School of Mathematical, Physical
and Computational Sciences

University of Reading

Department of Mathematics and Statistics

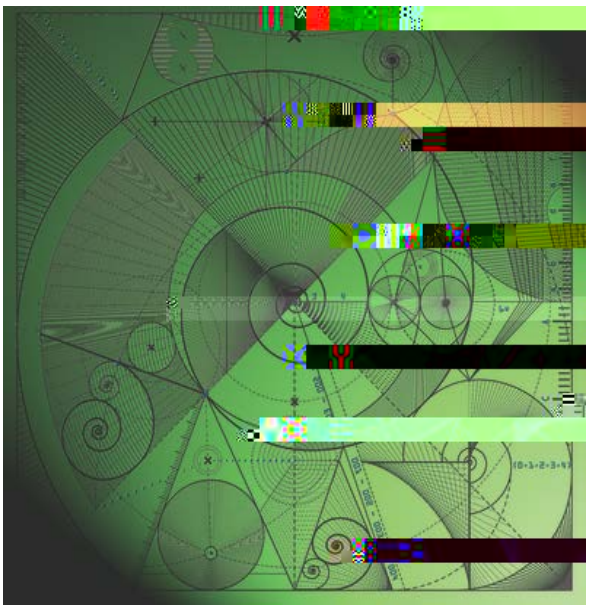
Preprint MPCS-2019-08

14 October 2019

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

by

M.J. Baines



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

$$u_t = Lu; \tag{1}$$

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

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

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

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

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

In an Eulerian form equivalent to the conservation law (3), the function $u(x;t)$ 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

$$\frac{d}{dx}(uv) = Lu \tag{5}$$

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

$$(u(x;t)v(x;t)\big|_{a(t)}^{x(t)} = \int_{a(t)}^{x(t)} L(u)\,d\xi$$

so that

$$v(x(t);t) = v(a(t);t) \quad \frac{1}{u(x(t);t)} \int_{a(t)}^{x(t)} L(u)\,d\xi \tag{6}$$

2.1 A reference space

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

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

where

$$v(\xi, \tau) = v(x(\xi, \tau), \tau)$$

By the chain rule

$$v(\xi, \tau) = v(x(\xi, \tau), \tau)$$

We consider two equations in the reference space.

1. Differentiating (7) with respect to ξ,

$$\frac{\partial x_\xi}{\partial \tau} = v_\xi \tag{8}$$

from which the moving coordinate $x(\xi, t)$ is retrieved from x_ξ using

$$x(\xi, \tau) = x_0 + \int_{\tau_0}^{\tau} x_{\xi 0}\, d\tau' \tag{9}$$

where $x(\xi, \tau) = x_0$ at $\tau = \tau_0$.

2. The Lagrangian conservation form (3) transforms to

$$u(\xi, \tau)\, |x_\xi| = b(\xi), \text{ independent of } \tau; \tag{10}$$

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

2.1.1 An initial value problem

Substitution of $u(\xi, \tau)$ from (10) into (8) yields an ODE for x_ξ. Given initial conditions on x_ξ and u (and hence $b(\xi)$) at $\tau = \tau_0$ equation (8) is an initial value problem for x_ξ possessing a unique solution under the conditions of Picard's Theorem.

From equation (8),

$$\frac{\partial \log x_\xi}{\partial \tau} = \frac{v_\xi}{x_\xi} = v_x \tag{11}$$

since $v = \dot{x} = v_x$ at time $t = \quad$. Integrating (11) from ^{0} to $\quad$, equation (11) has the formal solution

$$\dot{x} = \dot{x}^0 \exp \int_0^{\quad} v_x \, d \, ^0 \tag{12}$$

where $x = x^0$ at $\quad = \, ^0$. The solution $u(\; ; \;)$ on the moved domain is then generated from (10) by

$$u(\; ; \;) j\dot{x} j = b(\;) = u^0(\;) j\dot{x}^0 j \tag{13}$$

where $b(\;)$ is independent of $\quad$ (therefore de ned by the initial data), ensuring conservation of mass. The sign of $\dot{x}$ is determined by the sign of $\dot{x}^0$ from (12).

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

2.1.2 Scale-invariance and self-similarity

If the PDE problem is scale-invariant such that the scal -0.8F1 9.9626 Tf 182a9626 Tf 182a962

$$u(x,t) = \frac{b(x)}{|x|}u^0(x)\exp\left(\int_0^t \lambda(s)\,ds\right) \qquad \frac{u(x,t)}{u(x,t^0)} = \frac{u(x,t^0)}{u(x,0)};$$

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 u and 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

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

$$\log(\rho)^{n+1} = \log(\rho)^n + \qquad (v_x)^n \tag{18}$$

Equation (10) is time-independent and ensures that

$$u^{n+1}(\ )|\rho|^{n+1} = u^n(\ )|\rho|^n \tag{19}$$

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

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

2.3 Spatial approximation

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

$$a^n = x_0^n < x_1^n < \cdots < x_N^n = b^n$$

with corresponding nodal solutions u_i^n $(i = 0, \ldots, N)$, and define

$$x_i^n = x^n(\xi_i); \quad u_i^n = u^n(\xi_i); \quad v_i^n = v^n(\xi_i)$$

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

At the initial time, node-based u_i^0 and x_i^0 are obtained by sampling the initial data, while at later times u_i and x_i^n are given by the method itself.

In preparation we approximate masses α_k (kept constant over the time step) in each interval k between points x_i^n and x_{i-1}^n by the trapezium rule

$$\alpha_k = \frac{1}{2}(u_i + u_{i-1})(x_i - x_{i-1})$$

An approximate velocity v_i^n at each point 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

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

2.3.1 Fully discrete numerical schemes

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

$$(\Delta x_k)^{n+1} = (\Delta x_k)^n \exp\{(v_x)_k\}, \tag{20}$$

and

$$(\alpha_k)^{n+1} \, |\Delta x_k|^{n+1} = (\alpha_k)^n \, |\Delta x_k|^n \tag{21}$$

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

For problems in which $u(x,t)$ is specified at only one boundary $(\alpha_k)^{n+1}$ 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

$$(u_i)^{n+1} \, |\Delta x_{i-1/2}| + |\Delta x_{i+1/2}|^{n+1} = (u_i)^n \, |\Delta x_{i-1/2}| + |\Delta x_{i+1/2}|^n \tag{22}$$

for all interior nodes, yielding a node-based u_i^{n+1} directly.

It follows from (20) and (21) or (22) that the signs of Δx_k and u_i are preserved over a time step.

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

$$x_{i\ 1}^{n+1} = x_i^{n+1} \quad (\Delta x_{i\ 1=2})^{n+1} \tag{23}$$

When applied in successive intervals away from an anchor point x_0 (needed for uniqueness), equation (23) yields all the x_i^{n+1} exactly. Since the signs of the Δx_k are preserved the x_i

2.3.2 Fully discrete 1-D algorithm

Algorithm 2

At each time step n, given x_i^n and u_i^n,

obtain a discrete velocity v_i^n from a numerical approximation of (6) and deduce $(v_x)_k = (\partial v/\partial x)_k$

and $(\partial x_k)^{n+1}$ from (20) and hence x_i^{n+1} from (23)

determine $(u_k)^{n+1}$ from (21), then

(a) for problems in which u is given at only one boundary, assign u_k^{n+1} to 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 $(u_i)^{n+1}$ from (22).

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

$$\sum_k \left(u_k \, |\partial x_k| \right)^{n+1} = \sum_k \left(u_k \, |\partial x_k| \right)^{n} \tag{26}$$

where u_k is assigned to an endpoint of the k'th interval. In case (b) it is conservative by summing (22) over all intervals, in the sense that

$$\sum_i \left(u_i \, |x_{i\,1/2} + x_{i+1/2}| \right)^{n+1} = \sum_i \left(u_i \, |x_{i\,1/2} + x_{i+1/2}| \right)^{n} \tag{27}$$

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

2.4 1-D summary

Analytically, the method is sign-preserving in u and x (therefore monotonicity-preserving in 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, rst-order-in-time, and sign-preserving into u and x_0 over a time step for arbitrary and v. It is conservative, inherits scale-invariance, and propagates self-similar solutions over a time step to order .

In the fully-discrete case the algorithm of section 2.3.2 is explicit, rst-order-in-time, non-tangling in x_i and sign-preserving into u_i over a time stepx

3 The conservation method in 2-D

Suppose that the function $u(x;t)$ is a solution of the generic PDE

$$u_t = Lu;$$

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

$$\int_{R(t)} u(\ ;t)\,dx \qquad (28)$$

Equation (32) cannot be differentiated in the same way as (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 ρ and α as

$$\rho(\alpha, \tau)\, |J(\mathbf{x}, \tau)| = \rho_0(\alpha), \text{ independent of } t \tag{33}$$

where $J(\mathbf{x}, \tau)$ is the Jacobian of the transformation generated by (32), thus generalising equation (10). (Given a function $\rho(\alpha, \tau)$ at any time τ, equation (33) leads to a Monge-Ampere equation for a function $\mathbf{x}(\alpha, \tau)$, 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 t be discretised as $t^n = n\Delta t$, $n = 0, 1, 2, \ldots$, where Δt is the time step, and define

$$\mathbf{x}_i^n = \mathbf{x}(\alpha_i, t^n); \quad \rho_i^n = \rho(\alpha_i, t^n); \quad v_i^n = v(\alpha_i, t^n)$$

In the conservation-based scheme of [1, 4, 15] the moving coordinate $\mathbf{x}_i^n$ 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 in 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 $A_k(t)$, $(k = 1, \ldots, K)$ and moving nodes $\mathbf{x}_i(t)$ with corresponding moved solution values $u_i(t)$, $(i = 1, \ldots, N)$.

We begin by deriving a finite volume approximation to the velocity potential $\phi_i(t)$ at the nodes of the simplex from (31) in the case where $Lu = \nabla \cdot f$ where 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 Ω_i surrounding node i,

$$\int_{\Omega_i} \nabla \cdot (u \nabla \phi) \, dx = \int_{\partial \Omega_i} u \frac{\partial \phi}{\partial n} dS = \int_{\Omega_i} f \, dx \tag{34}$$

Equation (34) can be approximated by

$$\sum_{e_{ij}} \left(u_m \frac{(\phi_j - \phi_i)}{s_{ij}} + f_n \right) \bar{s}_{ij} = 0 \tag{35}$$

where ϕ_i is the value of ϕ sampled at the node x_i; e_{ij} is the edge joining node x_i to node x_j; u_m is the midpoint value of u on the edge e_{ij}; f_n is the component of f at x_j in the direction from x_i to x_j and $\bar{s}_{ij}$ is the average of the lengths of the boundary edges opposite node x_i on either side of node x_j.

The linear system consisting of (35) for all interior nodes is square and sparse and, given boundary conditions on Γ_1 or $\partial\phi/\partial n$, is solved for the interor ϕ values by a standard method. Boundary conditions on Γ_1 correspond to specifying tangential velocities and those on Γ_2 to normal velocities.

3.3.2 Approximating triangle area velocities

Having found ϕ_i at the nodes, we calculate an approximation to the velocity 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

$$V_k = (\nabla \phi)_k = \frac{1}{2A_k} \begin{pmatrix} 1 & 1 & 1 \\ y_1 & y_2 & y_3 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix}_k \tag{36}$$

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 v_i at the nodes are then obtained from the gradients $V_k = (r\ )_k$ in triangles k surrounding node i with areas A_k by the barycentric interpolant

$$v_i = \frac{\sum_k A_k^{-1} V_k}{\sum_k A_k^{-1}} \tag{38}$$

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 given by (35) we obtain triangle-based velocities V_k from (36) and node-based velocities 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 s measured along the edge, and denote by $(\ )$ the difference in the argument along the edge (so that s is the length of the edge).

At each end of the edge define v_e to be the component of the velocity v at an end of the edge in the direction of the edge coordinate s, and let v_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 s is

$$(s)^{n+1} = (s)^n \exp\{$$

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

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

where $(\overline{U}_1; \overline{V}_1)$, $(\overline{U}_2; \overline{V}_2)$, $(\overline{U}_3; \overline{V}_3)$ are the components of the v7(ts8Td [(051) -236 11.d [8(elo36.8
< tSP]TJ/F198235coen0.05/FT-G 6EnTd [(J7 11.959067(32663

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

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

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 $\mathbf{x}_i$ lies in the convex hull of the centroids $\mathbf{x}_g$ of the triangles containing $\mathbf{x}_i$, also preventing tangling.

Boundary nodes

Boundary nodes are treated either as Dirichlet conditions by moving the nodes with the known boundary velocities 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

$$\sum_{j=1}^{J_b} \frac{\mathbf{x}_m - \mathbf{x}_b}{s_{bj}} = \sum_{j=1}^{J_b} \mathbf{e}_j$$

where J_b is the number of edges emanating from node b and $\mathbf{e}_j$ is the unit vector from $\mathbf{x}_b$ to $\mathbf{x}_m$. In the case of areas equation

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

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

The solutions $\mathbf{x}_i$ of (42) or (43) (or their iterates $\mathbf{x}_i^{(p+1)}$ 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 $^{n+1}$, the moved solution 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

$$(u_m s_e)^{n+1} = c_e = (u_m s_e)^n \qquad (46)$$

where x_m is the midpoint of the edge and c_e is time-independent, leading to

$$(u_m)^{n+1} = \frac{(u_m s_e)^n}{(s_e)^{n+1}}$$

The moved solution u_i^{n+1} at the node x_i is then the barycentric interpolant

$$u_i^{n+1} = \frac{\sum_{j=1}^{J_i} (s_j^{n+1})^{-1} u_m^{n+1}}{\sum_{j=1}^{J_i} (s_j^{n+1})^{-1}} \qquad (47)$$

of the u_m^{n+1} over those edges containing node i, preserving the sign of u_i^{n+1} and ensuring continuity of the u_i^{n+1} when one of the s_j^{n+1} is vanishingly small.

From areas:

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

$$(u_g A_k)^{n+1} = C_k = (u_g A_k)^n \qquad (48)$$

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

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

The moved solution u_i^{n+1} at the node i is then the barycentric interpolant

$$u_i^{n+1} = \frac{\sum_{k=1}^{K_i} A_k^{n+1})^{-1} \dot{u}_g^{n+1}}{\sum_{k=1}^{K_i} (A_k^{n+1})^{-1}} \tag{49}$$

of the $\dot{u}_g^{n+1}$ over triangles containing node x_i, (cf. (24)), preserving the sign of $\dot{u}_i^{n+1}$ and ensuring continuity of the u_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 s or triangle-areas A in time from (39) or (41), and construct the nodes from (42) or (43),
3. evaluate the $\dot{u}_e$ or $\dot{C}_k$ 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 $\dot{u}_e$ or $\dot{C}_k$, although held constant over a time step, are recalculated at the beginning if each step.

The algorithm is non-tangling in x_i and positivity-preserving in u_i, irrespective of t or the accuracy of r^2. It is also conservative in the sense that, for edge lengths from (46),

$$\sum_j \dot{u}_m s_e$$

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 u_i and non-tangling in x_i for arbitrary t and Δ.

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 u_i through oscillations in r_x^2. By smoothing r_x^2 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 xed 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 rst order in the times step. Lastly we carried out the spatial approximation using nite di erences, 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-

[2] M.J.Baines, M.E.Hubbard, P.K.Jimack and A.C.Jones , Scale-invariant moving nite elements for nonlinear partial di erential 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 nite 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 di erential 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 di erential 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 nite di erence 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 nite di erence scheme that preserves scaling symmetry for a class of nonlinear di usion 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 Di usion , 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).