School of Mathematical, Physical
and Computational Sciences

# Department of Mathematics and Statistics

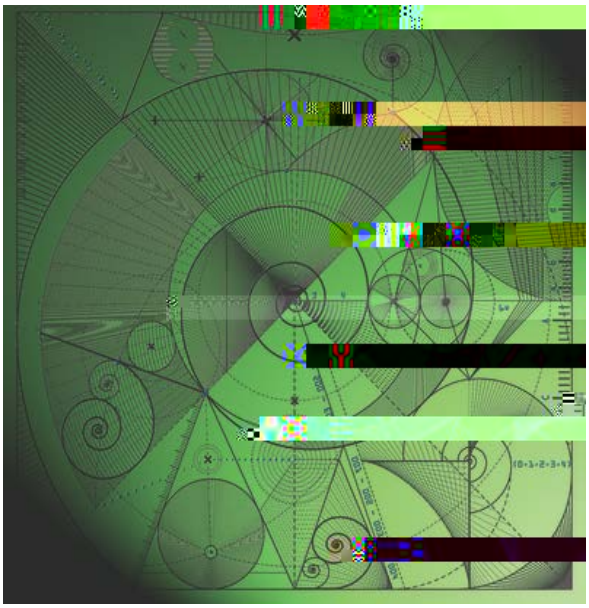
Preprint MPCS-2019-09

14 October 2019

# Non-tangling moving-mesh methods for PDE problems in one and two dimensions II: problems with prescribed boundary fluxes

by

M.J. Baines



# Non-tangling moving-mesh methods for PDE problems in one and two dimensions II: problems with prescribed boundary uxes

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 PDE problems with prescribed boundary uxes 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 di erential equations (PDEs) are potentially more powerful than  xed 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, 16] a velocity  eld is determined from a local Eulerian conservation law which is used to deform the domain,  nding the moved solution by local Lagrangian conservation. Applications can be found in [1, 2, 3, 4, 14, 17, 15, 16, 9, 5, 6, 10, 7, 18]. The method inherits the scale-invariance of the PDE problem, handles  ux-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 a separate step.

The paper is organised as follows. In section 2 the relative conservation method is reviewed in one dimension, a semi-discrete scheme analysed, and a fully discrete version described that uses an 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 2.4.3.

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 relative Lagrangian conservation. The algorithms are given in section 3.3.5.

Conclusions are drawn in section 4.

## 2 The relative conservation method in 1-D

Suppose that the functionu(x; t) satis es the generic PDE

$$u_t =$$

where v

2. The Lagrangian conservation form (6) transforms to

$$\frac{1}{(\ )}\rho(\ ;\ )\,|x_\ |=\rho_0(\ ),\ \text{independent of} \tag{14}$$

for all , where $\rho(\ ;\ )=\rho(x(\ ;\ );\ )$ and $x_\ $ is the Jacobian of $x$ with respect to . The sign of $x_\ $ is determined by (12).

## 2.2 An initial value problem

Substitution of $\rho(\ ;\ )$ from (14) into (12) yields an ODE system for $x_\ $ and $(\ )$. Given initial conditions on $x_\ $ and $\rho$ at $=\ ^0$ (and hence $\rho_0(\ )$ together with (5), equations (12) and () then constitute an initial value problem for the two unknowns $x_\ $ and $(\ )$ possessing a unique solution under the conditions of Picard's Theorem.

From equation (12),

$$\frac{@\log x_\ }{@}=\frac{v_\ }{x_\ }=v_x \tag{15}$$

since $v_\ =x_\ =v_x$. Integrating (15) from $^0$ to , equation (15) has the formal solution

$$x_\ =x_\ ^0\exp\int_{\ ^0}^{\ } v_x\,d\ ^0 \tag{16}$$

where $x_\ =x_\ ^0$ at $=\ ^0$. The solution $\rho(\ ;\ )$ on the moved domain is then generated from (14) by

$$\frac{1}{(\ )}\rho(\ ;\ )\,|x_\ |=\rho_0(\ )=\frac{1}{(\ ^0)}\rho^0(\ )\,|x_\ ^0| \tag{17}$$

where $\rho_0(\ )$ is independent of (therefore de ned by the initial data), ensuring conservation of mass. The sign of $x_\ $ is determined

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

$$\psi(x,t) = \qquad ; \qquad u(x,t) = \qquad (\ ); \qquad (\ ) = \ ^{+}\ \# \qquad (19)$$

where $(\ )$ satis es an ODE derived from (1) and $\#$ is constant.

From (19),

$$v(x,t) = \frac{\partial \psi}{\partial} = \qquad ^{1}$$

so that $v = \qquad ^{1}$, leading to $v_x = v = \psi = = ;$ independent of .

Hence, from (16) and (17),

$$\psi = \psi^{0} \exp \int_{0}^{} ( = \ ^{0}) d\ ^{0} \qquad ) \qquad \frac{\psi(x,t)}{} = \frac{\psi(x,\ ^{0})}{(\ ^{0})}$$

$$u(x,t) = \frac{b(\quad)}{|x|} \quad (\ )$$

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 $x^n(\ )$ and avoids node overtaking for any time step.

Approximating the integrand in (16) by its value at $^n$, an explicit first-order-accurate exponential scheme for $x^n$ is

$$x^{n+1} = x^n \exp\{ \quad (v_x)\}^n \tag{20}$$

The moving coordinate $x^{n+1}(\ )$ is obtained at time step $^{n+1}$ using (13) in the form

$$x(\ ^{n+1}) = \int_0^{\ ^{n+1}} x_0 \, d\ ^0 \tag{21}$$

where $^0$ is an anchor point necessary for uniqueness. From (20) the time evolution of $x$ is completely characterised by $(v_x)^n$.

Note that the scheme (20) preserves the sign of over a time step irrespective of or $(v_x)^n$.

The scheme (20) may also be obtained by applying the first-order-accurate explicit Euler scheme to (15), i.e.

$$\log x^{n+1} = \log x^n + \quad (v_x)^n \tag{22}$$

The total mass $^{n+1}$ is found from the corresponding explicit exponential scheme

$$^{n+1} = \ ^n \exp\{ \quad \_^n = \ ^n\}; \tag{23}$$

preserving its sign.

Equation (14) is time-independent and ensures that

$$\frac{1}{\ ^{n+1}} u^{n+1}(\ ) \, |x_\ |^{n+1} = \frac{1}{\ ^n} u^n(\ ) \, |x_\ |^n \tag{24}$$

yielding $u^{n+1}$. The sign of $(x_\ )^{n+1}$ is determined by the sign of $x(\ )^n$ in (20).

Once $(v_x)^n$ has been found from (10), equation (20) (with (21)) determines $x^{n+1}(\ )$, equation (23) yields $^{n+1}$ and equation (24) gives $u^{n+1}(\ )$.

It is straightforward to verify that the scale invariance (18) is inherited from the PDE problem by the semi-discrete schemes (20) and (24). However, the argument in section 2.2.1 for the propagation of self-similar solutions no longer holds over a time step, since $v_x$ is no longer proportional to $1=$ . Self-similar solutions are therefore propagated over a time step only to order .

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

### 2.3.1 A semi-discrete-in-time algorithm

Algorithm 1

At time $^n$, given $x^n(\ )$ and $u^n(\ )$, the semi-discrete algorithm is as follows.

At each time step

Algorithm 1

obtain $v^n(x)$ from (10) and hence $(v_x)^n$

calculate $\_$ from (5) and hence $^{n+1}$ from (23)

determine $x^{n+1}$ from (20) and hence $x^{n+1}(\ )$ from (21)

deduce $u^{n+1}$ from (24)

The algorithm is sign-preserving in $x^n$, $u^n$ and over a time step irrespective of $t$ or the accuracy of $v_x^n$. It is conservative (a seen) by integration of (24) over the domain, inherits scale-invariance from the PDE problem, and propagates self-similar solutions over a time step to order .

We now extend the algorithm to spatial approximation.

## 2.4 Spatial approximation

Suppose that at time leveln 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 $u_i^n$ $(i = 0$ nd helmesh

Approximate relative masses $b_k$ (kept constant over the time step) are found in each interval $k$ between points $x_i^n$ and $x_{i-1}^n$ from the trapezium rule

$$b_k = \frac{1}{2}(\frac{1}{2}($$

When applied in successive intervals away from an anchor point $x_0$ (needed for uniqueness), equation (28) yields all the $x_i^{n+1}$ exactly. Since the signs of the $\Delta x_k$ are preserved the $x_i$ remain ordered in a time step.

Another way of calculating the nodes $x_i$ from the intervals $\Delta x_k$ with the same outcome as the recursive step (28) (one that we shall later generalise to 2-D) is to solve the set of equations

$$\frac{\Delta x_{i+1} - \Delta x_i}{\Delta x_{i+1/2}} = \frac{\Delta x_i - \Delta x_{i-1}}{\Delta x_{i-1/2}} \tag{29}$$

for all interior $i$ (both sides equal to unity). Boundary conditions for (29) are either Neumann, imposed by setting the left or right hand side of (29) equal to unity, or Dirichlet, using node locations calculated from known boundary velocities. Equation (29) can be rewritten as the barycentric interpolant

$$\Delta x_i = \frac{(\Delta x_{i-1/2})^{-1} \Delta x_{i-1/2} + (\Delta x_{i+1/2})^{-1} \Delta x_{i+1/2}}{(\Delta x_{i-1/2})^{-1} + (\Delta x_{i+1/2})^{-1}} \tag{30}$$

for all interior $i$, its solution ensuring continuity when one of the $\Delta x_{i-1}$ is vanishingly small. The averaged position $x_i$ in (30) always lies between the midpoints of adjacent intervals so there can be no node overtaking.

Given $(v_x)_k^n$, equation (25) yields $(\Delta x_i)^{n+1}$, equation (23) gives $^{n+1}$, and equations 28) or (30) lead to $(x_i)^{n+1}$. The moved solution $(u_i)^{n+1}$ is given either by equation (26) (with

### 2.4.3 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 (10) and deduce $(v_x)_k = (\ v)_k / (\ x)_k$

calculate $\_$ from (5) and hence $^{n+1}$ from equation (23)

nd $(\ x_k)^{n+1}$ from (25) and hence $x_i^{n+1}$ from (28)

determine $(u_k)^{n+1}$ from (26), then

(a) for problems in which $u$ is given at one boundary only, assign $u_k^{n+1}$ to $u_i^{n+1}$, working away from the boundary w2way ary condi1

## 2.5 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 2.3.1 is explicit, rst-order-in-time, and sign-preserving in $u$, , and $x$ 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 2.4.3 is explicit, rst-order-in-time, non-tangling in $x_i$ and sign-preserving in $u_i$ over a time step for arbitrary $t$ and $v_x$. The algorithm is conservative in the sense of (31) or (32), propagates a self-similar scaling solution to order .The algorrithm is conservative in a particular sense, inherits the scale invariance of the problem and propagates a self-similar scaling solution to order .The algorithm is conservative in a particular sense, inherits the scale invariance of the problem and propagates a self-similar scaling solution to order .

Computations con rm the predictions of the theory.

As with any time-stepping scheme, temporal accuracy is undermined for large . However, even if is large and $v_i$ inaccurate, in a time step the $u_i$ keeps the same sign and the $x_i$ remain ordered.

## 2.6 Oscillations

Although $(x)_k$ remains positive it is not necessarily monotonic and approximation errors may lead to spurious oscillations in $v_x$ (see (22). By smoothing $v_x$ we introduce extra numerical di usion to counteract the oscillations at the expense of spatial accuracy.

The introduction of a Laplacian smoother,

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

to $(v_x)_k$ suppresses sawtooth oscillations in $v_x()_k$ and is equivalent to adding second order numerical di usion. 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 una ected by the smoothing.

# 3 The relative conservation method in 2-D

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

$$u_t = L\,u; \tag{33}$$

in a moving domain $R(t)$, where $L$ is a purely spatial operator, with given ux boundary conditions.

De ne the total mass to be

$$(t) = \int_{R(t)} u(\ ;t)\,dx \tag{34}$$

and introduce the relative density function

$$\bar{u}(x;t) = \frac{u(x;t)}{(t)}$$

satisfying

$$\int_{R(t)} \bar{u}(\ ;t)\,dx = 1 \tag{35}$$

from (34). By the Reynolds Transport Theorem the rate of change of the total mass (t) is

$$\frac{d}{dt} = \_= \frac{d}{dt}\int_{R(t)} u(\ ;t)\,dx = \int_{R(t)} u_t\,dx + \oint_{@R(t)} \frac{@(uv)}{@n}\,dS \tag{36}$$

where $v(x;t)$ is the Eulerian velocity, in which $\bar{\ }$ is given by (36) and $(t)$ by its integral. Then, from equations (33), (38) and (34), the velocity $v$ satisfies the time-independent PDE

$$r \quad (uv) = \quad \frac{\bar{\ }}{(t)} + L u$$

in $R(t)$.

For uniqueness put $v = r$ , where $(x;t)$ is a velocity potential satisfying the Poisson equation

$$r \quad (ur \quad) = \quad \frac{\bar{\ }}{(t)} + L u \tag{39}$$

For positive $u$ equation (39) has a unique solution for $(x;t)$ (and hence $v(x;t)$) under suitable Dirichlet or Neumann boundary conditions on .

## 3.1 A reference space

Introduce a Lagrangian moving coordinate $x( \ ; \ )$, where is a xed reference coordinate and $= t$, such that

$$\frac{@x}{@} = v( \ ; \ ) = v(x( \ ; \ ); \ ) \tag{40}$$

Equation (40) cannot be di erentiated in the same way as in (12) in 1-D since (11) is unidirectional, but we can obtain a multidimensional equivalent of (14).

The local Lagrangian conservation law (37) is expressed in terms of and as

$$\frac{1}{(t)} u( \ ; \ ) jJ(x; \ )j = b( \ ); \text{ independent of } t \tag{41}$$

where $J(x; \ )$ is the Jacobian of the transformation generated by (40), thus generalising (14). Given a function $u( \ ; \ )$ at any time , equation (41) leads to a Monge-Ampere equation for a function $x( \ ; \ )$, using a di erent potential function [13].

In spite of the di culties with generalisation of (40) and (41), if we specify

## 3.2 Spatial approximation in 2-D

Let the time variable be discretised as $^n = n$ , $n = 0; 1; 2; :::$, where is the time step, and de ne

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

In the conservation-based scheme of [1, 4, 16] the moving coordinate $x_i^n$ is advanced in time from equation (40) using the rst-order-accurate explicit Euler scheme. Although the nodes are moved in the correct direction to rst 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 $A_k( )$, $(k = 1; :::; K)$ and moving nodes $x_i( )$ with corresponding moved solution values $u_i( )$, $(i = 1; :::; N)$.

We begin by deriving a nite volume approximation to the velocity potential $_i( )$ at the nodes of the simplex from (39) in the case where $Lu = r \quad f$ where $f$ is a ux function and $g$ is a source term.

### 3.3.1 Approximating the velocity potential

Assuming that $Lu$ takes the form $r \quad f + g$ where $f$ is a ux function and $g$ is a source term, in a nite volume approach by integrating (39) over the boundary of a patch of triangles $_i$ surrounding node $i$,

$$\int_{ _i} r \quad (ur \quad ) \quad f) dx = \int_{ _i} g + \frac{\_}{(t)} \quad dx \qquad (42)$$

Using the divergence theorem equation (42) can be approximated by

$$\sum_{e_{ij}} \left( u_m \frac{( _j \quad _i)}{s_{ij}} \quad f_n \right) \overline{s}_{ij} = \int g + \frac{\_}{(t)} \quad dx \qquad (43)$$

where $\phi_i$ is the value of $\phi$ 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 $\overline{s}_{ij}$ 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 (43) for all interior nodes is square and sparse and, given boundary conditions on $\phi$ or $\partial\phi/\partial n$, is solved for the interor values by a standard method. Boundary conditions on $\phi$ correspond to specifying tangential velocities and those on $\partial\phi/\partial n$ to normal velocities.

### 3.3.2 Approximating triangle velocities

Having found $\phi_i$ at the nodes, we calculate an approximation to the velocity $V_k$ in a triangle as the gradient of a piecewise linear function through $\phi_i$ values at the vertices. In a triangle $k$ having values $\phi_1$, $\phi_2$, $\phi_3$ at vertices $(x_1, y_1)$, $(x_2, y_2)$, $(x_3, y_3)$, the velocity is

$$V_k = (\nabla\phi)_k = \frac{1}{2A_k} \begin{pmatrix} 1 & 1 & 1 \\ y_1 & y_2 & y_3 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix}_k \tag{44}$$

where $A_k$ is the area of the triangle given by

$$A_k = \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix}_k \tag{45}$$

### 3.3.3 Approximating the nodal velocities

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

$$v_i = \frac{\sum_k A_k^{-1} V_k}{\sum_k A_k^{-1}} \tag{46}$$

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 $\phi$ given by (43) we obtain triangle-based velocities $V_k$ from (44) and node-based velocities $v_i$ from (46).

### 3.3.4 Moving the nodes

In order to combat mesh tangling, instead of moving the nodes by the nodal velocities (46) 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

$$(s)^{n+1} = (s)^n \exp\{ (V_e/s)\}^n ; \qquad (47)$$

which preserves the sign of the edge length $(s)$ over a time step irrespective of or $(V_e/s)$ (and therefore that of $(s)$).

Since all the edge lengths remain of edge $v$

Since all the triangle areas remain of the same sign, mesh tangling is avoided.

The rate of change of the total mass is given by (5), then the total mass $^{n+1}$ obtained from the explicit exponential scheme

$$^{n+1} = {}^{n} \exp\{ t \, \dot{} = {}^{n}\} \tag{50}$$

### 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 $^{n+1}$. Unlike in one dimension, there is no unique mesh that is consistent with given edge lengths or triangle areas, each problem being overdetermined in general.

We generalise the approach of (30) to 2-D using either edge-lengths or triangle-areas, as follows.

Edge-lengths

Suppose that the interior node $x_i$ of the simplex is connected to $J_i$ nodes $x_j$, $(j = 1; : : : ; J_i)$, then generalisations of (29) and (30) to 2-D are

$$\sum_{j=1}^{J_i} \frac{(x_m - x_i)}{s_{ij}} = 0 \qquad \text{and} \qquad x_i = \frac{\sum_{j=1}^{J_i} (s_{ij})^{-1} x_m}{\sum_{j=1}^{J_i} (s_{ij})^{-1}} \tag{51}$$

for all interior nodes $x_i$, where the $x_m$ are the midpoints of the edges joining $x_i$ to $x_j$ and $s_{ij}$ is the (positive) length of the edge connecting node $i$ to node $j$. Positivity of the weights in the second of (51) ensures that $x_i$ lies in the convex hull of the midpoints $x_m$ of the edges through $x_i$, thus preventing tangling.

Triangle-areas

Similarly, if $k$ ===i sTd 73 045eno(lo)-277(Tnd[(T)so)-27751

Boundary nodes

Boundary nodes are treated either as Dirichlet conditions by moving the nodes with the known boundary velocities $v_i$ of (46), or as Neumann conditions by applying modified forms of (51) or (52), as follows. In the case of edges, the boundary node equation (51) is modified to

$$\sum_{j=1}^{J_b} \frac{x_m - x_b}{s_{bj}} = \sum_{j=1}^{J_b} e_j$$

where $J_b$ is the number of edges emanating from boundary node $b$ and $e_j$ is the unit vector from $x_b$ to $x_m$, while in the case of areas the boundary node

## 3.4 Finding the moved solution

Having obtained locations of the nodes at time $t^{n+1}$, the moved solution $u^{n+1}$ at these nodes is found from approximations to the Lagrangian conservation law (41), as follows.

From edges:

An approximate form of the Lagrangian conservation law (6) along an edge $e$ is

$$\frac{1}{\Delta t^{n+1}}(u_m \Delta s_e)^{n+1} = b_e = \frac{1}{\Delta t^n}(u_m \Delta s_e)^n \tag{55}$$

where $m$ is the midpoint of the edge and $b_e$ is time-independent, leading to

$$(u_m)^{n+1} = \frac{\Delta t^{n+1}}{\Delta t^n} \frac{(\Delta s_e)^n}{(\Delta s_e)^{n+1}} u_m \tag{56}$$

where $\Delta t^{n+1}$ is given by (50). The moved solution $u_i^{n+1}$ at the node $x_i$ is then the barycentric interpolant

$$u_i^{n+1} = \frac{\sum_{j=1}^{J_i} ((\Delta s_j)^{n+1})^{-1} u_m^{n+1}}{\sum_{j=1}^{J_i} ((\Delta s_j)^{n+1})^{-1}} \tag{57}$$

of the $u_m^{n+1}$ over those edges containing node $x_i$

of the $u_g^{n+1}$ over triangles containing node $x_i$, (cf .(29)), preserving the sign of $u_i^{n+1}$ 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 algorithm for the solution of the PDE problem with prescribed $u_x$ boundary conditions on a simplex is as follows.

Algorithm 3

At each time step:

1. Determine the velocity potential at the nodes from (43) and the nodal velocities from (44) and (46),
2. advance the edge-lengths $s$ or triangle-areas $A$ in time from (47) or (49), preserving their sign, and construct the nodes from (51) or (52),
3. calculate $^{n+1}$ from (50),
4. evaluate $u_e$ or $C_k$ from the right hand side of (55) or (58) and retrieve the solution on the moved mesh using (56) and (57) or (59) and (60).

Note that, due to the calculation of the nodes, the $u_e$ or $C_k$, 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 (55),

$$\sum_j u_m \, s_e$$

over all edges is constant over a time step, or for triangle areas, from (58),

$$\sum_k u_g A_k$$

over all areas is constant over a time step.

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

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).

[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{Ampere equation, SIAM J. Sci. Comput., 31 (5), pp. 3438-3465 (2009).

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

[15] T.E. Lee, M.J. Baines S. Langdon, and M.J. Tindall , A moving mesh approach for modelling avascular tumour growth, Appl. Numer, Math., 72, pp 99{114, (2013).

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

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

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