School of Mathematical, Physical
and Computational Sciences

University of Reading

Department of Mathematics and Statistics

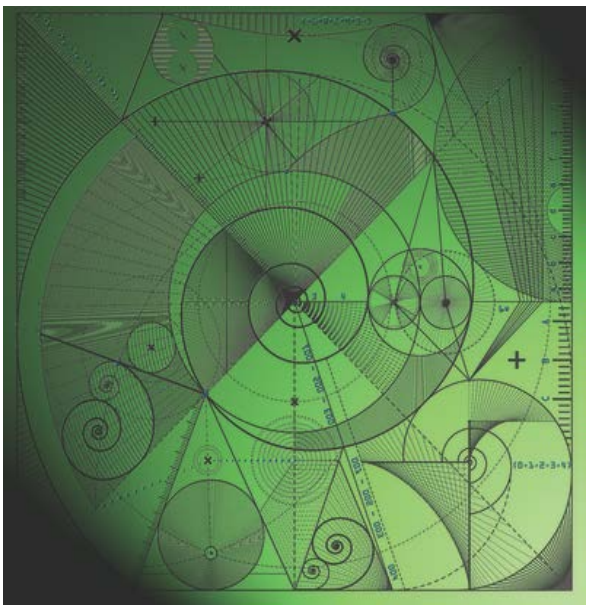
Preprint MPCS-2019-08

14 October 2019

Non-tangling moving-mesh methods for PDE problems in one and two dimensions I: mass-conserving problems

by

M.J. Baines



Non-tangling moving-mesh methods for PDE problems in one and two dimensions I: mass-conserving problems

M.J.Baines

Department of Mathematics and Statistics,
University of Reading, P.O.Box 220, Reading, RG6 6AX, UK

Abstract

Non-tangling moving-mesh algorithms based on local conservation for scalar mass-conserving PDE problems are derived in one and two dimensions, in the latter case on a linear simplex. Mesh tangling is prevented for any time step by applying an explicit sign-preserving exponential time-stepping scheme to intervals in 1-D or edges/areas in 2-D. The nodes are found in a separate step respecting the integrity of the mesh.

1 Introduction

Moving-mesh methods for the approximate solutions of time-dependent partial differential equations (PDEs) are potentially more powerful than fixed mesh methods, capable of providing high resolution locally, sustaining scale invariance and propagating self-similarity [12, 8].

In the velocity-based conservation method of [1, 4, 15] a velocity field is determined from a local Eulerian conservation law which is used to deform the domain, finding the moved solution by local Lagrangian conservation. Applications can be found in [1, 2, 3, 4, 14, 16, 15, 9, 5, 6, 10, 7, 17]. The method inherits the scale-invariance of the PDE problem, handles flux-driven moving boundary conditions, whether external or internal, in a natural way without the need for interpolation, and is capable of propagating self-similar scaling solutions at the nodes.

Numerically, the default time-stepping scheme has been explicit Euler. However, it is a requirement of the conservation method that the solution remains positive and the mesh untangled, which in many cases demands a very small time-step, therefore limiting the practicality of the method.

In this paper a variant of the method is described which ensures a positive solution on an untangled mesh for any time step. This is achieved by applying a sign-preserving exponential time stepping scheme to intervals in 1-D and to edges or areas of a simplex in 2-D, with the nodal positions found in a separate step.

The paper is organised as follows. In section 2 the conservation method is reviewed in one dimension, a semi-discrete scheme analysed, and a fully discrete version described that uses the explicit exponential time-stepping scheme on interval lengths, leading to sign-preserving solutions on ordered meshes for any time step and culminating in the algorithms given in section 2.3.2.

In section 3 the two-dimensional conservation method is reviewed and the features described in 1-D generalised to 2-D on a simplex of triangles by applying the sign-preserving exponential time-stepping scheme to either edge-lengths or triangle-areas. The nodes are found by an averaging procedure which retains the integrity of the simplex and solutions on the moved mesh obtained from Lagrangian conservation. The algorithms are given in section 3.3.5.

Conclusions are drawn in section 4.

2 The conservation method in 1-D

Suppose that the function $u(x,t)$ satisfies the generic PDE

$$u_t = \mathcal{L}u, \tag{1}$$

in an interval $(a(t), b(t))$, where $\mathcal{L}$ is a purely spatial operator, with boundary conditions such that the total mass

$$\int_{a(t)}^{b(t)} u(\xi,t)\,\mathrm{d}\xi \tag{2}$$

is independent of time. Such problems arise in in the study of nonlinear diffusion, for example.

At any time t let the points $x(t)$ of the domain $(a(t), b(t))$ move with a velocity $v(x,t)$ so as to satisfy the partial Lagrangian conservation law

$$\int_{a(t)}^{x(t)} u(\xi,t)\,\mathrm{d}\xi = c(x), \text{ independent of } t, \tag{3}$$

consistent with (2) if $c(b) = 1$.

In an Eulerian form equivalent to the conservation law (3), the function $\overline{u}(x,t)$ satisfies

$$u_t + (uv)_x = 0, \tag{4}$$

where $v(x,t)$ is the Eulerian velocity . Thus, from equations (1) and (4), at any time t the velocity $v(x,t)$ satisfies the ODE

$$-\frac{\mathrm{d}}{\mathrm{d}x}(uv) = \mathcal{L}u \tag{5}$$

After integration from an anchor point (taken to be $a(t)$ without loss of generality) to a general point $x(t)$, equation (5) yields

$$-\left(\overline{u}(x,t)v(x,t)\right|_{a(t)}^{x(t)} = \int_{a(t)}^{x(t)} \mathcal{L}(u)\,\mathrm{d}\chi$$

so that

$$v(x(t),t) = v(a(t),t) - \frac{1}{u(x(t),t)}\int_{a(t)}^{x(t)} \mathcal{L}(u)\,\mathrm{d}\chi \tag{6}$$

2.1 A reference space

Introduce a Lagrangian moving coordinate $\widehat{x}(\xi,\tau)$, where ξ is a fixed reference coordinate, such that

$$\frac{\partial \widehat{x}}{\partial \tau} = \widehat{v}(\xi,\tau), \qquad \tau = t, \tag{7}$$

where

$$\widehat{v}(\xi,\tau) = v(\widehat{x}(\xi,\tau),\tau)$$

By the chain rule

$$\widehat{v}(\xi,\tau) = v(\widehat{x}(\xi,\tau),\tau)$$

We consider two equations in the reference space.

1. Differentiating (7) with respect to ξ,

$$\frac{\partial \widehat{x}_\xi}{\partial \tau} = \widehat{v}_\xi \tag{8}$$

from which the moving coordinate $\widehat{x}(\xi,t)$ is retrieved from $\widehat{x}_\xi$ using

$$\widehat{x}(\xi,\tau) = \widehat{x}_0 + \int_{\xi^0}^{\xi} \widehat{x}_{\xi'}\,\mathrm{d}\xi' \tag{9}$$

where $\widehat{x}(\xi,\tau) = \widehat{x}_0$ at $\xi = \xi_0$.

2. The Lagrangian conservation form (3) transforms to

$$\widehat{u}(\xi,\tau)\,|\widehat{x}_\xi| = \widehat{c}(\xi), \text{independent of } \tau, \tag{10}$$

for all ξ, where $\widehat{u}(\xi,\tau) = u(\widehat{x}(\xi,\tau),\tau)$ and $\widehat{x}_\xi$ is the Jacobian of $\widehat{x}$ with respect to ξ. The sign of $\widehat{x}_\xi$ is determined by (8).

2.1.1 An initial value problem

Substitution of $\widehat{u}(\xi,\tau)$ from (10) into (8) yields an ODE for $\widehat{x}_\xi$. Given initial conditions on $\widehat{x}$ and $\widehat{u}$ (and hence $\widehat{c}(\xi)$) at $\tau = \tau^0$ equation (8) is an initial value problem for $\widehat{x}_\xi$ possessing a unique solution under the conditions of Picard's Theorem.

From equation (8),

$$\frac{\partial \log \widehat{x}_\xi}{\partial \tau} = \frac{\widehat{v}_\xi}{\widehat{x}_\xi} = v_x \tag{11}$$

since $\widehat{v}_\xi/\widehat{x}_\xi = v_x$ at time $t = \tau$. Integrating (11) from τ^0 to τ, equation (11) has the formal solution

$$\widehat{x}_\xi = \widehat{x}^0_\xi \exp\left\{\int_{\tau^0}^{\tau} v_x \,\mathrm{d}\tau'\right\} \tag{12}$$

where $\widehat{x}_\xi = \widehat{x}^0_\xi$ at $\tau = \tau^0$. The solution $\widehat{u}(\xi,\tau)$ on the moved domain is then generated from (10) by

$$\widehat{u}(\xi,\tau)\,|\widehat{x}_\xi| = \widehat{c}(\xi) = \widehat{u}^0(\xi)\,|\widehat{x}^0_\xi| \tag{13}$$

where $\widehat{c}(\xi)$ is independent of τ (therefore defined by the initial data), ensuring conservation of mass. The sign of $\widehat{x}_\xi$ is determined by the sign of $\widehat{x}^0_\xi$ from (12).

The conservation method determines v (thus v_x) from (6), obtains $\widehat{x}_\xi$ from (12) (thus $\widehat{x}$ from (9)). and finds $\widehat{u}$ from (13).

2.1.2 Scale-invariance and self-similarity

If the PDE problem is scale-invariant such that the scaling transformation

$$t \to \lambda t', \quad \widehat{x} \to \lambda^\beta x', \quad \widehat{u} \to \lambda^\gamma u' \tag{14}$$

leaves the problem invariant [11], it is straightforward to verify that the solution of the initial value problem maintains the same scale invariance.

Self-similar scaling solutions are found by seeking an ansatz of the form

$$\widehat{x}(\xi,\tau) = \tau^\beta \xi, \qquad \widehat{u}(\xi,\tau) = \tau^{-\beta}\eta(\xi), \tag{15}$$

where $\eta(\xi)$ satisfies an ODE derived from (1).

From (15),

$$\widehat{v}(\xi,\tau) = \frac{\partial \widehat{x}}{\partial \tau} = \beta\tau^{\beta-1}\xi$$

so that $\widehat{v}_\xi = \beta\tau^{\beta-1}$, leading to

$$v_x = \frac{\widehat{v}_\xi}{\widehat{x}_\xi} = \frac{\beta}{\tau},$$

independent of ξ. Hence, from (12) and (13),

$$\widehat{x}_\xi = \widehat{x}^0_\xi \exp\left\{\int_{\tau^0}^{\tau} (\beta/\tau')\mathrm{d}\tau'\right\} \quad \Rightarrow \quad \frac{\widehat{x}_\xi(\xi,\tau)}{\tau^\beta} = \frac{\widehat{x}_\xi(\xi,\tau^0)}{(\tau^0)^\beta}$$

$$\widehat{u}(\xi,\tau)=\frac{\widehat{c}(\xi)}{|\widehat{x}_\xi|}\widehat{u}^0(\xi)\exp\left\{-\int_{\tau^0}^{\tau}(\beta/\tau')\mathrm{d}\tau'\right\}\quad\Rightarrow\quad\frac{\widehat{u}(\xi,\tau)}{\tau^{-\beta}}=\frac{\widehat{u}(\xi,\tau^0)}{(\tau^0)^{-\beta}},$$

so that self-similar solutions are propagated exactly in time.

Summarising, for mass conserving PDE problems of the form (1) the conservation method in the form presented here preserves the signs of $\widehat{x}_\xi$ and $\widehat{u}$, is scale-invariant and propagates self-similar scaling solutions exactly in time.

We now consider a semi-discrete-in-time approximation by applying the procedure over a time step.

2.2 A semi-discrete-in-time scheme

Discretising the time variable τ as $\tau^n=n\Delta\tau$, $(n=0,1,2,\ldots)$, where $\Delta\tau$ is the time step, define

$$\widehat{x}^n(\xi)=\widehat{x}(\xi,\tau^n),\quad\widehat{x}^n_\xi=\widehat{x}_\xi(\tau^n),\quad\widehat{v}^n_\xi=\widehat{v}_\xi(\xi,\tau^n),\quad v^n_x=\widehat{v}^n_\xi/\widehat{x}^n_\xi$$

The velocity v remains equal to (6) and $\widehat{c}(\xi)$ is as in (10).

In the conservation-based scheme of [1, 4, 15] the moving coordinate $\widehat{x}^n(\xi)$ is advanced in time from equation (7) using the first-order-accurate explicit Euler scheme. While this scheme moves the nodes correctly to first order, there is no control over the positivity of $\widehat{x}_\xi$ (thus monotonicity of $\widehat{x}(\xi)$) and the method can break down for a required time step. In this paper we use a modified time-stepping scheme, the sign-preserving exponential scheme applied to mesh intervals, which preserves the monotonicity of $\widehat{x}^n(\xi)$ and avoids node overtaking for any time step.

Approximating the integrand in (12) by its value at τ^n, an explicit first-order-accurate exponential scheme for $\widehat{x}^n_\xi$ is

$$\widehat{x}^{n+1}_\xi=\widehat{x}^n_\xi\exp\{\Delta\tau\,(v_x)\}^n \tag{16}$$

The moving coordinate $\widehat{x}^{n+1}(\xi)$ is obtained at time step τ^{n+1} using (9) in the form

$$\widehat{x}(\xi^{n+1})=\int_{\xi^0}^{\xi^{n+1}}\widehat{x}_{\xi'}\,\mathrm{d}\xi' \tag{17}$$

where ξ^0 is an anchor point necessary for uniqueness. From (16) the time evolution of $\widehat{x}_\xi$ is completely characterised by $(v_x)^n$.

Note that the scheme (16) preserves the sign of $\widehat{x}_\xi$ over a time step irrespective of $\Delta\tau$ or $(v_x)^n$.

The scheme (16) may also be obtained by applying the first-order-accurate explicit Euler scheme to (11), i.e.

$$\log \widehat{x}_\xi^{n+1} = \log \widehat{x}_\xi^n + \Delta\tau \, (v_x)^n \tag{18}$$

Equation (10) is time-independent and ensures that

$$\widehat{u}^{n+1}(\xi) \, |\widehat{x}_\xi|^{n+1} = \widehat{u}^n(\xi) \, |\widehat{x}_\xi|^n \tag{19}$$

yielding $\widehat{u}^{n+1}$. The sign of $(\widehat{x}_\xi)^{n+1}$ is determined by the sign of $(\widehat{x}_\xi)^n$ in (16).

Once $(v_x)^n$ has been found from (6), equation (16) (with (17)) determines $\widehat{x}^{n+1}(\xi)$ and equation (19) gives $\widehat{u}^{n+1}(\xi)$.

It is straightforward to verify that the scale invariance (14) is inherited from the PDE problem by the semi-discrete scheme (16) and (19). However, the argument in section 2.1.2 for the propagation of self-similar solutions no longer holds over a time step, since v_x is no longer proportional to $1/\tau$. Self-similar solutions are therefore propagated over a time step only to order $\Delta\tau$.

We now state the semi-discrete-in-time algorithm.

2.2.1 A semi-discrete-in-time algorithm

Algorithm 1

At time τ^n, given $\widehat{x}^n(\xi)$ and $\widehat{u}^n(\xi)$, the semi-discrete-in-time algorithm is as follows.

At each time step:

- obtain $v^n(x)$ from (6) and hence $(v_x)^n$
- determine $\widehat{x}_\xi^{n+1}$ from (16) and hence $\widehat{x}^{n+1}(\xi)$ from (17)
- deduce $\widehat{u}^{n+1}$ from (19)

The algorithm is sign-preserving in both $\widehat{x}_\xi$ and $\widehat{u}$ over a time step irrespective of Δt or the accuracy of v_x^n. It is conservative (as seen by integration of (19) over the domain), inherits scale-invariance from the PDE problem, and propagates self-similar solutions over a time step to order $\Delta\tau$.

We now extend the algorithm to spatial approximation.

2.3 Spatial approximation

Suppose that at time level n the domain (a^n, b^n) is discretised using mesh points

$$a^n = \widehat{x}_0^n < \widehat{x}_1^n < \ldots < \widehat{x}_N^n = b^n$$

with corresponding nodal solutions $\widehat{u}_i^n$ $(i = 0, ..., N)$, and define

$$\widehat{x}_i^n = \widehat{x}^n(\xi_i), \quad \widehat{u}_i^n = \widehat{u}^n(\xi_i), \quad \widehat{v}_i^n = \widehat{v}^n(\xi_i)$$

Also let $\delta(\cdot)_k$ denote the difference in the argument across an interval k.

At the initial time, node-based $\widehat{u}_i^0$ and $\widehat{x}_i^0$ are obtained by sampling the initial data, while at later times $\widehat{u}_i$ and $\widehat{x}_i^n$ are given by the method itself.

In preparation we approximate masses $\widehat{c}_k$ (kept constant over the time step) in each interval k between points $\widehat{x}_i^n$ and $\widehat{x}_{i-1}^n$ by the trapezium rule

$$\widehat{c}_k = \frac{1}{2}(\widehat{u}_i + \widehat{u}_{i-1})(\widehat{x}_i - \widehat{x}_{i-1})$$

An approximate velocity v_i^n at each point $\widehat{x}_i^n$ is then found from a composite trapezium rule approximation applied to (6). We approximate v_x in each interval k using finite differences as

$$(v_x)_k = \frac{(\delta v)_k}{(\delta \widehat{x})_k}$$

2.3.1 Fully discrete numerical schemes

Fully discrete forms of the schemes (16) and (19) are then

$$(\delta \widehat{x}_k)^{n+1} = (\delta \widehat{x}_k)^n \exp\{\Delta\tau\,(v_x)_k\}, \tag{20}$$

and

$$(\widetilde{u}_k)^{n+1}\,|\delta \widehat{x}_k|^{n+1} = (\widetilde{u}_k)^n\,|\delta \widehat{x}_k|^n \tag{21}$$

where $\widetilde{u}_k$ is an interval-based value of $\widehat{u}$, yet to be assigned to an end of the interval. The sign of $(\delta \widehat{x}_k)^{n+1}$ is determined by the sign of $(\delta \widehat{x}_k)^n$ in (20).

For problems in which $u(x,t)$ is specified at only one boundary $(\widehat{u}_k)^{n+1}$ is assigned to the endpoint of the interval pointing away from that boundary. For problems in which $u(x,t)$ is specified at both boundaries (21) is replaced by

$$(\widehat{u}_i)^{n+1}\,|\delta \widehat{x}_{i-1/2} + \delta \widehat{x}_{i+1/2}|^{n+1} = (\widehat{u}_i)^n\,|\delta \widehat{x}_{i-1/2} + \delta \widehat{x}_{i+1/2}|^n \tag{22}$$

for all interior nodes, yielding a node-based $\widehat{u}_i^{n+1}$ directly.

It follows from (20) and (21) or (22) that the signs of $\delta\widehat{x}_k$ and $\widehat{u}_i$ are preserved over a time step.

In order to obtain the node-based coordinate $\widehat{x}_i^{n+1}$ from the interval-based $(\delta\widehat{x})_k^{n+1}$ in (20) we use the discretised form of equation (17),

$$\widehat{x}_{i\pm1}^{n+1} = \widehat{x}_i^{n+1} \pm (\delta\widehat{x}_{i\pm1/2})^{n+1} \tag{23}$$

When applied in successive intervals away from an anchor point $\widehat{x}_0$ (needed for uniqueness), equation (23) yields all the $\widehat{x}_i^{n+1}$ exactly. Since the signs of the $\delta\widehat{x}_k$ are preserved the $\widehat{x}_i$ remain ordered in a time step.

Another way of calculating the nodes $\widehat{x}_i$ from the intervals $\delta\widehat{x}_k$ with the same outcome as the recursive step (23) (one that we shall later generalise to 2-D) is to solve the set of equations

$$\frac{\widehat{x}_{i+1} - \widehat{x}_i}{\delta\widehat{x}_{i+1/2}} = \frac{\widehat{x}_i - \widehat{x}_{i-1}}{\delta\widehat{x}_{i-1/2}} \tag{24}$$

for all interior i (both sides being equal to unity). Boundary conditions for (24) are either Neumann, imposed by setting the left or right hand side of (24) equal to unity at the boundary, or Dirichlet, using boundary node locations calculated from the known boundary velocities. Equation (24) can be rewritten as the barycentric interpolant

$$\widehat{x}_i = \frac{(\delta\widehat{x}_{i-1/2})^{-1}\widehat{x}_{i-1/2} + (\delta\widehat{x}_{i+1/2})^{-1}\widehat{x}_{i+1/2}}{(\delta\widehat{x}_{i-1/2})^{-1} + (\delta\widehat{x}_{i+1/2})^{-1}} \tag{25}$$

for all interior i, its solution ensuring continuity when one of the $\delta\widehat{x}_{i\pm1}$ is vanishingly small. The averaged position $\widehat{x}_i$ in (25) always lies between the midpoints of adjacent intervals so there can be no node overtaking.

Given $(v_x)_k^n$, equation (20) yields $(\delta\widehat{x}_i)^{n+1}$, equation 23) or (25) leads to $(\widehat{x}_i)^{n+1}$. The moved solution $(\widehat{u}_i)^{n+1}$ is determined either by equation (21) (with assignment to an adjacent node), or by equation (22) directly.

Equations (20), (21), (22), and (23) inherit the scale invariance of the original problem.

In the case of a self-similar scaling solution the argument for the propagation of a self-similar scaling solution over a time step to order $\Delta\tau$ relies on the approximation to v_x being proportional to $1/\tau$ and independent of x. In the fully discrete method, although v_x is accurately represented by the discretisation, the temporal approximation remains of order Δt.

We now state the fully discrete algorithm.

2.3.2 Fully discrete 1-D algorithm

Algorithm 2

At each time step τ^n, given $\widehat{x}_i^n$ and $\widehat{u}_i^n$,

- obtain a discrete velocity $\widehat{v}_i^n$ from a numerical approximation of (6) and deduce $(v_x)_k = (\delta\widehat{v})_k/(\delta\widehat{x})_k$
- find $(\delta\widehat{x}_k)^{n+1}$ from (20) and hence $\widehat{x}_i^{n+1}$ from (23)
- determine $(\widehat{u}_k)^{n+1}$ from (21), then

  (a) for problems in which u is given at only one boundary, assign u_k^{n+1} to $\widehat{u}_i^{n+1}$, working away from the boundary where the condition is given,

  (b) for problems in which u is given at both boundaries, determine interior nodal values $(\widehat{u}_i)^{n+1}$ from (22).

The algorithm is order-preserving in $\widehat{x}_i$ and preserves the sign of $\widehat{u}_i$. For case (a) of determining $\widehat{u}^{n+1}i$ it is conservative by summing (21) over all intervals, in the sense that

$$\left(\sum_k \widehat{u}_k \left|\delta\widehat{x}_k\right|\right)^{n+1} = \left(\sum_k \widehat{u}_k \left|\delta\widehat{x}_k\right|\right)^{n} \tag{26}$$

where $\widehat{u}_k$ is assigned to an endpoint of the k'th interval. In case (b) it is conservativeby summing (22) over all intervals, in the sense that

$$\left(\sum_i \widehat{u}_i \left|\delta\widehat{x}_{i-1/2} + \delta\widehat{x}_{i+1/2}\right|\right)^{n+1} = \left(\sum_i \widehat{u}_i \left|\delta\widehat{x}_{i-1/2} + \delta\widehat{x}_{i+1/2}\right|\right)^{n} \tag{27}$$

The algorithm is scale-invariant under the transformation (14) and propagates self-similar solutions over a time step to order $\Delta\tau$.

2.4 1-D summary

Analytically, the method is sign-preserving in $\widehat{u}$ and $\widehat{x}_\xi$ (therefore monotonicity-preserving in $\widehat{x}$) for arbitrary v. It is conservative, inherits scale invariance, and propagates self-similar solutions exactly in time.

In the semi-discrete-in-time case the algorithm of section 2.2.1 is explicit, first-order-in-time, and sign-preserving in $\widehat{u}$ and $\widehat{x}_\xi$ over a time step for arbitrary $\Delta\tau$ and v. It is conservative, inherits scale-invariance, and propagates self-similar solutions over a time step to order $\Delta\tau$.

In the fully-discrete case the algorithm of section 2.3.2 is explicit, first-order-in-time, non-tangling in $\widehat{x}_i$ and sign-preserving in $\widehat{u}_i$ over a time step for arbitrary Δt and v_x. The algorithm is conservative in the sense of (26) or (27), inherits the scale invariance of the problem and propagates a self-similar scaling solution to order $\Delta\tau$.

Computations confirm the predictions of the theory.

As with any time-stepping scheme, temporal accuracy is undermined for large $\Delta\tau$. However, even if $\Delta\tau$ is large and v_i inaccurate, in a time step the $\widehat{u}_i$ keeps the same sign and the $\widehat{x}_i$ remain ordered.

2.5 Oscillations

Although $(\delta\widehat{x})_k$ remains positive it is not necessarily monotonic and approximation errors may lead to spurious oscillations in v_x (see (18). By smoothing v_x we introduce extra numerical diffusion to counteract the oscillations at the expense of spatial accuracy.

The introduction of a Laplacian smoother,

$$\frac{1}{4}\left\{(v_x)_{k+1} + 2(v_x)_k + (v_x)_{k-1}\right\}$$

to $(v_x)_k$ suppresses sawtooth oscillations in $(v_x)_k$ and is equivalent to adding second order numerical diffusion. More than one application of the smoother may be required to render v_x monotonic and suppress the oscillations.

The positivity of the moved solution and the ordering of the mesh are unaffected by the smoothing.

3 The conservation method in 2-D

Suppose that the function $u(\mathbf{x}, t)$ is a solution of the generic PDE

$$u_t = \mathcal{L}u,$$

in a moving domain $\mathcal{R}(t)$, where $\mathcal{L}$ is a purely spatial operator, such that the total mass integral

$$\int_{\mathcal{R}(t)} u(\boldsymbol{\xi}, t)\, \mathrm{d}\mathbf{x} \tag{28}$$

is independent of time.

At any time t let the points $\mathbf{x}(t)$ of the domain $\mathcal{R}(t)$ move with a velocity $\mathbf{v}(\mathbf{x}, t)$ so as to satisfy the local Lagrangian conservation law

$$\int_{\Omega(t)} u(\boldsymbol{\chi}, t)\, \mathrm{d}\boldsymbol{\chi} = c(\Omega), \text{ independent of } t \tag{29}$$

consistent with (28) if $c(\mathcal{R}) = 1$.

Equivalently, $u(\mathbf{x}, t)$ satisfies the Eulerian form of the conservation law 29), i.e.

$$u_t + \nabla \cdot (u\mathbf{v}) = 0 \tag{30}$$

where $\mathbf{v}(\mathbf{x}, t)$ is the Eulerian velocity. Then, from equations (28) and (30) the velocity $\mathbf{v}$ satisfies the time-independent PDE

$$-\nabla \cdot (u\mathbf{v}) = \mathcal{L}u$$

in $\mathcal{R}(t)$.

For uniqueness put $\mathbf{v} = \nabla\phi$, where $\phi(\mathbf{x}, t)$ is a velocity potential satisfying the Poisson equation

$$-\nabla \cdot (u\nabla\phi) = \mathcal{L}u \tag{31}$$

For positive u equation (31) has a unique solution for $\phi(\mathbf{x}, t)$ (and hence $\mathbf{v}(\mathbf{x}, t)$) under suitable Dirichlet or Neumann boundary conditions on ϕ.

3.1 A reference space

Introduce a Lagrangian moving coordinate $\widehat{\mathbf{x}}(\boldsymbol{\xi}, \tau)$, where $\boldsymbol{\xi}$ is a fixed reference coordinate and $\tau = t$, such that

$$\frac{\partial \widehat{\mathbf{x}}}{\partial \tau} = \widehat{\mathbf{v}}(\boldsymbol{\xi}, \tau) = \mathbf{v}(\widehat{\mathbf{x}}(\boldsymbol{\xi}, \tau), \tau) \tag{32}$$

Equation (32) cannot be differentiated in the same way as i(8) in 1-D since (7) is unidirectional, but we can obtain a multidimensional equivalent of (10).

The local Lagrangian conservation law (29) is expressed in terms of $\boldsymbol{\xi}$ and τ as

$$\widehat{u}(\boldsymbol{\xi},\tau)\,|J(\widehat{\mathbf{x}},\boldsymbol{\xi})| = \widehat{c}(\boldsymbol{\xi}),\ \text{independent of } t \tag{33}$$

where $J(\widehat{\mathbf{x}},\boldsymbol{\xi})$ is the Jacobian of the transformation generated by (32), thus generalising equation (10). (Given a funcion $\widehat{u}(\boldsymbol{\xi},\tau)$ at any time τ, equation (33) leads to a Monge-Ampere equation for a function $\widehat{\mathbf{x}}(\boldsymbol{\xi},\tau)$, using a different potential function [13]).

In spite of the difficulties with generalisation of (32) and (33), if we specify that the 2-D mesh is a linear simplex we may build spatial approximations for edges and areas that allow us to take advantage of the 1-D properties.

3.2 Spatial approximation in 2-D

Let the time variable τ be discretised as $\tau^n = n\Delta\tau$, $n = 0, 1, 2, ...$, where $\Delta\tau$ is the time step, and define

$$\widehat{\mathbf{x}}_i^n = \widehat{\mathbf{x}}(\boldsymbol{\xi}_i,\tau^n), \quad \widehat{u}_i^n = \widehat{u}(\boldsymbol{\xi}_i,\tau^n), \quad \widehat{\mathbf{v}}_i^n = \widehat{\mathbf{v}}(\boldsymbol{\xi}_i,\tau^n)$$

In the conservation-based scheme of [1, 4, 15] the moving coordinate $\widehat{\mathbf{x}}_i^n$ is advanced in time from equation (32) using the first-order-accurate explicit Euler scheme. Although the nodes are moved in the correct direction to first order in time there is then no control over mesh tangling or positivity of the solution and the scheme may break down for required time steps. Instead, in this paper we use the explicit sign-preserving exponential scheme of section 2 which is sign-preserving and prevents node tangling kn 1-D for any time step. There are two ways of doing this on a 2-D simplex, either through edge lengths or triangle areas.

3.3 Spatial discretisation on a 2-D simplex

Consider a 2-D simplex consisting of non-overlapping triangles of area $A_k(\tau)$, $(k = 1, ..., K)$ and moving nodes $\widehat{\mathbf{x}}_i(\tau)$ with corresponding moved solution values $\widehat{u}_i(\tau)$, $(i = 1, ..., N)$.

We begin by deriving a finite volume approximation to the velocity potential $\Phi_i(\tau)$ at the nodes of the simplex from (31) in the case where $\mathcal{L}u = \nabla\cdot\mathbf{f}$ where $\mathbf{f}$ is a flux function.

3.3.1 Approximating the velocity potential

In a finite volume approach, integrating (31) over the boundary of a patch of triangles Π_i surrounding node i,

$$\int_{\Pi_i} \nabla \cdot (u \nabla \phi) - \mathbf{f}) \, \mathrm{d}\mathbf{x} = \oint_{\partial \Pi_i} u \frac{\partial \phi}{\partial n} \mathrm{d}S - \int_{\Pi_i} \mathbf{f} \, \mathrm{d}\mathbf{x} \tag{34}$$

Equation (34) can be approximated by

$$\sum_{e_{ij}} \left\{ \overline{u}_m \frac{(\Phi_j - \Phi_i)}{\delta \widehat{s}_{ij}} + f_n \right\} \overline{\delta s}_{ij} = 0 \tag{35}$$

where Φ_i is the value of ϕ sampled at the node $\mathbf{x}_i$; e_{ij} is the edge joining node $\mathbf{x}_i$ to node $\mathbf{x}_j$; $\overline{u}_m$ is the midpoint value of u on the edge e_{ij}; f_n is the component of $\mathbf{f}$ at $\mathbf{x}_j$ in the direction from $\mathbf{x}_i$ to $\mathbf{x}_j$ and $\overline{\delta s}_{ij}$ is the average of the lengths of the boundary edges opposite node $\mathbf{x}_i$ either side of node $\mathbf{x}_j$.

The linear system consisting of (35) for all interior nodes is square and sparse and, given boundary conditions on ϕ or $\partial\phi/\partial n$, is solved for the interor Φ values by a standard method. Boundary conditions on ϕ correspond to specifying tangential velocities and those on $\partial\phi/\partial n$ to normal velocities.

3.3.2 Approximating triangle area velocities

Having found Φ_i at the nodes, we calculate an approximation to the velocity $\mathbf{V}_k$ in a triangle as the gradient of a piecewise linear function through Φ_i values at the vertices. In a triangle k having values Φ_1, Φ_2, Φ_3 at vertices (x_1, y_1), (x_2, y_2), (x_3, y_3), the velocity is

$$\mathbf{V}_k = (\nabla \Phi)_k = \frac{1}{2A_k} \left(\begin{vmatrix} 1 & 1 & 1 \\ \Phi_1 & \Phi_2 & \Phi_3 \\ y_1 & y_2 & y_3 \end{vmatrix}, \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ \Phi_1 & \Phi_2 & \Phi_3 \end{vmatrix} \right)_k \tag{36}$$

where A_k is the area of the triangle given by

$$A_k = \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix}_k \tag{37}$$

3.3.3 Approximating the nodal velocities

The velocities $\mathbf{v}_i$ at the nodes are then obtained from the gradients $\mathbf{V}_k = (\nabla\Phi)_k$ in triangles k surrounding node i with areas A_k by the barycentric interpolant

$$\mathbf{v}_i = \frac{\sum_k A_k^{-1} V_k}{\sum_k A_k^{-1}} \tag{38}$$

where the sum is over all triangles k surrounding node i, having the property of continuity at a node when one of the A_k is vanishingly small.

Summarising the calculation of the nodal velocities, from the velocity potential Φ given by (35) we obtain triangle-based velocities $\mathbf{V}_k$ from (36) and node-based velocities $\mathbf{v}_i$ from (38).

3.3.4 Moving the nodes

In order to combat mesh tangling, instead of moving the nodes by the nodal velocities (38) directly, we update local edge lengths or triangle areas using their rates of change, enabling implementation of the sign-preserving properties of the explicit exponential time-stepping scheme seen in section 2.

Edge-length velocities

Consider an edge e of the simplex having a coordinate $\widehat{s}$ measured along the edge, and denote by $\delta(\cdot)$ the difference in the argument along the edge (so that $\delta\widehat{s}$ is the length of the edge).

At each end of the edge define v_e to be the component of the velocity $\mathbf{v}$ at an end of the edge in the direction of the edge coordinate $\widehat{s}$, and let δv_e be the difference between the two v_e's at the endpoints of the edge.

Then the explicit exponential time-stepping scheme for updating the edge length $\delta\widehat{s}$ is

$$(\delta\widehat{s})^{n+1} = (\delta\widehat{s})^n \exp\left\{\Delta\tau \left(\delta v_e/\delta\widehat{s}\right)\right\}^n, \tag{39}$$

which preserves the sign of the edge length $\delta\widehat{s}$ over a time step irrespective of $\Delta\tau$ or $\delta v_e/\delta\widehat{s}$ (and therefore that of $\widehat{s}$).

Since all the edge lengths remain of the same sign, mesh tangling is impossible.

Triangle-area velocities

Alternatively, we can use triangles instead of edges. The rate of change of the triangle area A_k given by (37) for a triangle with vertices (x_1, y_1),

(x_2, y_2), (x_3, y_3), is

$$\dot{A}_k = \left(\frac{\partial A}{\partial t}\right)_k = \frac{1}{2}\left\{ \begin{vmatrix} 1 & 1 & 1 \\ \overline{U}_1 & \overline{U}_2 & \overline{U}_3 \\ y_1 & y_2 & y_3 \end{vmatrix} + \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ \overline{V}_1 & \overline{V}_2 & \overline{V}_3 \end{vmatrix} \right\}_k \tag{40}$$

where $(\overline{U}_1, \overline{V}_1)$, $(\overline{U}_2, \overline{V}_2)$, $(\overline{U}_3, \overline{V}_3)$ are the components of the vertex velocities $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$ in the two Cartesian coordinate directions.

Discretising (40) in time by the explicit exponential scheme (see (39)), we obtain the update

$$A_k^{n+1} = A_k^n \exp\{\Delta t\, \dot{A}/A\}_k^n, \tag{41}$$

preserving the sign of A_k over a time step irrespective of Δt or $(\dot{A})_k$. Since all the triangle areas remain of the same sign, mesh tangling is impossible.

3.3.5 Finding the node locations

It remains to locate the nodes from either the edge lengths or the triangle areas at the new time level τ^{n+1}. Unlike in one dimension, there is no unique mesh that is consistent with given edge lengths or triangle areas, each requirement being overdetermined in general.

We generalise the approach of (25) to 2-D using edge-lengths or triangle-areas, as follows.

Edge-lengths

Suppose that an interior node $\mathbf{x}_i$ of the simplex is connected to J_i surrounding nodes $\mathbf{x}_j$, $(j = 1, \ldots, J_i)$, then generalisations of (24) and (25) to 2-D are

$$\sum_{j=1}^{J_i} \frac{(\widehat{\mathbf{x}}_m - \widehat{\mathbf{x}}_i)}{\delta \widehat{s}_{ij}} = 0 \qquad \text{and} \qquad \widehat{\mathbf{x}}_i = \frac{\sum_{j=1}^{J_i} (\delta \widehat{s}_{ij})^{-1} \widehat{\mathbf{x}}_m}{\sum_{j=1}^{J_i} (\delta \widehat{s}_{ij})^{-1}} \tag{42}$$

for all interior nodes $\widehat{\mathbf{x}}_i$, where the $\widehat{\mathbf{x}}_m$ are the midpoints of the edges joining $\widehat{\mathbf{x}}_i$ to $\widehat{\mathbf{x}}_j$ and $\delta \widehat{s}_{ij}$ is the (positive) length of the edge connecting node i to node j. Positivity of the weights on the right hand side of the second of (42) ensures that $\widehat{\mathbf{x}}_i$ lies in the convex hull of the midpoints $\widehat{\mathbf{x}}_m$ of the edges through $\widehat{\mathbf{x}}_i$, thus preventing tangling.

Triangle-areas

Similarly, if $k = 1, \ldots, K_i$ denote the triangles of the simplex surrounding node $\mathbf{x}_i$, having centroids $\mathbf{x}_g$, then generalisations of (24) and (25) are

$$\sum_{k=1}^{K_i} \frac{(\widehat{\mathbf{x}}_g - \widehat{\mathbf{x}}_i)}{A_k} = 0 \qquad \text{and} \qquad \widehat{\mathbf{x}}_i = \frac{\sum_{k=1}^{K_i} A_k^{-1} \widehat{\mathbf{x}}_g}{\sum_{k=1}^{K_i} A_k^{-1}} \tag{43}$$

where the A_k are the (positive) areas of the triangles surrounding node i. Positivity of the weights in the second of (43) ensures that $\widehat{\mathbf{x}}_i$ lies in the convex hull of the centroids $\widehat{\mathbf{x}}_g$ of the triangles containing $\widehat{\mathbf{x}}_i$, also preventing tangling.

Boundary nodes

Boundary nodes are treated either as Dirichlet conditions by moving the nodes with the known boundary velocities $\mathbf{v}_i$ of (38), or as Neumann conditions by applying modified forms of (42) or (43), as follows. In the case of edges, equation (42) for a boundary node b is modified to

$$\sum_{j=1}^{J_b} \frac{\widehat{\mathbf{x}}_m - \widehat{\mathbf{x}}_b}{\delta \widehat{s}_{bj}} = \sum_{j=1}^{J_b} \widetilde{\mathbf{e}}_j$$

where J_b is the number of edges emanating from node b and $\widetilde{\mathbf{e}}_j$ is the unit vector from $\mathbf{x}_b$ to $\mathbf{x}_m$. In the case of areas equation (43) for a boundary node is modified to

$$\sum_{k=1}^{K_b} \frac{\widehat{\mathbf{x}}_g - \widehat{\mathbf{x}}_b}{A_k} = \sum_{k=1}^{K_b} \widetilde{\mathbf{e}}_g$$

where K_b is the number of triangles k that contain the boundary node and $\widetilde{\mathbf{e}}_g$ is the unit vector from $\widehat{\mathbf{x}}_b$ to the centroid $\widehat{\mathbf{x}}_g$ of triangle k.

3.3.6 Solving for the nodal positions

Because of the overlap of the patches each of the square sets of equations (42) or (43) leads to a sparse nonlinear matrix system for the $\widehat{\mathbf{x}}_i$, which can be solved by standard methods, either direct or iterative.

An iteration for the second of (42) in the case of edges (where p is the iteration index) is

$$\widehat{\mathbf{x}}_i^{(p+1)} = \widehat{\mathbf{x}}_i^{(p)} + \left(\frac{\sum_{j=1}^{J_i} (\delta \widehat{s}_{ij})^{-1} (\widehat{\mathbf{x}}_m - \widehat{\mathbf{x}}_i)}{\sum_{j=1}^{J_i} (\delta \widehat{s}_{ij})^{-1}} \right)^{(p)} = \left(\frac{\sum_{j=1}^{J_i} (\delta \widehat{s}_{ij})^{-1} \widehat{\mathbf{x}}_m}{\sum_{j=1}^{J_i} (\delta \widehat{s}_{ij})^{-1}} \right)^{(p)}, \tag{44}$$

while in the case of areas the second of (43) has the iteration

$$\widehat{\mathbf{x}}_i^{(p+1)} = \widehat{\mathbf{x}}_i^{(p)} + \left(\frac{\sum_{k=1}^{K_i} A_k^{-1} (\widehat{\mathbf{x}}_g - \widehat{\mathbf{x}}_i)}{\sum_{k=1}^{K_i} A_k^{-1}} \right)^p = \left(\frac{\sum_{k=1}^{K_i} A_k^{-1} \widehat{\mathbf{x}}_g}{\sum_{k=1}^{K_i} A_k^{-1}} \right)^{(p)} \tag{45}$$

The solutions $\widehat{\mathbf{x}}_i$ of (42) or (43) (or their iterates $\widehat{\mathbf{x}}_i^{(p+1)}$ of (44) or (45)), together with the boundary node equations, lie in a non-overlapping subpatch (either the convex hull of midpoints of edges or the convex hull of centroids of surrounding triangles) of each patch, thus avoiding mesh tangling.

3.4 Finding the moved solution

Having obtained the locations of the nodes at time τ^{n+1}, the moved solution $\widehat{u}^{n+1}$ at these nodes is found from approximations to the Lagrangian conservation law (33), as follows.

From edges:

An approximate form of the Lagrangian conservation law (29) along an edge e is

$$(\widehat{u}_m \, \delta\widehat{s}_e)^{n+1} = \widehat{c}_e = (\widehat{u}_m \, \delta\widehat{s}_e)^n \tag{46}$$

where $\mathbf{x}_m$ is the midpoint of the edge and $\widehat{c}_e$ is time-independent, leading to

$$(\widehat{u}_m)^{n+1} = \frac{(\widehat{u}_m \, \delta\widehat{s}_e)^n}{(\delta\widehat{s}_e)^{n+1}}$$

The moved solution $\widehat{u}_i^{n+1}$ at the node $\mathbf{x}_i$ is then the barycentric interpolant

$$\widehat{u}_i^{n+1} = \frac{\Sigma_{j=1}^{J_i} (\delta s_j^{n+1})^{-1} \, \widehat{u}_m^{n+1}}{\Sigma_{j=1}^{J_i} (\delta s_j^{n+1})^{-1}} \tag{47}$$

of the $\widehat{u}_m^{n+1}$ over those edges containing node $\mathbf{x}_i$, preserving the sign of $\widehat{u}_i^{n+1}$ and ensuring continuity of the δu_i^{n+1} when one of the δs_j^{n+1} is vanishingly small.

From areas:

An approximate form of the Lagrangian conservation law (29) in a triangle k is

$$(\widehat{u}_g A_k)^{n+1} = \widehat{C}_k = (\widehat{u}_g A_k)^n \tag{48}$$

where $\widehat{C}_k$ is time-independent and $\widehat{u}_g$ is the value of $\widehat{u}$ at the centroid of triangle k, leading to

$$(\widehat{u}_g)^{n+1} = \frac{(\widehat{u}_g\, A_k)^n}{(A_k)^{n+1}}$$

The moved solution $\widehat{u}_i^{n+1}$ at the node i is then the barycentric interpolant

$$\widehat{u}_i^{n+1} = \frac{\Sigma_{k=1}^{K_i} A_k^{n+1})^{-1}\, \widehat{u}_g^{n+1}}{\Sigma_{k=1}^{K_i} (A_k^{n+1})^{-1}} \tag{49}$$

of the $\widehat{u}_g^{n+1}$ over triangles containing node $\mathbf{x}_i$, $(cf.\ (24))$, preserving the sign of $\widehat{u}_i^{n+1}$ and ensuring continuity of the δu_i^{n+1} when one of the A_k^{n+1} is vanishingly small.

3.5 Fully discrete 2-D algorithms

The 2-D algorithms for the solution of mass conserving PDE problems on the simplex is as follows.

Algorithm 3

At each time step:

1. Determine an approximate velocity potential at the nodes from (35) and the consequent nodal velocities from (36) and (38).

2. advance the edge-lengths δs or triangle-areas A in time from (39) or (41), and construct the nodes from (42) or (43),

3. evaluate the $\widehat{c}_e$ or $\widehat{C}_k$ from the right hand side of (46) or (48) and retrieve the solution on the moved mesh using (47) or (49).

Note that, due to the calculation of the nodes, the $\widehat{c}_e$ or $\widehat{C}_k$, although held constant over a time step, are recalculated at the beginning if each step.

The algorithm is non-tangling in $\widehat{\mathbf{x}}_i$ and positivity-preserving in $\widehat{u}_i$, irrespective of Δt or the accuracy of $\nabla^2\Phi$. It is also conservative in the sense that, for edge lengths from (46),

$$\sum_j \widehat{u}_m\, \delta s_e$$

over all edges is over a time step, or for triangle areas, from (48),

$$\sum_k u_g \, A_k$$

over all areas is constant over a time step.

The sign of the moved solution is preserved and the mesh remains untangled in a time step under second-order smoothing.

3.6 2-D summary

The difficulties in discretising the Lagrangian conservation law or in generating an untangled mesh when applying the velocity-based conservation method in 2-D are avoided by applying a sign-preserving time stepping scheme to edge lengths or triangle areas of a 2-D simplex to calculate positive updates, followed by their transfer to nodes by a projection, preserving the integrity of the simplex. The algorithm is explicit, first-order-in-time, sign preserving in $\widehat{u}_i$ and non-tangling in $\widehat{\mathbf{x}}_i$ for arbitrary Δt and Φ.

Computations confirm the predictions of the theory.

3.7 Oscillations

Although there is no mesh tangling for any time step the edge-lengths or triangle-areas may not be sufficiently smooth spatially and numerical approximations may lead to spurious oscillations in the moved solution $\widehat{u}_i$ through oscillations in $\nabla^2_{\mathbf{x}}\phi$. By smoothing $\nabla^2_{\mathbf{x}}\phi$ until it is monotonic, numerical diffusion can be introduced to counteract the oscillations. More than one application of the smoother may be required to suppress the oscillations.

4 Conclusions

In this paper we have described a variant of the velocity-based conservation method for scalar mass-conserving PDEs in one and two dimensions which, unlike the standard approach, ensures that the mesh remains untangled and the sign of the moved solution is preserved for any time step.

The essence of the paper is the replacement of the explicit Euler time stepping scheme used in most implementations of the method by an explicit

exponential time-stepping scheme, preserving the signs of moving intervals in 1-D or moving edge-lengths/triangle-areas on a simplex of triangles in 2-D. Transfer to nodes is by an additional step which preserves the integrity of the mesh.

We began in 1-D with an analysis of the conservation method, deriving ihe procedure from a mapping between fixed and moving domains and exhibiting the properties of exact propagation of scaling invariance and self-similarity. We then introduced discretisation in time, superseding the standard explicit Euler scheme by an explicit exponential time stepping scheme that preserves the monotonicity of the mesh and the sign of the solution, as well as keeping scale invariance and propagating self-similarity to first order in the times step. Lastly we carried out the spatial approximation using finite differences, applying the explicit exponential time stepping scheme to intervals rather than nodes, ensuring preservation of node-ordering and the sign of the solution. The nodes were retrieved uniquely from the intervals, either by a simple recurrence or by the solution of a second order equation. The fully discrete 1-D algorithm was stated in section 2.3.2.

In section 3 we reviewed the conservation method in 2-D and highlighted the limitations in the numerical implementation of the method in higher dimensions. We then specialised the mesh to a 2-D simplex and applied the 1-D properties to the evolution of edge lengths and triangle areas using a projection to map to the nodes without disturbing the integrity if the mesh, in this way generating a fully-discrete explicit method with a non-tangling mesh and sign-preserving solution for any time step. The algorithm was stated in section 3.5.

Further work includes the generalisation of the approach to non mass-conserving PDEs with given flux boundary conditions, to finite element approximation, and to three dimensions.

References

[1] M.J.Baines, M.E.Hubbard, and P.K.Jimack, *A moving-mesh finite element algorithm for the adaptive solution of time-dependent partial differential equations with moving boundaries*, Appl. Numer. Math., 54, pp. 450-469, 2005.

[2] M.J.Baines, M.E.Hubbard, P.K.Jimack and A.C.Jones, *Scale-invariant moving finite elements for nonlinear partial differential equations in two dimensions*, Appl. Numer. Math., 56, pp. 230-252, 2006.

[3] M.J.Baines, M.E.Hubbard, P.K.Jimack and R.Mahmood, *A moving-mesh finite element method and its application to the numerical solution of phase-change problems*, Commun. Comput. Phys., 6, pp. 595-624, 2009.

[4] M.J.Baines, M.E.Hubbard, and P.K.Jimack, *Velocity-based moving mesh methods for nonlinear partial differential equations*, Commun. Comput. Phys., 10, pp. 509-576, 2011.

[5] M.J.Baines, *Explicit time stepping for moving meshes*, J. Math Study, 48, pp. 93-105 (2015).

[6] M.J.Baines, *The numerical propagation of scaling symmetries of scale-invariant partial differential equations: the S-property for mass-conserving problems* Mathematics Report 2/2016, Department of Mathematics and Statistics, University of Reading, UK (2016).

[7] M.J.Baines, *A positivity- and monotonicity-preserving moving-mesh finite difference scheme based on local conservation*, Mathematics Report 1/2017, Department of Mathematics and Statistics, University of Reading, UK (2017).

[8] M.J.Baines and N.Sarahs, *A moving-mesh finite difference scheme that preserves scaling symmetry for a class of nonlinear diffusion problems*. J. Comp. Appl. Maths, 340, 380-389, ISSN 0377-0427, doi.org/10.1016/j.cam.2018.02.040 (2018).

[9] N.Bird, *High Order Nonlinear Diffusion*, PhD thesis, Department of Mathematics and Statistics, University of Reading, UK, (2015).

[10] B.Bonan, M.J.Baines, N.K.Nichols, and D.Partridge, *A moving-point approach to model shallow ice sheets: a study case with radially symmetrical ice sheets* The Cryosphere, 10, 1-14, doi: 10.5194/tc-10-1-2016, (2016).

[11] C.J. Budd and M.D. Piggott, *Geometric integration and its applications*, Found. Comput. Math., Handbook of Numerical Analysis XI, ed. P.G. Ciarlet and F. Cucker, Elsevier, pp. 35-139, 2003.

[12] C.J. Budd, W. Huang, and R.D. Russell, *Adaptivity with moving grids*, Acta Numerica, 18, pp. 111-241 (2009).

[13] C.J. Budd and J.F. Williams, *Moving mesh generation using the parabolic Monge–Ampère equation*, SIAM J. Sci. Comput., 31 (5), pp. 3438-3465 (2009).

[14] T.E.Lee, *Modelling time-dependent partial differential equations using a moving mesh approach based on conservation*, PhD thesis, University of Reading, UK, (2011).

[15] T.E. Lee, M.J. Baines, and S. Langdon, *A finite difference moving mesh method based on conservation for moving boundary problems*, J. Comp. Appl. Math, 288, pp.1-17 (2015).

[16] A.V. Lukyanov, M.M. Sushchikh, M.J. Baines, and T.G. Theofanous, *Superfast Nonlinear Diffusion: Capillary Transport in Particulate Porous Media*, Phys. Rev. Letters, 109, 214501 (2012).

[17] A.R Watkins, *A Moving Mesh Finite Element method and its Application to Population Dynamics*. PhD thesis, University of Reading, UK (2017).