Report no. 05/21

MMPDE vs. SGWMFE

Experiments in one dimension

Abigail Wacher

Faculty of Aerospace Engineering, Technion, Haifa, Israel

We compare experimentally Moving Mesh Partial Differential Equations (Huang et al. (1994)) to the String Gradient Weighted Moving Finite Element Method (Wacher et al. (2003)) applied to the viscous Burgers equation and to the porous medium equation in one dimension. The methods are tested on a travelling wave solution, the Barenblatt solution of the porous medium equation and on one of the waiting time solution discovered by Lacey (1983).

Oxford University Computing Laboratory
Numerical Analysis Group
Wolfson Building
Parks Road
Oxford, England OX1 3QD

October, 2005

1 String Gradient Weighted Moving Finite Elements

The moving mesh method we will be describing shortly is the String Gradient Weighted Moving Finite Element method (SGWMFE) developed in Wacher et al. (2003). The SGWMFE formulation for systems of partial differential equations was originally proposed by Miller (1997) as an alternative formulation of the Gradient Weighted Moving Finite Element (GWMFE) method, which was developed in detail in Carlson and Miller (1998a) and Carlson and Miller (1998b) by Carlson and Miller for one and two-dimensional problems. Carlson and Miller (1998a) report on the design and implementation of a robust and versatile GWMFE code in one dimension applied to various PDEs and PDE systems. Sample problems for which the code was tested in that paper are: 1) a convection-diffusion boundary layer problem, 2) Burgers equation with diffusion term, plus a strong nonlinear source term and also with no diffusion, 3) drift-diffusion equations for semiconductors, 4) Sod's shock tube problem, and 5) a steady-state convection problem. There they found that GWMFE efficiently produces accurate results for problems which form steep moving fronts. The corresponding two-dimensional paper Carlson and Miller (1998b) does the same as the one-dimensional paper including the additional application problems: 1) Non-linear arsenic diffusion, 2) The Buckley-Levett "black oil" equations, and 3) motion by mean curvature which was implemented in the one-dimensional case in the paper Miller (1997) by Miller. The results therein show that the method is intended for "problems with sharp moving fronts where one needs to *resolve* the fine-scale structure of the front to compute the correct answer" greatly improving on the first Moving Finite Element (MFE) method developed originally by K.

Miller and R.N. Miller in Miller and Miller (1981) in one dimension, and by K. Miller in Miller (1981) for two dimensions.

What follows is the SGWMFE formulation for a scalar equation in one dimension, previously investigated for systems of partial differential equations in Wacher et al. (2003) and Wacher (2004). We note that in the case of a single PDE the SGWMFE and GWMFE reduce to the same set of equations, however here we use the SGWMFE approach to developing the system of equations following Wacher (2004). The theory is presented in such a way that it should be clear how SGWMFE is generalised to systems of equations with any number of components, in multiple dimensions, however see Wacher (2004) for the detailed extensions.

1.1 Formulation of the SGWMFE method

Given a partial differential equation as in (1.1), SGWMFE treats the solution graph for the equation as an evolving one-dimensional manifold $(x, u(x,t))$. To begin, consider the example of a PDE,

$$u_t = L(u) \tag{1.1}$$

for the unknown function $u(x,t)$ on a one-dimensional spatial interval Ω. L is a general first or second order nonlinear differential operator in space.

Consider a re-parameterisation with a moving coordinate $x(\tau,t)$, where τ is a one-dimensional parameter whose domain is arbitrary but bounded. Under the re-parameterisation the solution manifold becomes an evolving parameterised one-dimensional manifold immersed in two dimensions with the position vector

$$\mathbf{u}(\tau,t) = (x(\tau,t), u(\tau,t)), \tag{1.2}$$

for the evolving re-parameterised points of the solution graph.

At each parameterised point on the evolving manifold split the velocity vector $\dot{\mathbf{u}} = (\dot{x}, \dot{u})$ into its tangential, $[\dot{\mathbf{u}}]_T$, and its normal, $[\dot{\mathbf{u}}]_N$, parts. Noting that solving for the tangential part $[\dot{\mathbf{u}}]_T$ makes no changes to the solution manifold since any points along the manifold moving tangentially stay on the manifold, thus maintaining it the same solution manifold (not necessarily the graph of a function). The original PDE (1.1), is written in the following vector form:

$$\begin{pmatrix} 0 \\ u_t \end{pmatrix} = \begin{pmatrix} 0 \\ L(u) \end{pmatrix}. \tag{1.3}$$

Taking the normal component of equation (1.3) results in the same vector regardless of the parameters used to describe the velocity of the solution manifold, that is

$$\begin{pmatrix} \dot{x} \\ \dot{u} \end{pmatrix}_N = \begin{pmatrix} 0 \\ u_t \end{pmatrix}_N. \tag{1.4}$$

For a proof of equation (1.4) see Wacher (2004). The equation for the normal velocity is then written

$$\begin{pmatrix} \dot{x} \\ \dot{u} \end{pmatrix}_N = \begin{pmatrix} 0 \\ L \end{pmatrix}_N, \tag{1.5}$$

which is a system of two PDEs for the two unknown variables $x(\tau, t), u(\tau, t)$. Equation (1.5) is only a parameterisation of equation (1.1) as long as the solution manifold is the graph of a function.

It is convenient to use here the projection matrix P which projects any given vector, $\mathbf{F}$, into its normal part, $[\mathbf{F}]_N$, at a given point on the solution manifold, $(x, u(x))$. $[\mathbf{F}]_N$ is obtained by subtracting, from $\mathbf{F}$, the tangential component $[\mathbf{F}]_T$, where the tangential component is given by

$$[\mathbf{F}]_T = \mathbf{t}\mathbf{t}^{\mathrm{T}}\mathbf{F}, \tag{1.6}$$

where $\mathbf{t} = (1, u_x)/\sqrt{1+u_x^2}$ is the unit tangent vector to the manifold at this point. Hence

$$[\mathbf{F}]_N = \mathbf{F} - [\mathbf{F}]_T = (\mathbf{I} - \mathbf{t}\mathbf{t}^{\mathrm{T}})\mathbf{F} = P\mathbf{F}, \tag{1.7}$$

where

$$P = \frac{1}{1+u_x^2}\begin{pmatrix} u_x^2 & -u_x \\ -u_x & 1 \end{pmatrix}. \tag{1.8}$$

Equation (1.5) is then discretized by letting the SGWMFE approximation be an evolving, piecewise linear manifold with its two-dimensional nodal positions $\mathbf{u}_j = (x_j(t), u_j(t))$ as unknowns. Multiplying equation (1.7) by the well known nodal "hat" basis function α^j, and integrating over the spatial domain gives:

$$\int \begin{pmatrix} \dot{x} \\ \dot{u} \end{pmatrix}_N \alpha^j \mathrm{d}s = \int \begin{pmatrix} 0 \\ L \end{pmatrix}_N \alpha^j \mathrm{d}s, \tag{1.9}$$

at each node j.

1.2 Time derivative terms

Using equation (1.9) the integrals of the time derivatives (the left hand side of the PDE system) in $cell_i$ contribute to the i^{th} node by:

$$\int_{cell_i} \begin{pmatrix} 0 \\ u_t \end{pmatrix}_N \alpha^i \mathrm{d}s = \int_{cell_i} P \begin{pmatrix} \dot{x} \\ \dot{u} \end{pmatrix} \alpha^i \mathrm{d}s. \tag{1.10}$$

The integral is then obtained by using Simpson's quadrature rule found in many introductory mathematical and physical books such as Jeffreys and Jeffreys (1946),

$$\int_{cell_i} P\dot{\mathbf{u}}\alpha^i \mathrm{d}s = P(\frac{1}{3}\dot{\mathbf{u}}_i + \frac{1}{6}\dot{\mathbf{u}}_{i-1})\Delta s_i, \tag{1.11}$$

where

$$\Delta s_i = \| \mathbf{u}_i - \mathbf{u}_{i-1} \|_2 = \sqrt{(x_i - x_{i-1})^2 + (u_i - u_{i-1})^2}, \tag{1.12}$$

and noting that the term u_x is constant on each i^{th} cell. This implies that $\mathrm{d}s = \sqrt{1+u_x^2}\mathrm{d}x$ and the elements of the matrix P are also constant on the i^{th} cell. Further, since α^i and $\dot{\mathbf{u}}$ are both linear functions on the cell, then the integrand above is at most a quadratic polynomial on the i^{th} cell, which is the interval of integration. Since Simpson's rule is exact for polynomials of up to third order the expression for the integral in equation (1.11) is exact.

Once the ODE system has been constructed we use the numerical integration code developed in Carlson and Miller (1998a). The integration method used is the Backward Differentiation Formula 2 (BDF2), an implicit ODE solver with adaptive time stepping. For details of the implementation of this code see Carlson and Miller (1998a).

1.3 Flux terms

Consider restricting the non-linear operator L to have the particular form of a flux function:

$$u_t = -f_x(u, v). \tag{1.13}$$

For a scalar function $f(x, u(x))$ it will be useful to denote by f_i its value at the i^{th} node and by $[f]_i$, the average over the cell. Thus $f_i = f(x_i, u_i)$ and $[f]_i = \frac{1}{\Delta x_i}\int_{cell_i} f\mathrm{d}x$. Using this notation and noting that $\mathrm{d}s = \sqrt{1+u_x^2}\mathrm{d}x$, the integral contributions from the flux term in $cell_i$ onto the i^{th} node can be written

$$\begin{aligned}
\int_{cell_i} P\begin{pmatrix} 0 \\ -f_x \end{pmatrix}\alpha^i \mathrm{d}s &= P\begin{pmatrix} 0 \\ 1 \end{pmatrix}\int_{cell_i}(-f_x\alpha^i)\mathrm{d}s & (1.14)\\
&= \begin{pmatrix} -u_x \\ 1 \end{pmatrix}\frac{\int_{cell_i}(-f_x\alpha^i)\mathrm{d}x}{\sqrt{1+u_x^2}} & (1.15)\\
&= \begin{pmatrix} -u_x \\ 1 \end{pmatrix}\frac{[f]_i - f_i}{\sqrt{1+u_x^2}}. & (1.16)
\end{aligned}$$

1.4 Constant coefficient diffusion terms

Now consider a term with a non-linear operator L to have an artificial diffusivity term:

$$u_t = \epsilon u_{xx}, \tag{1.17}$$

where ϵ determines the magnitude of the artificial diffusion (here it is a constant). Using piecewise linear basis functions means that diffusive terms vanish in the interior of any cell but are undefined at nodes. One way of dealing with this problem is to use the mathematical technique of mollification as in Carlson and Miller (1998a). The first derivative, while being constant over the main body of the cell, is assumed to vary smoothly between cell values in a small neighbourhood of width 2δ at each node, see Figure 1. Then take the limit $\delta \to 0$. Thus in any integral involving diffusion terms it is only necessary to take into account the small neighbourhoods near each node since u_{xx} is still identically zero for most part of each cell. The use of the mollification technique is presented below using the same principles as in Carlson and Miller (1998a) and Miller (1997). For further reading on the use of mollification for MFE and GWMFE see Baines (1994).

Denote the value of u_x on $cell_i$ as m_i, then mollify by defining a variable $\sigma(x)$ and then, for instance at the right end of the cell, map a neighbourhood of width 2δ of x_i to $-1 \leq \sigma(x) \leq 1$, map u_x to $m_i \leq u_x \leq m_{i+1}$. Then the integral contribution on the i^{th} node due to diffusive terms is:

$$\epsilon \int_{-\delta}^{\delta} P \begin{pmatrix} 0 \\ u_{xx} \end{pmatrix} \alpha^i \mathrm{d}s = \epsilon \int_{-\delta}^{\delta} \frac{1}{\sqrt{1+u_x^2}} \begin{pmatrix} -u_x u_{xx} \\ u_{xx} \end{pmatrix} \mathrm{d}x, \tag{1.18}$$

where α^i has been replaced by 1 since the integral is taken over the infinitesimal neighbourhood of x_i which is the point where $\alpha^i = 1$. The

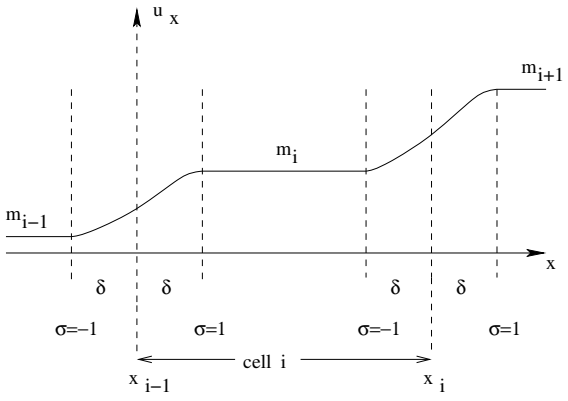

Figure 1: Mollification: the value of u_x on $cell_i$ is m_i, the value of u_x on $cell_{i-1}$ is m_{i-1} and the value of u_x on $cell_{i+1}$ is m_{i+1}. The value of u_x is assumed to vary smoothly in a small neighbourhood of each end of the cell of width 2δ and then $\delta \to 0$.

integral is now rewritten using the mapping

$$u_x = m_c + \Delta m_c \sigma(x), \tag{1.19}$$

where $m_c = \dfrac{m_i + m_{i+1}}{2}$, and $\Delta m_c = \dfrac{m_{i+1} - m_i}{2}$. With the first component of (1.18) in mind, let

$$I_A = \int_{-\delta}^{\delta} \frac{u_x u_{xx}}{\sqrt{1+u_x^2}} \mathrm{d}x. \tag{1.20}$$

Noting that when describing the mollification of u_x it is important to understand that the function u_x that is being mollified is the approximation of the actual unknown variable. Substituting equations (1.19) into (1.20), so that

$$\mathrm{d}u_x = u_{xx}\mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}\sigma} u_x \, \mathrm{d}\sigma,$$

then under change of variable for the integral, the neighbourhood size

falls out, and by letting

$$a = 1 + m_c^2, \tag{1.21}$$

$$b = 2m_i \Delta m_c, \tag{1.22}$$

$$c = \Delta {m_c}^2, \tag{1.23}$$

results in the following simplified expression for I_A in terms of σ:

$$I_A = \frac{b}{2} \int_{-1}^{1} \frac{\mathrm{d}\sigma}{\sqrt{a + b\sigma + c\sigma^2}} + c \int_{-1}^{1} \frac{\sigma \mathrm{d}\sigma}{\sqrt{a + b\sigma + c\sigma^2}}. \tag{1.24}$$

Similarly, applying the same technique to the second component leads to a similar integral expression, see Wacher (2004).

Despite knowing analytic expressions for these integrals, the expressions for the integrals can be subject to severe roundoff errors and great care has to be taken in their evaluation, as in Carlson and Miller (1998a) and Carlson and Miller (1998b) where they develop formulas for these integrals to avoid roundoff error. For the results shown in this report, sixteen point gaussian quadrature formula for these integrals was used.

1.5 Non-uniform diffusion terms in conservative form

Now consider the semi-linear diffusion terms from a non-linear operator L of the form $(au_x)_x$, where $a = a(x, u)$. The contribution from this term in the i^{th} cell onto its i^{th} node can be written

$$\begin{aligned}\int P \begin{pmatrix} 0 \\ a(x,u)u_x \end{pmatrix}_x \alpha^i \mathrm{d}s &= P_i \begin{pmatrix} 0 \\ m_i \end{pmatrix} \int_{cell_i \setminus nbd_i} a_x \alpha^i \mathrm{d}s \\ &+ \int_{nbd_i} P \begin{pmatrix} 0 \\ a_x u_x \end{pmatrix} \alpha^i \mathrm{d}s \\ &+ a_i \int_{nbd_i} P \begin{pmatrix} 0 \\ u_{xx} \end{pmatrix} \alpha^i \mathrm{d}s \\ &+ P_{i+1} \begin{pmatrix} 0 \\ m_{i+1} \end{pmatrix} \int_{cell_{i+1} \setminus nbd_i} a_x \alpha^i \mathrm{d}s. \end{aligned} \tag{1.25}$$

where m_i and m_{i+1} are the values of u_x on $cell_i$ and $cell_{i+1}$ respectively.

The second term on the right hand side of equation (1.25) is identically zero since the integrand is bounded in the infinitesimal neighbourhood of the i^{th} node. Note that in the third term on the right hand side of equation (1.25), the value a_i has been factored out. This is because $a(x,u)$ is replaced by a_i, its value near the i^{th} node, where u_{xx} has its infinitesimal support. See Section 4.5 of Carlson and Miller (1998a) for a similar procedure. The integrand in this same term is obtained from the theory for the constant coefficient Laplacian terms. The integrands in the other two terms on the right hand side of equation (1.25), the first and fourth terms, can be derived using integration by parts as was done for the flux terms, leading to the first and third terms in equation (1.26). As before let m_i be the value of u_x on $cell_i$,

and let m_{i+1} be the value of u_x on $cell_{i+1}$.

$$\begin{aligned}\int P \begin{pmatrix} 0 \\ a(x,u)u_x \end{pmatrix}_x \alpha^i \mathrm{d}s &= m_i \begin{pmatrix} -m_i \\ 1 \end{pmatrix} \frac{\int_{cell_i \setminus nbd_i} a_x \alpha^i \mathrm{d}x}{\sqrt{1+m_i^2}} \\ &+ a_i \int_{nbd_i} P \begin{pmatrix} 0 \\ u_{xx} \end{pmatrix} \alpha^i \mathrm{d}s \\ &+ m_{i+1} \begin{pmatrix} -m_{i+1} \\ 1 \end{pmatrix} \frac{\int_{cell_{i+1} \setminus nbd_i} a_x \alpha^i \mathrm{d}x}{\sqrt{1+m_{i+1}^2}}. \end{aligned} \tag{1.26}$$

where $\int_{cell_i \setminus nbd_i} a_x \alpha^i \mathrm{d}x = a_i - [a]_i$ and $\int_{cell_{i+1} \setminus nbd_i} a_x \alpha^i \mathrm{d}x = [a]_{i+1} - a_i$, using integration by parts as in the flux terms in equations (1.14) to (1.16). The second term in equation (1.26) is the Laplacian term with constant coefficient a_i, identical to what has been derived in the theory previously.

1.6 Regularization

Regularization of the mass matrix (resulting from the MFE, GWMFE or SGWMFE discretizations) is used to avoid the mass matrix from becoming ill-conditioned from possible degenerate nodes or cells. The first type of possible degeneracy is that discussed in Wathen and Baines (1985), for the MFE method where the same phenomena occurs, whereby the i^{th} block of the mass matrix is analyzed, and it is found that when the slopes of two adjacent cells are equal, then the first row of the mass matrix contains only zero elements. This type of degeneracy is called parallelism, that is when two cells are collinear, the central node joining the two cells is unnecessary and as a result the matrix is singular when this happens.

The other type of degeneracy, also discussed in Wathen and Baines

(1985), is the "shock" type degeneracy which happens when the cell locations in the MFE method become identical. It can be seen by looking at the mass matrix, if two nodes are in the same place then the elements of the mass matrix corresponding to those nodes will be identical. The result then is that two blocks in the mass matrix become identical thus making the matrix rank deficient.

The common approach to avoid the mass matrix becoming ill-conditioned is to add regularization terms. See Carlson and Miller (1998a) and Miller (1997) for regularization of GWMFE in one dimension, and Wacher (2004) for SGWMFE. The terms added are much smaller than the error tolerance used to solve the discretized ODE system, however the terms are of the same form as the terms in the mass matrix so that when there is a degenerate node the regularization terms in that row dominate so as to make the system nonsingular.

The regularization coefficients we use, added to the diagonal terms of the mass matrix, are of the form C/x_i or $C/\sqrt{1+x_i^2}$, where C is chosen so that it is well below the truncation error, thus not affecting the accuracy of the solutions beyond the tolerance required. For all experiments in this report a local truncation error tolerance on the residuals was set to 10^{-8}, but not all the regularization parameters used are the same. For the Porous Medium Equation with the Barenblatt solutions no regularization was needed and thus none was used. For the Waiting Time solutions the regularization terms used were of the form $5(10)^{-10}/x_i$, and for the viscous Burgers equation a regularization term of the form $10^{-13}/\sqrt{1+x_i^2}$ was used.

1.7 Summary

The equations for SGWMFE/GWMFE were presented in this section with a projection matrix. An advantage that has been identified pre-

viously is that the equations resulting from the formulation using this projection matrix make a more elegant extension to larger numbers of equations than is the case for the original formulation of the GWMFE method, though both approaches naturally reduce to the same set of equations for scalar PDEs.

2 Moving Mesh Partial Differential Equations

2.1 The Equidistribution Principle and MMPDEs in One Dimension

In one space dimension, good grids can be constructed using the equidistribution principle. Let $x = x(\xi)$ be a strictly increasing map from the *computational domain* $[0, 1]$ onto the *physical domain* $[a, b]$. It equidistributes the monitor function $g = g(x) > 0$ if for every $\xi \in [0, 1]$

$$\int_a^{x(\xi)} g(s)\,\mathrm{d}s = \xi \int_a^b g(s)\,\mathrm{d}s. \tag{2.1}$$

Differentiation of (2.1) twice with respect to ξ gives the equivalent formulation,

$$(g\,x_\xi)_\xi = 0, \quad x(0) = a, x(1) = b. \tag{2.2}$$

If the monitor function g is some measure of the local computational effort required and x equidistributes g, then more grid points are concentrated where needed. As a standing assumption, we let $g = g(x, t)$ be continuous on the space-time domain $[a, b] \times [0, T]$, strictly positive with $g_0 = \min_{x,t} g(x, t)$, and $\int g\,\mathrm{d}x = 1$.

By solving the *physical* PDE and (2.2) simultaneously at every time step, the equidistribution principle can be used to generate a moving mesh. This solution process would be relatively expensive and the mesh

obtained unsmooth which, apart from requiring small time steps, can lead to a deterioration in the convergence rate. Several authors (see Huang et al. (1994); Huang and Russell (1997) and references therein) introduced relaxations of this process by introducing mesh speed in different ways. A very general approach which is also easily generalized to higher dimensions was introduced in Huang and Russell (1999). By moving the mesh in the steepest descent direction of a *mesh functional*, parabolic *moving mesh partial differential equations*(MMPDEs) are obtained which can provide an efficient and stable moving mesh and a reliable moving mesh method.

With the right choice of parameters, one obtains the MMPDE

$$x_t = \frac{1}{\tau}(g x_\xi)_\xi, \quad x(0) = a, x(1) = b, \tag{2.3}$$

where $\tau > 0$ can be seen as a time scale or a relaxation parameter. In Huang et al. (1994), this MMPDE is labeled MMPDE5. It is shown in Ortner (2003) that its solution approximately equidistributes the monitor function in a sense that is made precise.

2.2 Monitor Functions Based on Geometric Properties

So far we have only assumed that we have a monitor function which somehow measures the local computational effort required. We now introduce several choices, which have been successfully used in the past and also a few new ideas. Two general classes are considered. In this section, we introduce monitor functions based on geometric properties of the solution like the arclength or the curvature. We call these *geometric* monitor functions.

For better readability, in the following presentation the scaling $\int_0^1 g\,dx = 1$ is not included. Furthermore, if a monitor function con-

sists of a convex combination of two other monitors g_1 and g_2, i.e., $g = \alpha g_1 + (1-\alpha)g_2$, then we implicitly assume that they are scaled to satisfy $\int g_i \,\mathrm{d}x = 1$. For example, we write $g = \alpha e(u) + (1-\alpha)f(u)$ instead of $g = \alpha e(u)/\int e(u)\,\mathrm{d}x + (1-\alpha)f(u)/\int f(u)\,\mathrm{d}x$.

One of the most popular monitor functions is the arclength,

$$g_{\mathrm{AL}} = \sqrt{1+u_x^2}. \tag{2.4}$$

Its main characteristic is its robustness, its wide applicability and interpolation error constants which are likely to be independent of perturbation parameters or similar. These constants usually hold, however, only for first order convergence.

Intuitively, when using piecewise linear splines for the approximation space, the first choice would be a monitor function based on the second derivative. Although great accuracy can be achieved, equally great care has to be taken as Blom and Verwer Blom and Verwer (1989) show in numerous experiments. In Ortner (2003), a monitor function is introduced, which is constructed as a combination of the arclength and jumps in the gradient, which are closely related to the second derivative. Let the linear spline u have values u_i at gridpoints x_i then the *jump* monitor function is defined as the linear spline of

$$g_{\mathrm{JMP},i} = \alpha\, g_{\mathrm{AL}}(x_i) + (1-\alpha)\left|[u_x]_{x=x_i}\right| \tag{2.5}$$

where $[u_x]$ denotes the jump in the gradient. Small choices of α turned out to produce similar problems as those observed in Blom and Verwer (1989). A good value for most problems is $\alpha = 0.7$.

Finally, let us also note that for equations like the porous medium equation, monitor functions can be constructed which take into account specific features of the equation, say, the conservation of mass. For an example see Baines et al. (2003).

2.3 Basic Implementation of the MMPDE method

The implementation of a moving mesh method based on the MMPDEs described in Section 2.1, and in Ortner (2003), involves solving a coupled system of partial differential equations of which at least one is nonlinear, as is the coupling between the two.

In this section, an implementation of a finite element moving mesh method based on a decoupling of the physical and the moving mesh PDEs is described. For the physical PDE, only second order parabolic problems with Dirichlet boundary conditions are considered.

The physical model problem which encompasses all examples considered in this report is

$$\begin{aligned} u_t - \nabla \cdot \mathbf{F}(u) - \nabla \cdot (A(u,x,t)\nabla u) &= f \quad \text{in } \Omega_T, \\ u &= u_0 \quad \text{at } t = 0, \\ u &= u_1 \quad \text{on } \partial\Omega. \end{aligned}$$

2.3.1 The Lagrangian Formulation

Following the notation in Ortner (2003), we define the functional

$$B(u,t;v,w) = \int_\Omega \mathbf{F}'(u) \cdot \nabla v + (\nabla v)^\top A(u,x,t) \nabla w \, \mathrm{d}x$$

which is bilinear in (v,w). Assume that for every $t \in [0,T]$ we are given a triangulation with grid points $(x_i(t); i = 0,\ldots,M)$ and let $(\Phi_i(t); i = 0,\ldots,M)$ be the associated nodal basis of piecewise linear functions. Suppose that the first $\tilde{M}$ nodes are the interior nodes. Then the finite element test and solution spaces are respectively defined as

$$\begin{aligned} V_h(t) &= \text{span}\,\{\Phi_i(t) : i = 0,\ldots,\tilde{M}\} \\ S_h(t) &= \sum_{j=\tilde{M}+1}^{M} u_1(x_j,t)\Phi_j(t) + V_h(t) \end{aligned}$$

and the semidiscrete finite element method reads *For all* $0 < t \leq T$ *find* $u_h(t) \in S_h(t)$ *such that for all* $\varphi \in V_h(t)$,

$$(u_{h,t}, \varphi)_{L^2(\Omega)} + B(u_h, t; u_h, \varphi) = (f, \varphi)_{L^2(\Omega)}. \tag{2.6}$$

Consider for a moment the time discretization of (2.6) by the implicit Euler method. Assume a partition $0 = t_0 < t_1 < \cdots < t_N = T$ of the time-interval $[0, T]$ is given and set $k_n = t_n - t_{n-1}$. For $f \equiv 0$ the implicit Euler method reads

$$(u_h(t_n), \varphi)_{L^2(\Omega)} + k_n B(u_h(t_n), t_n; u_h(t_n), \varphi) = (u_h(t_{n-1}), \varphi)_{L^2(\Omega)} \quad \text{for all } \varphi \in V_h(t_n).$$

If the mesh is not constant (e.g. moving), then the term $(u_h(t_{n-1}), \varphi)$ on the right hand side has to be calculated by a projection of $u_h(t_{n-1})$ onto the space $S_h(t_n)$ with the new mesh.

An alternative way, a discrete equivalent of the so-called Lagrangian formulation of the PDE,

$$u_t - L(t)u = \frac{\mathrm{d}u}{\mathrm{d}t} - (\nabla u)^\top x_t - L(t)u = f(t),$$

is far more efficient even in the one-dimensional case. Here, $\mathrm{d}u/\mathrm{d}t$ stands for the derivative of $u(x(\xi, t), t)$ with respect to t. This form of the PDE is analyzed in greater generality in Jimack and A.J. (1991) and Cao et al. (1999).

Based on this formulation, we can write an alternative formulation of the semidiscrete Galerkin finite element method (2.6). Equation (2.6) is equivalent to

$$\sum_i u_i' \left(\Phi_i, \varphi\right)_{L^2(\Omega)} - \left(\frac{\partial x}{\partial t} \cdot \nabla u_h, \varphi\right)_{L^2(\Omega)} + B(t, u_h; u_h, \varphi) = (f, \varphi)_{L^2(\Omega)}. \tag{2.7}$$

The form (2.7) of the semidiscrete finite element method can be easily discretized in time by any ODE solver. One possible choice is described in Section 2.3.2.

2.3.2 Discretization in Time

To enable adaptive time-stepping for the physical PDE, we use the second order *singly diagonally implicit* Runge-Kutta (SDIRK2) method. This method was proposed for moving mesh equations in Beckett et al. (2001) and Beckett et al. (2002). Details about the derivation and stability properties can be found in Hairer and Wanner (1991).

Suppose the SDIRK2 method is employed to integrate the system

$$\dot{u} = f(t, u),$$

where $f : \mathbf{R} \times \mathbf{R}^m \to \mathbf{R}^m$ on the grid $t_0 < t_1 < t_2 < \ldots$. Then, with $k_n = t_n - t_{n-1}$ and $\gamma = (2 - \sqrt{2})/2$, the method is given by

$$\begin{aligned} v_1 &= f(t_{n-1} + \gamma k_n, u(t_{n-1}) + \gamma k_n v_1), \\ v_2 &= f(t_{n-1} + k_n, u(t_{n-1}) + (1-\gamma)k_n v_1 + \gamma k_n v_2), \\ u(t_n) &= u(t_{n-1}) + k_n((1-\gamma)v_1 + \gamma v_2). \end{aligned} \tag{2.8}$$

To obtain a local estimate of the error, the second-order SDIRK2 scheme can be combined with an appropriate first-order scheme. To maximize computational efficiency, we use

$$\hat{u}(t_n) = u(t_{n-1}) + k_n v_1 \tag{2.9}$$

where v_1 is that calculated in (2.8). For details about the time step control, see Beckett et al. (2001) or Ortner (2003). In this section, we briefly review a method of decoupling the physical from the moving mesh equations which reduces the effort for solving the MMPDE significantly.

For evolving the mesh, we use the implicit Euler method in time. For evolving the physical PDE, we use the SDIRK2 scheme (2.8). Suppose we have computed $\mathbf{x}(t_{n-1})$ and $u_h(t_{n-1})$. We use $u_h(t_{n-1})$ as a first approximation for $u_h(t_n)$ to compute an approximation of the

mesh $\mathbf{x}(t_n)$. We then use this approximation to the new mesh to compute an approximation of $u_h(t_n)$. This procedure is iterated as often as necessary to improve the approximations for $\mathbf{x}(t_n)$ and $u_h(t_n)$.

The arising nonlinear systems are solved either by a Newton method or by a fixed point iteration. If the nonlinear iteration does not converge, the stepsize is decreased until it is successful.

2.4 Artificial Smoothing of the Monitor Functions

The necessity of spatial smoothing is discussed in Huang and Russell (1997) in great detail. Smoothing the monitor function, e.g. by some local averaging procedure, can greatly improve performance and even accuracy. The reason is mainly that the iterations converge faster so that bigger time-steps can be taken, while at the same time a slightly displaced mesh will only insignificantly decrease accuracy. In fact, a smoother mesh might even improve it. Spatial smoothing has been fully studied and we shall not discuss it further. Apart from explaining analytically why spatial smoothing is important, the analysis in Ortner (2003) suggests that smoothness in time is just as important and might bring additional stability and performance. We impose this in two ways.

- At every time-step we take a weighted average between the computed monitor function and that from the last time-step, i.e., we use $g = \text{TMP\_SM}\, g_{\text{old}} + (1 - \text{TMP\_SM}) g_{\text{new}}$.

- Becket et al. Beckett et al. (2001) suggest using 4 mesh iterations in the alternating solution procedure. Instead we use 8 relaxed iterations, i.e., we choose a relaxation parameter $\text{MRELX} \in (0, 1]$ and take $x_{\text{new}} = x_{\text{old}} + \text{MRELX} \times d$ if d is the usual iteration step.

An intensive benchmark was carried out, producing solutions for several monitor functions and a wide variety of choices of MRELX and TMP_SM. For a very large range of smoothing parameter choices, the error changes on such a small scale that we can practically choose the parameters solely based on performance considerations. For all of the experiments in this work, we use: TMP_SM = 0.3, MRELX = 0.6.

2.5 Modifications to the Moving Mesh Method for the Solution of the Porous Medium Equation

The first modification is the implementation of the boundary movement. To achieve this, we simply add a subroutine to the code which determines the interface movement by approximating (3.2) by the trapezium rule method. This is done every time before a mesh iteration in the alternating solution procedure. The inner derivatives are determined by a linear extrapolation method. The extrapolation values are evaluated at the element centers.

For a simple solution, such as the selfsimilar test solution in closed form, we could simply use any of the monitor functions presented in Section 2.2. To be able to resolve the boundary movement, especially when it should be zero, we modify the *jump* monitor function. We define

$$g_{\mathrm{PM}} = 0.6 \times g_{\mathrm{JMP}} + 0.4 \times \left(\frac{x - (s_+ + s_-)/2}{s_+ - s_-} \right)^6 \frac{\max |u_x|^2}{1/M + |u_x|^2}. \quad (2.10)$$

Again, some additional scaling procedures are not considered in this definition. The *PM* monitor function has no specific interpretation. It is constructed to create a strong concentration of gridpoints at the boundary if the solution should be flat there, compared to other parts of the domain.

The two modifications described so far are sufficient to solve for

easy solutions, like (3.3) or the waiting time solution with zero initial interface speed.

3 Model Problems

3.1 The Viscous Burgers Equation

We consider the viscous Burgers equation

$$\begin{aligned} u_t + uu_x - \nu u_{xx} &= 0, \quad \text{in } (0,1) \times (0,T) \\ u(0,t) &= a(t), \\ u(1,t) &= b(t), \\ u(x,0) &= u_0(x). \end{aligned}$$

The problem we consider is when $a(t) = b(t) = 0$ and

$$u_0(x) = \sin(2\pi x) + \frac{1}{2}\sin(\pi x),$$

which is shown in Figure 2. This benchmark was chosen because it does not require a well-adapted initial mesh.

3.2 The Porous Medium Equation

We consider the following special case of the porous medium equation:

$$\begin{aligned} u_t &= (uu_x)_x \quad \text{in } \mathbf{R} \times [0,T] \\ u(0) &= u_0 \geq 0, \end{aligned} \tag{3.1}$$

where u_0 has compact support. It arises as a model for many physical phenomena, such as the spreading of a thin film of liquid under gravity or the percolation of gas through a porous medium. For further information see Lacey et al. (1982) and references therein. Here, we summarize those results which are used in this paper.

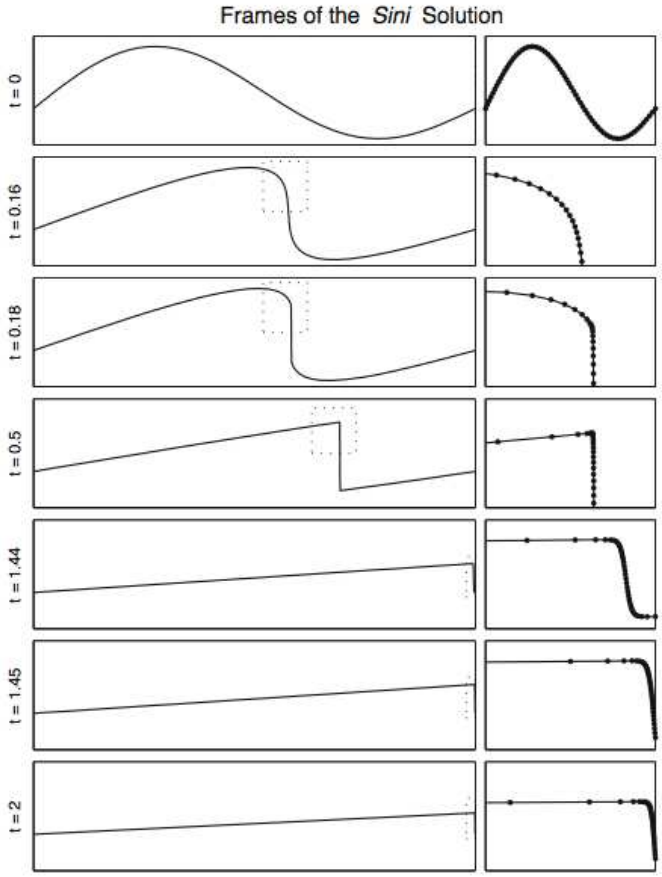

Figure 2: Frames of a solution of the viscous Burgers equation with $\nu = 10^{-3}$.

The porous medium equation (3.1) has a unique weak solution, i.e., a solution in the distributional sense. If the support of the initial condition u_0 is compact then the support of $u(\cdot, t)$ is compact for all t. If the solution u is positive in the interval $(s_-(t), s_+(t))$ and zero outside then the interfaces $s_\pm$ move at speeds

$$\frac{\mathrm{d}s_-}{\mathrm{d}t} = -\lim_{x \to s_- +} u_x(x) \quad \text{and} \quad \frac{\mathrm{d}s_+}{\mathrm{d}t} = -\lim_{x \to s_+ -} u_x(x). \tag{3.2}$$

Note that for the MMPDE method equation (3.2) is used to determine the boundary movement of the computational domain. For the SGWMFE method, this condition is not necessary as the boundary movement is part of the solution of the SGWMFE method. The SGWMFE methods result in a set of two PDEs at each node, one for the value of u and one for the positions of the nodes in the x-axis. This is the case also for the boundary nodes thus all that is necessary to apply at the boundary is the zero boundary condition for u and the nodes are allowed to freely move in the x-axis. An alternative way to determine the interface is to use the fact that the mass $\int_{\mathbf{R}} u(x)\,dx$ and the center of gravity $\int_{\mathbf{R}} xu(x)\,dx$ are conserved by the solution of (3.1). This approach turned out to be very unstable when used on top of the MMPDE method.

Next, we review some classes of solutions that we will test our methods on. The first is a family of similarity solutions, the Barenblatt solutions,

$$u(x,t) = \begin{cases} \frac{1}{6}t^{-1/3}(a^2 - x^2 t^{-2/3}), & |x| \le at^{1/3}, \\ 0 & |x| \ge at^{1/3}, \end{cases} \tag{3.3}$$

where a is a positive constant. This solution is plotted for several times in Figure 3.

The porous medium equation possesses solutions for which an interface can remain fixed for a finite time and then start moving. The time

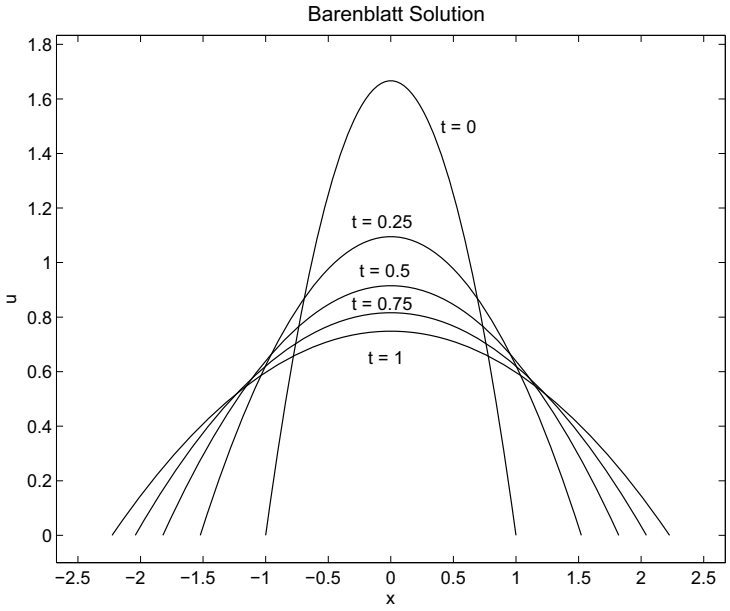

Figure 3: Frames of a Barenblatt solution of the porous medium equation with time shifted by 0.1, and $a = (0.1)^{-1/3}$.

t^* for which the interface remains stationary is called the waiting time. Solutions which exhibit such a behaviour are called waiting time solutions. The following facts are due to Lacey (1983). Suppose the initial condition has an interface at $x = x_0$ and the solution is positive to its left. Let $\alpha = \lim_{x \to x_0-} u_0(x)/(x - x_0)^2$, $\beta = \sup_{x<x_0} u_0(x)/(x - x_0)^2$, and let $t_\gamma = 1/(6\gamma)$ for all $\gamma > 0$. Then $t_\beta \leq t^* \leq t_\alpha$ and if $\beta = \alpha$, we know the exact waiting time. An example of an initial condition which satisfies this condition is $u_0(x) = \chi_{[-1,1]}(x) \sin(\pi(x + 1)/2)^2$. In the case $t_\alpha = t^*$, it is furthermore known that this interface is continuously differentiable, in particular, it starts moving at zero initial speed. We call this solution the *First Waiting Time solution*. Several frames of a numerical solution (using the MMPDE method) are plotted in Figure 4.

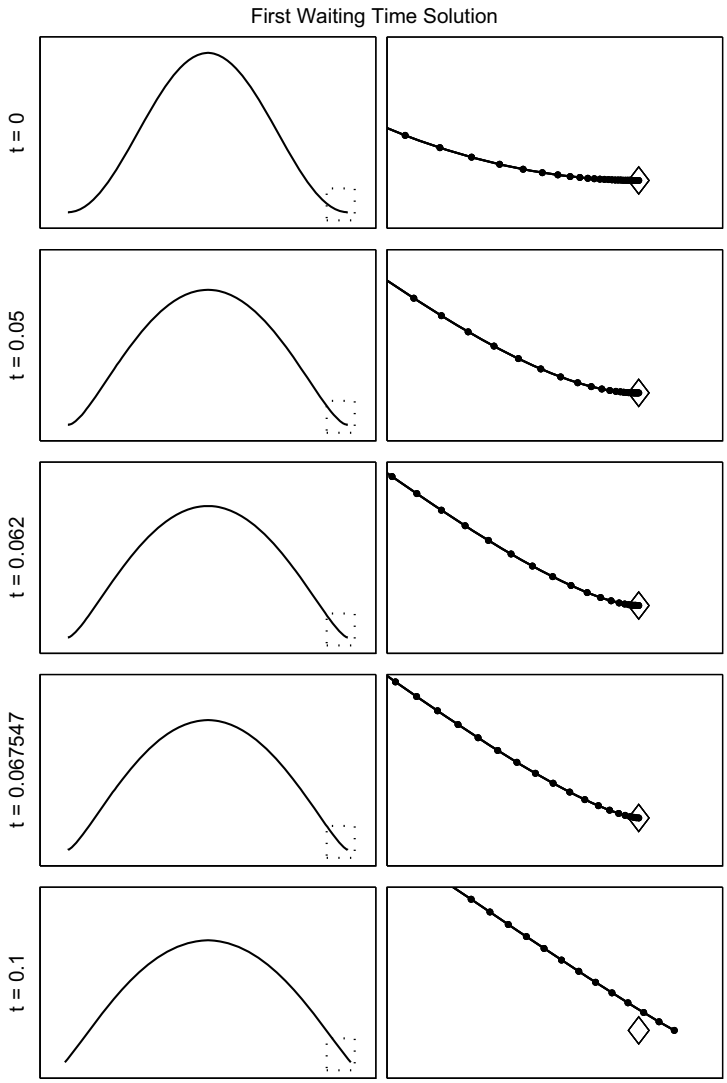

Figure 4: Frames of the First Waiting Time solution of the porous medium equation.

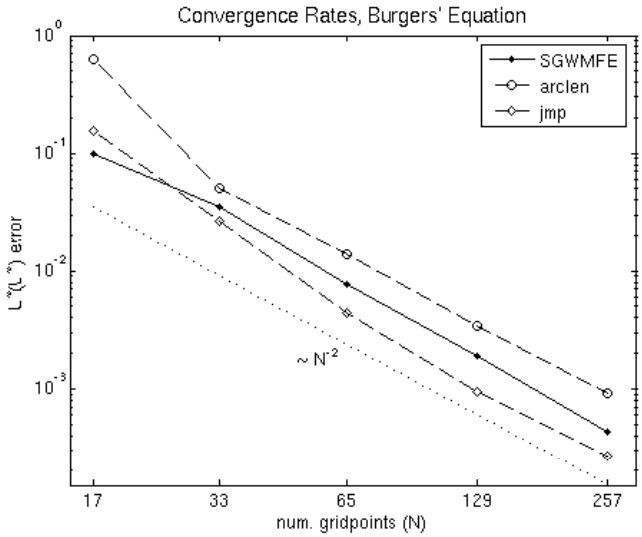

Figure 5: Convergence in $L^\infty(L^\infty)$ of MMPDE vs. SQWMFE solving Burgers' equation with $\nu = 10^{-3}$.

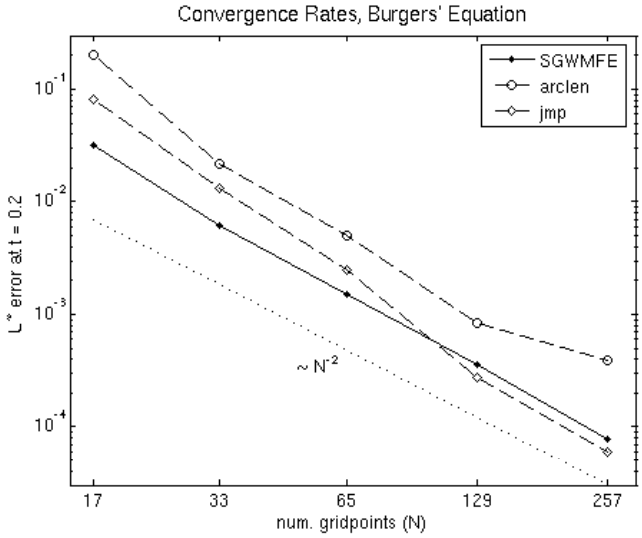

Figure 6: Convergence in L^∞ at a fixed time $t = 0.2$ of MMPDE vs. SQWMFE solving Burgers' equation with $\nu = 10^{-3}$.

4 Numerical results

The notation in the figures below is as follows: SGWMFE denotes computations with the string gradient weighted moving finite element method (see Section 1). Solutions computed by the MMPDE method (see Section 2) are denoted by the name of the monitor function used: 'arclen' for the arclength monitor function, 'jmp' for the second derivative-based monitor and 'pmmon' for the monitor which was specifically constructed for the porous medium equation. In Section 4.3, we only use the 'pmmon' monitor function. There, the notation for MMPDE computations depends on whether the interface is completely free ('pmmon') or is restricted to move on away from the mass ('pmmon, eb'). Compare also Section 2.5.

4.1 Benchmark 1: Nonlinear advection

In our first benchmark, we solve the viscous Burgers equation, described in Section 3.1. Frames of the solution using the MMPDE method are shown in Figure 2. We see from Figure 5 that the MMPDE method and the SGWMFE method are comparable in order of accuracy and both show a nearly second order accuracy taking the infinity norm over all integration time steps. In Figure 6 with the infinity norm at the particular time $t = 0.2$ similar slopes for the convergence rates are observed however one can see that the SGWMFE produces more accurate results particularly when less nodes are used, which also appears in the following two benchmark solutions.

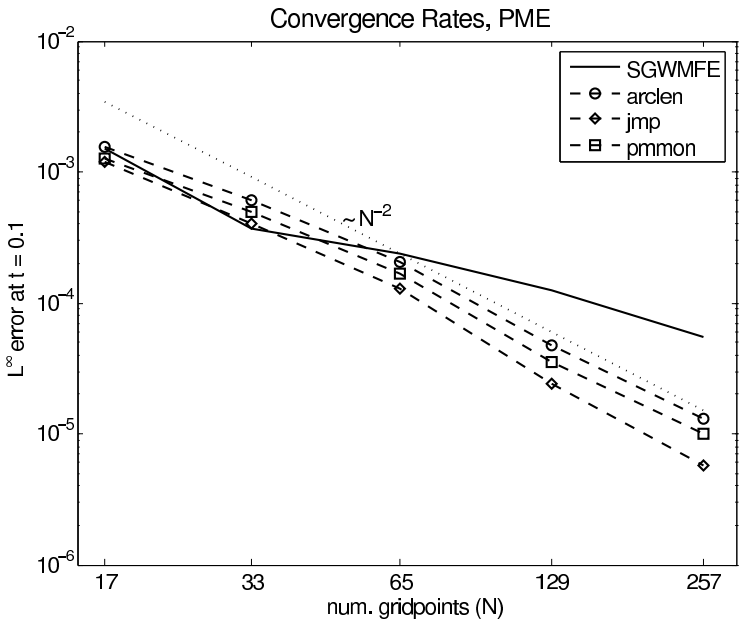

Figure 7: L^∞ errors in numerical solutions of the Porous Medium Equation at time $t = 0.1$.

4.2 Benchmark 2: Barenblatt solutions of the porous medium equation

Figures 7 (short time integration) and 8 (long time integration) show plots of the convergence rates relative to the number of nodes used. Both the MMPDE and the SGWMFE methods behave as expected for the smooth similarity solution. Comparable results (for the long time integration) were also obtained in Baines et al. (2003) for their moving mesh method.

4.3 Benchmark 3: A waiting time solution of the porous medium equation

In this benchmark, we examine whether the two numerical schemes are capable of reproducing the waiting time solution described in Section

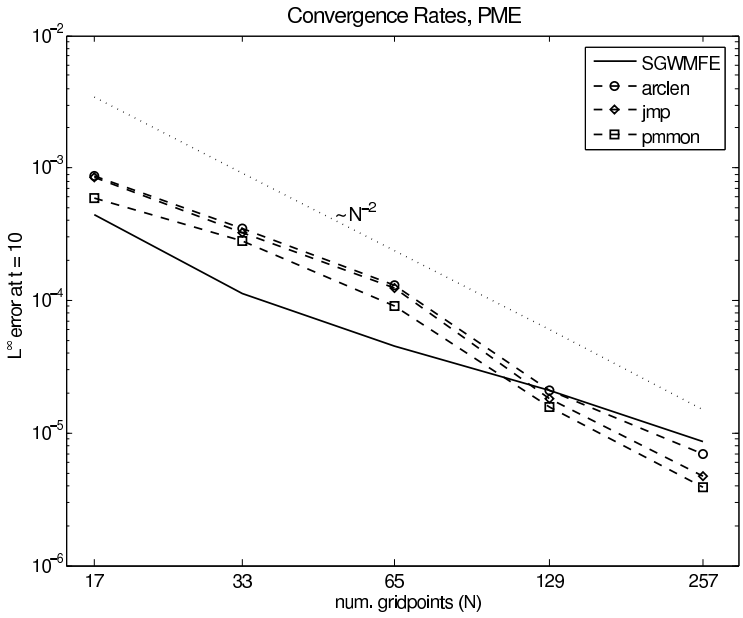

Figure 8: L^∞ errors in numerical solutions of the Porous Medium Equation at the final time $t = 10$.

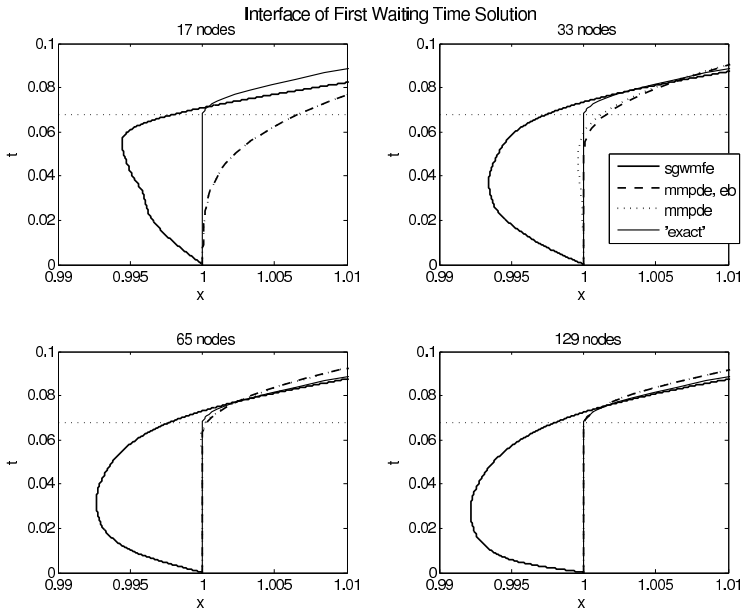

Figure 9: Right interface of the waiting time solution of the porous medium equation, using different numerical schemes.

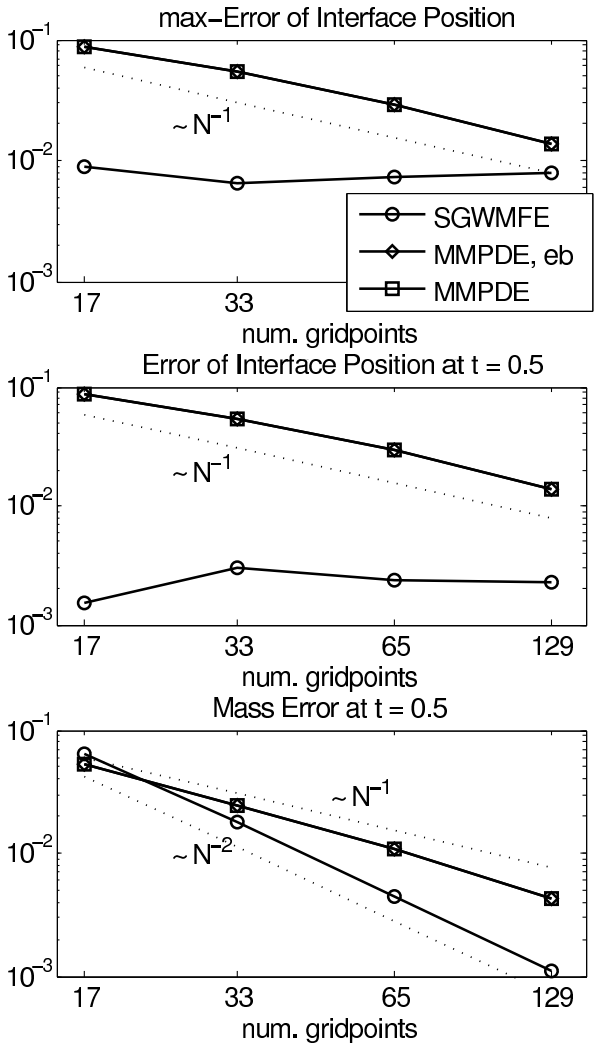

Figure 10: Convergence rates for: the maximum error ($0 \leq t \leq 1/2$) of the right interface approximations of the waiting time solution, the error of the right interface at the particular time $t = 1/2$, and the error in the mass integral of the solution at time $t = 1/2$.

3.2. Here, we are interested in the correct behaviour of the interface and subsequently the long time error.

Due to the lack of an analytical solution, we compare with a numerical solution, using the MMPDE approach with 512 mesh points. We use the shortcuts `sgwmfe`, `mmpde-eb` and `mmpde` to respectively denote the numerical solutions using the string gradient weighted moving finite element method, the MMPDE method with enforced positive interface speed and the pure MMPDE method. Figure 9 shows plots of the interfaces for the waiting time solution and different mesh sizes. The convergence rates of the approximations of the interfaces are plotted in Figure 10.

The results in Figures 9 show a number of things. One can observe that clearly SGWMFE does not keep its boundary node on the point $x = 1$ as does the enforced boundary implemented in the MMPDE method, this is because the boundary nodes in the SGWMFE method were not enforced to only move in one direction. This however does not imply a poor solution at the boundary since the solution is zero beyond the location of the boundary point until the boundary begins to move. If you look very closely, you will see that MMPDE without the enforced boundary also does this to a much smaller extent, see the plot with 33 nodes in Figure 9. From Figure 9 one can see that SGWMFE predicts the critical waiting time sooner than does the MMPDE method. This problem could be removed as with the MMPDE method by enforcing the boundary to only move in one direction. For the results shown in this paper this was not applied nor are the nodes initially optimally placed, so regardless of the initial positions you can see that in time the solutions from SGWMFE are more accurate than the MMPDE method once the boundary begins to move (beyond the waiting time). It is clear that for this start up problem an enforced velocity boundary

condition on the MMPDE method is much better at predicting the waiting time solution given that one uses a sufficient number of nodes.

Analyzing the MMPDE method on its own, it is clear that with the enforced velocity boundary condition the solutions are much more accurate, where as the MMPDE method without the imposed boundary undershoots the waiting time.

If we now focus our attention on Figures 10 we see that the accuracy of the numerical solutions behave quite differently in the waiting time interval than it does after the boundary is moving, once again concluding that as time progresses SGWMFE becomes more accurate than the MMPDE method. It should be noted that for this start up problem the time steps were chosen initially for both methods. Generally the time steps for SGWMFE are chosen dynamically, though for the MMPDE method implemented here the time steps are chosen initially since the interest here was on the waiting time solutions, and for comparison purposes the same time steps had to be imposed for the SGWMFE as well. Because of this, smaller time steps were chosen and the solution is sometimes more accurate than would have been required by the error tolerance.

5 Summary

From the numerical results shown in this report we conclude that for the viscous Burgers equation the SGWMFE/GWMFE and the MMPDE methods produce comparable results. Depending on the monitor function used for the MMPDE to solve Burgers' equation, the MMPDE can produce results that are slightly better or worse than those of SGWMFE/GWMFE. For both porous medium equation examples studied in this report it is observed that the longer time results obtained using

SGWMFE/GWMFE are more accurate than those produced using the MMPDE method. However for the startup problem of obtaining the critical time $t*$ (the waiting time) the MMPDE method predicts this more accurately.

Acknowledgement: I would like to thank Christoph Ortner for his efforts and the use of his codes to obtain the MMPDE simulations, as well as for parts of his MSc thesis which contributed to the MMPDE section of this report.

References

Baines M. (1994). *Moving Finite Elements*. Oxford University Press Inc, New York.

Baines M., Hubbard M., and Jimack P. (2003). *A Moving Finite Element Method using Monitor Functions*. Technical report, School of Computing, University of Leeds, Leeds. 2003.04.

Beckett G., Mackenzie J., Ramage A., and Sloan D. (2001). *On the Numerical Solution of One-Dimensional PDEs Using Adaptive Methods Based on Equidistribution*. *Journal of Computational Physics*, **167**:372–392.

Beckett G., Mackenzie J., Ramage A., and Sloan D. (2002). *Computational Solution of Two-Dimensional Unsteady PDEs Using Moving Mesh Methods*. *Journal of Computational Physics*, **182**:478–495.

Blom J.G. and Verwer J.G. (1989). *On the Use of the Arclength and Curvature Monitor in a Moving Grid Method which is Based on the Method of Lines*. Technical report, Tech. Report NM-N8902, CWI, Amsterdam.

Cao W., Huang W., and Russell R.D. (1999). *An r-Adaptive Finite Element Method Based Upon Moving Mesh PDEs*. *Journal of Computational Physics*, **149**:221–244.

Carlson N.N. and Miller K. (1998a). *Design And Application Of A Gradient-Weighted Moving Finite Element Code I: In One Dimension*. *SIAM J. Sci. Comput.*, **19**(3):728–765.

Carlson N.N. and Miller K. (1998b). *Design And Application Of A Gradient-Weighted Moving Finite Element Code II: In Two Dimensions*. *SIAM J. Sci. Comput.*, **19**(3):766–798.

Hairer E. and Wanner G. (1991). *Solving Ordinary Differential Equations II – Stiff and Algebraic Problems*. Springer Series in Computational Mathematics. Springer Verlag, Berlin Heidelberg.

Huang W., Ren Y., and Russell R.D. (1994). *Moving Mesh Partial Differential Equations (MMPDEs) Based on the Equidistribution Principle*. *SIAM Journal of Numerical Analysis*, **31(3)**:709–730.

Huang W. and Russell R.D. (1997). *Analysis of Moving Mesh Partial Differential Equations with Spatial Smoothing*. *SIAM Journal of Numerical Analysis*, **34(3)**:1106–1126.

Huang W. and Russell R.D. (1999). *Moving Mesh Strategy Based on a Gradient Flow Equation for Two-Dimensional Problems*. *SIAM Journal of Scientific Computing*, **20(3)**:998–1015.

Jeffreys H. and Jeffreys B.S. (1946). *Methods Of Mathematical Physics*. Cambridge University Press, Cambridge.

Jimack P. and A.J. W. (1991). *Temporal Derivatives in the Finite-Element Method on Continuously Deforming Grids*. *SIAM Journal of Numerical Analysis*, **28(4)**:990–1003.

Lacey A. (1983). *Initial Motion of a Free Boundary*. *IMA Journal of Applied Mathematics*, **31**:113–119.

Lacey A., Ockendon J., and Tayler A. (1982). *"Waiting Time" Solutions of a Nonlinear Diffusion Equation*. *SIAM Journal of Applied Mathematics*, **42(6)**:1252–1264.

Miller K. (1981). *Moving Finite Elements II*. *SIAM J. Numer. Anal.*, **18**(6):1033–1057.

Miller K. (1997). *A Geometrical-Mechanical Interpretation Of Gradient-Weighted Moving Finite Elements*. *SIAM J. Numer. Anal.*, **34**(1):67–90.

Miller K. and Miller R.N. (1981). *Moving Finite Elements I*. *SIAM J. Numer. Anal.*, **18**(6):1019–1032.

Ortner C. (2003). *Moving Mesh Partial Differential Equations*. MSc. thesis, Oxford University Computing Laboratory, Oxford.

Wacher A. (2004). *String Gradient Weighted Moving Finite Elements For Systems of Partial Differential Equations*. DPhil thesis, Oxford University Computing Laboratory, Oxford.

Wacher A., Sobey I., and Miller K. (2003). *String Gradient Weighted Moving Finite Elements For Systems of Partial Differential Equations I*. *Numerical Analysis internal report 03/15*.

Wathen A. and Baines M. (1985). *On The Structure Of The Moving Finite-Element Equations*. *IMA J. Numer. Anal*, **5**:161–182.