Applied Numerical Mathematics 95 (2015) 214–228

Contents lists available at ScienceDirect

Applied Numerical Mathematics

www.elsevier.com/locate/apnum

Semi-implicit finite volume level set method for advective motion of interfaces in normal direction

Peter Frolkovič *,[1,2], Karol Mikula [1,2], Jozef Urbán [2]

Department of Mathematics and Descriptive Geometry, Slovak University of Technology, Bratislava, Slovakia

ARTICLE INFO

Article history:
Available online 8 July 2014

Keywords:
Level set method
Advection equation
Finite volume method
Semi-implicit scheme

ABSTRACT

In this paper a semi-implicit finite volume method is proposed to solve the applications with moving interfaces using the approach of level set methods. The level set advection equation with a given speed in normal direction is solved by this method. Moreover, the scheme is used for the numerical solution of eikonal equation to compute the signed distance function and for the linear advection equation to compute the so-called extension speed [1]. In both equations an extrapolation near the interface is used in our method to treat Dirichlet boundary conditions on implicitly given interfaces. No restrictive CFL stability condition is required by the semi-implicit method that is very convenient especially when using the extrapolation approach. In summary, we can apply the method for the numerical solution of level set advection equation with the initial condition given by the signed distance function and with the advection velocity in normal direction given by the extension speed. Several advantages of the proposed approach can be shown for chosen examples and application. The advected numerical level set function approximates well the property of remaining the signed distance function during whole simulation time. Sufficiently accurate numerical results can be obtained even with the time steps violating the CFL stability condition.

1. Introduction

The level set methods are a popular numerical tool to treat applications with moving interfaces [27,26]. The advantages of level set methods for these applications are known and well discussed in literature, see e.g. [28,30,27,1,6,26,20,25,15,7]. To track a $(d-1)$-dimensional dynamic interface, one can use a d-dimensional (possibly fixed and rectangular) grid that is used to solve related PDEs. The moving interface is then described implicitly as the zero level set of the so-called level set function obtained by the solution of a specific advection equation. The advection velocity in this equation is usually given in the normal direction to the evolving interface. For the level set methods, the advection speed in normal direction must be defined everywhere in the d-dimensional computational domain, and it has to coincide with the prescribed normal velocity at the moving interface.

There exist various high quality numerical methods for solving advection level set equations, see e.g. [27,26,25,12,22]. However, there are several important points which must be solved when treating practical applications.

* Corresponding author.
E-mail address: peter.frolkovic@stuba.sk (P. Frolkovič).
[1] The author was supported by VEGA 1/1137/12.
[2] The author was supported by APVV 0184-10.

http://dx.doi.org/10.1016/j.apnum.2014.05.011
0168-9274/

First of all, the initial level set function must be given appropriately. A typical choice for such function is a signed distance to the interface which is usually obtained by solving the eikonal equation [27,26] with zero Dirichlet boundary condition at the interface. The position of the interface coincides only rarely with the nodes of the d-dimensional grid, therefore a special treatment in grid points near the interface is necessary. Usually a kind of "brute force" method for assigning the distances near the interface is applied [27,1]. To avoid such brute force method we propose to use an extrapolation approach for the grid nodes near the interface. Such extrapolation technique is used for boundary conditions on implicitly given interfaces in several numerical methods like immersed interface methods [17,20], ghost fluid methods [6,14] or Cartesian grid methods [18,5,25].

The next important point is the choice of advection velocity for the level set method. The so-called "natural velocity" [1], or its normal component, is usually given in the whole computational domain, see e.g. our application in Section 4.3. In many applications the natural velocity is given only partially like on one side of the interface [21,7] or even on the interface only. Therefore some procedure to extend the natural velocity away from the position of interface to the whole domain must be at disposal in general.

In any case, the usage of inappropriate velocity in the advection equation can yield solutions with very steep and/or flat gradient in subregions of the computational domain which may deteriorate the accuracy of numerical solutions [28, 1,16,15,10]. To avoid it, two main approaches are incorporated into the level set method. On the one hand, the so called reinitialization is used in order to describe the interface for whole simulation time with the signed distance function, see e.g. [28,16,10,15]. On the other hand, the so called "extension velocity" or "extension speed" [1] is constructed and used instead of the natural velocity to keep the evolving level set function (theoretically) equal to a signed distance during the whole interface evolution [30,1,7].

In our method we adopt the approach with the extension velocity. To compute it, an auxiliary linear advection equation with Dirichlet boundary condition (representing the natural speed) at the implicitly given interface is solved. Again, we prefer the extrapolation approach near the interface to treat such Dirichlet boundary conditions for the linear advection equation alternatively to some brute force method [30,1,2]. Consequently, both the (re-)initialization and the extension speed computation can be treated successfully by the extrapolation approach proposed in this paper.

The last but not least point which we discuss here is the precision and stability of the numerical solution obtained with the level set method. Clearly, high resolution methods without restrictive stability condition are desirable for practical applications. This is a general requirement when solving the advection equations, but even a must when treating the Dirichlet boundary conditions on implicitly given interface by an extrapolation technique.

When using the extrapolation with any explicit scheme having a natural CFL time step restriction (which is common for hyperbolic problems), one necessarily arrives to troubles, cf. the so called small cell problem in the Cartesian grid methods [18,5]. The reason is that the time step for the explicit scheme must be proportional to a shortest distance of the implicitly given interface to grid points which can be, in fact, arbitrary small. That makes standard explicit methods impractical for this type of problems and requires additional treatment inside the scheme [5,25,9,7]. On the other hand, the implicit (or semi-implicit) schemes can handle this phenomenon straightforwardly since they do not impose such restrictive conditions for coupling the temporal and spatial grid resolutions.

In this paper we use the second order semi-implicit finite volume method proposed in [22–24] and couple it with the extrapolation technique near the implicitly given interface. Moreover, we introduce to such scheme a different upwind type reconstruction of numerical solution on finite volume faces that is based on [12,13] which makes the method suitable for solving the stationary problems arising in the signed distance and extension speed computations. The proposed method is applied also to the advection level set equation for moving interface, therefore all three basic components of the level set method are treated in the same manner by using the proposed semi-implicit finite volume method.

The paper is organized as follows. In Section 2 we introduce partial differential equations (PDEs) used in the level set method and discretized by our semi-implicit finite volume scheme. In Section 3 we derive the proposed method for general advection equation and explain its usage in the case of moving interfaces. In Section 4 we present numerical experiments showing the properties of the proposed level set method.

2. PDEs for the level set method

Let $\Gamma(t) \subset R^d$ be a closed interface that evolves in time $t \in [0,T]$ in d-dimensional space. The movement of evolving interface $\Gamma(t)$ can be defined by describing the movement of all points located at the interface. More precisely, let $\Gamma(0)$ be a given initial position, then one requires

$$\Gamma(t) = \big\{X(t);\ X(0) \in \Gamma(0)\big\}, \quad t \in [0,T], \tag{1}$$

where a trajectory $X(t)$ describes the position of a particle at time t located initially at $X(0) \in \Gamma(0)$. We consider here only closed interfaces $\Gamma(t)$. By $\Omega(t) \subset R^d$ we denote a domain that is surrounded by the interface, i.e. $\partial\Omega(t) \equiv \Gamma(t)$.

We suppose that a velocity $\vec{V}$ is given in advance at any time $t \in [0,T]$ for each particle $\gamma \in \Gamma(0)$. Having such "natural" velocity $\vec{V}$ [1], the trajectories $X(t)$ can be determined by solving ordinary differential equations

$$\dot{X}(t) = \vec{V}\big(X(t),t\big), \quad t \in [0,T],\ X(0) \in \Gamma(0). \tag{2}$$

Note that the choice of trajectories $X(t)$ in (1) to define $\Gamma(t)$ is not unique. There exist many vector functions $\vec{V}$ in (2) that determine an identical evolving interface $\Gamma(t)$ in (1). Therefore, we consider a unique choice of $\vec{V}$ in (2) by prescribing only the velocity in the direction normal to the interface. This is standard in level set methods, while in Lagrangian numerical approaches also a tangential movement of particles play an important role, cf. e.g. [3]. So, we restrict further considerations only to the velocities in normal direction and thus we assume that $\vec{V}$ is given in the form

$$\vec{V}(\gamma,t) = S(\gamma,t)\vec{N}(\gamma), \quad t \in [0,T],\ \gamma \in \Gamma(t). \tag{3}$$

Here $\vec{N}$ denotes a normal outward vector to $\partial\Omega(t)$ and the function S represents the natural speed in normal direction known in advance.

Next we describe the evolving interface $\Gamma(t)$ using level set method [27,26]. To do so we suppose that $\Gamma(t) \subset (-L,L)^d \subset R^d$ for some fixed $L > 0$, $t \in [0,T]$, and we search for a level set function $\phi = \phi(x,t)$, $x \in (-L,L)^d$, $t \in [0,T]$, such that

$$x \in \Gamma(t) \quad \Leftrightarrow \quad \phi(x,t) = 0. \tag{4}$$

Moreover, we require the "sign property"

$$\phi(x,t) < 0 \quad \Leftrightarrow \quad x \in \Omega(t), \tag{5}$$

and the gradient $\nabla\phi(x,t)$ well-defined almost everywhere such that

$$\left|\nabla\phi(x,t)\right| \neq 0 \quad \text{almost everywhere in } (-L,L)^d \times [0,T]. \tag{6}$$

The idea is to find the function ϕ by solving the level set advection equation for the motion of interface in normal direction [27,26,11]

$$\partial_t\phi + s\vec{n}\cdot\nabla\phi = 0, \qquad \phi(x,0) = \phi^0(x), \quad \vec{n} = \frac{\nabla\phi}{|\nabla\phi|} \tag{7}$$

for $x \in (-L,L)^d$ and $t \in [0,T]$. Note that $\vec{n}(\gamma,t) = \vec{N}(\gamma)$ for $\gamma \in \Gamma(t)$, and to be compatible with (3) one has to require $s(\gamma,t) = S(\gamma,t)$ for $\gamma \in \Gamma(t)$.

As already noticed in Introduction, the choice of the speed function $s(x,t)$ and the initial function $\phi^0(x)$ in (7) is important. The appropriate choice will be explained in the sequel, similarly the boundary conditions will be commented later.

The function $\phi^0(x)$ in (7) must be chosen such that (4)–(6) are valid for $t = 0$. There exist infinitely many functions ϕ^0 that define implicitly an identical initial interface $\Gamma(0)$. In practice the signed distance function is a typical choice. This function can be found as a viscosity solution of the following nonlinear boundary value problem for the eikonal equation [27,26],

$$\left|\nabla\phi^0\right| = 1, \qquad x \in (-L,L)^d, \qquad \phi^0(\gamma) = 0, \quad \gamma \in \Gamma(0). \tag{8}$$

In our approach, we will look for the function ϕ^0 in (8) as a stationary solution of the following nonlinear evolutionary problem

$$\partial_\tau\Phi \pm \frac{\nabla\Phi}{|\nabla\Phi|}\cdot\nabla\Phi = \pm 1, \qquad \Phi(\gamma,\tau) = 0, \quad \gamma \in \Gamma(0), \qquad \Phi(x,0) = \Phi^0(x). \tag{9}$$

Here $\tau \geq 0$ is an artificial time relaxation variable. The choice of plus respectively minus sign in (9) depends on which side of the interface we solve the equation. The initial function $\Phi^0(x)$ for the eikonal equation with time relaxation can be set, e.g., to zero for all $x \in (-L,L)^d$. If a stationary solution of (9) is obtained at some time $\tau = \tilde{\tau}$, then one sets $\phi^0(x) = \Phi(x,\tilde{\tau})$.

Concerning the choice of function s in (7), there exist infinitely many speed functions that coincides with S at the interface. Following [30,1] we define the extension speed s at each fixed time t as a solution of the following boundary value problem

$$\pm\vec{n}\cdot\nabla s = 0, \qquad s(\gamma,t) = S(\gamma,t), \quad \gamma \in \Gamma(t). \tag{10}$$

Note that Eq. (10) must be solved simultaneously with (7). The choice of plus respectively minus sign in (10) depends on which side of the interface we solve the equation. The extension speed s can be seen as a constant extrapolation of the prescribed natural speed S along the vector field $\vec{n}$ away from the interface, cf. [30,1,2,9].

The choice of s according to (10) is motivated by a property that solving the coupled problem (7) and (10) one can prove (at least for smooth cases [30,1]) that $|\nabla\phi(x,t)| = 1$ if ϕ^0 in (7) fulfills (8). Consequently, if the level set function ϕ is initially the signed distance function to $\Gamma(0)$, then it remains the signed distance function to $\Gamma(t)$ also for $t > 0$.

In our approach, at any fixed time t we look for the function s in (10) as a stationary solution of the following evolutionary problem

$$\partial_\tau w \pm \vec{n}\cdot\nabla w = 0, \qquad w(\gamma,\tau) = S(\gamma,t), \quad \gamma \in \Gamma(t), \tag{11}$$

where the choice of plus respectively minus sign is equivalent to a choice of the sign in (10). If a stationary solution of (11) is obtained at some relaxation time $\tilde{\tau}$, then one sets $s(x,t) = w(x,\tilde{\tau})$. As an initial condition for $w(x,0)$ in the advection equation (11) at some fixed physical time t_2, one can use $w(x,0) = s(x,t_1)$ for some $t_1 < t_2$.

Summarizing, our level set method for the advective motion of interfaces consists in solving numerically the evolutionary equations (7), (9), and (11). In the case of Eqs. (9) and (11) we look for stationary solutions, Eq. (7) describes the dynamically moving interface. It is clear that all of these equations can be included into a general nonlinear initial-boundary value problem of this form

$$\partial_t u + \vec{v} \cdot \nabla u = c, \qquad u(x,0) = u^0(x), \quad x \in D,\ t \geq 0 \tag{12}$$

for the unknown function $u = u(x,t)$. The domain D will be explained in the sequel. The vector field $\vec{v}$ in (12) may be given, as in (11), or it may depend on the solution u, as in (7) and (9). The right hand side c represents a real constant, which is equal to ± 1 in case of (9) and zero in other two cases.

In the next section we will suggest a numerical discretization scheme of (12) that will be common for all the problems (7), (9), and (11).

Remark 1. Concerning the domain D, one has $D = (-L,L)^d$ when Eq. (7) is modeled by (12). Eqs. (9) and (11) shall be considered independently in two (sub)domains, $D = \Omega(t)$ and $D = (-L,L)^d \setminus \Omega(t)$. The boundary $\Gamma(t) = \partial\Omega(t)$ is then given implicitly as the zero level set of $\phi(x,t) = 0$ for both subdomains.

To define proper boundary conditions for (12), the boundary ∂D must be split to two parts with respect to the flow regime characterized by $\vec{v}$ at ∂D. The outflow part requires no boundary conditions. The inflow part type requires some Dirichlet boundary conditions to be prescribed.

Concerning the implicitly given part $\Gamma(t)$ of boundary ∂D, the correct choice of plus respectively minus sign in (9) and (11) gives always the inflow regime at $\Gamma(t) \subset \partial D$. Therefore the Dirichlet boundary conditions $\phi^0(\gamma) = 0$ for (8) and $s(\gamma) = S(\gamma,t)$ for (10) at $\Gamma(t)$ are appropriate.

3. Numerical methods

In this section we begin with the derivation of a general finite volume numerical scheme for solving the advection equation (12). Afterwards, a brief overview of important particular forms of this scheme is given, and the new semi-implicit version is presented. Finally, the extrapolation technique near the interface is explained for Dirichlet boundary conditions on implicitly given boundary.

3.1. General numerical scheme

To derive the general numerical scheme, we follow [22,24] where much more details are given for finite volume discretization used here, see also references there. Later, when we present examples on structured Cartesian grids, we specify the finite volume mesh in details.

Let x_p be a representative point of the finite volume p with measure m_p. Let the index set $N(p)$ contains all neighboring finite volumes q of p with a nonzero $d-1$ measure of $e_{pq} := \bar{p} \cap \bar{q}$. We will use also some auxiliary finite volumes $q' \in N(p)$ lying outside of D such that $e_{pq'} \subset \partial D$. Furthermore, let us consider a division of time interval $[0,T]$ to $0 = t^0 < \ldots < t^n < t^{n+1} < \ldots < t^N = T$ and, for simplicity, let us consider uniform time step size $\Delta t = t^{n+1} - t^n$, $n = 0,1,\ldots,N-1$. We search for the numerical solution of (12) in the form $u_p^{n+1} \approx u(x_p, t^{n+1})$.

The first step in the derivation of general scheme is considering Eq. (12) as a balance law with a source term given by the divergence of $\vec{v}$,

$$\partial_t u + \nabla \cdot (u\vec{v}) - u \nabla \cdot \vec{v} = c. \tag{13}$$

Then we define integrated fluxes

$$a_{pq} = \int_{e_{pq}} \vec{v} \cdot \vec{\nu}_{pq} ds, \tag{14}$$

where $\vec{\nu}_{pq}$ is the normal vector to e_{pq} pointing from p to q. Note that $a_{pq} = -a_{qp}$. If $\vec{v}$ depends on t we fix it at $t^{n+1/2}$, or we fix it at t^n in nonlinear cases, but we do not emphasize it in the notation for simplicity.

Using definition (14) one can define the following approximation of divergence by an averaging technique at the representative point x_p

$$\nabla \cdot \vec{v}(x_p) \approx \frac{1}{m_p} \int_p \nabla \cdot \vec{v} dx = \frac{1}{m_p} \sum_{q \in N(p)} \int_{e_{pq}} \vec{v} \cdot \vec{\nu}_{pq} ds = \frac{1}{m_p} \sum_{q \in N(p)} a_{pq}. \tag{15}$$

Similarly one obtains

$$\nabla \cdot (u\vec{v})(x_p) \approx \frac{1}{m_p} \sum_{q \in N(p)} \int_{e_{pq}} u\vec{v} \cdot \vec{\nu}_{pq} ds \approx \frac{1}{m_p} \sum_{q \in N(p)} u_{pq} a_{pq}, \tag{16}$$

where u_{pq} represents an averaged value of the solution on the edge e_{pq}. Then using central finite difference approximation for the time derivative and considering the approximations (15) and (16) at $(x_p, t^{n+1/2})$ we obtain the following *general numerical scheme* for solving (12), cf. also [12,13,22–24],

$$u_p^{n+1} + \frac{\Delta t}{m_p} \sum_{q \in N(p)} a_{pq}\left(u_{pq}^{n+1/2} - u_p^{n+1/2}\right) = u_p^n + c\ \Delta t, \tag{17}$$

where $u_p^0 = u^0(x_p)$. Note that the scheme (17) is called "flux-based" level set method in [11–13] due to the flux-based approximation of (15).

To obtain a particular form of the general scheme (17), one has to propose the approximative values $u_{pq}^{n+1/2}$ and $u_p^{n+1/2}$.

3.2. Particular discretization schemes

The first order accurate (explicit in time) upwind method can be obtained from (17) by setting

$$u_{pq}^{n+1/2} = \begin{cases} u_p^n & a_{pq} > 0 \\ u_q^n & a_{pq} < 0, \end{cases} \qquad u_p^{n+1/2} = u_p^n, \tag{18}$$

see [11] for details. Note that one obtains

$$u_{pq}^{n+1/2} - u_p^{n+1/2} = \begin{cases} 0 & a_{pq} > 0 \\ u_q^n - u_p^n & a_{pq} < 0. \end{cases} \tag{19}$$

The method (18) has the CFL type stability restriction on the choice of Δt, namely

$$\frac{\Delta t}{m_p} \sum_{q \in N(p)} \max\{-a_{pq}, 0\} \le 1. \tag{20}$$

Let us consider the one-dimensional case of (12) for a constant velocity $v > 0$ and let $c = 0$, $D = (-L, L)$ and $x_i \equiv x_p$. For variable space discretization step h_i one obtains with (18) that

$$u_i^{n+1} = u_i^n - \Delta t v \frac{u_i^n - u_{i-1}^n}{h_i}. \tag{21}$$

The restriction (20) then takes the form $\Delta t v \le \min_i h_i$. As explained in Remark 1, if D is split according to a position of the interface $\Gamma(t) \in (-L, L)$ one has to add a new grid point at the position of the interface $\Gamma(t)$ which can be situated in any distance h_i from x_i. Because h_i can be arbitrary small, the scheme (18), without any modification, can hardly be used in practice when considering problems with an implicitly given moving interface. In fact, it is true for any explicit in time method unless some modification is used in order to weaken the CFL time step restriction, cf. the so-called "small cell problem" [18,4].

The second order accurate (explicit in time) schemes for the numerical solution of level set advection equation were considered in [12,13]. A piecewise linear reconstruction in space and the first order Taylor expansion in time is used to define the values $u_{pq}^{n+1/2}$ and $u_p^{n+1/2}$. These methods have again the CFL stability restriction on the choice of time step Δt analogous to (20). As it is not trivial to extend them to a form without such restriction [9,7], they are again not practical when considering problems with an implicitly given moving interface.

The first order accurate (fully implicit in time) upwind method for solving (12) can be written in the form (17) by defining $u_{pq}^{n+1/2} = u_p^{n+1}$ if $a_{pq} > 0$, $u_{pq}^{n+1/2} = u_q^{n+1}$ if $a_{pq} < 0$ and $u_p^{n+1/2} = u_p^{n+1}$. Such method has no stability restriction on the choice of Δt and it can produce numerical solutions with no unphysical oscillations. The scheme leads to an algebraic system of equations that must be solved to obtain u_p^{n+1}. Nevertheless, the resulting system matrix is an M-matrix with a special structure, so it can be solved efficiently. The main disadvantage of the first order accurate (fully implicit in time) upwind method is a significant artificial diffusion effect on numerical solutions that decreases the accuracy of results [19,8].

A new approach to derive the second order accurate (semi-implicit in time) methods is developed in [22–24]. The methods are called inflow-implicit/outflow-explicit schemes in [23,24]. They can be written in the form of (17) by setting

$$u_{pq}^{n+1/2} - u_p^{n+1/2} = \begin{cases} u_{pq}^n - u_p^n & a_{pq} > 0 \\ u_{pq}^{n+1} - u_p^{n+1} & a_{pq} < 0. \end{cases} \tag{22}$$

The method developed in this paper is based on the inflow-implicit/outflow-explicit approach (22), too.

Several choices how to define the values u_{pq}^* in (22) for $t^* = t^n$ and $t^* = t^{n+1}$ are discussed in [22–24] with the simplest one being a linear interpolation of values u_p^* and u_q^*. This choice leads to

$$u^*_{pq} = \frac{1}{2}(u^*_p + u^*_q). \tag{23}$$

Note that the value u^*_{pq} in (23) is defined independently of the sign of a_{pq}. The one-dimensional case of this scheme for $v > 0$ and $h_i \equiv h$ is given by

$$u_i^{n+1} + \frac{1}{2}\Delta t v \frac{u_i^{n+1} - u_{i-1}^{n+1}}{h} = u_i^n - \frac{1}{2}\Delta t v \frac{u_{i+1}^n - u_i^n}{h}, \tag{24}$$

that turns in the stationary case to the central finite difference form

$$v\frac{u_{i+1} - u_{i-1}}{2h} = 0.$$

The method (23) has an advantage of producing the algebraic system of equations with an M-matrix. To obtain numerical solution with no unphysical oscillations some stabilization on the right hand side and corresponding adjustment of the left hand side must be used in general. To that goal, other reconstruction techniques respecting the upwind principle different from (23) are considered e.g. in [24].

In this paper we use another type of upwind type reconstructions, both in explicit and implicit part of the scheme (22). For all numerical examples presented in this paper, which are sufficiently general, such new approach does not require any stabilization in the explicit part of the scheme.

The suggested method uses the upwind type piecewise linear reconstruction of numerical solution in space and time in order to define the values u^*_{pq} in (22). Such approach is inspired by the second order explicit level set methods [12,13]. To apply it here, some approximation of gradients ∇u^*_p must be constructed using standard techniques, see e.g. [19,12,13,22]. It allows us to define

$$u^*_{pq} = \begin{cases} u^*_p + \nabla u^*_p \cdot (x_{pq} - x_p) & a_{pq} > 0 \\ u^*_q + \nabla u^*_q \cdot (x_{pq} - x_q) & a_{pq} < 0 \end{cases} \tag{25}$$

where the point x_{pq} is a barycenter of the finite volume face e_{pq}. One can consider (22) and (25) as our newly suggested semi-implicit finite volume level set method of the form (17).

In numerical experiments presented in this paper we use structured Cartesian grids. Thus we discuss here in more details a particular form of (22) and (25) when used on Cartesian grids where it is sufficient to define it only for one-dimensional case.

The grid can be denoted in 1D as $-L = x_{1/2} < x_1 < \ldots < x_{i-1/2} < x_i < x_{i+1/2} < \ldots < x_I < x_{I+1/2} = L$ and $h \equiv x_{i+1} - x_i \equiv x_{i+1/2} - x_{i-1/2}$ for simplicity. Furthermore, one has $x_p \equiv x_i$ and $u^*_{pq} = u^*_{i-1/2}$ or $u^*_{pq} = u^*_{i+1/2}$. Analogously $a_{pq} = -v_{i-1/2}$ or $a_{pq} = v_{i+1/2}$. In order to define (25) in 1D case we set

$$u^*_{i-1/2} = \begin{cases} u^*_{i-1} + \frac{u^*_i - u^*_{i-2}}{2h}\frac{h}{2} = \frac{1}{4}(u^*_i + 4u^*_{i-1} - u^*_{i-2}) & v_{i-1/2} > 0 \\ u^*_i + \frac{u^*_{i+1} - u^*_{i-1}}{2h}\frac{-h}{2} = \frac{1}{4}(u^*_{i-1} + 4u^*_i - u^*_{i+1}) & v_{i-1/2} < 0. \end{cases} \tag{26}$$

For one-dimensional advection equation with positive constant velocity v the method (22) and (26) takes the form

$$u_i^{n+1} + \frac{1}{2}\Delta t v \frac{3u_i^{n+1} - 4u_{i-1}^{n+1} + u_{i-2}^{n+1}}{2h} = u_i^n - \frac{1}{2}\Delta t v \frac{u_{i+1}^n - u_{i-1}^n}{2h}. \tag{27}$$

For a stationary situation, the scheme (27) takes the form

$$v\frac{u_{i+1} + 3u_i - 5u_{i-1} + u_{i-2}}{4h} = 0, \tag{28}$$

that is clearly of upwind type for $v > 0$.

The upwind scheme (26) with (22) does not give an M-matrix in general, because of the positive sign of the coefficient before u^{n+1}_{i-2}, see (27). Nevertheless, for all examples presented here, representing the level set motion in normal direction, no stabilization or other modifications of the scheme is used.

Remark 2. We comment briefly two issues that must be added to the description of our numerical method. Firstly, we note that the velocity $\vec{v}$ in (12) depends on $\nabla\phi$ when the motion in normal direction is considered. In such cases we take $\vec{v}_{pq} = \vec{v}(\nabla\phi^n_{pq})$, where the approximation of $\nabla\phi^n_{pq}$ shall be proposed. We use the so called diamond cell strategy that can be written in two-dimensional case of Cartesian grids for e.g. $\phi^n_{pq} \equiv \phi^n_{i+1/2\,j}$ using the standard notation

$$\nabla\phi^n_{i+1/2j} \approx \left(\frac{\phi^n_{i+1} - \phi^n_i}{h}, \frac{\phi^n_{ij+1} + \phi^n_{i+1j+1} - \phi^n_{ij-1} - \phi^n_{i+1j-1}}{4h}\right), \tag{29}$$

see [22] and the references there for more details.

Finally we note that for the applications and examples considered here we have the outflow type of boundary conditions on fixed parts of $\partial D \subset \partial(-L,L)^d$. To treat them easily we create auxiliary layer of finite volumes outside of D and next to the fixed part of D as described in Section 3.1. Afterwards we associate with auxiliary finite volumes q' some explicit in time values $u^n_{q'}$ obtained by linear extrapolation. To that goal, e.g. in the one-dimensional case, one needs in general only the values $u^n_0 = 2u^n_1 - u^n_2$ and $u^n_{I+1} = 2u^n_I - u^n_{I-1}$.

3.3. Extrapolation near the interface

A special treatment is necessary for our scheme (17) when solving Eq. (12) for $t \in (t^n, t^{n+1})$ on the subdomain $D = \Omega(t^n)$ or $D = (-L,L)^d \setminus \Omega(t^n)$. In both cases we have Dirichlet boundary conditions on the boundary segment $\partial\Omega(t^n) \subset \partial D$ that is given only implicitly. We extend (17) with the so called extrapolation near the interface similarly to the idea of immersed interface method [17,20,9] and analogous approaches [18,5,25,6,14].

We explain the extrapolation near the interface for fixed structured Cartesian grid of $(-L,L)^d$. To solve Eq. (12) using such grid but restricted only to e.g. $D = \Omega(t^n) \subset (-L,L)^d$, we construct algebraic system of equations corresponding to the scheme (22) and (26) only for u^{n+1}_p such that $\phi^n_p < 0$. Analogously, when considering the problem in $D = (-L,L)^d \setminus \Omega(t^n)$, we search for a numerical solution only in grid nodes x_p such that $\phi^n_p > 0$. Of course, if $\phi^n_p = 0$ one can use directly the Dirichlet boundary conditions to define u^{n+1}_p and incorporate them into the algebraic system.

For grid nodes x_p near the interface one can have that the value ϕ^n_q for some $q \in N(p)$ has different sign then the value ϕ^n_p. Consequently, the unknown u^{n+1}_q is not included in the algebraic system, but it may be presented in (17). Analogously, the value u^n_q needs not to be available in (17). To treat such situations for the scheme (26) we use the following approach.

To evaluate $u^*_{i-1/2}$ in the i-th discrete equation, the values from u^*_{i-2} up to u^*_{i+1} at time level n or $n+1$ are required. The idea of extrapolation near the interface is to obtain any value u^*_{i-2} up to u^*_{i+1}, that is not available but required in (26), using a linear extrapolation. This is necessary if any grid node from x_{i-2} up to x_{i+1} lies outside of D.

Any point that lies in a subinterval (x_{i-1}, x_i) can be defined by

$$x_{i-\alpha} = \alpha x_{i-1} + (1-\alpha)x_i, \quad \alpha \in [0,1]. \tag{30}$$

Similarly, any linear interpolation of two values given at x_{i-1} and x_i, e.g. u^*_{i-1} and u^*_i, can be defined analogously to obtain $u^*_{i-\alpha}$. Consequently, if one quantity in (30) is unknown and the rest is given, one can use (30) as an equation to determine the unknown quantity.

The most important usage of (30) is to determine $\alpha \in (0,1)$ such that $\phi^n_{i-\alpha} = 0$. This can happen only when the approximation of $\phi(x,t^n)$ changes its sign between two grid nodes x_{i-1} and x_i. Clearly,

$$\alpha = \frac{\phi^n_i}{\phi^n_i - \phi^n_{i-1}}, \quad \text{if } \phi^n_i \phi^n_{i-1} < 0. \tag{31}$$

Now if the value u^*_{i-1} is unavailable in (26) and the value $u^*_{i-\alpha}$ is given by some Dirichlet boundary conditions, one can use

$$u^*_{i-1} = \frac{\alpha-1}{\alpha} u^*_i + \frac{1}{\alpha} u^*_{i-\alpha} = \frac{\phi^n_{i-1}}{\phi^n_i} u^*_i + \frac{\phi^n_i - \phi^n_{i-1}}{\phi^n_i} u^*_{i-\alpha}, \quad \text{if } \phi^n_i \phi^n_{i-1} < 0. \tag{32}$$

For $t^* = t^n$ the relation (32) can be used directly to define u^n_{i-1}, because the values u^n_i and $u^n_{i-\alpha}$ are available. At the time level $t^* = t^{n+1}$ the value u^{n+1}_i is unknown, so the definition (32) determines the contributions to the diagonal of matrix and the right hand side of the i-th algebraic discrete equation.

We remark that (32) is used also for the $(i+1)$-th discrete equation, when $\phi^n_{i+1}\phi^n_i > 0$ and $\phi^n_i \phi^n_{i-1} < 0$. Moreover, if $\phi^n_i \phi^n_{i-1} < 0$ and the value u^*_{i-2} is required by (26) then

$$u^*_{i-2} = 2u^*_{i-1} - u^*_i, \tag{33}$$

where u^*_{i-1} is determined from (32).

Using analogous approach to (32) when defining the value $u^*_{i+1/2}$ one has all values available to use (26) for the grid nodes near the interface.

We note that the extrapolation near the interface is used also in the computations of $\nabla\phi^n_{pq}$, see Remark 2. In that case the linear extrapolation is used also in diagonal directions, compare with (29).

Remark 3. Before presenting numerical experiments we describe the final form of our inflow-implicit/outflow-explicit upwind scheme with the extrapolation near the interface in the case of two-dimensional structured Cartesian grid. We use notations as mentioned in Remark 2 and in the text before it.

In what follows we use $k,l = -1,0,1$ such that $|k| + |l| = 1$. We rewrite (17) using $u^n_{ij} \equiv u^n_p$, $u^n_{i+k,j+l} \equiv u^n_{pq}$ and so on. Let $\vec{v} = (v,w)$, then we define the coefficients

$$a^{in}_{i+k/2,j} = \min\{0, kv_{i+k/2,j}\}, \qquad a^{out}_{i+k/2,j} = \max\{0, kv_{i+k/2,j}\}, \quad k = -1, 1$$
$$a^{in}_{i,j+l/2} = \min\{0, lw_{i,j+l/2}\}, \qquad a^{out}_{i,j+l/2} = \max\{0, lw_{i,j+l/2}\}, \quad l = -1, 1.$$

The numerical scheme can be now written in the form

$$u^{n+1}_{ij} + \frac{\Delta t}{4h}\sum_{k,l} a^{in}_{i+k/2,j+l/2}\left(-3u^{n+1}_{ij} + 4\bar{u}^{n+1}_{i+k,j+l} - \bar{u}^{n+1}_{i+2k,j+2l}\right)$$
$$= u^{n}_{ij} - \frac{\Delta t}{4h}\sum_{k,l} a^{out}_{i+k/2,j+l/2}\left(u^{n}_{i+k,j+l} - \bar{u}^{n}_{i-k,j-l}\right) + c\Delta t, \tag{34}$$

where the "bars" may indicate extrapolated values. If we solve the level set advection equation (7), the values with bar are simply the values without it, i.e. $\bar{u}^{*}_{i+k,j+l} = u^{*}_{i+k,j+l}$ for $* = n$ or $* = n+1$ and so on.

If we solve the signed distance function using (9) or the extension velocity using (11), i.e. the extrapolation near the interface is required, then for $\phi_{ij} \neq 0$ analogously to (32) we define

$$\bar{u}^{*}_{i+k,j+l} = \begin{cases} u^{*}_{i+k,j+l} & \phi^{n}_{ij}\phi^{n}_{i+k,j+l} > 0 \\ \frac{\phi^{n}_{i+k,j+l}}{\phi^{n}_{ij}} u^{*}_{ij} + \frac{\phi^{n}_{ij} - \phi^{n}_{i+k,j+l}}{\phi^{n}_{ij}} u^{*}_{i+\alpha k,j+\alpha l} & \text{otherwise.} \end{cases} \tag{35}$$

Concerning the value $\bar{u}^{n+1}_{i+2k,j+2l}$ in (34), one has to use (35) with $\bar{u}^{*}_{i+2k,j+2l} = \bar{u}^{*}_{(i+k)+k,(j+l)+l}$ if $\phi^{n}_{i}\phi^{n}_{i+k,j+l} > 0$, otherwise

$$\bar{u}^{n+1}_{i+2k,j+2l} = 2\bar{u}^{n+1}_{i+k,j+l} - u^{n+1}_{ij}.$$

Finally we note that we do not allow Dirichlet boundary conditions on outflow type of boundary, therefore $a^{out}_{i+k/2,j+l/2} = 0$ if $\phi^{n}_{i}\phi^{n}_{i+k,j+l} < 0$.

4. Numerical experiments

The following numerical experiments were computed using the semi-implicit finite volume scheme (34). Firstly, the most difficult particular equation of the abstract form (12) is computed, the eikonal equation (8). Secondly, the example taken from [1] is studied to illustrate the advantages of using the extension velocity computed by (10) instead of the natural velocity. Finally, an example of the form (7) with discontinuous natural velocity is computed that is inspired by numerical simulation of forest fire front propagation. Again, the results computed by our scheme with the extension and the natural velocity are presented and compared.

In what follows, the domain D is always a square. For each particular example the coordinates of the left bottom corner of D and the length L of the side are specified. We consider always a uniform grid with $h = L/I$ where I is chosen.

The algebraic system of equation is always solved with fast sweeping Gauss–Seidel iterations [29] using the maximum of 4 iterations (each one with 4 sweeps) or the reduction of the residuum below 10^{-20}.

When computing the signed distance function or the extension speed, we consider the time dependent form of Eq. (12) and search its stationary solution, see (9) and (11). The number of (artificial time) relaxation steps is always commented in each example.

4.1. Distance functions

In this section we compute signed distance functions to three different interfaces using the scheme (34) with the extrapolation near the interface. For two examples the exact solution is available and compared with the obtained numerical solution using the L_1 discrete error E_1 at $t = t^N$ by

$$E_1(I) = h^2 \sum_{p} \left| u\left(x_p, t^N\right) - u^N_p \right|. \tag{36}$$

For the last example the computed numerical solution is compared with the one obtained by a "brute force" method.

The domain D is the unit square with the left bottom corner having coordinates $(0, 0)$. We consider uniform grids with $h = 1/I$ where $I = 20$, 40, 80, 160 and 320. The time step is chosen $\Delta t = 4h$ and the number of relaxation steps is chosen $N = I$.

In the first example we search for the distance function to a circle that can be seen as a simple representative smooth solution. The middle point of circle is chosen $(0.51, 0.52)$ and the radius equals 0.26. The initial function (or the "initial guess") Φ^0 is simply the exact distance function multiplied by 0.5. The initial function and the obtained numerical solutions on the coarsest grid with $n = 1$, $n = 6$ and $n = N = 20$ can be viewed in Fig. 1. The experimental order of convergence (EOC) using (36) can be viewed in Table 1. One can observe a clear second order accuracy for this example.

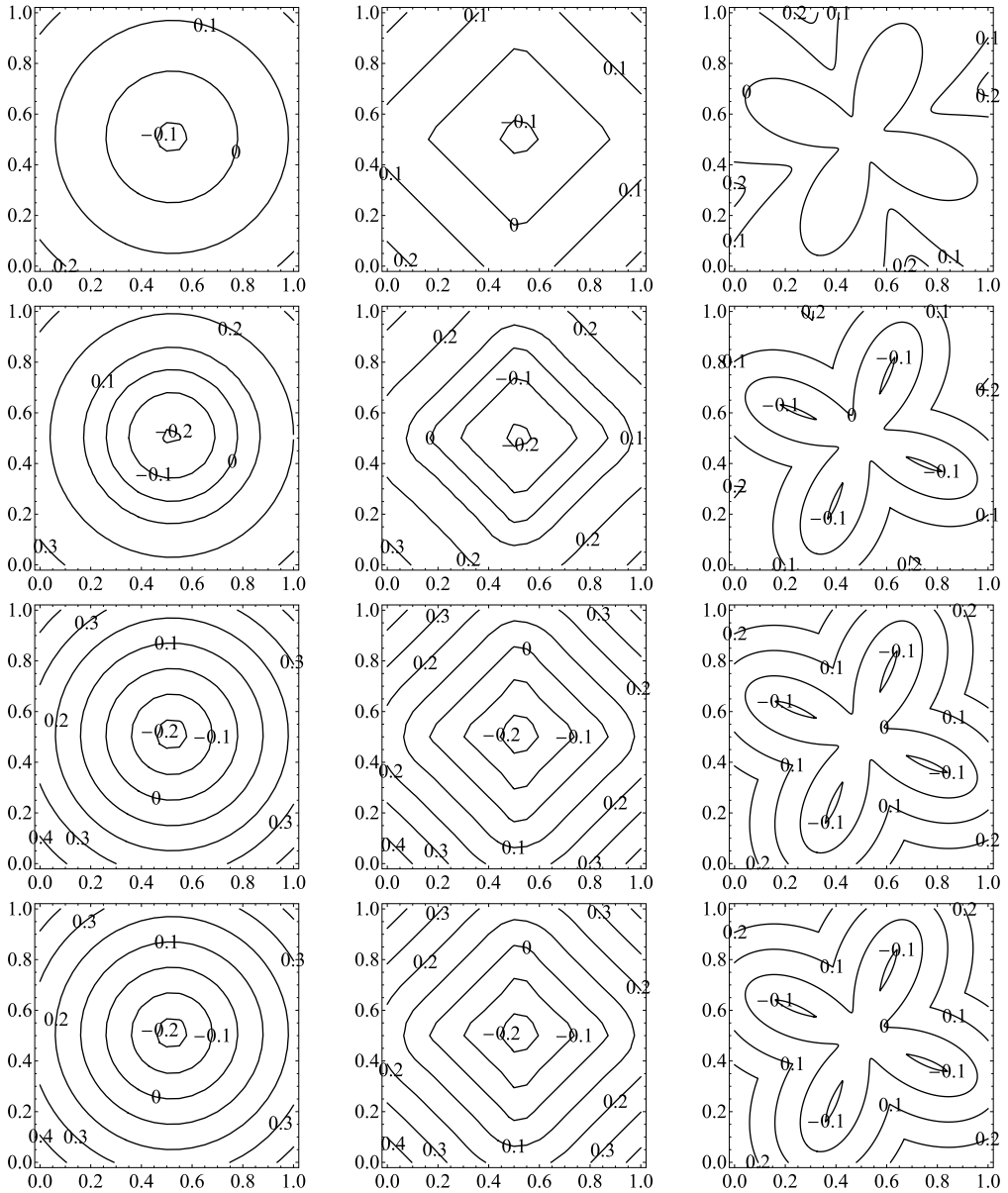

Fig. 1. The initial condition (the 1st row) and the results after n relaxation steps (the 2nd, 3rd and 4th row). The first and second column is obtained with the coarse grid $I = 20$ and $n = 1, 6$, and 20, the third column with the fine grid $I = 320$ and $n = 8, 20$, and 80. The last row contains practically stationary solutions, i.e. the approximated signed distance functions with E_1 errors given in Table 1.

Table 1
The discrete E_1 errors (36) and the EOCs for the distance function to the circle (2nd and 3rd column), the square (4th and 5th column) and the quatrefoil (6th and 7th column).

I, N	$E_1(I)$	EOC	$E_1(I)$	EOC	$E_1(I)$	EOC
20	9.31E–4	–	2.54e–2	–	6.15e–3	–
40	2.31E–4	2.01	8.22e–3	1.63	1.72e–3	1.83
80	5.72E–5	2.01	2.98e–3	1.46	6.19e–4	1.48
160	1.43E–5	2.00	1.22e–4	1.28	2.20e–4	1.49
320	3.56E–6	2.01	5.31e–4	1.20	7.89e–5	1.48

The second example is the distance function to a (rotated) square with analogous data to the previous example, see Fig. 1. The exact distance function contains "knicks" and can be viewed as a simple representative non-smooth solution. The EOC for this example is converging to 1 from above, see Table 1. Note that for $I = 320$ the signed distance function for the circle and the square with the presented E_1 errors in Table 1 can be obtained after less than 100 relaxation steps.

Table 2
The localized discrete errors $E_l(I)$ from (37) and the corresponding EOCs for the advection of circle (3rd and 4th column) and the square (5th and 6th column).

I	N	$E_l(I)$	EOC	$E_l(I)$	EOC
20	2	3.06E–4	–	7.88e–3	–
40	4	3.59E–5	3.09	3.74e–3	1.08
80	8	8.62E–6	2.06	1.87e–3	1.00
160	16	2.36E–6	1.87	9.52e–4	0.94
320	32	5.75E–7	2.04	4.60e–4	1.05

In the third example we choose an interface in the form of rotated "quatrefoil", where no analytical solution for the signed distance function is available. The initial guess $\Phi^0(x)$ in (9) is taken as

$$\Phi^0(x) = -0.049 + \frac{0.1\sqrt{(x-0.5)^2+(y-0.5)^2}}{r_L},$$
$$r_L = 0.6 + 0.4\sin\left(4\,\mathrm{arctg}\left(\frac{0.5-y}{0.5-x}\right)\right).$$

The initial function and the computed signed distance function after 8, 20 and 320 relaxation steps for the finest grid $I = 320$ can be seen in Fig. 1.

To estimate the EOC for this example, we compute the distance function using a simplified "brute force" method. This method computes an approximative distance for each grid node to the polygonal line. The polygonal line is given as the zero level set of the linear interpolation of nodal values $\Phi^0(x_p)$ on this grid. The brute force method is simplified (approximative) in the sense that the distance of a point x_p to the polygonal interface is computed only from the distances of x_p to the points that defines the polygon. In Table 1 this estimate suggests that such EOC is around 1.5. Note that for $I = 320$ the signed distance function with the presented E_1 error in Table 1 can be obtained after less than 80 relaxation steps.

Finally we illustrate the qualitative behavior of our numerical scheme (34) concerning the approximation of advected interface when solving the level set advection equation (7). To do so, we take the signed distance function to the circle and the square as in previous experiments and let it evolve in normal direction with the speed 1 up to $T = 0.1$. Afterwards we find all intersection points of approximated interface with the edges of finite volumes and compute the absolute difference of their distance, say d_k, and the exact distance 0.1 to the initial interface. Finally, we define the localized discrete error E_l by

$$E_l(I) = \frac{1}{K(I)}\sum_{k=1}^{K(I)} |d_k - 0.1|, \tag{37}$$

where $K(I)$ is the number of intersection points corresponding to the grid with $h = 1/I$. The values $E_l(I)$ for consecutively refined grids can be found in Table 2 together with the corresponding EOCs. One can observe second order accuracy for the advected circle and the first order accurate results for the advected square.

4.2. Comparison of natural and extension velocity

In the following example [1] we compute the level set advection equation (7) for the motion of an interface in normal direction by semi-implicit finite volume scheme (34). Firstly, we use an explicitly given natural speed from [1], then the extension speed computed from (11) by (34) with the extrapolation near the interface is used.

The domain is $D = (-7.5, 7.5)^2$. The natural speed $S = S(x, y)$ in normal direction is given by

$$S = \left(\left(\sqrt{x^2+y^2} - 3\right)^2 + 1\right)\left(2 + \sin(4\theta)\right). \tag{38}$$

The angle $\theta = \theta(x, y)$ is made between the vector (x, y) and the positive x axis. Note that S attains its maximum at corners of D being approximately 118, and the minimum is 1, see also Fig. 2.

The initial level set function $\phi^0(x, y)$ is the distance function to the circle with radius 3 and the center at $(0, 0)$. Note that $S(x, y)$ varies between 1 and 3 for (x, y) such that $\phi^0(x, y) = 0$.

Firstly, we compute the advection equation (7) with $s \equiv S$ being the natural speed in normal direction. We start with $I = 50$, so $h = 15/I$, and $\Delta t = 0.9/N$ where N, the number of time steps, is chosen.

To test the computations with very large grid Courant number, we start with $N = 1$, see Fig. 2 for the numerical solution at $t = 0.9$. Although the numerical solution is obtained with only one time step, no unphysical oscillations can be observed.

Of course, using $N = 1$ leads to a large time discretization error. Doing several experiments it occurs that $N = 256$ gives a satisfactory numerical approximation of the zero level set at $t = 0.9$. It means choosing $N = 512$ does not bring visible changes to its position. The numerical solution for $N = 256$ at $t = 0.9$ is presented in Fig. 2.

To study also the space discretization error, we compare the position of zero level set for the numerical solutions obtained with $I = 50$, 100, 200, and 400, see Fig. 3. Note that the position of approximative interface for $I = 50$ is relatively far

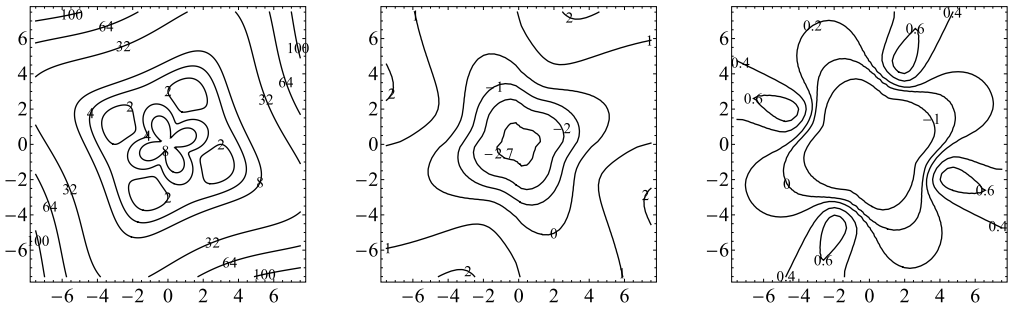

Fig. 2. The function S from (38) (the left picture). The advected numerical level set function at $t = 0.9$ for $I = 50$ obtained with only one time step (the middle picture) and the one obtained with $N = 256$ time steps.

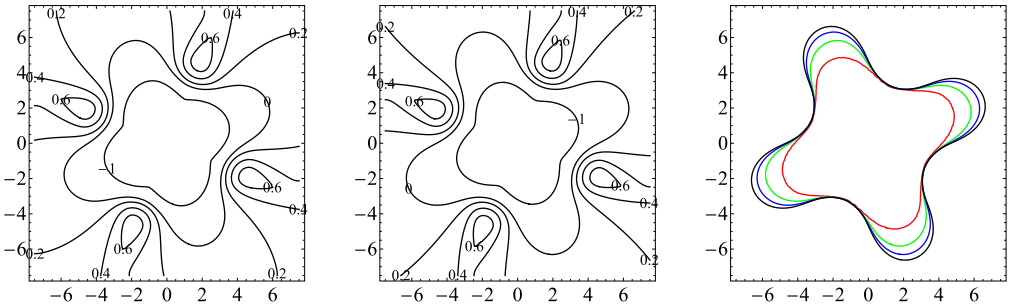

Fig. 3. The advected numerical level set function at $t = 0.9$ using the natural speed for $I = 100$ (the left picture) and 200 (the middle picture), compare also with the right picture in Fig. 2 for $I = 50$. In the right picture one can see the grid convergence of the zero level set for the numerical solutions of (7) for $I = 50$ (the shortest curve), 100, 200 and 400 (the longest curve).

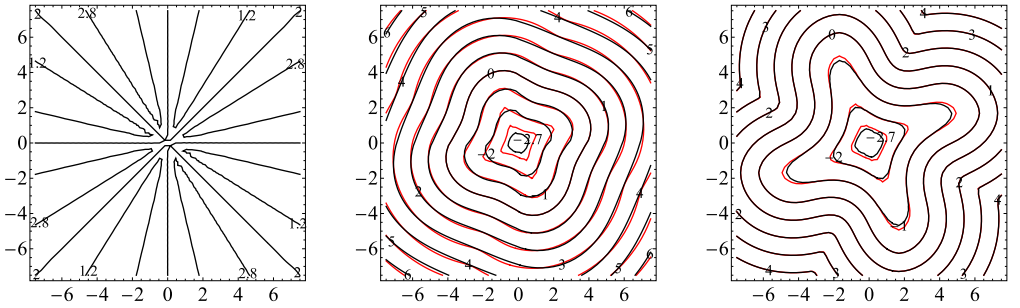

Fig. 4. The numerical extension speed $s(x, y, 0)$ (the left picture), the advected numerical level set at $t = 0.9$ using the extension speed with only one time step (the middle picture) and the analogous numerical solution using $N = 256$ time steps. In the last two pictures the advected level set function is plotted with black color and the signed distance function with red color (for color pictures see the online version of this article). The grid resolution is $I = 50$.

away from the positions for finer grids. Interestingly, the numerical solutions for all grids can be computed with $N = 256$, because there is no visible difference when obtaining them with $N = 512$.

Next, we compute the advection equation (7) using the extension speed. The extension speed s is computed from (11) by using the scheme (34) with the extrapolation near the interface and a given number of relaxation steps. Enough many relaxation steps are used to obtain a good approximation of the stationary solution $w(x, y, \tilde{\tau})$ of (11) at $t = 0$, see Fig. 4 for the result where $s(x, y, 0) = w(x, y, \tilde{\tau})$. Afterwards, to compute an approximation of the extension speed $s(x, y, t)$ for $t > 0$ only 8 relaxation steps are used to solve (11) with $\Delta\tau = h$.

To illustrate the behavior of numerical advected level set function for very large time step, we use $N = 1$ to obtain the numerical solution of (7) at $t = 0.9$, see Fig. 4. Moreover, we compare it with the computed signed distance function using the method as presented in the previous Section 4.1. Although the time discretization error is large, no unphysical oscillations can be observed, and the advected level set function retains well the property of staying the distance function, see Fig. 4.

To reduce the time discretization error, we compute the example with $N = 256$. To estimate the space discretization error we compute the example for $I = 100$ (8 relaxation steps) and 200 (16 relaxation steps) and 400 (32 relaxation steps). The comparison of the zero level set for advected numerical level set function can be found in Fig. 5. Much better precision

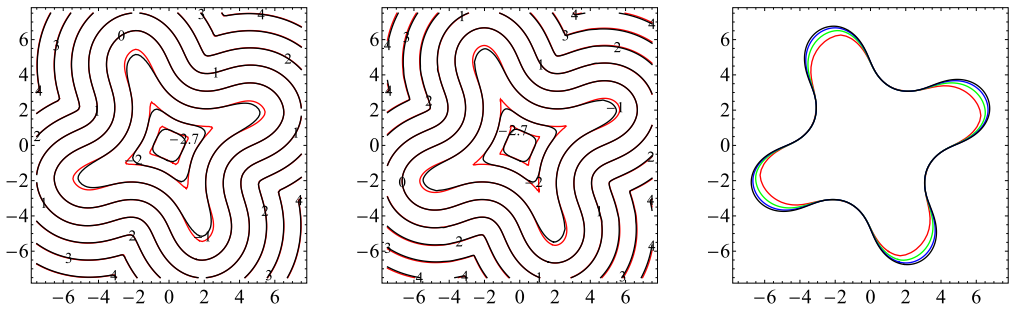

Fig. 5. The numerical advected level set function (the black contour lines) at $t = 0.9$ using the extension speed computed from (11) for $I = 100$ (the left picture) and for $I = 200$ (the middle picture). Moreover, the result in these pictures is compared with related numerical signed distance function (the red contour lines in online version of this paper). In the right picture one can see the zero level set of numerical solutions for $I = 50$ (the shortest curve), 100, 200 and 400 (the longest curve).

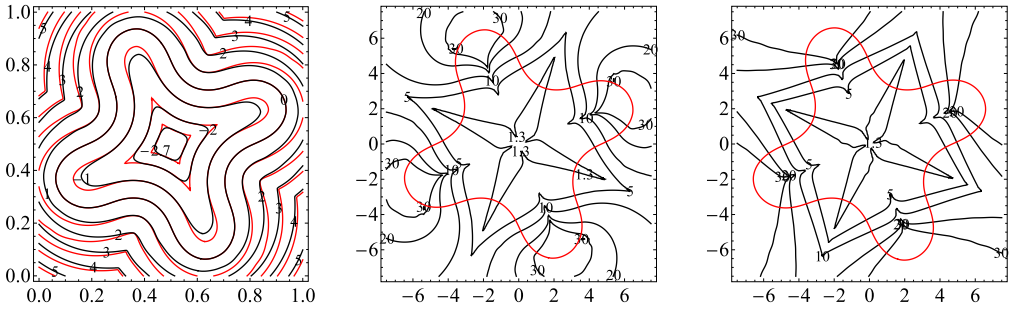

Fig. 6. The numerical advected level set function at $t = 0.9$ computed for $I = 200$ with only 2 relaxation time steps to compute the extension speed (the left picture, compare with Fig. 5). The related numerical extension speed used in the last advection step (the middle picture, the position of interface is plotted, too). In the right picture one can see the analogous numerical extension speed when using 16 relaxation time steps.

is obtained for the position of zero level set at $t = 0.9$, especially for $I = 50$, when using the extension speed than for the computations with the natural speed, compare the right picture in Fig. 5 with the right picture in Fig. 3.

Finally, we compute the example for $I = 200$ when the number of relaxation time steps to compute (11) is only 2. As it can be seen in Fig. 6, the advected numerical level set function does not preserve the property being the distance function so well, but the approximation is still very good. For an illustration we plot in Fig. 6 also the obtained extension speed that is used in the last advection time step for two different number of relaxation time steps.

4.3. Forest fire propagation

The final example is motivated by some real application when a propagation of fire in forest is considered [3]. Our aim is to test the presented numerical method in such settings to understand better its properties and not to compete with other numerical approaches like the one used in [3].

The input data for such application are forestry typological maps that can be transfered to grayscale pictures. The gray level of each pixel characterizes how fast can burn the underlying forest. Particularly, the black color means a nonburnable material that we quantify with $S = 0$, the white color denotes the most burnable material that we quantify with $S = 1$. In such way we obtain the function S that characterizes the natural speed of fire propagation. Note that S has a piecewise constant form. The picture of a typological map with the resolution of 160^2 pixels that we use in our computations can be seen in Fig. 7. The data are obtained from Slovak state forest company Vojenské lesy a majetky SR, Malacky [3].

To model the fire propagation we suppose that at each time a sharp interface (the front) can be recognized. Any front separates locally a completely burnt area from the one that is not yet burnt. The interface (the fire fronts) can be made of several closed curves. The number of curves can change in time, it means the fire fronts can split or merge. For the speed of moving interface (the fire fronts) we consider here the simplest variant that the fronts expand only in the normal direction with the speed S at the current position of fire.

Typical results of related simulations are presented in Fig. 7 where two approaches are compared. The left picture shows the results obtained with the extension speed s obtained from (11), the right one using the natural speed S. The behavior of both methods, when considered only for the evolving zero level set, is similar, but some differences can be recognized by a closer look. We comment first the evolution of the fire in this experiment.

The initial position of the interface is defined by two small circles that can be viewed formally as two separate origins of the fire, see Fig. 7. The blue color (see the online version of paper) represents the interface at a consecutive time point when these two fire fronts are going to merge. At the same time, one small nonburnable area is almost enclosed by the

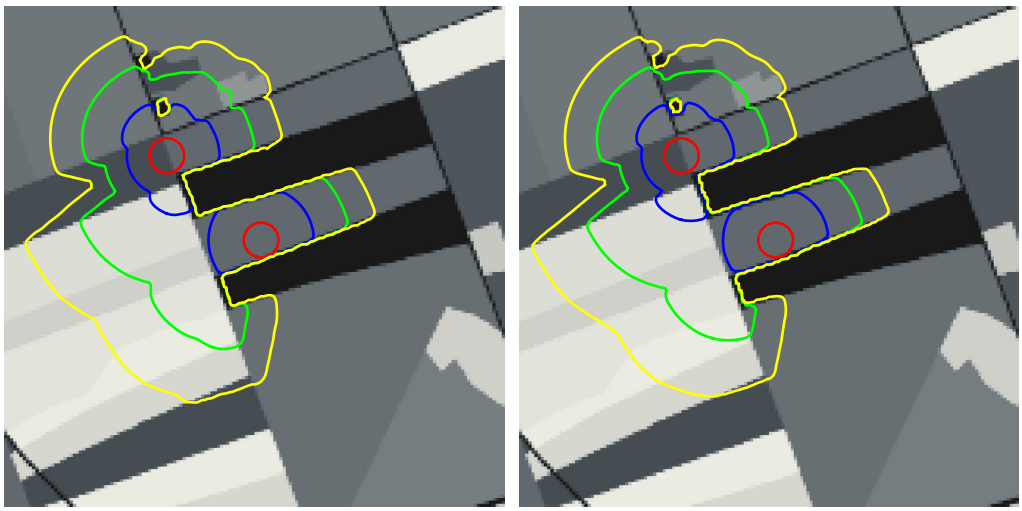

Fig. 7. The fire fronts plotted at the initial position (two small circles) and at three different consecutive times. The left picture uses the extension speed, the right pictures uses the natural speed. A close look shows that using the extension speed a sharper resolution of nonburnable areas is obtained.

fire. Finally, the yellow curves represent the position of interface at the last time point when it is split into two separate fire fronts where one front is enclosing small nonburnable area.

Concerning the parameters of this numerical experiment, the finite volume mesh coincides with pixels and the time step equals the size of pixel. The actual position of interface is plotted after 0, 25, 50 and 70 time steps. When computing the extension speed, eight relaxation time steps for the numerical solution of (11) are used.

Next we comment some differences in the approximation of evolving interface when using the extension or natural speed. Analogously to [1] we can confirm the so called subgrid accuracy effect when using the extension speed. The subgrid accuracy is an issue for instance when the input data (like the function S) have piecewise constant form, so large jumps of values can occur. When using the extension speed, the natural speed S is evaluated exactly at some positions of zero level set to treat Dirichlet boundary conditions, see the values $x_{i-\alpha}$ in (30) with (31). Consequently, the correct values of the speed S at the interface are extended to grid points surrounding $x_{i-\alpha}$ to define the extension speed s.

On the other hand, when using the natural speed S with any discretization methods for the numerical solution of advection equation (12), the speed S is evaluated only in integration points (e.g. the grid points) independently on the fire position. Consequently, one can confirm more accurate resolution when using the extension speed. This can be observed also in Fig. 7, where, for instance, the position of fire fronts enter less into the nonburnable areas with the extension speed than with the natural speed.

Finally, we compare in Fig. 8 the advected numerical level set function at the last time step when using the two different approaches. As expected, using the extension speed the numerical advected level set function is close to the distance function. One has to stress that this approximation is good only in the unburnt area (the positive distance) and not away from the interface into the burnt area (the negative distance). This is due to the fact that the initial interface attains only small negative values (the minus initial radius). The numerical method shall not produce the values outside of the range given by the initial values (and boundary values) that is also observed in this numerical experiment. If for any reason also the approximation of (larger) negative distances is important, one can replace at some time point the advected numerical level set function with the signed distance function obtained by the approach of Section 4.1.

In numerical experiment with the natural speed one cannot expect that the advected level set function resembles the signed distance function with the property $|\nabla\phi| = 1$. By contrast, the steep gradients of advected numerical level set function can be observed at boundaries of nonburnable areas and the flat ones at the fast burning areas. As more sophisticated mathematical models of forest fire propagation can include a dependence on the curvature of interface [3], such distortion can decrease significantly the accuracy of results.

5. Conclusions

In this paper the semi-implicit finite volume level set is proposed. It combines the idea of flux-based level set method [12,13] with the inflow/implicit and outflow/explicit time discretization in [22,24]. The method is successfully applied to numerical solutions of advection level set equation for the motion in normal direction. For the presented examples the time step is restricted only by accuracy issues, and no CFL stability restriction is required.

Following the idea of [30,1,9], the method is applied also to two auxiliary problems, the computation of signed distance function and the construction of extension speed. The signed distance function is used in the initial conditions for the advection level set equation, and the extension speed is used as the speed in normal direction in the same advection level set equation.

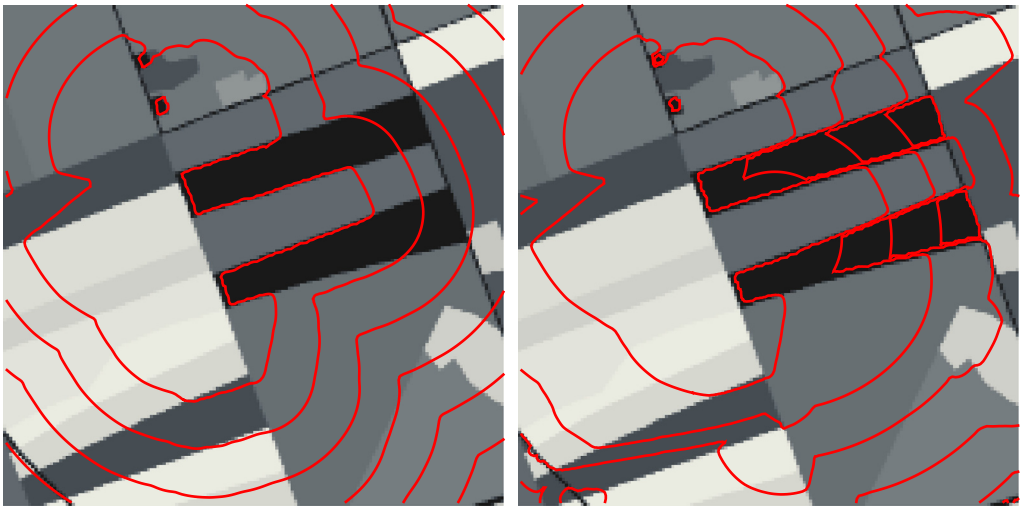

Fig. 8. The contour lines of numerical advected level set function for the last time point when using the extension speed (the left picture) or the natural speed (the right picture). In the left picture the distance function is well preserved (at least for positive values), in the right picture a large distortion of level set function occurs with steep gradients between nonburnable and burnable areas.

To use the method for these two auxiliary problems when some Dirichlet boundary conditions are defined on a boundary given only implicitly, we use the extrapolation near the interface similarly to the immersed interface method [17,9]. In this approach one can again clearly profit from the fact that no CFL restriction is required in our semi-implicit method.

Using the semi-implicit finite volume level set method we expect a positive contribution to the accuracy and stability of computations even when some nontrivial applications like two-phase flows [10], moving groundwater table [7], or the forest fire propagation [3] are treated. Such applications can require the computations of curvature [10,3] and/or the computation of some additional PDEs to compute the natural speed [10,7].

References

[1] D. Adalsteinsson, J. Sethian, The fast construction of extension velocities in level set methods, J. Comput. Phys. 148 (1999) 2–22.
[2] T.D. Aslam, A partial differential equation approach to multidimensional extrapolation, J. Comput. Phys. 193 (2003) 349–355.
[3] M. Balažovjech, K. Mikula, M. Petrášová, J. Urbán, Lagrangian methods with topological changes for numerical modelling of forest fire propagation, in: A. Handlovičová, et al. (Eds.), Proceeding of Algoritmy 2012, Slovak University of Technology, Bratislava, 2012, pp. 42–52.
[4] M.J. Berger, R.J. LeVeque, An adaptive Cartesian mesh algorithm for the Euler equations in arbitrary geometries, in: 9th AIAA Computational Fluid Dynamics Conference, 1989, pp. 1–7.
[5] D. Calhoun, R.J. LeVeque, A Cartesian grid finite-volume method for the advection–diffusion equation in irregular geometries, J. Comput. Phys. 157 (2000) 143–180.
[6] R. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys. 152 (1999) 457–492.
[7] P. Frolkovič, Application of level set method for groundwater flow with moving boundary, Adv. Water Resour. 47 (2012) 56–66.
[8] P. Frolkovič, Maximum principle and local mass balance for numerical solutions of transport equation coupled with variable density flow, Acta Math. Univ. Comen. 67 (1998) 137–157.
[9] P. Frolkovič, Flux-based level set method for extrapolation along characteristics using immersed interface formulation, in: P. Struk (Ed.), Magia, Slovak University of Technology, Bratislava, 2010, pp. 15–26.
[10] P. Frolkovič, D. Logashenko, G. Wittum, Flux-based level set method for two-phase flows, in: R. Eymard, J.M. Herard (Eds.), Finite Volumes for Complex Applications, ISTE and Wiley, 2008, pp. 415–422.
[11] P. Frolkovič, K. Mikula, Flux-based level set method: a finite volume method for evolving interfaces, Appl. Numer. Math. 57 (2007) 436–454.
[12] P. Frolkovič, K. Mikula, High-resolution flux-based level set method, SIAM J. Sci. Comput. 29 (2007) 579–597.
[13] P. Frolkovič, C. Wehner, Flux-based level set method on rectangular grids and computations of first arrival time functions, Comput. Vis. Sci. 12 (2009) 297–306.
[14] F. Gibou, R.P. Fedkiw, L.T. Cheng, M. Kang, A second-order-accurate symmetric discretization of the Poisson equation on irregular domains, J. Comput. Phys. 176 (2002) 205–227.
[15] S. Gross, A. Reusken, Numerical Methods for Two-Phase Incompressible Flows, Springer, 2011.
[16] M. Herreros, M. Mabssout, M. Pastor, Application of level-set approach to moving interfaces and free surface problems in flow through porous media, Comput. Methods Appl. Math. 195 (2006) 1–25.
[17] R. Leveque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.
[18] R.J. LeVeque, Cartesian grid methods for flow in irregular regions, in: K.W. Morton, M.J. Baines (Eds.), Num. Meth. Fl. Dyn. III, Clarendon Press, 1988, pp. 375–382.
[19] R.J. LeVeque, Finite Volume Methods for Hyperbolic Problems, Cambridge University Press, 2002.
[20] Z. Li, K. Ito, The Immersed Interface Method: Numerical Solutions of PDEs Involving Interfaces and Irregular Domains, SIAM, 2006.
[21] P. Macklin, J. Lowengrub, Evolving interfaces via gradients of geometry-dependent interior Poisson problems: application to tumor growth, J. Comput. Phys. 203 (2005) 191–220.

[22] K. Mikula, M. Ohlberger, A new level set method for motion in normal direction based on a semi-implicit forward–backward diffusion approach, SIAM J. Sci. Comput. 32 (2010) 1527–1544.
[23] K. Mikula, M. Ohlberger, Inflow-implicit/outflow-explicit scheme for solving advection equations, in: J. Fořt, et al. (Eds.), Finite Volumes for Complex Applications VI, Springer Verlag, 2011, pp. 683–692.
[24] K. Mikula, M. Ohlberger, J. Urban, Inflow-implicit/outflow-explicit finite volume methods for solving advection equations, Universitaet Münster, February 2012, Preprint 01/12.
[25] C. Min, F. Gibou, A second order accurate level set method on non-graded adaptive Cartesian grids, J. Comput. Phys. 225 (2007) 300–321.
[26] S. Osher, R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Springer, 2003.
[27] J. Sethian, Level Set Methods and Fast Marching Methods, Cambridge University Press, 1999.
[28] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[29] H. Zhao, A fast sweeping method for eikonal equations, Math. Comput. 74 (2004) 603–628.
[30] H.K. Zhao, T. Chan, B. Merriman, S. Osher, A variational level set approach to multiphase motion, J. Comput. Phys. 127 (1996) 179–195.