FINITE DIFFERENCES METHOD AND ADAPTIVE GRIDS IN THE METHOD OF LINES FOR PARTIAL DIFFERENTIAL EQUATION

Sangare Boureima
UnitÃ© de Formation et de Recherche en Sciences et Techniques (UFR/ST)
Université Polytechnique de Bobo Dioulasso, Burkina Faso
E-mail address: mazou1979@yahoo.fr

ABSTRACT
In this paper a time-dependent moving-grid method is described to numerically solve time-dependent partial differential equations (PDEs) in one space dimensions involving fine scale structures such as steep moving fronts, emerging steep layers, pulses and shocks. Smoothing in the spatial direction is employed to control grid clustering and expansion. Additional smoothing in the temporal direction ensures a smooth progression of the grid points in time by preventing the points from responding too quickly to current values of the weight functions. In particular, we focus attention on finite differences scheme and adaptative grids using Method Of Line (MOL) toolbox within MATLAB. The numerical simulation includes various spatial approximation schemes based on finite differences and slope limiters. Several finite difference schemes, are compared. The performance of the algorithm is demonstrated with illustrative example including, a model of flame propagation, a methanisation in a reactor problem, and a classical Korteweg-de-Vries equation.

Keywords: *Moving mesh, Partial differential equations (PDEs), Finite Difference Methods (FDMs), Method Of Lines (MOL), Monitor Functions*
2000 Mathematics Subject Classification: 65L12, 65M20, 65N40.

1. INTRODUCTION

During the last decade, dynamically-moving grid methods, also characterized by the term refinement, have shown to be very useful for solving parabolic and hyperbolic partial differential equations (PDEs) involving fine scale structures such as steep moving fronts, emerging steep layers, pulses and shocks. In one space dimension moving-grid methods have been applied successfully to a large class of PDE systems (see e.g. [1, 2, 3] and [4]). For instances there are many possibilities to treat the one-dimensional boundary and to discretize the spatial domain, each having their own difficulties for specific PDEs.

One of the most popular approaches to the numerical solution of PDE models is the method of lines (MOL), which proceeds in two separate steps:

- approximation of the spatial derivatives using finite differences, elements or volumes;

- time integration of the resulting semi-discrete (discrete in space, but continuous in time) equations using an appropriate ODE solver.

MATLAB is now widely available in industry and academia and provides a very convenient basis for the development of MOL tools, allowing compact vector/matrix operations, and requiring minimum programming expertise.

In recent papers [5], [6], the authors have reported on the development of a collection of MATLAB functions (called MATMOL) implementing various finite difference schemes, flux limiters, static and dynamic spatial grid adaptation strategies.

In this paper a time-dependent moving-grid method for 1D models is described that produces adaptive grids which move smoothly in time. Smoothing is also applied in the spatial direction to control grid clustering and expansion. The discretization of the PDE is carried out in two stages through the method-of-lines. First, the PDE is transformed to a PDE in a moving frame by a coordinate transformation. This transformed PDE is semi-discretized using central differences for the spatial derivatives. Then, the moving-grid PDEs are defined that describe the dynamics of the grid. The moving-grid PDEs are based on an equidistribution principle along coordinate lines in the two spatial directions. Additional parameters are included to control the smoothness and adaptivity of the grid. Specifically, our aim is to provide a moving grid algorithm that has robust performance and can be used easily by non-expert users.

The paper is organized as follows. The next section introduces the MOL strategy and a moving grid algorithm, originally proposed in [12]. In section 3 we introduce some new options in the adaptive grids method. Section 4 presents Matlab numerical simulation of the algorithm proposed in section 2. The fifth section is devoted to numerical experiments. Finally, section 6 draws some conclusions remarks and future works.

2. OUTLINE OF FINITE DIFFERENCES METHOD AND ADAPTIVE GRIDS IN THE METHOD OF LINES ALGORITHM

2.1 Problem class and semi-discrete ODE or DAE system

We consider the partial differential equations (PDEs) problem

$$x_t = f(z, t, x, x_z, x_{zz}, \dots),\ z \in \Omega,\ t \geq 0 \qquad (1)$$

$$0 = b(z, t, x, x_z, \dots),\ z \in \Gamma,\ t > 0 \qquad (2)$$

$$x_0(z) = x(t = 0, z),\ z \in \Omega \cup \Gamma \qquad (3)$$

where

- $x \in \mathbb{R}^{n_{pde}}$ is the vector dependent variables,
- z is the vector of spatial independent variables,
- t is an initial value independent variable,
- $x_t = \frac{\partial x}{\partial t},\ x_z = \frac{\partial x}{\partial z}$

This problem represent a system of partial differential equation PDEs defined in a spatial domain Ω, their associated boundary conditions (BCs) defined on the boundary surface Γ of Ω, and initial conditions (ICs) defined on the complete spatial domain.

Following the MOL principle, finite differences, or other techniques such as spectral methods etc., can be used to approximate the spatial derivatives appearing in PDEs (1) and BCs (2) and the corresponding linear transformation can be conveniently implemented using the concept of a differentiation matrix D.

$$\tilde{x}_z = D_1\tilde{x},\ \tilde{x}_{zz} = D_2\tilde{x},\ \dots \qquad (4)$$

Replacing of (4) into (1)-(3) gives the semi-discrete ODE or DAE system

$$\tilde{x}_t = f(z, t, \tilde{x}, \tilde{x}_z, \tilde{x}_{zz}, \dots),\ z \in \Omega,\ t \geq 0 \qquad (5)$$

$$0 = b(z, t, \tilde{x}, \tilde{x}_z, \dots),\ z \in \Gamma,\ t > 0 \qquad (6)$$

$$\tilde{x}_0(z) = x(t = 0, z),\ z \in \Omega \cup \Gamma \qquad (7)$$

$\tilde{x}$ is the approximate solution.

This ODE/DAE system can be integrated in time using one of the solvers available in the MATALB ODE. At this stage, the numerical solution procedure is only semi-automatic in the sense that the time stepsize is automatically adjusted by the solver, but the spatial grid is chosen a priori and fixed. However, when considering PDE problems with steep moving fronts, it can be advantageous to concentrate the grid points in spatial regions of high activity and to move them continuously in time, i.e., to use dynamic grid adaptation. Not only the number of grid points can be significantly reduced, as fewer grid points are used in regions of low solution activity, but larger time steps can usually be taken as the moving fronts are less likely to cross grid points.

2.2 An adaptive method of lines algorithm

The algorithm considered in this study has been originally proposed in [9, 12] and is based on the Lagrangian formulation of the PDE problem $(1)-(3)$. Consider the continuous time trajectories of the grid points

$$Z_0 < Z_1 <\ldots< Z_i(t) <\ldots< Z_N(t) < Z_{N+1} \quad (8)$$

which are, as yet, unknown. Introduce, along $z(t) = Z_i(t)$, the total temporal derivative of x is given by

$$\dot{x} = x_z\dot{z} + x_t = \dot{z}x_z + \mathcal{L}(x, Z_i(t), t), \quad 1 \le i \le N, \quad (9)$$

and spatially discretize, for each fixed t, the space operators $\frac{\partial}{\partial z}$ and $\mathcal{L}$ so as to obtain the semi-discrete system

$$\dot{X}_i = \dot{Z}_i \frac{X_{i+1}-X_{i-1}}{Z_{i+1}-Z_{i-1}} + L_i, \quad t > 0, \quad 1 \le i \le N. \quad (10)$$

As usual, $X_i(t)$ represents the semi-discrete approximation to the exact PDE solution u at the point $(z,t) = (Z_i(t), t)$ and L_i the finite difference replacement of $\mathcal{L}(x,z,t)$ at this point. Note that the standard, central difference approximation for x_z is used.

It is supposed that L_i is also based on standard, 3-point central differencing. Further it is of interest to observe that at this stage of development the only errors introduced are the space discretization errors. With the associated grid functions

$$Z = [Z_1, \ldots, Z_N]^T,$$

$$X = [X_1^T, \ldots, X_N^T]^T,$$

$$L = [L_1^T, \ldots, L_N^T]^T,$$

$$D = [D_1^T, \ldots, D_N^T]^T$$

with $X_i = [X_{i,1}, X_{i,2}, \ldots, X_{i,NPDE}]$ and

$$D_i = \frac{X_{i+1}-X_{i-1}}{Z_{i+1}-Z_{i-1}}$$

We reformulate (6) in the more compact form

$$\dot{X} = \dot{Z}oD + L, \quad t > 0, \quad X(0) \quad given, \qquad (11)$$

which represents the semi-discrete system to be numerically integrated in time. The notation $\dot{Z}oD$ meaning that $\dot{Z}_i$ is to be multiplied with all components of the vector D_i.

Equation (11) which contains the two unknowns X and Z must be coupled with a second equation in order to form a complete system. This equation will be obtained by the moving grid process.

The ODEs defining the grid point movement, i.e., $z = g(t)$, can be derived based on some physical a priori knowledge, such as a flow-related quantity, or so as to equally distribute a monitor function $m(x)$ such as the arc-length of the solution (several other monitor functions can be considered, e.g., based on the solution curvature).

The spatial node movement is based on the equidistribution principle: the grid points Z_i, $1 \le i \le N$ are moved so that a specified quantity, also called the $monitor\ function\ m(x)$, is equally distributed over the spatial domain. Equation which govern this equally distribution can be traduced by the following equality called $equidistribution\ equation$

$$\int_{Z_{i-1}}^{Z_i} m(x)dz = \int_{Z_i}^{Z_{i+1}} m(x)dz, \;\; 2 \le i \le N-1, \quad (12)$$

Or in a discrete form

$$\frac{\eta_{i-1}}{M_{i-1}} = \frac{\eta_i}{M_i}; \quad 2 \le i \le N-1, \quad (13)$$

where η_i are called $point\ concentrated\ values$ and are defined are follows;

$$\eta_i = \frac{1}{(\Delta Z_i)}, \quad \Delta Z_i = Z_{i+1} - Z_i, \quad 0 \le i \le N. \quad (14)$$

For example, the following monitor function based on the arc-length of the solution (case of $\alpha = 1$)

$$m(x) = \sqrt{\alpha + \| x_z \|_2^2} = \sqrt{\alpha + \| \frac{\partial x(z,t_{k+1})}{\partial z} \|_2^2} \quad (15)$$

yields, employing upwind differencing,

$$M_i = \sqrt{\alpha + \frac{1}{NPDE} \sum_{j=1}^{NPDE} \frac{(X_{i+1}^j - X_i^j)^2}{(Z_{i+1} - Z_i)^2}} \quad (16)$$

The positive parameter α is introduced to modify the relative importance of values of ΔZ_i and of ΔX_i.

It is well-known that the grid prescribed by equation (16) must be smoothed in order to avoid oscillations and distortions. Of course, other choices for the monitor function could be used. Others alternative monitor functions will be considered in the following.

2.2.1 The grid-smoothing procedures

The spatial grid-smoothing is effected by replacing the point concentrations η_i by their numerically "anti-diffused" counterparts $\tilde{\eta}_i$, defined as

$$\tilde{\eta}_0 = \eta_0 - \kappa(\kappa+1)(\eta_1 - \eta_0), \;\; \tilde{\eta}_i = \eta_i - \kappa(\kappa+1)(\eta_{i+1} - 2\eta_i + \eta_{i-1}), \;\; 1 \le i \le N-1, \;\; \tilde{\eta}_N = \eta_N - \kappa(\kappa+1)(\eta_{N-1} - \eta_N),$$

which results in the following system:

$$\frac{\tilde{\eta}_{i-1}}{M_{i-1}} = \frac{\tilde{\eta}_i}{M_i}, \quad 0.3cm1 \le i \le N, \quad (17)$$

Equation (17) insures that the adjacent point concentrations are restricted such that

$$\frac{\kappa}{\kappa+1} \le \frac{\eta_{i-1}}{\eta_i} \le \frac{\kappa+1}{\kappa} \quad (18)$$

which is the spatial regularization condition. Parameter κ determines the minimum and maximum interval lengths.

2.2.2 The temporal smoothing procedures

When combined with spatial grid smoothing, the temporal grid-smoothing is affected by replacing the system of algebraic equations (17) by the following system of differential equations:

$$\frac{\tilde{\eta}_{i-1} - \tau\dot{\tilde{\eta}}_{i-1}}{M_{i-1}} = \frac{\tilde{\eta}_i - \tau\dot{\tilde{\eta}}_i}{M_i}, \quad \tau > 0, \;\; 1 \le i \le N, \quad (19)$$

The introduction of the derivatives of the point concentrations serves to prevent the grid movement from adjusting solely to new monitor values. The positive parameter τ acts as a time-constant preventing abrupt changes in the grid movement and allows to avoid temporal oscillations and hence to obtain a smoother progression of $Z(t)$. Experience

shows that spatial smoothing is more important than temporal smoothing.

2.3 The complete semi-discrete system

2.3.1 The moving grid equation in terms of nodal values

Inserting $\eta_i = \frac{1}{(\Delta Z_i)}$ and $\dot{\eta}_i = -\frac{\Delta \dot{Z}_i}{\Delta Z_i^2}$ into (19) leads to the moving grid equation system which is represented in the form of the nonlinear ODE system:

$$\tau \mathcal{B}(Z,X)\dot{Z} = g(Z,X), \qquad (20)$$

where $\mathcal{B}$ is the $N \times N$ penta-diagonal matrix associated to the left-hand side part of (16) and g the right-hand.

2.3.2 The complete semi-discrete system and its numerical integration

System (11) and system (20) together form the complete semi-discrete system that is numerically integrated in time,

$$\tau \mathcal{B}\dot{Z} = g, \quad t > 0, \quad Z(0) \quad given, \qquad (21)$$

$$\dot{X} - \dot{Z}oD = L, \quad t > 0, \quad X(0) \quad given, \qquad (22)$$

giving at each time t the associated grid Z and the corresponding solution X. For many integrators the coupled system (21)-(22) must be rewritten as follows:

$$\mathcal{M}(t,X)\dot{X} = f(t,X) \qquad (23)$$

where vector $X = [\dots, X_i^1, X_i^2, \dots, X_i^{NPDE}, Z_i, \dots]$ is the global unknown, $\mathcal{M}$ represents the mass matrix and f is the right-hand member of this global system [11].

3. INTRODUCTION OF SOME NEW OPTIONS IN THE ADAPTIVE GRIDS METHOD

3.1 Initial grid adaptation

Since the initial conditions (ICs) play a major role in accuracy of numerical solutions, it may be wise to have the best approximation possible from the start. Thus, when the initial condition depends spatial variable z, we apply the initial grid adaption described in [10].

3.2 Monitor function

In the method described by P. A. Zegeling, [12], only monitor function $m = m(x_z)$, based on the arc length of the solution, was used. However in the scientific literature, other monitors functions $m = m(x_{zz})$ based on the curvature of the solution exists. In total, therefore, we have the following two monitors functions, depending on the derivatives of the solution x we have planned and integrated in our informatics program.

$$m_1(x) = \sqrt{\alpha + \| x_z \|_2^2} \qquad (24)$$

$$m_2(x) = \sqrt{\alpha + \| x_{zz} \|_\infty} \qquad (25)$$

3.3 Approximation of the monitor function

In the description of the method of moving grid in [12], x_z was approximated by finite differences progressive, two points, and the standard used was calculated by taking the average of N_{pde} G components, giving the following formula:

$$M_i = \sqrt{\alpha + \frac{1}{N_{pde}} \sum_{j=1}^{N_{pde}} \frac{(X_{i+1}^j - X_i^j)^2}{(Z_{i+1} - Z_i)^2}}, \qquad (26)$$

$$M_i = \sqrt{\alpha + max_j \frac{(X_{i+1}^j - X_i^j)^2}{(Z_{i+1} - Z_i)^2}}, \quad 0 \le i \le N. \quad (27)$$

We can, for approximating derivatives of order 1, choose a derivative operator digital D_1 calculating x_z ie $(x_z = D_1 X)$ and use again the average or maximum as follows:

$$M_i = \sqrt{\alpha + \frac{1}{N_{pde}} \sum_{j=1}^{N_{pde}} \frac{(X_{z,i+1}^j - X_{z,i}^j)}{2}}, \quad 0 \le i \le N, \quad (28)$$

or

$$M_i = \sqrt{\alpha + max_j \frac{(X_{z,i+1}^j - X_{z,i}^j)}{2}}, \quad 0 \le i \le N. \quad (29)$$

The choice of the operator D_1 is enough large if we recall the different stencils: centered, decentered, decentered biased. In the case of hyperbolic PDEs with a flux term $f(x)$, we may consider in the monitor function, the term $f(x)$ rather than the term x and approach the derivative $f(x)_z$ by different approximation formulas of the first derivatives or slopes limiters or flux.

For the monitor function m_2, the options are the follow:

$$M_i = \sqrt{\alpha + \frac{1}{N_{pde}} \sum_{j=1}^{N_{pde}} \left| \frac{(X_{zz,i+1}^j - X_{zz,i}^j)}{2} \right|}, \quad 0 \le i \le N. \quad (30)$$

and

$$M_i = \sqrt{\alpha + max_j \left| \frac{(X_{zz,i+1}^j - X_{zz,i}^j)}{2} \right|}, \quad 0 \le i \le N. \quad (31)$$

In these formulas the second derivative x_{zz} is calculated by the equation $X_{zz} = D_2 X$ where D_2 denotes an operator of numerical derivation for the second derivatives. We can use the operator of three bullet points or five bullet points.

3.4 Option in approximation of the moving grid equation

Recall that after the Lagrangian formulation of the problem, the total derivative of the solution x wrote

$$\dot{x} = x_z \dot{z} + x_t = \dot{z} x_z + \mathcal{L}(x, Z_i(t), t), \ 1 \le i \le N, \quad (32)$$

then

$$\dot{X}_i = \dot{Z}_i \frac{X_{i+1} - X_{i-1}}{Z_{i+1} - Z_{i-1}} + L_i, \ t > 0, \ 1 \le i \le N, \quad (33)$$

formula in which x_z was approximated by finite differences centered at 3 points. Had been introduced there a grid function $D = [D_1^T, \cdots, D_N^T]^T$ defined by

$$D_i = \frac{X_{i+1} - X_{i-1}}{Z_{i+1} - Z_{i-1}} \quad (34)$$

Another possibility is to consider finite differences centered at 5 points, or decentered finite differences that reflect the direction of flow. A formula centered finite differences to 5 points on a non- uniform grid will function following grid, obtained from a Taylor series:

$$D_i = -(X_{i+2} - X_i)\frac{(z_{i+1}-z_i)(z_{i-2}-z_i)}{(z_{i+2}-z_{i+1})\dots(z_{i+2}-z_{i-2})} +$$

$$(X_{i+1} - X_i)\frac{(z_{i+2}-z_i)(z_{i-2}-z_i)}{(z_{i+1}-z_{i+2})\dots(z_{i+1}-z_{i-2})} +$$

$$(X_{i-1} - X_i)\frac{(z_{i+2}-z_i)(z_{i-2}-z_i)}{(z_{i-2}-z_{i+2})\dots(z_{i-1}-z_{i-2})} +$$

$$(X_{i-2} - X_i)\frac{(z_{i+2}-z_i)(z_i-z_i)}{(z_{i-2}-z_{i+2})\dots(z_{i+2}-z_{i-2})},\ 2 \le i \le N. \quad (35)$$

As finite difference formula we can consider the finite difference progressive 2-point function giving by the grid function.

$$D_i = \frac{X_{i+1}-X_i}{Z_{i+1}-Z_i},\quad 0 \le i \le N+1 \qquad (36)$$

We can always try without insisting too, the approximation to 5 bullet points in the partial differential equation semi-discretized.

3.5 Coupling of moving grid method with pente limiters

In the case of hyperbolic PDEs with a flux term $f(x)$, we may, to approximate the partial derivative $f(x)_z$, use limiters slopes or stream instead of operators D_1 numerical derivation. Recall the main limiting slopes available in MATMOL:

- *koren-slope-limiter-fz*;
- *kurg-centered-slope-limiter-fz*;
- *minmod-slope-limiter-fz*;
- *smart-slope-limiter-fz*;
- *superbee-slope-limiter-fz*;
- *vanleer-slope-limiter-fz*.

After identifying the flow depending on the problem and adapted arguments, the call of limiter enable to approach $f(x)_z$. We can also use this approximation $f(x)_z$ in the monitor instead of x_z function. For a given problem, it is advisable to use and compare the effect of different limiting slopes to retain the most effective.

3.6 Option in approximation operators of PDEs

To approximate the partial derivatives of order 1, i.e x_z, D_1, we use generally the operator D_1 decentered or biased decentered for the term of convection. However, if the phenomenon known a propagation in two directions, operators D_1 centered is used. When the problem contains a flux term $f(x)$, it is recommended to use limiters slopes or stream. As regards the derivatives of order 2 and higher which typically model the phenomenon of diffusion or dissipation, centered operators are best adapted. Finally, note that for the derivatives of order n and greater than two, operators D_n can be used or the technique of *branch* cascade from a lower order operator, eg $x_{zzz} = D_1(D_1(D_1x))$.

4. MATLAB NUMERICAL SIMULATIONS OF THE ALGORITHM

4.1 Solver ODE15s

In MATLAB there are many, ODE solver. There syntaxe is:

$$M(t,Y)Y' = f(t,Y) \qquad (37)$$

ODE15s is chosen for this specialisation in resolution of $stiffs$ problem. This syntaxe is follow:

$$[T,Y] = ODE15s(ODEFUN,\ \ TSAN,\ \ Y0,\ \ OPTIONS)$$

where

• vectors T et Y represent respectively instant t_i and solutions $Y(t_i)$ corresponding,

• ODEFUN means the function describing the ODE system is the time interval of integration and intermediate times at which the solution is desired Y_0 is the initial vector

• OPTIONS represents the different options selected for the solver.

The options are set using the function ODESET and we used the following form:

$$ODESETOPTIONS = ('reltol'valeurRelTol'AbsTol'valeurAbsTol,'Mass',$$
$$MatricedeMass'MStateDependance','strong','JPattern',JPat'MvPattern'MvPat'\ MassSingular','no','stats','on')$$

where $reltol$ and $AbsTol$ denote the relative and absolute error which we can find justification in MATLAB, and for the other options tolerances. We insist on the $MvPattern$ options and $JPattern$ because they play a very important role and must be programmed. The introduction of these options in the code is a trick that allows the ODE integrator to reduce the number of operations and gain computation time.

4.1.1 Le MvPattern

If the mass matrix $M(t,\ Y)$ of the system (37) depends of Y we associate the sparse matrix S defined by

$$S_{ij} = \begin{cases} 1 \ \ if \ \ exist \ \ one \ \ value \ \ of \ \ k \ \ for \ \ which \\ \ \ M_{ik} \ \ depend \ \ of \ \ Y_j, \\ 0 \ \ else. \end{cases}$$

It expresses the incomplete nature of $\frac{\partial M}{\partial Y}$.

4.1.2 Le JPattern

The Jacobian sparsity pattern or JPattern is sparse matrix S defined by

$$S_{ij} = \begin{cases} 1 \ \ if \ \ f_i \ \ depend \ \ of \ \ Y_j, \\ 0 \ \ else. \end{cases}$$

where $f_i = f(t_i, Y_i)$. It expresses the incomplete nature of $\frac{\partial f}{\partial Y}$.

The interested reader is referred to the MATLAB documentation for more information on these differents concepts.

4.2 Mass Matrix implementation

In the previous chapter we have shown that the method of moving grid leads to the resolution of the following coupled system

$$\tau \boldsymbol{B}\dot{Z} = g, \quad t > 0, \qquad (38)$$

$$\dot{X} - \dot{Z}oD = L, \quad t > 0, \tag{39}$$

$$Z(0) = Z^0, \tag{40}$$

$$X(0) = X^0, \tag{41}$$

where Z is the grid computing and X the corresponding solution. To use the solver ODE15s we wrote the system in the following general form :

$$A(t,Y)\dot{X} = b(t,Y) \tag{42}$$

where

$$X = [X_1^1, X_1^2, \dots X_1^{Npde}, Z_1, \dots, X_i^1, X_i^2, \dots X_i^{Npde},$$

$$Z_i, \dots, X_N^1, X_N^2, \dots X_N^{Npde}, Z_N]^T \tag{43}$$

is the global unknown vector, **A** the mass matrix of the global system and **b** the second associate member.

Compared to formula (37), the mass matrix **M** is replaced by **A**, the global vector Y by X and the second member f by b.

The construction of the unknown vector X shows that the matrix A is square and order $(NPDE + 1) \times N$ times and has a penta-diagonal-block. The blocks, order $(NPDE + 1)$ are of the form.

$$A_{ii} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & -D_{i1} \\ 0 & 1 & 0 & \dots & 0 & -D_{i2} \\ 0 & 1 & 0 & \dots & 0 & -D_{i3} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & -D_{iNpde} \\ 0 & 0 & 0 & \dots & 0 & -\tau B_{i,i} \end{bmatrix} 1 \le i \le N \tag{44}$$

and the A_{ij} blocks given by

$$A_{ij} = \begin{bmatrix} 0 & \dots & 0 & 0 \\ 0 & \dots & 0 & 0 \\ 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & 0 \\ 0 & \dots & 0 & \tau B_{i,j} \end{bmatrix}, \tag{45}$$

$$1 \le i \le N, \quad 1 \le j \le N, \quad j \ne i, \quad |i - j| \le 2$$

$$A_{ij} = 0, \tag{46}$$
$$1 \le i \le N, \quad 1 \le j \le N, \quad j \ne i, \quad |i - j| > 2$$

with

$$D_{ij} = \frac{X_{i+1}^j - X_{i-1}^j}{Z_{i+1} - Z_{i-1}}, \quad 1 \le i \le N, \quad 1 \le j \le N_{pde}$$

and

$$B_{i,i-2} = -\frac{\mu}{M_{i-1}(\Delta Z_{i-2})^2}$$

$$B_{i,i-1} = +\frac{\mu}{M_i(\Delta Z_{i-1})^2} + \cdots + \frac{\mu}{M_{i-1}(\Delta Z_{i-2})^2}$$

$$B_{i,i} = -\frac{\mu}{M_i(\Delta Z_{i-1})^2} + \cdots + \frac{\mu}{M_{i-1}(\Delta Z_i)^2}$$

$$B_{i,i+1} = +\frac{\mu}{M_i(\Delta Z_{i+1})^2} + \frac{\mu}{M_{i-1}(\Delta Z_i)^2}$$

$$B_{i,i+2} = -\frac{\mu}{M_i(\Delta Z_{i+1})^2} \quad 3 \le i \le N-2$$

$$B_{i,j} = 0,$$

$$1 \le i \le N, \quad 1 \le j \le N, \quad j \ne i, \quad |i-j| > 2$$

The above formulas allow you to write a first program called $mass$ and calculates the global matrix **A** of the system. Sparse matrix **As** associated also square order $(N_{pde}+1) \times N$, penta-diagonal-block is given by the following blocks of order $(N_{pde}+1)$.

$$As_{ii} = \begin{bmatrix} 0 & . & . & . & 0 \\ 0 & . & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & . & . & . & 0 \\ 1 & . & . & . & 1 \end{bmatrix}, \qquad (47)$$

$$As_{i,i-1} = As_{i,i+1} = \begin{bmatrix} 1 & 0 & . & . & 0 & 1 \\ 0 & 1 & 0 & . & 0 & 1 \\ . & . & . & . & . & \\ . & . & . & . & . & \\ . & . & . & . & . & \\ 0 & . & . & . & 1 & 1 \\ 1 & . & . & . & . & 1 \end{bmatrix}, \qquad (48)$$

$$As_{i,i-2} = As_{i,i+2} = \begin{bmatrix} 0 & . & . & . & 0 & 0 \\ 0 & 0 & . & . & 0 & 0 \\ . & . & . & . & . & \\ . & . & . & . & . & \\ . & . & . & . & . & \\ 0 & . & . & . & 0 & 0 \\ 0 & . & . & . & 0 & 1 \end{bmatrix}, \quad (49)$$

$$3 \leq i \leq N-2$$

$$As_{i,j} = 0, \quad (50)$$

$$1 \leq i \leq N, \quad 1 \leq j \leq N, \quad j \neq i, \quad |i-j| > 2$$

The above formulas allow us to write a second program called *MvPat* and calculates the sparse matrix **As** associated with **A**.

4.3 Implementation of second system member

The second member **b** of the global system, vector $(pdeN_1) \times N$ components, is made of the second member **g** of the system (37) of the moving grid and the second member L from the semi-discretized system giving $\dot{X}$. Thus

$$b = [\dot{X}_1^1, \ldots \dot{X}_1^{Npde}, g_1, \dot{X}_2^1, \ldots, \dot{X}_2^{Npde}, g_2, \ldots, X_i^1, \ldots]^T$$

with

$$g_1 = \frac{1}{M_0}\left[-\frac{1+\mu}{\Delta Z_0} + \frac{\mu}{\Delta Z_1}\right] + \frac{1}{M_1}\left[-\frac{\mu}{\Delta Z_0} + \cdots - \frac{\mu}{\Delta Z_2}\right]$$

$$g_i = \frac{1}{M_i}\left[-\frac{\mu}{\Delta Z_{i+1}} + \cdots - \frac{\mu}{\Delta Z_{i-1}}\right] -$$

$$\frac{1}{M_{i-1}}\left[-\frac{\mu}{\Delta Z_i} + \cdots - \frac{\mu}{\Delta Z_{i-2}}\right], \quad 0.2cm2 \leq i \leq N-1$$

$$g_N = \frac{1}{M_N}\left[\frac{1+\mu}{\Delta Z_N} - \frac{\mu}{\Delta Z_{N-1}}\right] + \quad 1cm \quad (51)$$

$$\frac{1}{M_{N-1}}\left[-\frac{\mu}{\Delta Z_{N-2}} + \frac{1+2\mu}{\Delta Z_{N-1}} - \frac{\mu}{\Delta Z_N}\right] \quad 0.7cm \quad (52)$$

As for the $\dot{X}_i^j, \quad 1 \leq i \leq N, \quad 1 \leq j \leq N_{pde}$, they are obtained as follows: is replaced in the second members of (39), the partial spatial derivatives by finite difference approximation used for these derivatives. It is at this level that involves operators from numerical derivation tools MATMOL.

We consider for example the following system: find three functions

$u = u(z,t),\ v = v(z,t),\ w = w(z,t)$ solutions of

$$u_t = -v_z + \varepsilon u_{zz}, \quad 0 < z < 1, \quad t > 0; \quad (53)$$

$$v_t = -\frac{\partial}{\partial z}[(\gamma-1)w - 0.5(\gamma-3)\frac{v^2}{u}] + \quad (54)$$

$\varepsilon v_{zz},\ 0 < z < 1,\ t > 0;$

$$w_t = -\frac{\partial}{\partial z}[(\gamma w - 0.5(\gamma-1)\frac{v^2}{u})\frac{v}{u}] + \quad (55)$$

$\varepsilon w_{zz},\ 0 < z < 1,\ t > 0;$

If we decide to approach the partial derivatives of order 1 (resp. 2) by centered finite differences to 3 (resp. 5) points, blocks $(i-2)$ $(i-1)$ i $(i+1)$ $(i+2)$ for i^{nd} line-th block of the matrix $JPat$ have the following structures:

$$\begin{array}{cccccccccccc}
1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\underbrace{0}_{i-2} & \underbrace{0}_{i-2} & \underbrace{0}_{i-2} & \underbrace{1}_{i-2} & \underbrace{1}_{i-1} & \underbrace{1}_{i-1} & \underbrace{1}_{i-1} & \underbrace{1}_{i-1} & \underbrace{1}_{i} & \underbrace{1}_{i} & \underbrace{1}_{i} & \underbrace{1}_{i}
\end{array}$$

$$\begin{array}{cccccccc}
1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
\underbrace{1}_{i+1} & \underbrace{1}_{i+1} & \underbrace{1}_{i+1} & \underbrace{1}_{i+1} & \underbrace{0}_{i+2} & \underbrace{0}_{i+2} & \underbrace{0}_{i+2} & \underbrace{1}_{i+2}
\end{array}$$

$$3 \le i \le N-2$$

Other columns blocks j of this line, e.g,

$$1 \le j \le i-3, \quad i+3 \le j \le N$$

are formed only from "0". Blocks-lines 1 and 2 and the lines blocks $(N-1)$ and N have the following specific structures due to special approximations border nodes derived:

• Line-bloc 1

$$\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\underbrace{1}_{1} & \underbrace{1}_{1} & \underbrace{1}_{1} & \underbrace{1}_{1} & \underbrace{1}_{2} & \underbrace{1}_{2} & \underbrace{1}_{2} & \underbrace{1}_{2} & \underbrace{0}_{3} & \underbrace{0}_{3} & \underbrace{0}_{3} & \underbrace{1}_{3}
\end{array}$$

$$\begin{array}{cccccccccccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \underbrace{0 \quad 0 \quad 0 \quad 0}_{4} & & & & \underbrace{0 \quad 0 \quad 0 \quad 1}_{5} & & & & \underbrace{0 \quad 0 \quad 0 \quad 1}_{6} & & & \end{array}$$

The column blocks such as $j \geq 7$ are invested only of "0".

• Line-bloc 2

$$\begin{array}{cccccccccccc} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \underbrace{1 \quad 1 \quad 1 \quad 1}_{1} & & & & \underbrace{1 \quad 1 \quad 1 \quad 1}_{2} & & & & \underbrace{1 \quad 1 \quad 1 \quad 1}_{3} & & & \end{array}$$

$$\begin{array}{cccccccccccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \underbrace{0 \quad 0 \quad 0 \quad 1}_{4} & & & & \underbrace{0 \quad 0 \quad 0 \quad 0}_{5} & & & & \underbrace{0 \quad 0 \quad 0 \quad 0}_{6} & & & \end{array}$$

Similarly here column blocks j such as $j \geq 7$ are invested only of "0".

• Line-bloc $(N-1)$

$$\begin{array}{cccccccccccc} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \underbrace{0 \quad 0 \quad 0 \quad 0}_{N-5} & & & & \underbrace{0 \quad 0 \quad 0 \quad 0}_{N-4} & & & & \underbrace{0 \quad 0 \quad 0 \quad 1}_{N-3} & & & \end{array}$$

$$\begin{array}{cccccccccccc} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \underbrace{1 \quad 1 \quad 1 \quad 1}_{N-2} & & & & \underbrace{1 \quad 1 \quad 1 \quad 1}_{N-1} & & & & \underbrace{1 \quad 1 \quad 1 \quad 1}_{N} & & & \end{array}$$

The column blocks j such as $j \geq (N-6)$ are invested only of "0".

• Line-bloc N

$$\begin{array}{cccccccccccc} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ \underbrace{0 \quad 0 \quad 0 \quad 0}_{N-5} & & & & \underbrace{0 \quad 0 \quad 0 \quad 0}_{N-4} & & & & \underbrace{0 \quad 0 \quad 0 \quad 0}_{N-3} & & & \end{array}$$

$$\begin{array}{cccccccccccc} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \underbrace{0 \quad 0 \quad 0}_{N-2} & 1 & \underbrace{1 \quad 1 \quad 1}_{N-1} & 1 & \underbrace{1 \quad 1 \quad 1}_{N} & 1 \end{array}$$

Similarly here column blocks j such as $j \geq (N-6)$ are invested only of "0".

5. NUMERICAL EXPERIMENTS

5.1 A methanisation in a reactor problem

5.1.1 Description of the problem

A fixed bed reactor is considered wherein the hydrogenation of small amounts methane gives $CO2$, according to chemical equation.

$$CO_2 + 4H_2 \rightarrow CH_4 + 2H_2O$$

Hydrogen was injected into the reactor initially supplied with carbon dioxide following a stepwise pattern. The initial temperature of the reactor is also adjusted stepwise and we study the evolution of the concentration of carbon dioxide and evolution of the temperature during the experiment, [7, 8].

Mathematically, the problem is to determine two functions $C = C(z,t)$ and $T = T(z,t)$ solutions of

$$C_t = -vC_z + DC_{zz} - r, \quad 0 < z < L, \quad t > 0; \text{(56)}$$

$$T_t = -\varepsilon v \frac{\rho_g c_{pg}}{\rho c_p} T_z + \frac{\lambda}{\rho c_p} T_{zz} + \frac{2k_w}{d\rho c_p}(T_w - T) + \frac{(-\Delta H)}{\rho c_p} r, \qquad (57)$$

$$0 < z > L, \ t > 0;$$

where $c(kmol/m^3)$ is the concentration of reactive species, $T(K)$ the temperature, $D = 5 \times 10^{-3} m^2/s$ the diffusivity constant, $v = 1m/s$ the gas velocity, $k = 29732s^{-1}$ the rate constant, $E/R = 8000K$ the activation temperature, $\rho\bar{c_p} = 400kJ/(m^3K)$ the heat capacity of the fixed bed, $\lambda = 2.06 \times 10^{-3} kW/(mK)$ the effective axial heat conductivity, $\rho_g c_{pg} = 0.5kJ/(m^3K)$ the heat capacity of the gas, $\varepsilon = 0.8$ the bed void fraction, $(-\Delta H_R) = 206000kJ/kmol$ the heat of reaction, $L = 1m$ the reactor length, [8].

For weakly exothermic reactions, operation of the fixed-bed catalytic reactor with periodic flow reversals is of particular interest. This way, the front and end parts of the catalyst bed act as regenerative heat exchangers for feed and effluent, allowing the reactions to be operated autothermally at high temperatures. After a start-up phase (here $t_{start} = 1500s$), where the gas enters at high temperature ($c_{in} = 1.21 \times 10^{-4} kmol/m^3$ at $T_{in} = 873K$), the feed temperature is decreased ($T_{in} = 293K$) and periodic flow reversal allows the reactor to be operated autothermally, [10, 11].

$$C(z,0) = C_0(z) = 0, \quad 0 < z < L; \qquad (58)$$

$$T(z,0) = T_0(z) = 300, \quad 0 < z < L; \qquad (59)$$

$$C_z(0,t) = \frac{v}{D}(C - C_{in}), \quad t > 0; \qquad (60)$$

$$C_z(L,t) = 0, \quad t > 0; \tag{61}$$

$$T_z(0,t) = \varepsilon v \frac{\rho_g c_{pg}}{\lambda}(T - T_{in}), \quad t > 0; \tag{62}$$

$$T_z(L,t) = 0, \quad t > 0; \tag{63}$$

where the reaction rate is given by

$$r = k_r \frac{Cexp(-\frac{E}{RT})}{1+k_c C};$$

the initial concentration C_{in} of CO_2 varies by level,

$$C_{in}(t) = 0 \rightarrow 2.5 \ \ moles;$$

The initial temperature T_{in} varies also by level,

$$T_{in}(t) = 300 \rightarrow 500K.$$

and periodic changes in the sign of the gas velocity v. From a numerical point of view, this requires the use of appropriate differentiation matrix.

5.1.2 Numerical solution

The problem has no analytical solutions. Reference solution can be computed on a fixed uniform grid with $N = 1500$ nodes.

Figure 1 show the evolution of the concentration and temperature at $t = 0{,}100, \ldots, 1000$. Time integration is performed using the solver $ode15s$ with $RelTol = 10^{-3}$ and $AbsTol = 10^{-6}$. The elements of the concentration and temperature vectors are interlaced so as to confer a banded structure to the Jacobian matrix, which can be specified using a function JPattern.

The best results were obtained with the following choice of moving grid parameters and digital derivation operators: $\alpha = 1$, $\kappa = 2$, $\tau = 10^{-2}$, $D_1 = four - point - biased - upwind$ and

$D_2 = three - point - centered.$

For approximating the derivatives of order 1 we used finite differences to four points biased and downstream derivatives of order 2 centered finite differences to 3 points, [7].

The numerical results obtained with a grate $N = 100$ compared points to those of a fixed uniform grid, even at $N = 1500$ nodes show a very good precision numerical solutions with a gain in computing time of the order of 12.8 (427,141 seconds against 5 458.7 seconds).

5.2 A model of flame Propagation

5.2.1 Problem description

Our second example is a model of flame propagation consisting of two coupled equations for mass density and temperature. This is a problem that simulates several characteristics of flame propagation. A heat source gives rise to a flame immediately two steep wave fronts propagate in opposite directions, one representing the mass density and the other temperature [8, 10, 11]. The problem of flame propagation was also discussed in [3].

The problem consists of a system of two partial differential equations (PDEs) supplemented by initial conditions (ICs) and boundary conditions (BCs). Specifically, it is determined two functions $u = u(z,t)$ and $v = v(z,t)$ solutions:

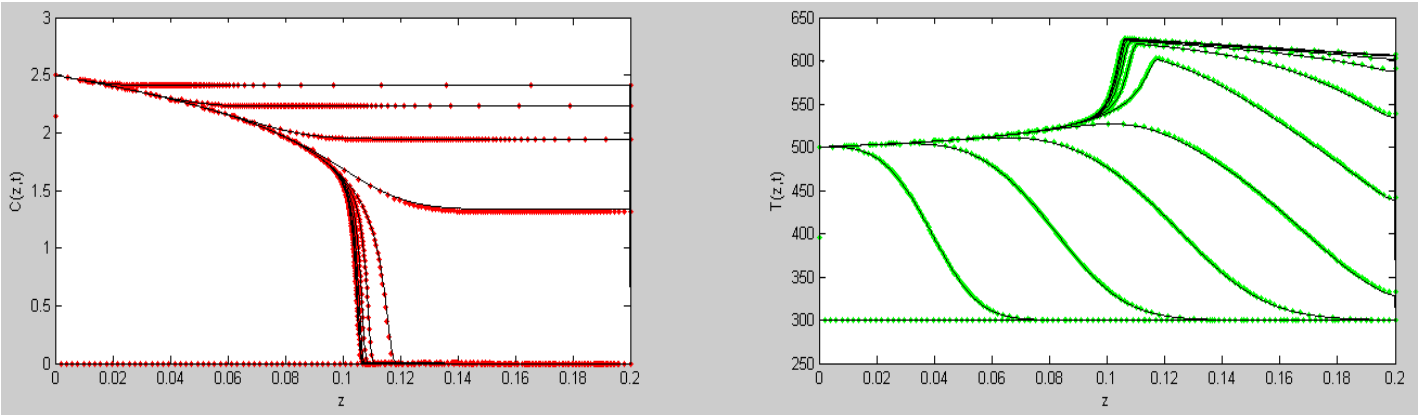


Figure 1: Methanisation problem: the concentration and temperature times $t = 0,\ 100,\ \ldots,1000$; comparison of the solutions obtained with a moving grid to $N = 100$ nodes and those of a uniform fixed grid with $N = 1500$ nodes.

$$u_t = u_{zz} - uf(v),\ \ 0 < z < 1,\ \ 0.3cm0 < t \leq 0.006; \qquad (64)$$

$$v_t = v_{zz} - uf(v),\ \ 0 < z < 1,\ \ 0 < t \leq 0.006; \qquad (65)$$

The initial condition are given by

$$u_t = u_{zz} - uf(v),\ \ 0 < z < 1,\ \ 0 < t \leq 0.006; \qquad (66)$$

$$v_t = v_{zz} - uf(v),\ \ 0 < z < 1,\ \ 0 < t \leq 0.006; \qquad (67)$$

and the boundary conditions are

$$u_z(0,t) = u_z(1,t) = 0,\ \ t \geq 0; \qquad (68)$$

$$v_z(0,t) = 0,\ \ t \geq 0; \qquad (69)$$

$$v(1,t) = 0.2 + \frac{t}{0.0002},\ \ t \leq 0.0002; \qquad (70)$$

$$v(1,t) = 1.2,\ \ t \geq 0.0002. \qquad (71)$$

where the function u denotes a mass density, v temperature and $f(v)$ a heat source given by

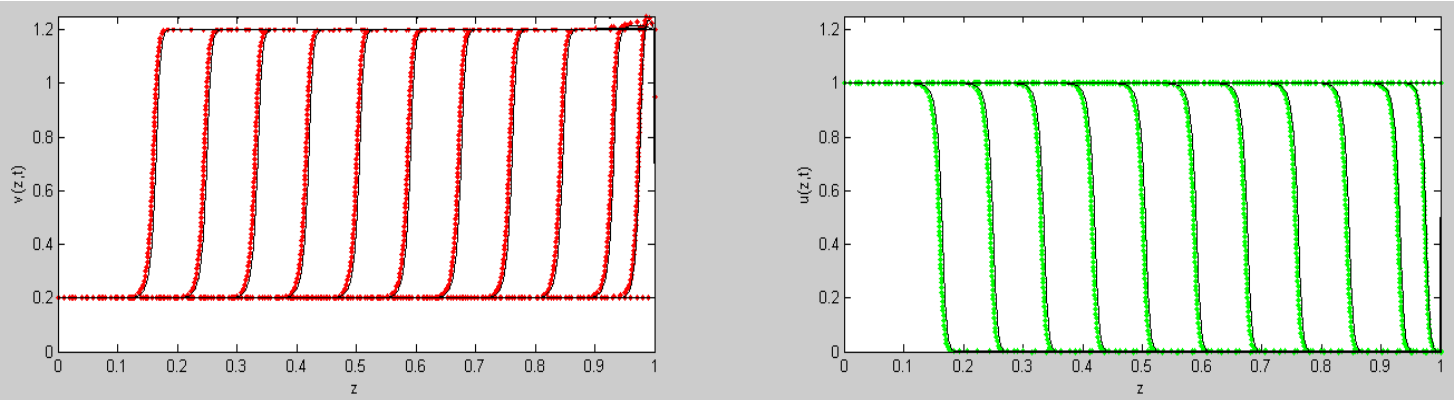
$$f(v) = 3.52 \times 10^6 exp(\frac{-4}{v})$$

5.2.2 Numerical solution

The problem (64)-(71) has no analytical solutions. The problem is considered at the following times: $t = 0.15 \times$

10^{-3}, 0.3×10^{-3}, $0.6 \times 10^{-3}: 0.6 \times 10^{-3}: 0.6 \times 10^{-2}$. The heat source located at $z = 1$ generates a flame front that propagates from right to left at an almost constant speed. From a numerical point of view, the main challenge in this problem is to accurately capture the ignition phase and to subsequently reproduce the correct propagation speed of the flame front. To this end, a relatively large number of points, e.g., $N = 1500$, is needed when using finite differences on fixed uniform grids. The problem is solved on the time interval $[0,1]$ and time integration is performed using ode15s with $AbsTol = 10^{-3}$ and $RelTol = 10^{-3}$. A moving grid, which concentrates the nodes where they are needed, could therefore be advantageous in this case. The following tuning parameter values are selected: $\alpha = 0{,}5$, $\kappa = 1$ $\tau = 10^{-6}$ and $D_2 = three - point - centered$.

Time integration is performed using ode15s with $AbsTol = 10^{-3}$ and $RelTol = 10^{-3}$. Very satisfactory numerical results are obtained, which are represented in Figure 2.



Fgiure 2: Flame propagation model Dwyer-Sanders: fronts temperature and mass density at times $t = 0.15$, 0.3×103, $0.6 \times 103: 0.6 \times 103, 0.6 \times 102$, comparison of solutions for a uniform fixed grid with $N = 1500$ nodes and a moving grid with $N = 100$ nodes.

5.3 A classical Korteweg-de-Vries equation

5.3.1 Problem description

In mathematics, the equation of Korteweg[1] and de Vries (KdV for short) introduced in 1895 is a mathematical model to describe the small amplitude wave behavior in shallow waters. It is a well known example of partial differential equation nonlinear whose solutions are known exactly. These solutions can be calculated by inverse diffusion processing (same principle as the resolution of the heat equation), [1, 3]. Over the years, the KdV equation has found application in several areas such as plasma physics, mixtures of bubbles, the non-harmonic crystals, etc. [2, 3]. The general equation of KdV equation is:

$$u_t = -6uu_z - u_{zzz}, \quad -30 < z < 70, \quad t \geq 0; \quad (72)$$

with initial conditions

$$u(z,0) = 0.25sech^2(0.5\tfrac{3}{2}z), \quad -30 \leq z \leq 70; \quad (73)$$

and boundary conditions

$$u(-30,t) = 0, \quad t \geq 0; \quad (74)$$

$$u(70,t) = 0, \quad t \geq 0. \quad (75)$$

which combines the effect of nonlinearity and dispersion.

[1] Diederik Korteweg, 1848-1941, Dutch mathematician applied

$$u(z,t) = 0.5ssech^2[0.5\sqrt{(s)}(z - st)], \qquad (76)$$

5.3.2 Numerical solution

In the following, the propagation of a single soliton is investigated numerically. $u(z,t)$ of (76) is the analytical solution to $(72) - (75)$ and serves as a standard by which the accuracy of the numerical solution can be assessed. Particular attention must be paid to the selection of finite difference schemes for the approximation of the spatial derivatives. Here, the second form of the equation is used, [3]. The first-order derivative term is computed using a five-point biased upwind scheme, and the third-order derivative term is computed using stagewise differentiation, i.e., $\tilde{u}_z zz = D1(D1(D1\tilde{u}))$, with a three-point centered differentiation matrix D_1. Figure 3 shows the propagation of a single soliton. Time integration is performed using ode15s with $AbsTol = 10^{-3}$ and $RelTol = 10^{-3}$. The following tuning parameter values are selected: $\alpha = 10^{-2}$, $\kappa = 2$, $\tau = 10^{-4}$. The numerical results obtained with a moving grid $N = 100$ points compared to those of a uniform fixed grid with $N = 1500$ nodes show, Figure 3, a passable accuracy of numerical solutions.

6. CONCLUSIONS AND FUTURE WORKS

In this paper we have described a moving-grid method for one-dimensional time-dependent PDEs. Satisfactory numerical results are obtained for a given function with steep slopes in all directions, a scalar parabolic model describing a flame propagation front with a steep steady-state profile, a methanisation in a reactor problem, and a classical Korteweg-de-Vries equation. The effects of the different method parameters are clear. Smoothing both in time and in space direction is necessary to prevent irregular adaptive grids. Standard choices for the smoothing parameters can be given. There are some possibilities to modify and improve the moving-grid PDEs that control the grid movement. Further, more attention should be paid to make the time integration process more efficient. Finally, it is planned to test the method on systems of nonlinear PDEs with more general boundary conditions.

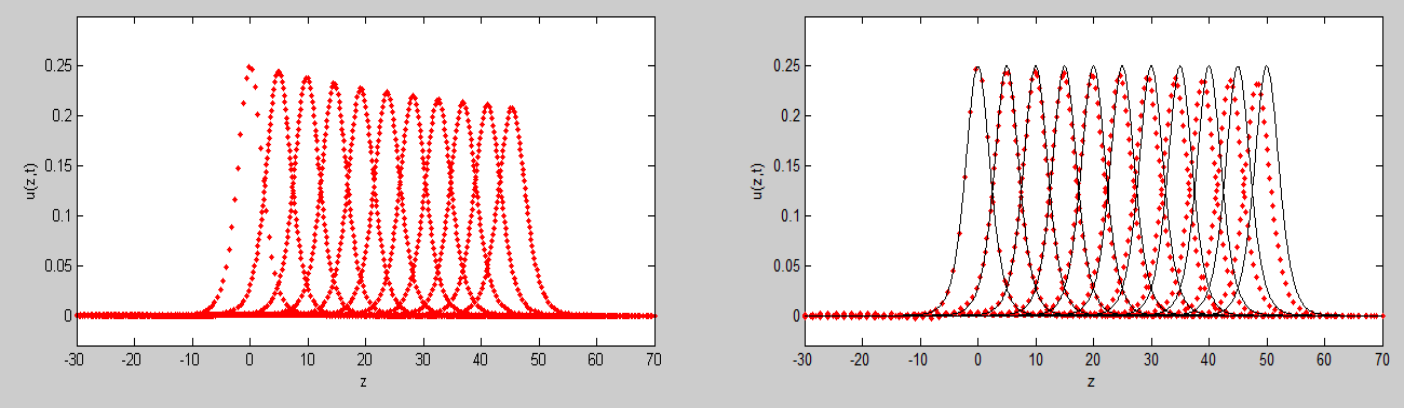


Figure 3: Korteweg-de Vries (KdV) equation: soliton at times $t = 0,\ 10,\ \ldots,\ 100$; comparison between exact solution with $N = 1500$ and numerical solution with a moving grid to $N = 100$ nodes.

ACKNOWLEDGEMENTS

We are indebted to professor SOME Longin of University of Ouagadougou for placing at our disposal some codes of MATLAB about numerical method. We would also like to thank the referees for their detailed comments and helpful suggestions.

REFERENCES

[1]. M. Bakker, *Software for semi-discretization of time-dependent partial differential equations in one space variable* (1977) Rep. NW 52/77, Mathematisch Centrum, Amsterdam, The Netherlands.

[2]. G. Beckett, J. A Mackenzie, A. Ramage and D. M. Sloan, *On The Numerical Solution of One-Dimensional PDEs Using Adaptive Methods based on Equidistribution*. Journal of Computational Physics vol. 167 (2001)

pp. 372-392.
[3]. A. Jeffrey and S. Xu, Exact solutions to the Korteweg-de Vries-Burgers Equation, Wave Motion, 11(1989) 559-564.
[4]. A. Kurganov and E. Tadmor. *New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations.* Journal of Computational Physics, 160:241-282, 2000.
[5]. N. Maman, *Algorithmes dâ€™adaptation dynamique de maillages en Ã©lÃ©ments finis. Application Ã des Ã©coulements rÃ©actifs instationnaires*, PhD thesis, UniversitÃ© Paul Sabatier de Toulouse, 1992.
[6]. B. SangarÃ©, O. Diallo and L. SomÃ©, *A New MATLAB Implementation and Analysis of A Moving Grid Method For Systems of One-Dimensional Time-Dependent Partial Differential Equations Based on The Equidistribution Principle*, Int. J. Appl. Math. 25(2012), no.1, 66-85.
[7]. A. Vande Wouwer, P. Saucez ,W. E. Schiesser, S. Thompson, *A MATLAB implementation of upwind finite differences and adaptive grids in the method of lines* Journal of Computational and Applied Mathematics 183 (2005) 245â€“258.
[8]. W.E. Schiesser, *The Numerical Method of Lines: Integration of Partial Differential Equations,* Academic Press, San Diego, 1991.
[9]. P. Saucez, L. Some and A. Vande Wouwer *Matlab Implementation of a Moving Grid Method based on the Equidistribution Principle* Fourth International Conference on Advanced COmputational Methods in ENgineering (ACOMEN 2008)
[10]. A. Vande Wouwer, Ph. Saucez, and W.E. Schiesser. *Simulation of distributed parameter systems using a matlab-based method of lines toolbox-chemical engineering applications.* Industrial Engineering and Chemistry Research, 43:3469â€“3477, 2004.
[11]. Verwer, J.G, Blom, R.M. Furzeland, P. A. Zegeling, *A moving grid method for one-dimensional PDEs based on the method of lines. In Adaptive methods for Partial Differential Equations*, J.E. Flaherty, P.J. Paslow, M.S. Shepard, and J.D. Vasilakis (Eds.), SIAM, Philadelphia, Pa., (1989) 160-175.
[12]. P. A. Zegeling, *Moving grid methods for time-dependent partial differential equations*, CWI TRACT 94 (1993), 47-60 and 92-95.