accuracy of order $O(k) + O(h^2)$.

Finally, we have

$$\begin{cases} w_0^{n+1} = (1 - \dfrac{2D_2 k}{G^n h^2} + k\gamma_2(1 - w_0^n - K_2 mtoW_0^n))w_0^n + 2\dfrac{D_2 k}{G^n h^2} w_1^n. \\ w_j^{n+1} = A_j^n w_{j-1}^n + B_j^n w_j^n + C_j^n w_{j+1}^n, \quad n \geq 0, \quad 0 < j \leq M-1. \\ w_M^{n+1} = 0. \end{cases} \tag{43}$$

where the coefficients are given by:

$$A_j^n = \frac{D_2 k}{G^n h^2} - \frac{z_j \mu_2 k(4w_{M-1}^n - w_{M-2}^n)}{4h^2 G^n},$$
$$B_j^n = 1 - \frac{2D_2 k}{G^n h^2} + k\gamma_2(1 - w_j^n - K_2 wtoM_j^n),$$
$$C_j^n = \frac{D_2 k}{G^n h^2} + \frac{z_j \mu_2 k(4w_{M-1}^n - w_{M-2}^n)}{4h^2 G^n}.$$

Step3. Update the value of $WtoM(t^n, y_i)$ *with the front information* G^{n+1} *and* H^{n+1}.

1 When $y_i = \sqrt{\frac{G^{n+1}}{H^{n+1}}}$, then we know that $WtoM(t^{n+1}, y_i) = 0$. Let $wtoM_i^{n+1} = 0$, $wtoM_l^{n+1} = 0$, for $l = i+1, ...M$. We consider the central approximation of the spatial derivatives $\frac{\partial^2 WtoM}{\partial y^2}$at y_j, for $j = 0, 1, 2, ..., i-1$, where $WtoM$ is updated by the backward Euler

$$\begin{cases} wtoM_0^{n+1} = (1 - \frac{2D_1 k}{H^n h^2} + k\gamma_1(1 - wtoM_0^n - K_1 m_0^n))wtoM_0^n + 2\frac{D_1 k}{H^n h^2} wtoM_1^n. \\ wtoM_j^{n+1} = a_j^n wtoM_{j-1}^n + b_j^n wtoM_j^n + c_j^n wtoM_{j+1}^n, \quad n \geq 0, \quad 0 < j \leq i-1. \\ wtoM_j^{n+1} = 0, \quad j = i, i+1, ...M. \end{cases} \tag{44}$$

2 When $y_i < \sqrt{\frac{G^{n+1}}{H^{n+1}}} < y_{i+1}$, denoting $R = \frac{\sqrt{G^{n+1}/H^{n+1}} - y_i}{h}$, we use the Lagrange interpolation to construct polynomial P^L from the value of h, R, $wtoM_{i-2}^{n+1}$, $wtoM_{i-1}^{n+1}$, $wtoM_i^{n+1}$ and $\sqrt{\frac{G^{n+1}}{H^{n+1}}}$, We consider to use the value of P^L at y_{i+1} instead of $wtoM_{j+1}^{n+1}$.

$$\begin{cases} wtoM_0^{n+1} = (1 - \frac{2D_1 k}{H^n h^2} + k\gamma_1(1 - wtoM_0^n - K_1 m_0^n))wtoM_0^n + 2\frac{D_1 k}{H^n h^2} wtoM_1^n. \\ wtoM_j^{n+1} = a_j^n wtoM_{j-1}^n + b_j^n wtoM_j^n + c_j^n wtoM_{j+1}^n, \quad n \geq 0, \quad 0 < j \leq i-1. \\ wtoM_i^{n+1} = a_i^n wtoM_{i-1}^n + b_i^n wtoM_i^n + c_i^n P^L(y_{i+1}). \\ wtoM_j^{n+1} = 0, \quad j = i+1, ...M. \end{cases} \tag{45}$$

3 When $\sqrt{\frac{G^{n+1}}{H^{n+1}}} \geq 1$ as shown in Figure 3, we consider the central approximation of the spatial derivatives $\frac{\partial^2 WtoM}{\partial y^2}$at y_j, for $j = 0, 1, 2, ..., M-1$, for $WtoM(t^{n+1}, y_M)$, we know that $W(t^{n+1}, \sqrt{\frac{H^{n+1}}{G^{n+1}}}) = WtoM(t^{n+1}, y_M)$, let us approximate $W(t^{n+1}, \sqrt{\frac{H^{n+1}}{G^{n+1}}})$ instead of approximating $WtoM(t^{n+1}, y_M)$. Suppose $z_i < \sqrt{\frac{H^{n+1}}{G^{n+1}}} \leq z_{i+1}$, we use the Lagrange interpolation to construct polynomial P from the value of z_i, z_{i-1}, z_{i+1}, W_{i-1}^n, W_i^n, and W_{i+1}^n, then $WtoM(t^{n+1}, y_M) = W(t^{n+1}, \sqrt{\frac{H^{n+1}}{G^{n+1}}}) \approx P(\sqrt{\frac{H^{n+1}}{G^{n+1}}})$.

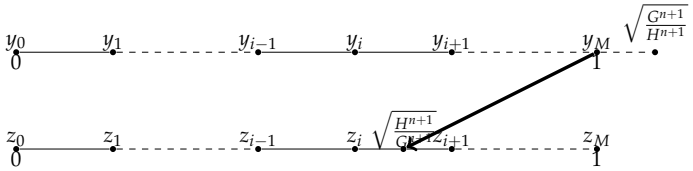

Figure 3. when $\sqrt{\frac{G^{n+1}}{H^{n+1}}} \geq 1$, $WtoM(t^{n+1}, y_M) = W(t^{n+1}, \sqrt{\frac{H^{n+1}}{G^{n+1}}})$.

Step4. Update the value of $MtoW(t^n, z_i)$ with the front information G^{n+1} and H^{n+1}.

1 When $z_i = \sqrt{\frac{H^{n+1}}{G^{n+1}}}$, then we know that $MtoW(t^{n+1}, z_i) = 0$. Let $mtoW_i^{n+1} = 0$, $mtoW_l^{n+1} = 0$, for $l = i+1, ...M$. We consider the central approximation of the spatial derivatives $\frac{\partial^2 MtoW}{\partial z^2}$at z_j, for $j = 0, 1, 2, ..., i-1$, where M is updated by the backward Euler

$$\begin{cases} mtoW_0^{n+1} = (1 - \frac{2D_2 k}{G^n h^2} + k\gamma_2(1 - mtoW_0^n - K_2 w_0^n))mtoW_0^n + 2\frac{D_2 k}{G^n h^2} mtoW_1^n. \\ mtoW_j^{n+1} = A_j^n mtoW_{j-1}^n + B_j^n mtoW_j^n + C_j^n mtoW_{j+1}^n, \quad n \geq 0, \quad 0 < j \leq i-1. \\ mtoW_j^{n+1} = 0. j = i, i+1, ..M. \end{cases} \tag{46}$$

2 When $z_i < \sqrt{\frac{H^{n+1}}{G^{n+1}}} < z_{i+1}$, denoting $R = \frac{\sqrt{H^{n+1}/G^{n+1}}-x_i}{h}$, we use the Lagrange interpolation to construct polynomial P^L from the value of h, R, $mtoW_{i-2}^{n+1}$, $mtoW_{i-1}^{n+1}$, $mtoW_i^{n+1}$ and $\sqrt{\frac{H^{n+1}}{G^{n+1}}}$, We consider to use the value of P^L at z_{i+1} instead of $mtoW_{i+1}^{n+1}$.

$$\begin{cases} mtoW_0^{n+1} = (1 - \frac{2D_2k}{G^n h^2} + k\gamma_2(1 - mtoW_0^n - K_2 w_0^n))mtoW_0^n + 2\frac{D_2 k}{G^n h^2} mtoW_1^n. \\ mtoW_j^{n+1} = A_j^n mtoW_{j-1}^n + B_j^n mtoW_j^n + C_j^n mtoW_{j+1}^n, \quad n \geq 0, \quad 0 < j \leq i-1. \\ mtoW_i^{n+1} = A_i^n mtoW_{i-1}^n + B_i^n mtoW_i^n + C_i^n P^L(z_{i+1}). \\ mtoW_j^{n+1} = 0. j = i+1, ..M. \end{cases} \tag{47}$$

3 When $\sqrt{\frac{H^{n+1}}{G^{n+1}}} \geq 1$ as illustrated in Figure 4, we consider the central approximation of the spatial derivatives $\frac{\partial^2 MtoW}{\partial z^2}$at z_j, for $j = 0, 1, 2, ..., M-1$, for $MtoW(t^{n+1}, z_M)$, we know that $M(t^{n+1}, \sqrt{\frac{G^{n+1}}{H^{n+1}}}) = MtoW(t^{n+1}, z_M)$, let us approximate $M(t^{n+1}, \sqrt{\frac{G^{n+1}}{H^{n+1}}})$ instead of approximating $MtoW(t^{n+1}, z_M)$. Suppose $y_i < \sqrt{\frac{G^{n+1}}{H^{n+1}}} \leq y_{i+1}$, we use the Lagrange interpolation to construct polynomial P from the value of y_i, y_{i-1}, y_{i+1}, M_{i-1}^n, M_i^n, and M_{i+1}^n, then $MtoW(t^{n+1}, z_M) = M(t^{n+1}, \sqrt{\frac{G^{n+1}}{H^{n+1}}}) \approx P(\sqrt{\frac{G^{n+1}}{H^{n+1}}})$.

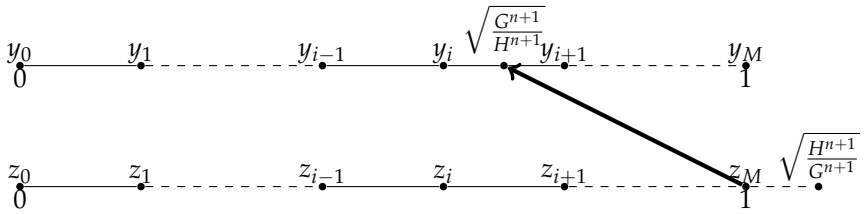

Figure 4. when $\sqrt{\frac{H^{n+1}}{G^{n+1}}} \geq 1$, $MtoW(t^{n+1}, Z_M) = M(t^{n+1}, \sqrt{\frac{G^{n+1}}{H^{n+1}}})$.

3. Level Set Method for 2D Two-Species Competition-Diffusion Model

A general 2D two-species competition-diffusive model for the densities of population of the species $U(t,x,y)$ and $V(t,x,y)$ depending on time t and spatial variable (x,y) states as follows:

$$\frac{\partial U}{\partial t} - D_1(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}) = \gamma_1 U(1 - U - K_1 V), \qquad t > 0, \quad (x,y) \in \Omega_1(t) \backslash \tau_1(t), \tag{48}$$

$$\frac{\partial V}{\partial t} - D_2(\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2}) = \gamma_2 V(1 - V - K_2 U), \qquad t > 0, \quad (x,y) \in \Omega_2(t) \backslash \tau_2(t), \tag{49}$$

together with the boundary conditions

$$U(t, \tau_1(t)) = 0, \qquad V(t, \tau_2(t)) = 0, \quad t > 0, \tag{50}$$

the Stefan conditions

$$\mathbf{v_1}(t,x,y) = \mu_1 |\nabla U(t,x,y)| \, \mathbf{n_1}(t,x,y) = -\mu \nabla U(t,x,y), \quad t > 0, \ (x,y) \in \partial\Omega_1(t), \tag{51}$$

$$\mathbf{v_2}(t,x,y) = \mu_2 |\nabla V(t,x,y)| \, \mathbf{n_2}(t,x,y) = -\mu \nabla V(t,x,y), \quad t > 0, \ (x,y) \in \partial\Omega_2(t), \tag{52}$$

where $\mathbf{v_1}(t,x,y)$ and $\mathbf{n_1}(t,x,y)$ are, respectively, the velocity vector of the boundary point $(x,y) \in \partial\Omega_1(t)$, and the unit outward normal of $\Omega_1(t)$ at $(x,y) \in \partial\Omega_1(t)$, $\mathbf{v_2}(t,x,y)$ and $\mathbf{n_2}(t,x,y)$

are, respectively, the velocity vector of the boundary point $(x,y) \in \partial\Omega_2(t)$, and the unit outward normal of $\Omega_2(t)$ at $(x,y) \in \partial\Omega_2(t)$, and the initial conditions

$$U(0,x,y) = U_0(x,y), \quad (x,y) \in \Omega_1(0), \tag{53}$$
$$V(0,x,y) = V_0(x,y), \quad (x,y) \in \Omega_2(0), \tag{54}$$

The initial functions $U_0(x,y)$ and $V_0(x,y)$ satisfies the following properties:

$$U_0(x,y) \in C^2(\Omega_1(0)), \quad U_0'(\mathbf{0}) = U_0(\tau_1(0)) = 0, \quad U_0(x,y) > 0, \quad (x,y) \in \overline{\Omega_1}(0). \tag{55}$$
$$V_0(x,y) \in C^2(\Omega_2(0)), \quad U_0'(\mathbf{0}) = V_0(\tau_2(0)) = 0, \quad V_0(x,y) > 0, \quad (x,y) \in \overline{\Omega_2}(0). \tag{56}$$

Here $\tau_1(t)$ and $\tau_2(t)$ are the unknown moving boundaries of two species $U(t,x,y)$ and $V(t,x,y)$ such that the population are distributed in the domain $\Omega_1(t)$ and $\Omega_2(t)$ separately. $D_1 > 0$ and $D_2 > 0$ are the dispersal rates. The parameters $\mu_1 > 0$ and $\mu_2 > 0$ involved in the Stefan conditions (51) and (52) are the proportionality constant between the population gradient at the front and the speed of the moving boundary of two species $U(t,x,y)$ and $V(t,x,y)$ respectively. Following the ideas of [19], here we use a level set approach to effectively capture the front at each new time step and a finite difference scheme to solve the heat equation everywhere away from the front. The idea behind the level set method is to construct a level set function ϕ, then move ϕ with the correct speed $\mathbf{v}$ at the front and followed by updating $u(t,x,y)$. The new position of the front is stored implicitly in ϕ. With this approach, we avoid the difficulties that arise from explicitly tracking the front and thus increase the efficiency to deal with complex interfacial geometries.

Step1. Initialize $U(t,x,y)$ and $\phi_1(t,x,y)$.

We construct a level set function ϕ_1, such that at any time t, the front $\tau_1(t)$ is equal to the zero level set of ϕ_1, i.e.,

$$\tau_1(t) = \{(x,y) \in \Omega_1 : \phi_1(t,x,y) = 0\}$$

Initially, $U(0,x,y) = U_0(x,y)$ and ϕ_1 is set equal to the signed distance function from the front of species U such that ϕ_1 is negative in $\Omega_1(0)$ and positive in $\Omega_1^c(0)$,

$$\phi_1(0,x,y) = \begin{cases} +d, & x \in \Omega_1^c(0), \\ 0, & x \in \tau_1(0), \\ -d & x \in \Omega_1(0). \end{cases} \tag{57}$$

where d is the distance from the front $\tau_1(t)$.

Given the normal speed, $\mathbf{v_1}$, at which the front $\tau_1(t)$ moves, we would construct a speed function, $F_1(t,x,y)$, which is a continuous extension of $|\mathbf{v_1}(\mathbf{t},\mathbf{x},\mathbf{y})|$ from the front $\tau_1(t)$ over the whole computational domain. The governing equation of ϕ_1 is then given by

$$\frac{\partial \phi_1}{\partial t} + F_1|\nabla \phi_1| = 0. \tag{58}$$

This equation will move ϕ_1 with the correct speed at the front by assuring that $\tau_1(t)$ will always coincide with the zero level set of ϕ_1 at time t.

We also use ϕ_1 to define the outward normal vector $\mathbf{n_1}$ corresponding to τ_1 by

$$\mathbf{n_1} = \nabla \phi_1 / |\nabla \phi_1|. \tag{59}$$

From Equations (51) and (59), we can rewrite the expression for $\tau_1'(t)$ as

$$\mathbf{v_1}(t) = -\mu_1|\nabla U|\mathbf{n_1} = -\mu_1|\nabla U|\frac{\nabla\phi_1}{|\nabla\phi_1|}. \tag{60}$$

Since F_1 is equal to $|\mathbf{v_1}(\mathbf{t})|$ along the interface, we can combine Equations (58) and (60) to get the following equation, which of course is only valid on the zero level set of ϕ_1:

$$\frac{\partial\phi_1}{\partial t} = \mu_1|\nabla U||\nabla\phi_1|, (x,y) \in \tau_1(t). \tag{61}$$

Next, we need to extend the velocity function V_1 to a neighborhood of $\tau_1(t)$.

Therefore, we get the velocity function over the computational domain

$$F_1(t,x,y) = \mu_1|\nabla U(t,x,y)|. \tag{62}$$

which is of course only valid on the zero level set of ϕ.

Step2. Compute the velocity filed $F_1(x,y,t)$ of U .

By introducing F_1 defined as an extension of $|\mathbf{v_1}(t,x,y)| = \mu_1|\nabla U(t,x,y)|$ from $(x,y) \in \tau_1(t)$, we can avoid unnecessary numerical difficulties when we solve Equations (60) and (61).

According to (58), (59) and (62), the level set equation turns into

$$\begin{aligned}\frac{\partial\phi_1}{\partial t} &= -F_1|\nabla\phi_1| \\ &= -\mu_1|\nabla U(t,x,y)||\nabla\phi_1| \\ &= -\mu_1|\nabla U(t,x,y)|\mathbf{n_1}\cdot\nabla\phi_1 \\ &= \mu_1\nabla U(t,x,y)\cdot\nabla\phi_1\end{aligned} \tag{63}$$

One issue in computing ∇U arises from the fact that its approximation is usually in the order $O(1)$ at points close to or on the front.

The approximation to ∇U at $\tau_1(t)$ is based upon approximations to the derivatives of U in four coordinate directions to cut down on grid orientation effects (please see Figure 5 for illustration). Each approximation to a derivative of U can be continuously extended away from the front by the advection equations

$$u_t^1 + S(\phi_1\frac{\partial\phi_1}{\partial x})u_x^1 = 0, \tag{64}$$

$$u_t^2 + S(\phi_1\frac{\partial\phi_1}{\partial y})u_y^2 = 0, \tag{65}$$

$$u_t^3 + S(\phi_1\frac{\partial\phi_1}{\partial\eta})u_\eta^3 = 0, \tag{66}$$

$$u_t^4 + S(\phi_1\frac{\partial\phi_1}{\partial\zeta})u_\zeta^4 = 0, \tag{67}$$

where $u^1 = \partial U/\partial x$, $u^2 = \partial U/\partial y$, $u^3 = \partial U/\partial\eta$ and $u^4 = \partial U/\partial\zeta$ on $\tau_1(t)$. Here S is equal to the sign function. Equation (64) through Equation (67) have the effect of continuously extending u^1, u^2, u^3, u^4 away from the front by advecting these fields in the proper upwind direction. Note that these equations will not degrade the value of V_1 on the front because ϕ_1 is zero on $\tau_1(t)$, hence, so are $S(\phi_1\frac{\partial\phi_1}{\partial x})$, $S(\phi_1\frac{\partial\phi_1}{\partial y})$, $S(\phi_1\frac{\partial\phi_1}{\partial\eta})$ and $S(\phi_1\frac{\partial\phi_1}{\partial\zeta})$.

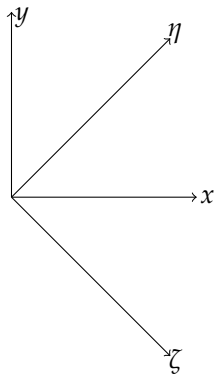

Figure 5. Four coordinate directions used to compute the normal velocity.

From (63), we end up solving for the right hand side of the equation

$$\frac{\partial \phi_1}{\partial t} = \frac{\mu_1}{2}(u^1_{i,j}(\phi_{1x})_{i,j} + u^2_{i,j}(\phi_{1y})_{i,j} + u^3_{i,j}(\phi_{1\eta})_{i,j} + u^4_{i,j}(\phi_{1\zeta})_{i,j}) \tag{68}$$

The spatial first derivatives of ϕ_1 are approximated by a second-order ENO scheme. We update ϕ_1 by solving (68) with a third-order Runge-Kutta scheme.

Step3: Update ϕ_1 to be a signed distance function for one time step.

From Equation (58) and (59), it is clear that the computation of the normal velocity, and normal vector at the front are all dependent upon the level set function ϕ_1. However, by Equation (58), the level set function will cease to be an exact distance function even after one time step. In order to keep the accuracy of $\mathbf{n_1}$, and F_1, we need to avoid having steep or flat gradients developed in ϕ_1. One way to avoid these numerical difficulties is to reinitialize ϕ_1 to be an exact distance function from the evolving front $\tau_1(t)$ at each time step.

In order to reinitialize the level set function, we use the reinitialization scheme of Sussman [37]

$$\frac{\partial \phi_1}{\partial t} = S(\phi_1^0)(1 - |\nabla \phi_1|), \tag{69}$$

where $\phi_1(0, x, y) = \phi_1^0$ and S again denotes the sign function. As in [37], the sign function S is smoothed by the equation.

The basic idea behind this method is that given a function ϕ_0 that is not a distance function, one can evolve it into a function ϕ that is an exact signed distance function from the zero level set of ϕ_0. This can be accomplished by iterating (69) to a steady state. As in [37], the sign function S is smoothed by the equation

$$S_\varepsilon(\phi_1^0) = \frac{\phi_1^0}{\sqrt{(\phi_1^0)^2 + \varepsilon^2}} \tag{70}$$

to avoid numerical difficulties while implemented.

By using this approach, we avoid having to explicitly find the contour $\phi_1^0 = 0$ and then resetting values of the front ϕ_1^0 at grid points. From Equation (69), it is clear that the original position of the front will not change, but at points away from $\tau_1(t)$, ϕ_1 will be evolved into a distance function.

Step4: Update $U(t,x,y)$.

After moving ϕ_1 by the correct velocity at the front and then reinitializing ϕ_1 to be an exact signed distance function from $\tau_1(t)$ in *Step 3*, next we update $U(t,x,y)$. Updating $U(t,x,y)$ essentially boils down to solving the nonlinear parabolic partial difference equation (48) over the whole computational domain in the following three cases:

- At points away from the front, which means the nearby four grid points are all inside the domain $\Omega_1(t)$, we solve the nonlinear parabolic partial difference equation (53) by combining the forward Euler method and the five-point stencil scheme.

  For example, we use the scheme (71) to update $U(t,x,y)$ at the grid point $(i+1,j)$ in Figure 6.

$$\frac{u^{n+1}_{i+1,j}-u^n_{i+1,j}}{\triangle t} - D_1\frac{u^n_{i,j}+u^n_{i+2,j}-4u^n_{i+1,j}+u^n_{i+1,j-1}+u^n_{i+1,j+1}}{h^2} = \gamma_1 u^n_{i+1,j}(1-u^n_{i+1,j}-K_1 v^n_{i+1,j}) \tag{71}$$

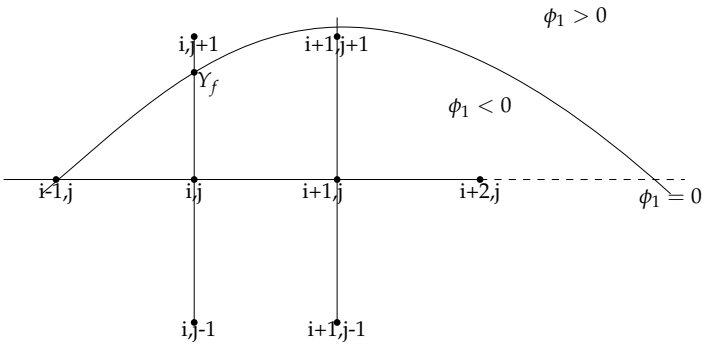

Figure 6. Illustration of updating U.

- For points near the front $\tau_1(t)$, some special care should be taken. We effectively capture the front using the level set function ϕ_1. We can use the one-sided different sign of ϕ_1 to incorporate the distances between a point on the front and grid points neighboring it in either the vertical or horizontal direction. For example, $Y_f = (-\frac{L}{2}+(i-1)h, y_f) \in \tau_1(t)$, we consider two grid points $(i,j+1)$ and (i,j) which border Y_f. In y-direction, we have $y_j \le y_f \le y_{j+1}$. We introduce

$$y_f - y_j = rh = -\frac{\phi_{1_{ij}}}{\phi_{1_{i,j+1}}-\phi_{1_{i,j}}}h$$

  and use $u^n_{i,j}$, $u^n_{i,j-1}$, $u^n_{i,j-2}$, r and $U(n\triangle t, -\frac{L}{2}+(i-1)h, y_f) = 0$ to construct interpolating polynomial P. When updating $u^{n+1}_{i,j}$, we use a standard five-point stencil combing forward Euler method by employing $P(-\frac{L}{2}+jh)$ instead of $u^n_{i,j+1}$, i.e.,

$$\frac{u^{n+1}_{i,j}-u^n_{i,j}}{\triangle t} - D_1\frac{u^n_{i-1,j}+u^n_{i,j-1}-4u^n_{i,j}+P(-\frac{L}{2}+jh)+u^n_{i+1,j}}{h^2} = \gamma_1 u^n_{i,j}(1-u^n_{i,j}-K_1 v^n_{i,j}) \tag{72}$$

  For the case when front interacts with x-axis, we use the same process in x-direction. In the special case where we cannot find enough grid points inside the domain to construct interpolating polynomial P, we employ the nearby grid points and intersect points of the front and x and y-axis to construct quadratic polynomial or straight line as the interpolating polynomial P to update U. For the extreme configuration, where there are only intersect points of the front and x and y-axis near the grid point, we update $U = 0$ at the grid point.

- If a grid point lies on the front, we set the value $U = 0$ at that point (in view of (53)). For example, we set $U^{n+1}_{i-1,j}$=0 for the grid point $(i-1, j)$.

Step5. Repeat Step 2 through Step 4 to ϕ_2 and F_2.

Step6. Repeat Step 2 through Step 6 to update ϕ_1 , ϕ_2 , U and V for the next time step.

4. Numerical Experiments

4.1. Numerical Tests of 1D Problem: Front-Fixing Method and Front-Tracking Method

Convergence test of front-fixing method

In the 1D two-species competition-diffusion model (3)–(9) with parameters values $(D_1, \mu_1, \gamma_1, K_1, S_1^0) = (0.4, 5, 2, 1, 0.4)$, $(D_2, \mu_2, \gamma_2, K_2, S_2^0) = (0.4, 10, 1, 2, 1)$, $U_0 = 2cos(\frac{\pi x}{2})$, and $V_0 = 4cos(\frac{\pi x}{2})$. Here we test the order of convergence in space with very refined temporal step size.

In Tables 1 and 2 the error (both L_2 and L_∞) and the convergence to the solution of front-fixing method is examined, with final time $t_{end} = 0.01$. The error is computed by the difference of the numerical solution with the exact solution. For all the examples below when the exact solution is not given, the solution with a very fine resolution will be considered as reference or "exact" solution. As expected, a second-order convergence in space for both u and v can be observed.

Table 1. Convergence analysis of the value of U and the front of U using the front-fixing method.

$N_x \times N_t$	L_2*Error*	Order	L_∞*Error*	Order
Accuracy test of U of front-fixing method				
101×10^6	1.195×10^{-4}		2.119×10^{-4}	
201×10^6	3.142×10^{-5}	1.93	5.424×10^{-5}	1.97
401×10^6	8.233×10^{-6}	1.93	1.314×10^{-5}	2.05
801×10^6	1.956×10^{-6}	2.07	3.983×10^{-6}	1.72
1601×10^6	Reference			
Accuracy test of the front of U of front-fixing method				
101×10^6	3.178×10^{-6}		9.366×10^{-5}	
201×10^6	7.880×10^{-7}	2.01	2.424×10^{-5}	1.95
401×10^6	1.880×10^{-7}	2.07	5.927×10^{-6}	2.03
801×10^6	3.800×10^{-8}	2.32	1.202×10^{-6}	2.30
1601×10^6	Reference			

Table 2. Convergence analysis of the value of V and the front of V using the front-fixing method.

$N_x \times N_t$	L_2*Error*	Order	L_∞*Error*	Order
Accuracy test of V of front-fixing method				
101×10^6	1.038×10^{-3}		2.115×10^{-3}	
201×10^6	2.776×10^{-4}	1.90	5.609×10^{-4}	1.91
401×10^6	6.861×10^{-5}	2.02	1.381×10^{-4}	2.02
801×10^6	1.398×10^{-5}	2.30	2.807×10^{-5}	2.30
1601×10^6	Reference			

Table 2. *Cont.*

$N_x \times N_t$	L_2*Error*	Order	L_∞*Error*	Order
Accuracy test of the front of V of front-fixing method				
101×10^6	2.890×10^{-5}		7.595×10^{-4}	
201×10^6	7.700×10^{-6}	1.91	2.123×10^{-4}	1.84
401×10^6	1.910×10^{-6}	2.01	5.454×10^{-5}	1.96
801×10^6	3.900×10^{-7}	2.29	1.141×10^{-5}	2.26
1601×10^6	Reference			

Convergence test of front-tracking method

In the 1D two-species competition-diffusion model (3)–(9) with parameters values $(D_1, \mu_1, \gamma_1, K_1, S_1^0) = (0.4, 5, 2, 1, 0.4)$, $(D_2, \mu_2, \gamma_2, K_2, S_2^0) = (0.4, 10, 1, 2, 1)$, $L = 1.2$, $U_0 = 2cos(\frac{\pi x}{2})$, and $V_0 = 4cos(\frac{\pi x}{2})$. Here we test the order of convergence in space with very refined temporal step size.

In Tables 3 and 4 the error (both L_2 and L_∞) and the convergence to the solution of front-tracking method is examined, with final time $t_{end} = 0.01$. The error is computed by the difference of the numerical solution with the exact solution. For all the examples below, when the exact solution is not given, the solution with a fine resolution will be considered as reference or "exact" solution. As expected, a second-order convergence in space for both u and v can be observed.

Table 3. Convergence analysis of the value of U and the front of U using the front-tracking method.

$N_x \times N_t$	L_2*Error*	Order	L_∞*Error*	Order
Accuracy test of U of front-tracking method				
61×10^5	5.637×10^{-4}		1.699×10^{-3}	
121×10^5	1.035×10^{-4}	2.45	3.260×10^{-4}	2.38
241×10^5	1.850×10^{-5}	2.48	6.019×10^{-5}	2.44
481×10^5	2.987×10^{-6}	2.63	9.833×10^{-6}	2.61
961×10^5	Reference			
Accuracy test of the front of U of front-tracking method				
61×10^5	1.222×10^{-4}		2.233×10^{-3}	
121×10^5	2.280×10^{-5}	2.42	5.672×10^{-4}	1.98
241×10^5	4.300×10^{-6}	2.39	1.296×10^{-4}	2.13
481×10^5	8.000×10^{-7}	2.50	2.494×10^{-5}	2.38
961×10^5	Reference			

Table 4. Convergence analysis of the value of V and the front of V using the front-tracking method.

$N_x \times N_t$	L_2*Error*	Order	L_∞*Error*	Order
Accuracy test of V of front-tracking method				
61×10^5	4.443×10^{-4}		1.373×10^{-3}	
121×10^5	7.882×10^{-5}	2.49	2.493×10^{-4}	2.46
241×10^5	1.396×10^{-5}	2.50	4.254×10^{-5}	2.55
481×10^5	2.378×10^{-6}	2.55	9.871×10^{-6}	2.11
961×10^5	Reference			

Table 4. *Cont.*

$N_x \times N_t$	$L_2 Error$	Order	$L_\infty Error$	Order
Accuracy test of the front of V of front-tracking method				
61×10^5	1.385×10^{-4}		3.268×10^{-3}	
121×10^5	2.721×10^{-5}	2.35	8.504×10^{-4}	1.94
241×10^5	6.000×10^{-6}	2.19	1.922×10^{-4}	2.15
481×10^5	1.200×10^{-6}	2.30	3.788×10^{-5}	2.34
961×10^5	Reference			

The Comparison of Front-fixing with Front-tracking for 1D model

In Figures 7 and 8, we use the front-fixing method and front-tracking method to simulate the 1D two-species competition-diffusion model (3)–(9) with parameters values $(D_1, \mu_1, \gamma_1, K_1, S_1^0) = (0.4, 5, 2, 1, 0.4)$, $(D_2, \mu_2, \gamma_2, K_2, S_2^0) = (0.4, 10, 1, 2, 1)$, $U_0 = 2cos(\frac{\pi x}{2})$, and $V_0 = 4cos(\frac{\pi x}{2})$ and spatial size $h = 0.00125$. It shows that the results of front-tracking method and the results of front-fixing method are consistent with each other.

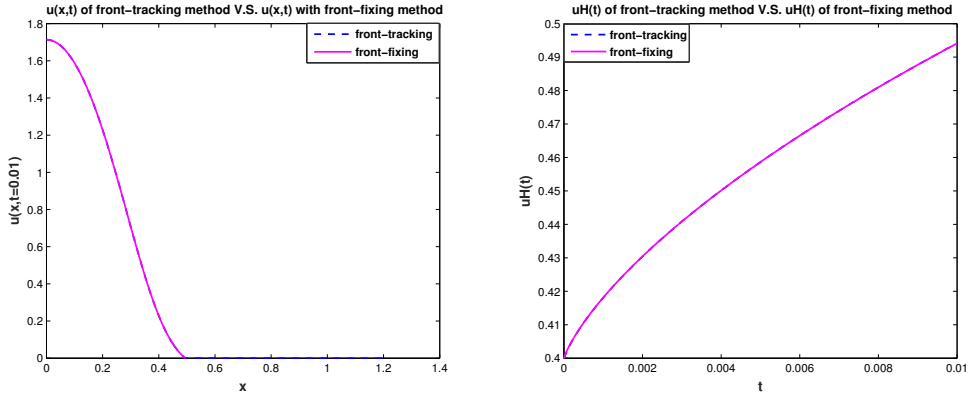

Figure 7. u(x,t=0.01) and uH(t):Front-tracking method vs. front-fixing method for 1D model.

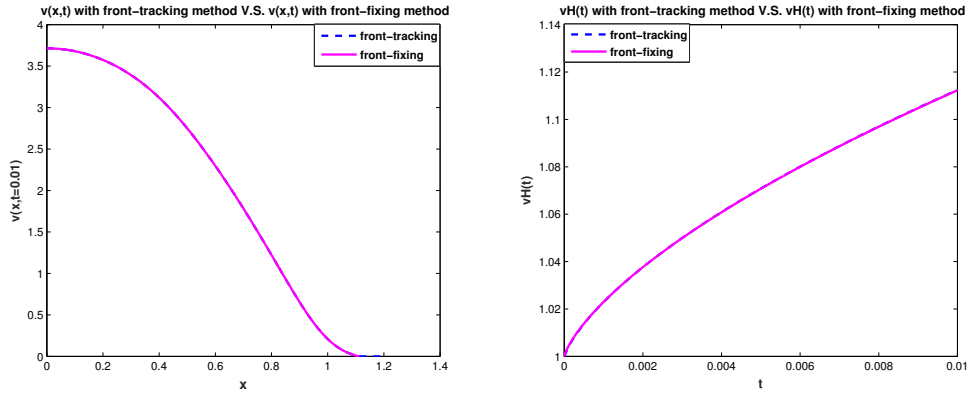

Figure 8. v(x,t=0.01) and vH(t):Front-tracking method vs. front-fixing method for 1D model.

4.2. Numerical Tests of Level Set Methods for 2D Model With Different Initial Configuration

Example 1. *In the 2D two-species competition-diffusion model (48)–(56) with parameters values* $(D_1, \mu_1, \gamma_1, K_1) = (4, 10, 1, 0.6)$, $(D_2, \mu_2, \gamma_2, K_2) = (0.4, 5, 3, 0.5)$, *the initial boundary of species U is set to be an equilateral triangle which centers at the origin point* $(0, 0)$ *with side length 1, while the initial*

boundary of species V is set to be a circle which centers at the origin point $(0,0)$ *with radius = 1.5. The initial values* $U_0(x,y)$, V_0 *(x,y) and the initial level set functions* $\phi_1^0(x,y)$, ϕ_2^0 *(x,y) are set as follows*

$$U_0(x,y) = \begin{cases} 40(\frac{\sqrt{3}}{2} - \frac{1}{\sqrt{3}} + y)(\sqrt{3}x - y + \frac{1}{\sqrt{3}})(-\sqrt{3}x - y + \frac{1}{\sqrt{3}}), & (x,y) \in \Omega_1(0), \\ 0 & (x,y) \in \Omega_1^c(0). \end{cases} \tag{73}$$

$$V_0(x,y) = \begin{cases} 45cos(\frac{\sqrt{x^2+y^2}\pi}{2}), & (x,y) \in \Omega_2(0), \\ 0 & (x,y) \in \Omega_2^c(0). \end{cases} \tag{74}$$

$$\phi_1^0(x,y) = \begin{cases} -min(\frac{\sqrt{3}}{2} - \frac{1}{\sqrt{3}} + y, (\sqrt{3}x - y + \frac{1}{\sqrt{3}})/2, (-\sqrt{3}x - y + \frac{1}{\sqrt{3}})/2), & (x,y) \in \Omega_1(0), \\ 0 & (x,y) \in \partial\Omega_1(0), \\ min(|\frac{\sqrt{3}}{2} - \frac{1}{\sqrt{3}} + y|, |\sqrt{3}x - y + \frac{1}{\sqrt{3}}|/2, |-\sqrt{3}x - y + \frac{1}{\sqrt{3}}|/2), & (x,y) \in \overline{\Omega}_1^c(0). \end{cases} \tag{75}$$

$$\phi_2^0(x,y) = -(1.5 - \sqrt{x^2+y^2}). \tag{76}$$

Figure 9 shows the simulation of the evolvement of two species and their moving boundaries along time with an equilateral triangle as the initial boundary of U and a circle as the initial boundary of V. In the figure of boundary line, the red curves represent the initial boundaries, and the blue curves simulate the evolvement of free boundaries.

From Figure 9, we can see that the triangle evolves into a circle during the simulation.

Example 2. *In the 2D two-species competition-diffusion model (48)–(56) with parameters values* $(D_1, \mu_1, \gamma_1, K_1) = (4, 20, 1, 0.6)$, $(D_2, \mu_2, \gamma_2, K_2) = (1, 5, 2, 0.5)$, *the initial boundary of species U is set to be a square with side length = 1, centered at (0,0), while the initial boundary of species V is set to be a circle which centers at the origin point* $(0,0)$ *with radius = 2. The initial values* $U_0(x,y)$, V_0 *(x,y) and the initial level set functions* $\phi_1^0(x,y)$, ϕ_2^0 *(x,y)are set as following*

$$U_0(x,y) = \begin{cases} 10(1-x)(1+x)(1-y)(1+y), & (x,y) \in \Omega_1(0), \\ 0 & (x,y) \in \Omega_1^c(0). \end{cases} \tag{77}$$

$$V_0(x,y) = \begin{cases} 50cos(\frac{\sqrt{x^2+y^2}\pi}{4}), & (x,y) \in \Omega_2(0), \\ 0 & (x,y) \in \Omega_2^c(0). \end{cases} \tag{78}$$

$$\phi_1^0(x,y) = \begin{cases} -min(1-|x|, 1-|y|), & (x,y) \in \Omega_1(0), \\ 0 & (x,y) \in \partial\Omega_1(0), \\ min(|1-|x||, |1-|y||) & (x,y) \in \overline{\Omega}_1^c(0), \end{cases} \tag{79}$$

$$\phi_2^0(x,y) = -(2 - \sqrt{x^2+y^2}). \tag{80}$$

Figure 10 shows the simulation of the evolvement of two species and their moving boundaries along time with a square as the initial boundary of U and a circle as the initial boundary of V. In the figure of boundary line, the red curves represent the initial boundaries, and the blue curves simulate the evolvement of free boundaries.

From Figure 10, we can see that the square evolves into a circle during the simulation.

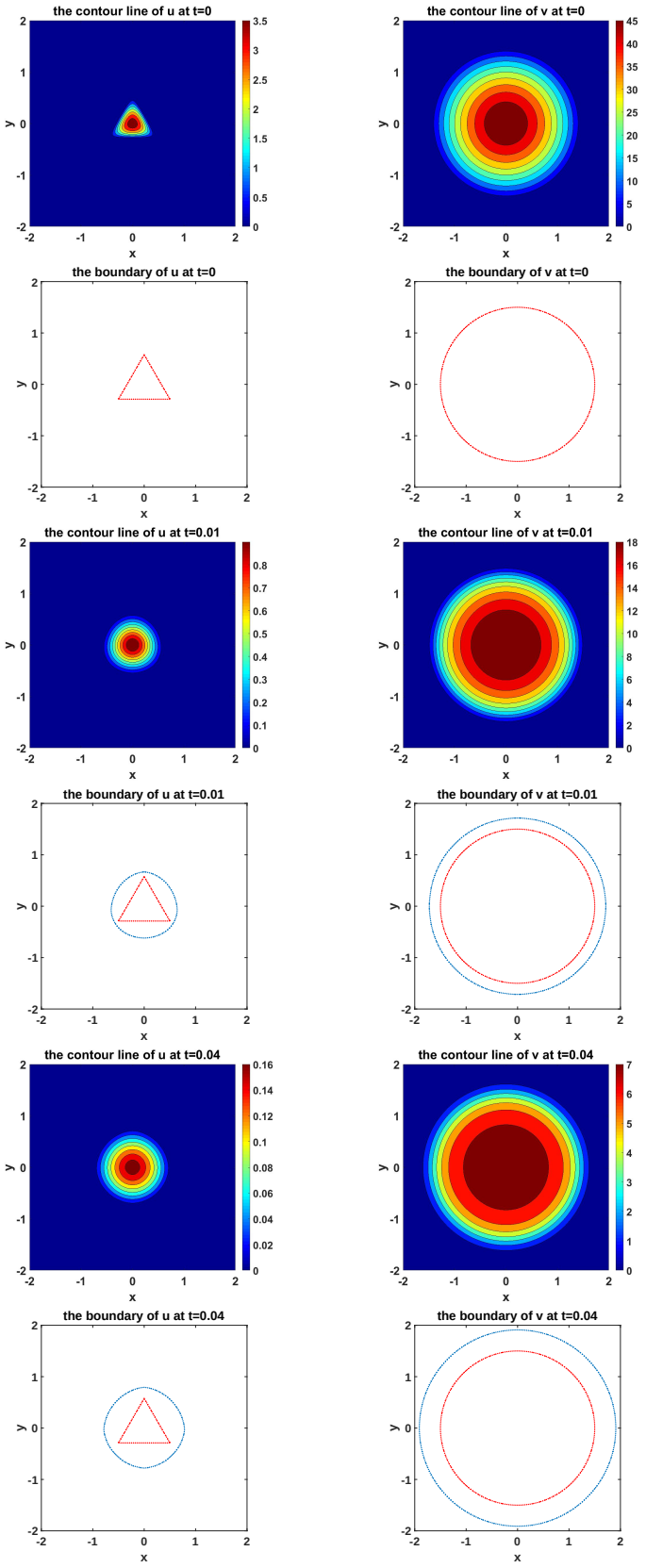

Figure 9. The simulated dynamics where initial boundary of U is an equilateral triangle and initial boundary of V is a circle. The snapshots are taken at the times $t = 0, 0.01, 0.04$, respectively.

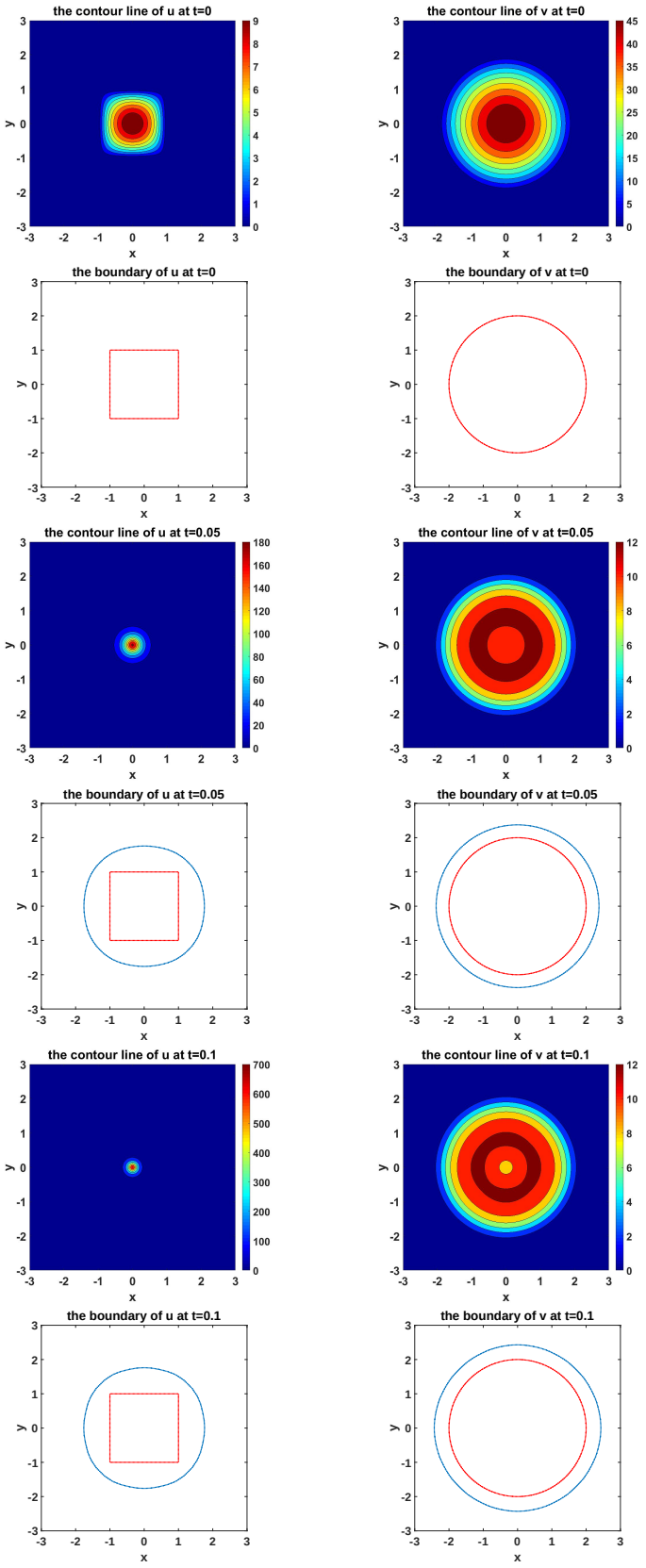

Figure 10. The simulated dynamics where initial boundary of U is a square and initial boundary of V is a circle. The snapshots are taken at the times $t = 0, 0.05, 0.1$, respectively.

Example 3. *In the 2D two-species competition-diffusion model (48)–(56) with parameters values* $(D_1, \mu_1, \gamma_1, K_1) = (1, 15, 1, 0.4)$, $(D_2, \mu_2, \gamma_2, K_2) = (2, 5, 1, 0.5)$, *the initial boundary of species U is set to be a circle which centers at* $(0, 0)$ *with radius = 2.5, while the initial boundary of species V is set to be a circle which centers at the origin point* $(0, 0)$ *with radius = 3.2. The initial values* $U_0(x, y)$, V_0 *(x,y) and the initial level set functions* $\phi_1^0(x, y)$, ϕ_2^0 *(x,y)are set as follows*

$$U_0(x,y) = \begin{cases} 40cos(\frac{\sqrt{x^2+y^2}\pi}{5}), & (x,y) \in \Omega_1(0), \\ 0 & (x,y) \in \Omega_1^c(0). \end{cases} \tag{81}$$

$$V_0(x,y) = \begin{cases} 20cos(\frac{\sqrt{x^2+y^2}\pi}{6.4}), & (x,y) \in \Omega_2(0), \\ 0 & (x,y) \in \Omega_2^c(0). \end{cases} \tag{82}$$

$$\phi_1^0(x,y) = -(2.5 - \sqrt{x^2+y^2}). \tag{83}$$

$$\phi_2^0(x,y) = -(3.2 - \sqrt{x^2+y^2}). \tag{84}$$

Figure 11 shows the simulation of the evolvement of two species and their moving boundaries along time with circles as the initial boundary of U and V. In the figure of boundary line, the red curves represent the initial boundaries, and the blue curves simulate the evolvement of free boundaries.

From Figure 11, we can see that the circles propagate as circles during the simulation.

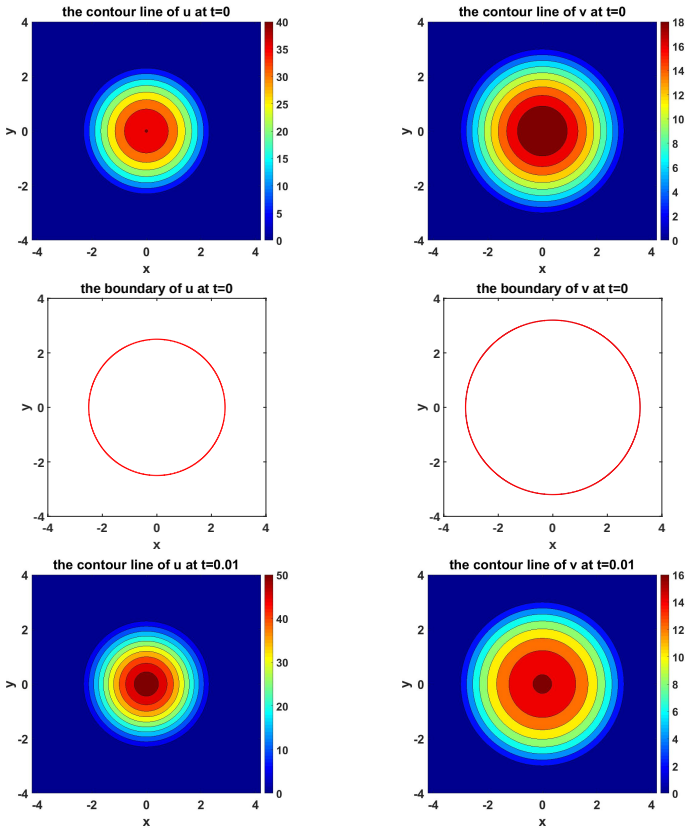

Figure 11. *Cont.*

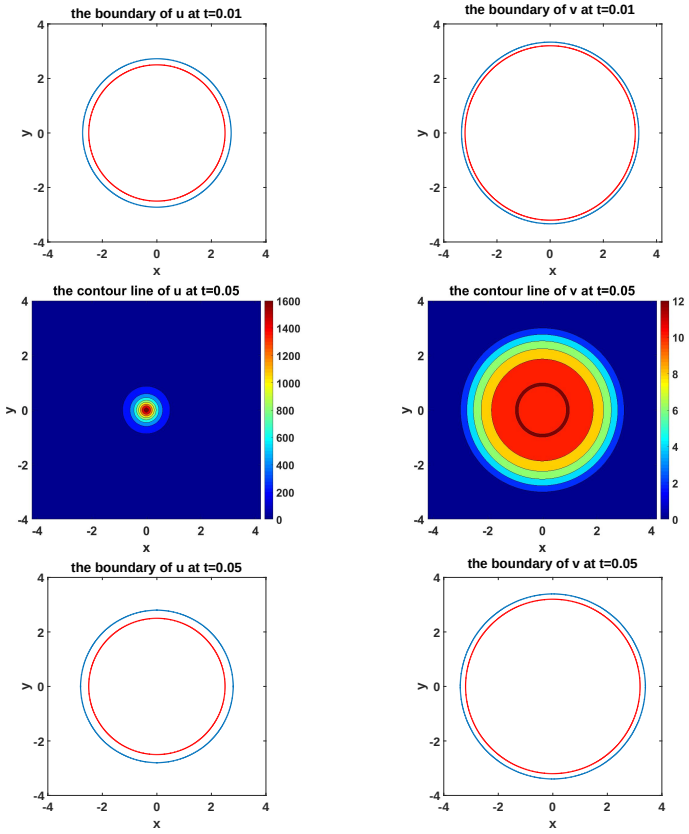

Figure 11. The simulated dynamics where initial boundaries of U and V are circles. The snapshots are taken at the times $t = 0, 0.01, 0.05$, respectively.

5. Conclusions

The system of reaction-diffusion equations with moving boundaries has been intensively studied analytically in recent years, however, very little numerical work has been done in this field due to numerical challenges in tracking free boundaries. In this paper, we first introduce a front tracking framework for 1D model, and compare it with a front-fixing method. Numerical experiments demonstrate that these two methods are consistent with each other. For 2D models, to overcome the difficulty of handling complicated topologically changes, we apply a level set approach to handle the moving boundaries. Numerical examples with different initial configurations demonstrate that the level set approach is able to robustly and efficiently capture different complicated geometries.

Although the level set method is very robust in handling topological changes, sometimes it is very hard to achieve high order accuracy, especially near the fronts. Currently we are extending the front tracking method to more accurately deal with topological changes for general 2D models, and to the systems of two competing species in which each species has its own moving boundary. The front will become more complicated and more challenging once two moving fronts are tangled together, and we would apply the reconstruction strategy to overcome these difficulties.

Author Contributions: X.L. and S.L. proposed and designed the numerical methods and S.L. performed the numerical experiments. X.L. and S.L. together wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, M.; Zhang, Y. Note on a two-species competition-diffusion model with two free boundaries. *Nonlinear Anal. Theory Methods Appl.* **2017**, *159*, 458–467. [CrossRef]
2. Guo, J.S.; Wu, C.H. Dynamics for a two-species competition—Diffusion model with two free boundaries. *Nonlinearity* **2014**, *28*, 1. [CrossRef]
3. Guo, J.S.; Wu, C.H. On a free boundary problem for a two-species weak competition system. *J. Dyn. Differ. Equ.* **2012**, *24*, 873–895. [CrossRef]
4. Wu, C.H. The minimal habitat size for spreading in a weak competition system with two free boundaries. *J. Differ. Equ.* **2015**, *259*, 873–897. [CrossRef]
5. Hilhorst, D.; Iida, M.; Mimura, M.; Ninomiya, H. A competition-diffusion system approximation to the classical two-phase Stefan problem. *Jpn. J. Ind. Appl. Math.* **2001**, *18*, 161. [CrossRef]
6. Wang, M.; Zhao, J. Free boundary problems for a Lotka-Volterra competition system. *J. Dyn. Differ. Equ.* **2014**, *26*, 655–672. [CrossRef]
7. Hosono, Y. The minimal speed of traveling fronts for a diffusive Lotka-Volterra competition model. *Bull. Math. Biol.* **1998**, *60*, 435–448. [CrossRef]
8. Conley, C.; Gardner, R. *An Application of the Generalized Morse Index to Travelling Wave Solutions of a Competitive Reaction-Diffusion Model*; Technical Report; Wisconsin Univ-Madison Mathematics Research Center: Madison, WI, USA, 1980.
9. Gardner, R.A. Existence and stability of travelling wave solutions of competition models: A degree theoretic approach. *J. Differ. Equ.* **1982**, *44*, 343–364. [CrossRef]
10. Lewis, M.A.; Li, B.; Weinberger, H.F. Spreading speed and linear determinacy for two-species competition models. *J. Math. Biol.* **2002**, *45*, 219–233. [CrossRef] [PubMed]
11. Li, B.; Weinberger, H.F.; Lewis, M.A. Spreading speeds as slowest wave speeds for cooperative systems. *Math. Biosci.* **2005**, *196*, 82–98. [CrossRef] [PubMed]
12. Weinberger, H.F.; Lewis, M.A.; Li, B. Analysis of linear determinacy for spread in cooperative models. *J. Math. Biol.* **2002**, *45*, 183–218. [CrossRef] [PubMed]
13. Du, Y.; Guo, Z. The Stefan problem for the Fisher–KPP equation. *J. Differ. Equ.* **2012**, *253*, 996–1035. [CrossRef]
14. Dong, X.; Li, H.; Derdowski, A.; Ding, L.; Burnett, A.; Chen, X.; Peters, T.; Dermody, T.; Woodruff, E.; Wang, J.; et al. AP-3 directs the intracellular trafficking of HIV-1 Gag and plays a key role in particle assembly. *Cell* **2005**, *120*, 663–674. [CrossRef] [PubMed]
15. Du, Y.; Lou, B. Spreading and vanishing in nonlinear diffusion problems with free boundaries. *J. Eur. Math. Soc.* **2015**, *17*, 2673–2724. [CrossRef]
16. Du, Y.; Lou, B.; Zhou, M. Nonlinear diffusion problems with free boundaries: convergence, transition speed, and zero number arguments. *SIAM J. Math. Anal.* **2015**, *47*, 3555–3584. [CrossRef]
17. Du, Y.; Matano, H.; Wang, K. Regularity and asymptotic behavior of nonlinear Stefan problems. *Arch. Ration. Mech. Anal.* **2014**, *212*, 957–1010. [CrossRef]
18. Liu, S.; Du, Y.; Liu, X. Numerical methods for a class of reaction-diffusion equations with free boundaries. **2018**, preprint.
19. Osher, S.; Fedkiw, R. *Level Set Methods and Dynamic Implicit Surfaces*; Springer Verlag: Berlin, Germany, 2002.
20. Osher, S.; Sethian, J. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.* **1988**, *79*, 12–49. [CrossRef]
21. Sethian, J. A fast marching level set method for monotonically advancing fronts. *Proc. Natl. Acad. Sci. USA* **1996**, *93*, 1591–1595. [CrossRef] [PubMed]
22. Sethian, J.A. *Level Set Methods and Fast Marching Methods*; Cambridge University Press: Cambridge, UK, 1999.
23. Xu, J.; Li, Z.; Lowengrub, J.; Zhao, H. A level-set method for interfacial flows with surfactant. *J. Comput. Phys.* **2006**, *212*, 590–616. [CrossRef]
24. Zhao, H.; Chan, T.; Merriman, B.; Osher, S. A variational level set approach to multiphase motion. *J. Comput. Phys.* **1996**, *127*, 179–195. [CrossRef]
25. Leveque, R.; Li, Z. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM J. Numer. Anal.* **1994**, *31*, 1019–1044. [CrossRef]
26. Peskin, C. The immersed boundary method. *Acta Numer.* **2003**, *11*, 479–517.

27. Wiegmann, A.; Bube, K. The immersed interface method for nonlinear differential equations with discontinuous coefficients and singular sources. *SIAM J. Numer. Anal.* **1998**, *35*, 177–200. [CrossRef]
28. Zhu, L.; Peskin, C. Simulation of a flapping flexible filament in a flowing soap film by the immersed boundary method. *J. Comput. Phys.* **2002**, *179*, 452–468. [CrossRef]
29. Chern, I.; Glimm, J.; McBryan, O.; Plohr, B.; Yaniv, S. Front tracking for gas dynamics. *J. Comput. Phys.* **1986**, *62*, 83–110. [CrossRef]
30. Glimm, J.; Li, X.; Liu, Y.; Zhao, N. Conservative front tracking and level set algorithms. *Proc. Natl. Acad. Sci. USA* **2001**, *98*, 14198–14201. [CrossRef] [PubMed]
31. Hilditch, J.; Colella, P. A front tracking method for compressible flames in one dimension. *SIAM J. Sci. Comput.* **1995**, *16*, 755–772. [CrossRef]
32. Hua, J.; Stene, J.; Lin, P. Numerical simulation of 3D bubbles rising in viscous liquids using a front tracking method. *J. Comput. Phys.* **2007**, *227*, 3358–3382. [CrossRef]
33. LeVeque, R.; Shyue, K. Two-dimensional front tracking based on high resolution wave propagation methods. *J. Comput. Phys.* **1996**, *123*, 354–368. [CrossRef]
34. Unverdi, S.; Tryggvason, G. A front-tracking method for viscous, incompressible, multi-fluid flows. *J. Comput. Phys.* **1992**, *100*, 25–37. [CrossRef]
35. Crank, J. *Free and Moving Boundary Problems*; Oxford Science Publications: Oxford, UK, 1984.
36. Landau, H.G. Heat conduction in a melting solid. *Q. Appl. Math.* **1950**, *8*, 81–94. [CrossRef]
37. Sussman, M.; Smereka, P.; Osher, S. A level set approach for computing solutions to incompressible two-phase flow. *J. Comput. Phys.* **1994**, *114*, 146–159. [CrossRef]