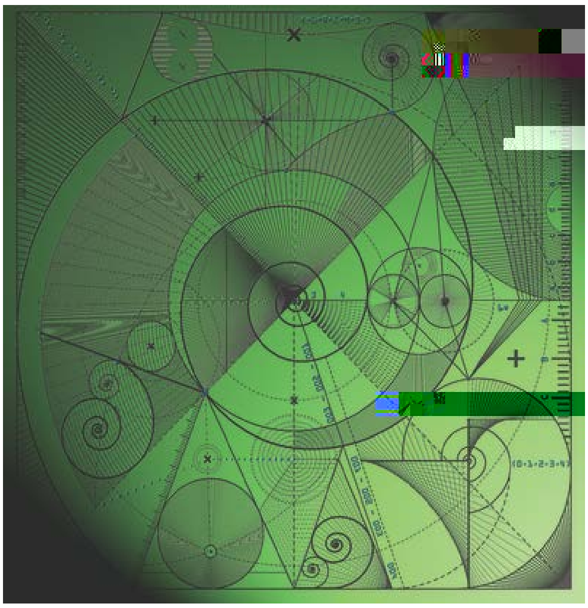


A nite di

solution. Numerical results for all our examples are presented in §4, and some conclusions are presented in §5.

We remark finally that our investigation is confined to initial-boundary-value problems for which the solution $u(x,t)$ is one-signed in the interior of the domain, which is sufficient for the validity of the method.

2. Conservation-based moving mesh methods

Let $u(x,t)$ be a positive solution of the generic time-dependent scalar PDE

$$\frac{\partial u(x,t)}{\partial t} = \mathcal{L}\,u(x,t), \qquad t > t^0, \quad x \in (a(t), b(t)), \tag{1}$$

where $\mathcal{L}$ is a purely spatial differential operator. In all of our examples we have a moving boundary at $x = b(t)$ at which we impose the following boundary conditions

$$u(b(t),t) = 0, \tag{2}$$

$$u(b(t),t)\frac{db}{dt} = 0. \tag{3}$$

The initial condition is

$$u(x,t^0) = u^0(x), \quad x \in (a \text{[illegible]})).$$

We introduce a time-dependent space coordinate $\tilde{x}($[illegible]h coincides instantaneously with the fixed coordinate x. Consider two such coordinat[illegible] and $\tilde{x}(x_2;t)$, in $(a(t),b(t))$, abbreviated to $\tilde{x}_1(t)$ and $\tilde{x}_2(t)$. The rate of change of the [illegible]n the subinterval $(\tilde{x}_1(t), \tilde{x}_2(t))$ is given by Leibnitz' Integral Rule in the form

$$\frac{d}{dt}\int_{\tilde{x}_1(t)}^{\tilde{x}_2(t)} u(s,t)\,ds = \int_{\tilde{x}_1(\text{[illegible]}}^{\tilde{x}_2(\text{[illegible]}} \left(\text{[illegible]}\,t) + \frac{\partial}{\partial s}(u(s,t)v(s,t)) \right) ds, \tag{4}$$

where

$$v(x,t) \text{[illegible]}_{=x} \tag{5}$$

is a local velocity. We denote the total m[illegible]

$$(t) := \int_{a(\text{[illegible]}} \text{[illegible]})\,dx. \tag{6}$$

2.1. A method based on preservation of p[illegible]asses

We begin by describing a solution meth[illegible]oblems that conserve the total integral (global mass) of the solution, i.e. for which [illegible]) and the Eulerian conservation law,

$$\frac{\partial u(x,t)}{\partial t} + \frac{\partial}{\partial x}(u(x,t)v(x,t)) = 0. \tag{8}$$

From (8) and the PDE (1) we have

$$L\,u(x,t) + \frac{\partial}{\partial x}(u(x,t)v(x,t)) = 0, \tag{9}$$

which, given $u(x,t$

3.1. The Porous Medium Equation

The PME is the simplest nonlinear diffusion problem which arises in a physically natural way, describing processes involving fluid flow, heat transfer or diffusion. It also occurs in mathematical biology and other fields [24]. We assume the initial data is symmetrical about its centre of mass, taken to be the origin, in which case the PME takes the form

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(u^n \frac{\partial u}{\partial x}\right); \qquad t > t^0; \; x \in (-b(t), b(t));$$

with $u(-b(t);t) = u(b(t);t) = 0$ and $u(-b(t);t)\,db/dt = 0$. For this problem the total mass (6) is conserved and the centre of mass is fixed in time [24], from which it follows that the solution retains the symmetry of the initial data for all time. We therefore model only half of the region, i.e. $x(t) \in [0; b(t)]$, with $a(t) = 0$ as the anchor point for all t. For the half problem we have

$$\frac{\partial u}{\partial x} = 0 \qquad \text{at} \quad x = 0; \tag{24}$$

by symmetry. From (10) the velocity is given by

$$v(x;t) = -\frac{1}{u(x;t)}\int_0^x \frac{\partial}{\partial s}\left(u(s;t)^n \frac{\partial u}{\partial s}\right) ds = -u^{n-1}\frac{\partial u}{\partial x} = -\frac{1}{n}\frac{\partial (u^n)}{\partial x}; \quad t > t^0; \; x \in [0; b(t)). \tag{25}$$

Given X_j^m and U_j^m, $j = 0;1;\ldots;N$, $m = 0;1;2;\ldots$, the finite difference algorithm of §2.1 is used to calculate the velocity V_j^m at each node j, $j = 0;1;\ldots;N$, then the new nodal positions X_j^{m+1}, and finally the approximate solution U_j^{m+1}. A standard discretisation of the velocity (25) at interior nodes is

$$V_j^m = -\frac{1}{n}\left(\frac{(U_{j+1}^m)^n - (U_{j-1}^m)^n}{X_{j+1}^m - X_{j-1}^m}\right); \qquad j = 1;2;\ldots;N-1;$$

which, although of second order on a uniform mesh, is only first order discretisation on a non-uniform mesh. An approximation which is second order on a non-uniform mesh (i.e. exact for quadratics) uses all three values U_{j-1}^m, U_j^m and U_{j+1}^m, and is

$$V_j^m = -\frac{1}{n}\left(\frac{\frac{1}{\Delta_+ X_j^m}\frac{\Delta_+(U_j^m)^n}{\Delta_+ X_j^m} + \frac{1}{\Delta_- X_j^m}\frac{\Delta_-(U_j^m)^n}{\Delta_- X_j^m}}{\frac{1}{\Delta_+ X_j^m} + \frac{1}{\Delta_- X_j^m}}\right); \qquad j = 1;2;\ldots;N-1; \tag{26}$$

where

$$\Delta_+(\,)_j = (\,)_{j+1} - (\,)_j \quad \text{and} \quad \Delta_-(\,)_j = (\,)_j - (\,)_{j-1}$$

(see [21]). We note that equation (26) is an inversely weighted sum, or linear interpolation, of the gradients $\Delta(U_j^m)^n / \Delta X_j^m$. The velocity at $x = 0$ is zero and at the moving boundary $x = X_N^m$ the velocity V_N^m is extrapolated by a polynomial approximation using three adjacent points. The new mesh is obtained at time $t^{m+1} = t^m + \Delta t$ by the explicit Euler time-stepping scheme (14).

The updated approximate solution U_j^{m+1} is given by (15), $j = 1;\ldots;N-1$. At $j = 0$ the approximate solution U_0^{m+1} is calculated using (26) with $X_{-1} = -X_1$, approximating the boundary condition (24). At the outer boundary, $U_N^{m+1} = 0$ from (2). Results are presented in §4.

3.2. Richards' Equation

Richards' equation is a nonlinear PDE which models the movement of moisture in an unsaturated porous medium [23]. In the present paper we model a particular form of Richards' equation, where the solution describes liquid flowing downwards through an unsaturated porous medium, making it applicable to the tracking of a contaminated liquid. The equation is of the form for $x \in [0;1]$, as in [9], giving initial total mass$(0) = 1=$

3.4. The Crank-Gupta problem with a modi ed boundary condition

There is no known analytical solution for the Crank-Gupta problem although approximate solutions have been given in [10]. Hence, in order to compare results to an exact solution we have modelled the Crank-Gupta PDE with a modi ed boundary values of $\hat{N}$, we denote the points of the mesh for a particular value of $\hat{N}$ by $x_{j;\hat{N}}(t)$, $j = 0, \dots, N$. We then compute both $x_{2^{\hat{N}-1}i;\hat{N}}(t)$ and $u_{2^{\hat{N}-1}i;\hat{N}}(t) \approx u(x_{2^{\hat{N}-1}i;\hat{N}}(t); t)$ for each $i = 0, \dots, 10$; this new notation allows comparison of $x_{j;\hat{N}}(t)$ and $u_{j;\hat{N}}(t)$ at eleven different points, determined by $j = 2^{\hat{N}-1}i$, $i = 0, \dots, 10$, for various $\hat{N}$. Where possible we compare the numerical outcomes with the exact solution and boundary position. When such a solution is not known, we compare with numerical results determined using other methods. In each case we denote our reference solution by $\bar{u}(x;t)$, and our reference boundary position by $\bar{x}(t)$.

Recalling that we have used explicit Euler time-stepping, in order to balance the spatial and temporal errors, we take $\Delta t = O\left(1/N^2\right)$, anticipating that the pointwise errors $|\bar{x}(t) - x_{N;\hat{N}}(t)|$ and $|\bar{u}(x_{2^{\hat{N}-1}i;\hat{N}}(t); t) - u_{2^{\hat{N}-1}i;\hat{N}}(t)|$ will decrease as $\hat{N}$ increases, for each $i = 0, \dots, 10$.

As a measure of the errors, we calculate the ℓ_2 norm of the error in our solution, and the relative error of our boundary position, as defined by

$$E^u_N := \sqrt{\frac{\sum_{i=0}^{10} |\bar{u}(x_{2^{\hat{N}-1}i;\hat{N}}(T); T) - u_{2^{\hat{N}-1}i;\hat{N}}(T)|^2}{\sum_{i=0}^{10} |\bar{u}(x_{2^{\hat{N}-1}i;\hat{N}}(T); T)|^2}}, \qquad E^x_N := \frac{|\bar{x}(T) - x_{N;\hat{N}}(T)|}{|\bar{x}(T)|},$$

for $\hat{N} = 1, 2, 3, 4, \dots$ (i.e. $N = 10, 20, 40, 80, \dots$). We investigate the hypothesis that

$$E^u_N$$

and the exact boundary position, is

$$\bar{x}(t) = b(t) = t^{1/(n+2)}\sqrt{\frac{2(n+2)}{n}}.$$

As stated above, to balance the spatial and temporal errors we use $\Delta t = O(1/N^2)$, precisely $\Delta t = 0.4 \times 4^{-\hat{N}}$. Convergence results for $n = 1$ are shown in Table 1. We see that E^u_N and E^x_N decrease as N increases. This suggests that as the number of nodes increases our approximations to both the solution and the boundary position are converging. The p and q values presented strongly indicate second-order convergence of both the numerical solution and numerical boundary position.

N	E^u_N	p_N	E^x_N	q_N
10	7.715×10^{-3}	-	1.451×10^{-3}	-
20	1.941×10^{-3}	2.0	3.066×10^{-4}	2.2
40	4.976×10^{-4}	2.0	7.138×10^{-5}	2.1
80	1.259×10^{-4}	2.0	1.730×10^{-5}	2.0
160	3.166×10^{-5}	2.0	4.262×10^{-6}	2.0
320	7.937×10^{-6}	2.0	1.058×10^{-6}	2.0

Table 1: Relative errors, E^u_N and E^x_N, for the porous medium equation with $n = 1$.

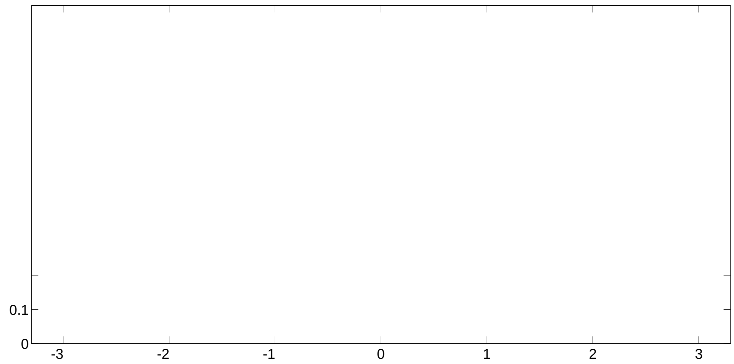
The results from the self-similar solutions for $n = 1, 2, 3$ and $N = 20$ are given in Figures 1–3. In each case we see that with only twenty nodes in our mesh, the boundary position (Figures 1(b)–3(b)) is computed very accurately (better than 1% relative error at $t = 5$ in each case). From (47) we note that the exact solution for $n = 2, 3$ has a steep gradient at the boundaries, as can be seen in Figures 2(a) and 3(a). Figures 1(c)–3(c) show exactly how the mesh moves. We observe a smooth even spread of the nodes, without mesh tangling, in all three cases.

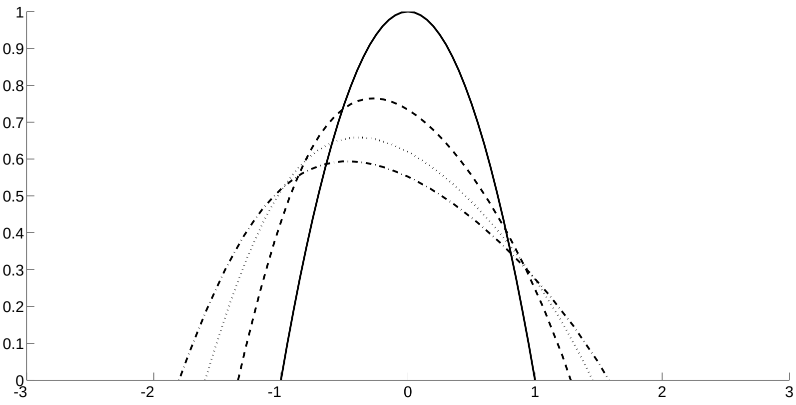
4.2. Richards' Equation

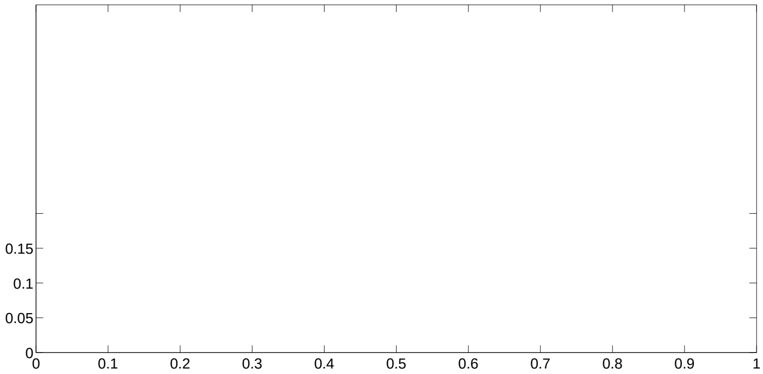
In this section we present results from applying the moving mesh method to Richards' equation, as described in §3.2. To test that the numerical solution from the moving mesh method converges we compare the solution with that from a very fine fixed mesh. All numerical results presented here are for $n = 3$. In the absence of an exact reference solution we do not compare the position of the boundary.

We solve for $t \in [0, 0.5]$ and compute results for $N = 10 \times 2^{\hat{N}-1}$, $\hat{N} = 1, \ldots, 4$. We compare the numerical solutions with a numerical solution calculated by solving Richards' equation on the fixed mesh $\bar{x}_{\bar{j}} \in [-4, 4]$, $\bar{j} = 0, 1, \ldots, 10000$, which is given by

$\bar{u}_{\bar{j}+\frac{1}{2}}$


0.1
0
-3
-2
-1
0
1
2
3


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-3
-2
-1
0
1
2
3


0.15
0.1
0.05
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

and compute results for $N = 10 \cdot 2^{\hat{N}-1}$, $\hat{N} = 1, \ldots, 6$. We compare the numerical outcomes with the exact solution (38), at $t = 0.1$,

$$\bar{u}(x_{j,\hat{N}}(0.1), 0.1) = e^{x_{j,\hat{N}}(0.1) - 0.9} - x_{j,\hat{N}}(0.1) - 0.1.$$

To balance the spatial and temporal errors we use $\tau = O\left(1/N^2\right) = 0.02 \cdot 4^{-\hat{N}}$.

Numerical results are shown in Table 4. We see that E^u_N decreases as N increases, and the

N	E^u_N	p_N
10	$7.581 \cdot 10^{-3}$	-

Crank-Gupta problem, which models oxygen-diusion through tissue. We examined the accuracy in all cases and found that the numerical solution converged with roughly second-order accuracy. Furthermore, for the Crank-Gupta problem, we found that preservation of mass fractions can lead to higher resolution at the boundary, which is desirable.

Throughout this paper we have used an explicit Euler time-stepping scheme. Other explicit time-stepping schemes we experimented with are the higher order methods built into Matlab (ODE23, ODE45, ODE15s); see [15] for details. There was little di erence in the results from all the Matlab solvers, indicating that none of the problems lead to a sti system of ODEs for the $\tilde{x}_j(t)$. We found that all the time-stepping schemes produced accurate and stable results, with no mesh tangling, provided that su ciently small time-steps were taken. It has been shown in [2] that the PME can also be solved by this moving mesh method with a semi-implicit time-stepping scheme using larger time steps.

We conclude that this moving mesh ap433749(i)-0.240155(n)0.433749(g)0.193594(e)0.443552(s)-0.355331

References

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

[2] Baines, M.J. and Lee, T.E. (2014) A large time-step implici