A MATLAB Method of Lines Template for Evolution Equations

H.S. Lee[a], C.J. Matthews[a], R.D. Braddock[a], G.C. Sander[b] and F. Gandola[a]

[a]Faculty of Environmental Sciences, Griffith University, Nathan, QLD, 4111

[b]Department of Civil and Building Engineering, Loughborough University, Leicestershire, LE11 3TU, UK

Abstract: Many environmental problems involve diffusion and convection processes, which can be described by Partial Differential Equations (PDEs). This paper will describe the development of a MATLAB template that generates a numerical solution to PDEs using the Method of Lines. The template will be applied to various problems within soil physics to demonstrate the versatility of the method. In particular, the template will generate a solution for one-dimensional infiltration, and two-dimensional infiltration over a complex geometry. Where possible, the results from the template will be compared against analytical solutions to determine the accuracy of the numerical solution. In addition, the paper will provide a discussion on possible extensions to the template and future directions.

Keywords: *Method of Lines, infiltration, soil physics, numerical solution.*

1. INTRODUCTION

Many environmental problems involve diffusion and convection processes and generally result in a partial differential equation (PDE), or evolution equation, which is parabolic in nature. Generally, such equations are difficult to solve analytically and numerical simulations are used to understand the particular problem more fully. Finite-difference and finite-element methods can be applied to these non-linear evolution models. Often their application leads to the need to solve large sets of non-linear algebraic equations, and considerable mathematical skill may be needed in implementing the technique (Ames, 1992).

The numerical method used here is the Method of Lines (MoL), which involves discretising the spatial domain while keeping the time component continuous. This replaces the PDE with a vector system of ordinary differential equations (ODEs), for which efficient and effective integrating packages have been developed (Schiesser, 1991; Shampine and Reichelt, 1994). The MATLAB package has strong vector and matrix handling capabilities, a good set of ODE solvers, and an extensive functionality which can be used to implement the MoL (Shampine and Reichelt, 1994).

Many of these evolution equations are similar in structure regardless of the dimension and the functional representations of the underlying diffusive and convective processes. A template has been developed in MATLAB to handle evolution equations, using the MoL and the extensive functionality of the language (Lee et al., 1998). The template has many inbuilt housekeeping features related to grid generation and the creation of the system of ODEs. Further, the coding of the essential ODEs is simplified, as the coding required closely follows the mathematical description of the problem.

This paper aims to show how this template can be used to handle evolution equations. The template is described and illustrated by application to problems arising in soil physics with complex mathematical descriptions of the diffusive and convective processes. The template is applied to problems in one and two dimensions, and to a problem with a complex geometric domain.

2. RICHARDS' EQUATION

The two-dimensional hydraulic pressure (h) form of Richards' equation is (Hillel, 1980)

$$C_w \frac{\partial h}{\partial t} = \frac{\partial}{\partial x}\left(K\frac{\partial h}{\partial x}\right) + \frac{\partial}{\partial z}\left(K\frac{\partial h}{\partial z}\right) - \frac{\partial K}{\partial z}, \qquad (1)$$

where h is the hydraulic pressure, K is the hydraulic conductivity, C_w (= $\partial\theta/\partial h$) is the water capacity, θ is the water content, and the spatial axes are x (horizontal) and z (vertical downwards). The hydraulic functions are usually developed by function fitting to experimental data (Touma et al., 1984). The term $-\partial K/\partial z$ in (1) is a convective term representing the effects of gravity, while the other terms represent non-linear diffusion. The detail of these terms, whilst important in soil physics, is not as important to this paper. In fact, the mathematical functions that represent these terms can be replaced with functions to suit other areas of modelling.

For one-dimensional flow in the vertical direction, (1) reduces to

$$C_w \frac{\partial h}{\partial t} = \frac{\partial}{\partial z}\left(K \frac{\partial h}{\partial z} \right) - \frac{\partial K}{\partial z}. \qquad (2)$$

Sander et al. (1988) provide an analytic solution to the water flow in the vertical direction for a specific form of the hydraulic functions and boundary conditions. This analytic solution is used to test the accuracy of a numerical solution obtained from the MoL to the same problem.

3. REPRESENTATION OF DERIVATIVES

Consider (2) on the grid $z_i = z^* + i\Delta z$, where Δz is fixed, z* is some (arbitrary) starting point, i = 0,1,2…n is a counter, and the grid has n+1 nodes. The vector **h** contains the value for h for each node within the grid. The derivatives $\partial h/\partial z$ at the nodes are calculated using a fourth order finite difference scheme from Schiesser (1991) giving

$$\mathbf{h}_z^d = A_z^d \mathbf{h}, \qquad (3)$$

where

$$A_z^d = \frac{1}{24\Delta z}\begin{bmatrix} -50 & 96 & -72 & 32 & -6 & & \\ -6 & -20 & 36 & -12 & 2 & & \\ 2 & -16 & 0 & 16 & -2 & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & 2 & -16 & 0 & 16 & -2 \\ & & -2 & 12 & -36 & 20 & 6 \\ & & 6 & -32 & 72 & -96 & 50 \end{bmatrix},$$

h_z^d is the vector of $\partial h/\partial z$ values on the grid. Note that A_z^d is a sparse n+1,n+1 matrix and the derivatves towards the boundaries are represented by a series of upwinding and downwinding finite differencing equations. This avoids the use of fictitious points to incorporate boundary conditions directly into the finite differencing scheme.

As given by Schiesser (1991), the technique to incorporate boundary conditions into the above scheme depends on the type of boundary condition involved. For Dirichlet type boundary conditions, the values are assigned to the respective boundary nodes before applying (3), so that the correct values of h are retained. For Neumann type boundary conditions, the conditions are imposed on the boundary nodes in $\mathbf{h}_z^d$ so that the correct first-order derivatives are involved in subsequent calculations. If mixed-type boundary conditions are involved, these are treated in the same manner as for the Neumann-type problem. The finite differencing scheme results in a single differential matrix for a given problem and does not involve any algebraic manipulation to include boundary conditions. Consequently, finite differencing schemes of any desired order are easily implemented within the template.

Note that (3) is strictly used to estimate derivatives for diffusive terms in Richards' Equation. To evaluate convective terms, an upwinding finite difference scheme (A_z^c) is used (Schiesser, 1991). The differential matrix for A_z^c has the same structure as A_z^d except that the central difference equations (from 4th line down) is replaced by the first upwinding difference equation given on the 2nd last line in A_z^d. Using A_z^c, the convective term is calculated as follows

$$\mathbf{K}_z^c = \mathbf{K_h} \square (A_z^c \mathbf{h}), \qquad (4)$$

where $\mathbf{K}_z^c$ is the vector $\partial K/\partial z$ values on the grid, $\mathbf{K_h}$ is a vector dK/dh values and □ is a MATLAB operator for element-by-element multiplication. Note that dK/dh is a function derived from K(h).

Consequently, (2) is discretised to the form

$$\mathbf{C}_w(\mathbf{h}) \square \frac{\mathbf{dh}}{dt} = A_z^d \left(\mathbf{K} \square \mathbf{h}_z^d \right) - \mathbf{K}_z^c, \qquad (5)$$

where $\mathbf{C}_w(\mathbf{h})$ is the vector of values of $C_w(h)$ at each node. (5) is now a system of ODEs, with each ODE representing a node on the grid. An ODE integrator is then used to integrate (5) forward in time. Note that (5) may be stiff requiring a stiff ODE solver which is provided in the MATLAB suite (Shampine and Reichelt, 1994). An initial condition appropriate to the physical problem being considered, is used to start the time integrator.

4. THE TEMPLATE

To solve evolution equations using the MoL, a template is design using MATLAB to automate the solution process. The functionality and matrix-based capabilities of MATLAB is well suited to this task. The conceptual steps of the template are shown on the left-hand side of Figure 1, while the right-hand side shows the corresponding subroutine structure and information pathways for the computer-based implementation.

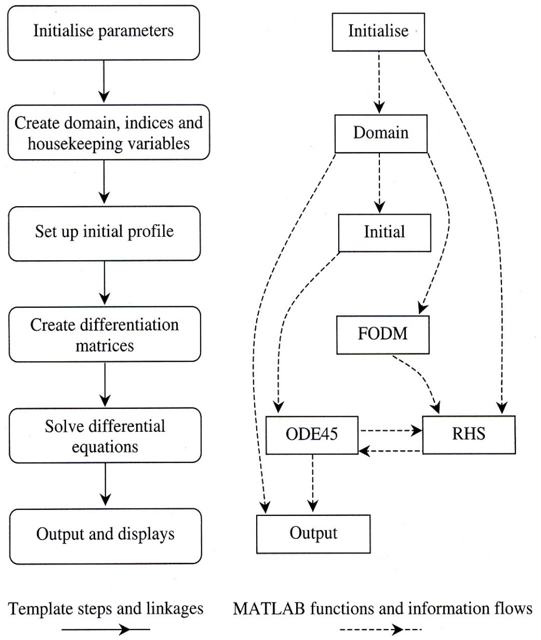
Initialise parameters
Create domain, indices and housekeeping variables
Set up initial profile
Create differentiation matrices
Solve differential equations
Output and displays
Initialise
Domain
Initial
FODM
ODE45
RHS
Output
Template steps and linkages
MATLAB functions and information flows

Figure 1. The template and corresponding MATLAB functions showing the information flows.

To illustrate the conceptual steps in Figure 1, consider the one-dimensional Richards' equation discussed in Section 3. The initialisation step defines the essential parameters for the model e.g. physical parameters (soil parameters and constant rainfall rate) and geometric parameters (total depth and number of nodes). The domain routine generates the spatial lattice, the node numbering system and node indexing vectors, which identify nodes associated with boundaries. This subroutine is particularly simple for the one-dimensional case, but becomes more complex for higher dimensions, particularly where boundaries are not all parallel to the co-ordinate axes. The routine 'initial' sets up the initial values of **h** on the grid. Note that the matrix-based structure of MATLAB enables mathematical formulae to be expressed directly into the code.

The 'FODM' (First-Order Differentiation Matrix) routine accepts information on the lattice structure to generate the matrices A_z^d and A_z^c. The differential matrices, A_z^d and A_z^c, are then passed to the 'RHS' routine, which calculates the right-hand side of (5). Values of **h** are passed from the ODE Integrator to the RHS routine to calculate the right-hand side of (5) giving an approximation for $\partial\mathbf{h}/\partial t$. The $\partial\mathbf{h}/\partial t$ vector is then returned to the ODE solver calculating **h** at the next time step. In general, ODE45 from the MATLAB library is used as the ODE solver in the template, and is based on a fourth and fifth-order pair of Runge Kutta formulae. However, changing the ODE solver within the template is a minor change since all ODE solvers have been standardised input/output within MATLAB.

The last step is to provide a graphical representation of the numerical output, which will usually involve a plot of h over the spatial domain. When an analytical solution is available, this paper will compare the numerical and analytical solutions by calculating the relative error $\varepsilon_i = |V_i - v_i|/V_i$ where V_i is the analytic solution and v_i is the numerical solution at node i. The functionality of the graphics package in MATLAB is large and well suited to obtaining information on various aspects of the output.

5. EXAMPLE WITH ONE-DIMENSIONAL EVOLUTION EQUATION

As an example, the numerical solution from the template was compared against the analytic solution of Sander et al. (1988). The Sander et al (1988) solution applies to constant flux infiltration into a semi-infinite domain. The initial condition is a constant water content that corresponds to a dry soil. For this example, Case 1 from Watson et al. (1995) will be considered.

Within the template, the physical and geometric parameters are initialised in 'Initialise' routine including the total number of nodes, N_t = 101. The domain function accepts the geometric parameters (L and N) and returns the step size Δz and a matrix G, that represents the domain grid. For this case, G will be a column vector of length N_t with the vector entries containing the node numbers. From G, the initial condition (**h0**) and the indexing vectors, **b1** and **b2,** for the top and bottom boundaries are set. The FODM function accepts G, **b1**, **b2** and Δz to create the differential matrices A_z^d and A_z^c. These matrices are used by the rhs function to calculate a discretised form of $\partial\Theta/\partial t$, in a similar fashion to (5). The ODE45 function accepts **h0** at the start of the simulation and uses the rhs function to approximate $\partial\Theta/\partial t$ at the required time interval. The approximation of $\partial\Theta/\partial t$ is used to obtain **h** at the next time step.

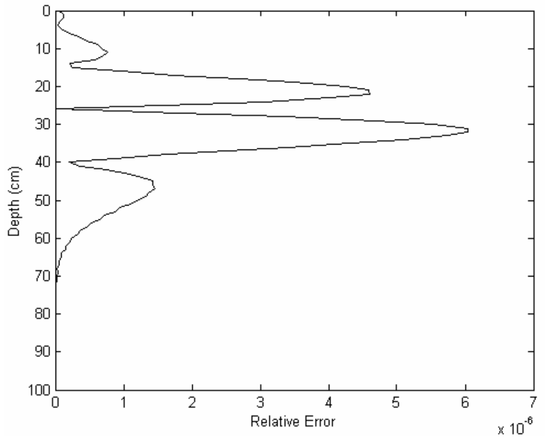

Figure 2. Plot of Depth vs Relative Error at $t = 36.25$ mins and $\Delta z = 1$cm.

The MoL template was used to generate a numerical solution at $t = 36.25$ mins. Relative error between the analytical and numerical solutions was calculated in terms of Θ and is shown in Figure 2. From Figure 2, the solution is accurate with relative errors of 10^{-6} over the domain.

6. A TWO-DIMENSIONAL EVOLUTION EQUATION

Weeks et al. (2002) considered the infiltration of water into and through a soil liner, which covered mine waste. They considered a piecewise linear dump and liner as an approximation to the shapes, which are used in practice (Hoekstra and Berkhout, 1994).

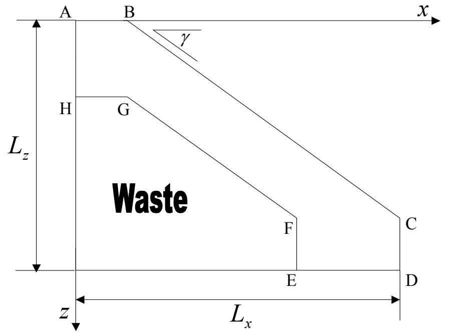

Figure 3. The piece-wise linear dump and cover liner domain.

In Figure 3, the cover liner is given by A, B, C, D, E, F, G and H, which overlays a mound of waste. A two-dimensional Richards' equation (1) is applied to the liner region, with a constant h initial condition. The boundary conditions are a rainfall flux (Q) over the surface A-B-C, a no flow condition along the bottom boundary E-F-G-H and along the boundaries A-H and C-D, and a natural drainage condition along E-D. Note that γ is the angle of the sloping face of the dump.

The shape of the cover liner is two-dimensional and complex, and contains edges which are not aligned parallel to the co-ordinate axes. The spatial discretisation is selected so that $\Delta z = \Delta x \tan(\gamma)$, ensuring that nodes lie along the sloping face. In addition, the lengths AH = ED $\tan(\gamma)$, CD = AB $\tan(\gamma)$ and $L_z = L_x \tan(\gamma)$ are related through the geometry. The soil characteristics are given by the hydraulic functions of Touma et al. (1984).

To apply the template to this problem, the initialisation routine needs to be extended to accept values associated with the new geometry, that is, the lengths L_z, AH, AB and the step size Δz. These values are passed to the domain routine, which has been extended to generate a two-dimensional grid over the domain and two numbering systems for the node indexing. As before, the domain routine will number the nodes in a column-wise manner (z-direction), which is the main node indexing for the template and is used to construct all indexing vectors. In addition, the domain routine generates a row-wise numbering system (x-direction), which is used as input data for the FODM routine to generate the differential matrices in the x-direction (A_x^d and A_x^c). As a result, the domain routine returns a two-dimensional grid matrix (**G**) and two additional indexing vectors, $\mathbf{k_c}$ and $\mathbf{k_r}$, that represent the column-wise and row-wise node numbering systems, respectively.

The FODM routine accepts G, $\mathbf{k_c}$ and $\mathbf{k_r}$ and the step sizes Δx and Δz and generates differential matrices for the x and z-directions, A_x^d and A_z^d, including the upwind matrices A_x^c and A_z^c. Note that both A_x^d and A_x^c are ordered in a row-wise fashion and can not be applied directly to **h**. To overcome this problem, the template creates a reordering vector **R**, which is an index vector that converts any domain vector from a column-wise number system to a row-wise numbering system. Note that **R** is constructed different to $\mathbf{k_c}$ or $\mathbf{k_r}$ and involves storing the column-wise numbering system in a row-wise fashion. Using MATLAB, the reordering vector is applied as follows

$$\mathbf{v_x}(\mathbf{R}) = A_x^d \mathbf{v}(\mathbf{R}) . \qquad (6)$$

In (6), **R** rearranges **v** to a row-wise numbering system and is applied to the differential matrix to evaluate $\mathbf{v_x}$. The second application of **R** ensures that the resultant vector keeps the original column-wise numbering system so that it can be applied to other indexing vectors.

Given that **R** is used correctly, (1) in MoL is effectively discretised to the form

$$C_w(\mathbf{h})\Box\frac{d\mathbf{h}}{dt} = A_x^d\left(\mathbf{K}\Box\mathbf{h}_x^d\right) + A_z^d\left(\mathbf{K}\Box\mathbf{h}_z^d\right) - \mathbf{K}_z^c \qquad (7)$$

The RHS routine needs to be modified to accommodate the above changes in evaluating (7). Also, boundary conditions need to be applied to a set of nodes, which is essentially automatic using MATLAB and the boundary indexing vectors.

To demonstrate flow behaviour from the cover liner model, the dimensions of the dump where set at L_z = 1m, AB = 0.4m and AH = 0.25m. The rainfall flux was set at $0.5K_s$ corresponding to a moderate rainfall rate. Figure 4 shows the contour lines of h at t = 18 minutes for a small waste dump. From Figure 4, a range of expected flow behaviour is evident with water infiltrating parallel to the sloping surface and the top flat section with a smooth transition between the two flow domains around the corner points B and G. The influence of the bottom no flow boundary is also evident with an accumulation of water along the boundaries HG and GF. Figure 4 also shows lateral divergence from the curvature of the contours around the corner points B and G and the influence of the natural drainage condition from the drier zone along boundary ED particular towards E. These types of flow behaviours are evident in Weeks et al (2002) and a comparison between the two numerical solutions for a more realistic size dump, L_z = 5 m, has shown good agreement (Matthews, 2002). Note that the output routine can also display the time development of the solution for the whole domain or for a particular node within the domain.

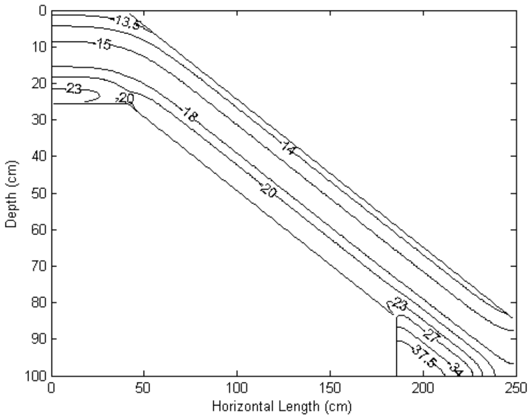

Figure 4. Contour plots of h(x,z,t) at t = 18 minutes for a cover liner with small dimensions.

7. DISCUSSION AND EXTENSIONS

The MATLAB template has been applied to problems, which involve complex functional representation of diffusion and convection processes. In each case, the template can be applied to generate the numerical solution. Changes to the principal routines are required to handle a two-space dimensional problem with a complicated geometry. All the code is available at the site www.gu.edu.au/school/eve/cmatthews.html.

In each of these applications, accurate numerical solutions can be obtained, provided that the solution profile does not contain strong slopes. The accuracy obtained does depend on the spatial step size and the error tolerances used in the integrator.

As another example, Gandola et al. (2001) applied the MoL template to a system of coupled PDEs for horizontal water (θ) and solute (c) transport. Gandola et al. (2001) compared the numerical solution against an analytical solution and obtained good agreement. The main change to the template is to represent both solution variables as a single column vector i.e. $[\boldsymbol{\theta},\mathbf{c}]'$ where **θ** and **c** are vectors that contain the values for θ and c for each node within the domain. This vector is needed to satisfy the input requirements for the ODE integrators within MATLAB. As a result, the RHS routine must split this vector into individual components, **θ** and **c**, to apply indexing vectors and differential matrices. The code for this problem can be also be found at the above website address.

It should be noted that the these applications do not contain source or sink terms such as may arise from chemical reactions, biological interactions between species, and even groundwater uptake by plants. Such effects can be readily added in the routine RHS for a particular problem. For example, assume that root uptake is given by the sink term

$$\mathbf{s} = \left(-S\left(z,t,\mathbf{h}\right)\right), \qquad (8)$$

where **s** is a vector of S values at the grid points **z**. As an example, **s** can added to (5) and discretised to give

$$C_w(\mathbf{h})\Box\frac{d\mathbf{h}}{dt} = A_z^d\left(\mathbf{K}\Box\mathbf{h}_z^d\right) - \mathbf{K}_z^c + \mathbf{s}, \qquad (9)$$

and the necessary alterations to the RHS routine are obvious. Note that if the sink term depended on either the water gradients, then the appropriate differentiation matrix would need to be included.

The dispersion of a pollutant in the air usually involves convection by a given wind field, as well as diffusion governed by eddy viscosity terms. The representation of such terms has its own physics, but the governing equations are essentially parabolic. They may contain source or sink terms due to settling of particulates or chemical reactions with other species. The template, can be extended to more than two species, since it equates to a system of coupled PDES. However, this can only be achieved if the interspecies interaction terms are available.

A prime feature of the template is that it provides an easy way to incorporate the finite differencing into the right-hand side of the partial differential equation, and thus to numerically evaluate these terms. In the MoL, this is then used by an integrator to advance the solution in time. The template can also be used in other common time-stepping techniques where time is also discretised, on a grid $t_j = j\Delta t$, j = 0, 1, 2 ..., and Δt is a prescribed time-step. Then (5) is discretised to the form

$$\mathbf{h}_{j+1} - \mathbf{h}_j = \Delta t\,\mathbf{F}\left(j, j+1, ...\right) \tag{10}$$

where $\mathbf{h}_j$ is the vector of h-values at the spatial nodes at time j, F is a function of the right-hand side discretised using the template at various time lines $j, j+1$, as well as possibly at other times. The choice of the function F depends on the classical method being used. As an example, the choice

$$\mathrm{F} = \frac{1}{2}\left\{\left[\mathrm{A}_z^d\left(\mathbf{K}\square\mathbf{h}_z^d\right) - \mathbf{K}_z^c\right]_j + \left[\mathrm{A}_z^d\left(\mathbf{K}\square\mathbf{h}_z^d\right) - \mathbf{K}_z^c\right]_{j+1}\right\} \tag{11}$$

gives the classical Crank Nicolson scheme (Ames, 1992). Where (5) is non-linear, i.e. K(h), then (10) for the Crank Nicolson scheme with F given in (11), is a set of non-linear algebraic equations for $\mathbf{h}_{j+1}$, and is usually solved using some variation of the Newton scheme. The generation of F in (11) is frequently done by hand, and requires technical skill. The template provides the means of easily constructing a computational form for carrying out the required functional evaluations. Where the Jacobian or Hessian matrix needs to be evaluated, appropriate differentiation matrices can be readily constructed and applied to F. Whilst the practical aspects of performing the calculations are handled by the template, the theoretical analysis of convergence and error properties still remains.

8. REFERENCES

Ames, W.E. (1992) Numerical Methods for Partial Differential Equations, Academic Press, New York.

Gandola, F., Sander, G. C. and Braddock, R. D. (2001) One Dimensional Transient Water and Solute Transport in Soils. MODSIM 2001, MSSANZ, ISBN 0 86740 525 0, pp. 505-510.

Hillel, D. (1998) Environmental Soil Physics, Academic Press, San Diego.

Hoekstra, S. E. and Berkhout, H. C. (1994) Geosynthetics for surface capping landfilling of waste: Barriers, E & FN SPON , New York, pp. 157-167.

Lee, H. S., Braddock, R. D. and Sander, G. C. (1998) Automating the Method of Lines for modeling moisture flow in the unsaturated zone. In Computational Techniques and Applications: CTAC-97, eds Noye, B. J., Teubner, M. D. and Gill, A. W., pp. 361-368. World Scientific Publishing Co.

Matthews, C. J. Modelling of a Multi-layered Cover Liner for a Waste Dump, PhD Thesis, Griffith University (submitted 2002).

Sander, G. C., Parlange, J.-Y., Kuhnel, V., Hogarth, W. L., Lockington, D. and O'Kane, J. P. J. (1988) Exact nonlinear solution for constant flux infiltration. J. Hydrol., **97**, 341-346.

Schiesser, W. E. (1991) The Numerical Method of Lines: Integration of Partial Differential Equations: ODEs, DAEs and PDEs, Academic Press, San Diego.

Shampine, L. F. and Reichelt, M. W. (1994) The MATLAB ODE, Suite Rept., 94-6, Math. Dept., SMU, Dallas, TX.

Touma, J., Vachaud, G. and Parlange, J.-Y. (1984) Air and water flow in a sealed, ponded vertical soil column, experiment and model. Soil Sci., **137**, 181-187.

Watson, K.K., Sardana, V.A. and Sander, G.C. (1995) Comparison of analytical and numerical results for constant flux infiltration. J. Hydro., **165**, 101-112.

Weeks, S. W., Sander, G. C., Braddock, R. D. and Matthews, C. (2002) Saturated and unsaturated water flow in individual porous media, EMA, accepted for publication.