Splitting Methods for Mixed Hyperbolic-Parabolic Systems

Alf Gerisch*, David F. Griffiths†, Rüdiger Weiner*, and Mark A. J. Chaplain†

*Institut für Numerische Mathematik, Fachbereich Mathematik und Informatik
Martin-Luther Universität Halle-Wittenberg
Postfach, 06099 Halle (Saale), Germany
Fax: +49 (0)345 5527004
E-mail: {gerisch, weiner}@mathematik.uni-halle.de

†Department of Mathematics
University of Dundee
Dundee, DD1 4HN, U.K.
Fax: +44 (0)1382 345516
E-mail: {dfg, chaplain}@mcs.dundee.ac.uk

In this paper we present two different approaches to the numerical solution of a system of coupled parabolic–hyperbolic partial differential equations (PDEs). The first approach uses a finite difference scheme based on a splitting of the PDE system. This method is computationally inexpensive but is prone to generating spurious oscillations in the solution in certain regions of the spatial domain. These lead to incorrect negative solution values. The second approach uses the method of lines with special attention being given to preserving the positivity of the solution. Here we use splitting techniques for the solution of the resulting system of stiff ordinary differential equations. The performance of the two approaches on a model problem from mathematical biology is compared and discussed and conclusions drawn.

Keywords: Mixed parabolic–hyperbolic PDE system, Finite difference approximation, Method of lines, Splitting methods

I. INTRODUCTION

We consider the hyperbolic-parabolic system of partial differential equations (PDEs)

$$\frac{\partial n}{\partial t} = -\nabla \cdot (nG_1(c)\nabla c) + g_1(n,c) \tag{1.1a}$$

This manuscript appears as Numerical Analysis Report NA/186, University of Dundee, December 1998.

$$\frac{\partial c}{\partial t} = \Delta c - \lambda c + g_2(n,c) \tag{1.1b}$$

on the interior of the unit square $(0,1)\times(0,1)$ in space and for $t \in (0, t_{final}]$ in time for the unknown functions $n(x,y,t)$ and $c(x,y,t)$ where ∇ denotes the vector $(\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ and Δ is the Laplace operator. The quantity n is convected by a velocity field which depends nonlinearly on the quantity c and its gradient. For $G_1(c) \geq 0$, n will be convected towards areas of larger c values. The quantity c is the solution of the diffusion equation with nonlinear source term (1.1b) in this case but might, in other applications, be described by a differential equation of different type (for example, an ordinary differential equation (ODE)). Beside this coupling of the equations, there is also a coupling of the system of PDEs via the source terms g_1 and g_2. The functions G_1, g_1 and g_2 are prescribed functions of n and c. The equations may be termed a "reaction-diffusion-convection" system.

The system of PDEs is supplied with initial conditions

$$n(x,y,0) = n_0(x,y), \quad c(x,y,0) = c_0(x,y) \tag{1.2}$$

and appropriate boundary conditions consisting of one condition for c at every point of the boundary and Dirichlet boundary conditions for n on inflow boundaries. There are no boundary conditions for n on outflow boundaries. In this paper we will concentrate on the following boundary conditions

$$n(1,y,t) = n_r(y) \quad \text{(assuming inflow on the right boundary) and} \tag{1.3}$$

$$c(0,y,t) = c_l(y), \quad c(1,y,t) = c_r(y), \quad c_y(x,0,t) = c_y(x,1,t) = 0. \tag{1.4}$$

The functions n_0, c_0, n_r, c_l and c_r are given such that initial and boundary conditions are consistent. The algorithms presented here have to be adjusted slightly to other boundary conditions.

Problems of this or similar kind and our motivation to consider them arise for instance in biomathematical models describing tumour induced angiogenesis [5, 1]. Our test problem is such a model and is derived and described in greater detail in [5, 8].

Example 1. Equations (1.1a, 1.1b) with

$$G_1(c) = \kappa, \tag{1.5}$$

$$g_1(n,c) = \mu n(1-n)\max\{0, c-c^\star\} - \beta n, \tag{1.6}$$

$$g_2(n,c) = -\frac{\alpha n c}{\gamma + c}, \tag{1.7}$$

initial conditions

$$n_0(x,y) = \begin{cases} 1 : x \geq 0.95 \text{ and } y \in \bigcup_{i=1}^{4} \left[y_i \pm \frac{35}{1000}\right], y_i = 0.2, 0.36, 0.5, 0.64, 0.8 \\ 0 : \text{otherwise} \end{cases} \tag{1.8}$$

$$c_0(x,y) = \cos\left(\frac{\pi}{2}x\right), \tag{1.9}$$

boundary conditions defined as

$$n_r(y) = n_0(1,y), \tag{1.10}$$

$$c_l(y) = 1, \qquad c_r(y) = 0, \tag{1.11}$$

and parameter values (typical for the application [5])

$$\alpha = 10,\ \beta = 4,\ \gamma = 1,\ \kappa = 0.7,\ \lambda = 1,\ \mu = 100,\ c^\star = 0.2\,. \tag{1.12}$$

There is a variety of approximate solution methods for pure hyperbolic or pure parabolic PDEs and we make use of these by decoupling the system (1.1a, 1.1b) through splitting techniques. In particular, we treat the hyperbolic equation (or parts arising from it) with explicit methods and the parabolic equation implicitly.

In Section II., a fully discrete splitting method is proposed. It is based on a splitting of the PDE system and solves in turn the easier subproblems (1.1a) for n and (1.1b) for c (independently of each other) with small time steps. ADI methods combined with Strang splitting are then used for the solution of the parabolic subproblem and MacCormack's method combined with Strang splitting is the main tool for advancing the hyperbolic equation. The time step size will be selected such that the Courant number in the hyperbolic subproblem satisfies a stability condition.

In Section III., we follow a method of lines (MOL) approach. Here we consider the case that n and c describe concentrations or densities, and as such, are always non-negative. We therefore expect the functions G_1, g_1 and g_2 and the boundary conditions to be of such a structure that the exact solution of the system (1.1a,1.1b) is non-negative for all times $t > 0$ provided the initial conditions for n and c are non-negative.

The first and second order spatial derivatives of c in the system are approximated by standard second order central differences but the convection operator in (1.1a) is approximated by a flux-limited second order discretization similar to the one proposed in [10]. The aim of the limiting procedure is that the non-negativity property of the PDE's solution is carried over to the solution of the ODE system obtained by the spatial approximation. The result is a large stiff system of ODEs which is solved by splitting the right hand side into the terms representing the convection and the diffusion/reaction of the underlying PDE system. Within this splitting, explicit Runge-Kutta methods (RKMs) are used to solve the resulting systems of ODEs with the right hand side from the convection discretization and implicit schemes are employed for ODE systems with the right hand side representing diffusion and reaction (these are themselves splitting schemes). It is shown that the method is second order in time.

Section IV. contains a discussion and comparison of the results of the two different algorithms applied to the test problems.

An equidistant linear grid covers the spatial domain having M intervals in each spatial direction with grid spacing $h = 1/M$. The index i refers always to grid values along the y-axis and the index j always to grid values along the x-axis. We define $x_j = jh$, $y_i = ih$ and denote the continuous time approximations to $n(x_j, y_i, t)$ and $c(x_j, y_i, t)$ by $N_{ij}(t)$ and $C_{ij}(t)$, respectively. The dependence on t will be omitted if it is clear from the context. The time scale will be covered by an irregular grid with grid spacing τ_k defined adaptively by an appropriate step size control which generates grid points $t_{k+1} = t_k + \tau_k$ for $k = 0, 1, \ldots$ and $t_0 = 0$. In the following, the variable time step size τ_k will be denoted by τ (without subscript). N_{ij}^k and C_{ij}^k denote the point approximations to $n(x_j, y_i, t_k)$ and $c(x_j, y_i, t_k)$, respectively. The index or superscript k always refers to points on the time scale.

For a given grid function U_{ij}, the standard second order central difference operator in the x-direction, δ_x^2, is defined as

$$\delta_x^2 U_{ij} = U_{i,j+1} - 2U_{ij} + U_{i,j-1}\,. \tag{1.13}$$

The central difference (Δ_{0x}), forward difference (Δ_x) and backward difference (∇_x) operators are defined as

$$\Delta_{0x}U_{ij} = U_{i,j+1} - U_{i,j-1},\ \Delta_x U_{ij} = U_{i,j+1} - U_{i,j},\ \nabla_x U_{ij} = U_{i,j} - U_{i,j-1}, \quad (1.14)$$

respectively. The operators $\delta_y^2, \Delta_{0y}, \Delta_y$ and ∇_y for the y-direction are defined in a similar manner.

II. THE FINITE DIFFERENCE APPROACH

The general idea of this approach is to solve the system (1.1a, 1.1b) by advancing each of the solutions n and c independently of each other (but in turn) by small time steps τ.

FD Algorithm

$k = 0, t_k = 0;$ /*Initialization*/
$C_{ij}^k = c_0(x_j, y_i), \quad (i = 0, 1, \ldots, M,\ j = 1, 2, \ldots, M-1);$
$N_{ij}^k = n_0(x_j, y_i), \quad (i = 0, 1, \ldots, M,\ j = 0, 1, \ldots, M-1);$
while $(t_k < t_{final})$ *do* /*Main loop*/
(1) Compute $C_{ij}^{k+1} \quad (i = 0, 1, \ldots, M,\ j = 1, 2, \ldots, M-1)$
from C_{ij}^k using (1.1b) with $n(x_j, y_i, t)$ fixed as N_{ij}^k;
(2) Compute $N_{ij}^{k+1} \quad (i = 0, 1, \ldots, M,\ j = 0, 1, \ldots, M-1)$
from N_{ij}^k using (1.1a) with $c(x_j, y_i, t) = C_{ij}^k$ or $= C_{ij}^{k+1}$
as required by the method employed;
(3) $t_{k+1} := t_k + \tau, k := k+1;$
Compute τ for next step;
end while;
End of FD Algorithm.

In the following subsections, we explain the steps (1) to (3) in greater detail.

A. Advancing the Solution of the Parabolic Equation

We consider the numerical solution of the equation (1.1b) from time t_k to t_{k+1} with a time step τ. The grid values N_{ij}^k and C_{ij}^k of n and c at time t_k are given and we fix the value of $n(x_j, y_i, t)$ in time to N_{ij}^k in (1.1b) such that a single PDE for the unknown function c results

$$\frac{\partial c}{\partial t} = \Delta c - \lambda c + g_2(N_{ij}^k, c). \quad (2.1)$$

To avoid the time step size being restricted by diffusion, an implicit method is used to compute the grid values C_{ij}^{k+1} of c at time t_{k+1}. The spatial derivatives in (2.1) are approximated by standard second order central differences. If the resulting large nonlinear equation system is solved by a method which makes use of the Jacobian of the system then the resulting linear algebraic systems have a bandwidth of $\approx 2M$ and will therefore be expensive to solve. We will therefore use an ADI method for the approximation of (2.1) to reduce the bandwidth. This faces us with the new problem of how to include the source term in the scheme. Our proposed solution is to apply a Strang splitting [16, p186] such that the right hand side of (2.1) is split into the two terms $\Delta c - \lambda c$ and $g_2(N_{ij}^k, c)$. This kind of splitting requires the solution of equation (2.1) with $g_2(N_{ij}^k, c) \equiv 0$, which is equivalent to

$$\frac{\partial s}{\partial t} = \Delta s, \text{ where } s(x,y,t) = c(x,y,t)e^{\lambda(t-t_k)}. \tag{2.2}$$

The values of s at time t_k are those of c at t_k but the boundary conditions for s are those of c multiplied by $e^{\lambda(t-t_k)}$. The procedure we have just described for Stage (1) of the FD algorithm is summarized in

FD Subalgorithm (1)

(1.1) Solve $c_t = g_2(N_{ij}^k, c)$ with initial condition $c(x_j, y_i, t_k) = C_{ij}^k$ from t_k to $t_k + \tau/2$ with a method of consistency order (at least) one in time and denote the result by $\bar{C}_{ij}$. (We use the implicit Euler method.);

(1.2) Solve $s_t = \Delta s$ with initial condition $S_{ij} = \bar{C}_{ij}$ and boundary conditions for s as those of c multiplied by $e^{\lambda(t-t_k)}$ from t_k to $t_k + \tau$ by an ADI method with second order accuracy in space and at least first order in time and denote the result by $\bar{S}_{ij}$;

(1.3) Rescale $\hat{C}_{ij} = e^{-\lambda\tau}\bar{S}_{ij}$;

(1.4) Solve $c_t = g_2(N_{ij}^k, c)$ with initial condition $c(x_j, y_i, t_k + \tau/2) = \hat{C}_{ij}$ from $t_k + \tau/2$ to $t_k + \tau$ with a method of consistency order (at least) one in time and denote the result by C_{ij}^{k+1} (We use the implicit Euler method.);

End of FD Subalgorithm (1).

Remark. *The implicit Euler step $\bar{C}_{ij} = C_{ij}^k + \frac{\tau}{2} g_2(N_{ij}^k, \bar{C}_{ij})$ in step (1.1) where N_{ij}^k is used as a constant (in time) approximation to $n(x_j, y_i, t)$ is the same as a step with the first order one-stage Radau-IA method [18, p224] applied to the problem with time-continuous approximation to $n(x_j, y_i, t)$. In step (1.4), we would prefer to use a better approximation to $n(x_j, y_i, t + \tau/2)$ than N_{ij}^k but these values are not given to us. The equation in the implicit Euler steps (1.1) and (1.4) for the source term $g_2(n,c)$ defined in Example I. can be solved exactly.*

Lemma 2.1. *The FD Subalgorithm (1) for the solution of equation (2.1) has a local accuracy of $\mathcal{O}(\tau + h^2)$.*

Proof. At least first order in time is used in each step and second order in space. ■

In step (1.2) of the algorithm we have to solve the following problem from t_k to $t_k + \tau$

$$\frac{\partial s}{\partial t} = \Delta s = s_{xx} + s_{yy}, \tag{2.3a}$$

$$s(x_j, y_i, t_k) = \bar{C}_{ij}^k \text{ (at all grid points)}, \tag{2.3b}$$

$$\begin{aligned} s(0,y,t) &= c_l(y)e^{\lambda(t-t_k)}, \quad s(1,y,t) = c_r(y)e^{\lambda(t-t_k)}, \\ s_y(x,0,t) &= s_y(x,1,t) = 0. \end{aligned} \tag{2.3c}$$

We use the Peaceman–Rachford ADI method for the solution of (2.3a-2.3c). It is second order accurate in time and space, and unconditionally stable (see [20, p165]). This method can be regarded as an approximate factorization of the two dimensional Crank–Nicolson method [20, p168]. The method can be written as

$$\left(1 - \frac{r}{2}\delta_x^2\right) S_{ij}^{1/2} = \left(1 + \frac{r}{2}\delta_y^2\right) \bar{C}_{ij} \tag{2.4a}$$

$$\left(1 - \frac{r}{2}\delta_y^2\right) \bar{S}_{ij} \;=\; \left(1 + \frac{r}{2}\delta_x^2\right) S_{ij}^{1/2}, \tag{2.4b}$$

where the mesh ratio $r = \frac{\tau}{h^2}$. The boundary conditions (2.3c) prescribe the values $S_{i0}^{1/2} = c_l(y_i)e^{\lambda\tau/2}$ and $S_{iM}^{1/2} = c_r(y_i)e^{\lambda\tau/2}$ (and this choice guarantees an overall second order method as shown in [20]). The remaining unknowns in (2.4a) are those with indices $i = 0, 1, \dots, M$ and $j = 1, 2, \dots, M-1$. A closer look at (2.4a) reveals that equations with different indices i on the left have no common unknowns and the linear system (2.4a) splits up into an $(M-1)$-dimensional tridiagonal system of linear equations for each i. This system reads for $i = 1, 2, \dots, M-1$

$$\texttt{tridiag}\left(-\frac{r}{2}, 1+r, -\frac{r}{2}\right) \begin{pmatrix} S_{i1}^{1/2} \\ S_{i2}^{1/2} \\ \vdots \\ S_{i(M-1)}^{1/2} \end{pmatrix} = \left(1 + \frac{r}{2}\delta_y^2\right) \begin{pmatrix} \bar{C}_{i1} \\ \bar{C}_{i2} \\ \vdots \\ \bar{C}_{i(M-1)} \end{pmatrix} + \mathbf{b}_i \tag{2.5}$$

with $\mathbf{b}_i$ of dimension $M-1$ containing the known boundary values from the left hand side,

$$\mathbf{b}_i = \left(\frac{r}{2}S_{i0}^{1/2},\ 0,\ \dots,\ 0,\ \frac{r}{2}S_{iM}^{1/2}\right)^T. \tag{2.6}$$

For $i = 0$ and $i = M$, two rows of fictitious grid points are introduced at y_{-1} and y_{M+1}, respectively. The no flux boundary conditions at $y = 0$ and $y = 1$ are approximated by central differences $\frac{\Delta_{0y}}{2h}$ resulting in

$$\bar{C}_{1j} - \bar{C}_{-1j} = 0 \quad\text{and}\quad \bar{C}_{(M+1)j} - \bar{C}_{(M-1)j} = 0. \tag{2.7}$$

These equations are used to eliminate the fictitious grid points in a standard way when imposing (2.4a) for $i = 0$ and $i = M$. This leads to linear systems with the same left hand side as in (2.5) and right hand sides defined by

$$\begin{pmatrix} (1-r)\bar{C}_{01} & +r\bar{C}_{11} \\ (1-r)\bar{C}_{02} & +r\bar{C}_{12} \\ & \vdots \\ (1-r)\bar{C}_{0(M-1)} & +r\bar{C}_{1(M-1)} \end{pmatrix} + \mathbf{b}_0 \text{ and } \begin{pmatrix} (1-r)\bar{C}_{M1} & +r\bar{C}_{(M-1)1} \\ (1-r)\bar{C}_{M2} & +r\bar{C}_{(M-1)2} \\ & \vdots \\ (1-r)\bar{C}_{M(M-1)} & +r\bar{C}_{(M-1)(M-1)} \end{pmatrix} + \mathbf{b}_M \tag{2.8}$$

for $i = 0$ and $i = M$, respectively.

It is important to remark that the coefficient matrix of all linear systems is the same and therefore only one LU factorization is necessary in order to solve all $M+1$ systems. Most of the computational effort is devoted to constructing the right hand sides of the systems. Assembling these $M+1$ vectors and the solution of the linear systems could be performed in parallel.

All necessary values of $S_{ij}^{1/2}$ are now computed and we can proceed with the solution of (2.4b). Equations of this system with different indices j on the left hand side are independent and the system again splits up. The Dirichlet boundary conditions supply the values $\bar{S}_{i0} = c_l(y_i)e^{\lambda\tau}$ and $\bar{S}_{iM} = c_r(y_i)e^{\lambda\tau}$ and we have to solve systems when $j = 1, 2, \dots, M-1$ is fixed. Fictitious grid points are introduced at the upper and lower boundary of the domain to deal with the no flux boundary conditions and these are approximated by central differences $\frac{\Delta_{0y}}{2h}$ at the new time level leading to

$$\bar{S}_{1j} - \bar{S}_{-1j} = 0 \quad \text{and} \quad \bar{S}_{(M+1)j} - \bar{S}_{(M-1)j} = 0. \tag{2.9}$$

For each $j = 1, 2, \ldots, M-1$, equation (2.4b) is employed for $i = 0, 1, \ldots, M$ and the fictitious grid points are eliminated with the equations just derived. Altogether, this results in the system

$$\begin{bmatrix} 1+r & -r & 0 & \ldots & 0 & 0 \\ -\frac{1}{2}r & 1+r & -\frac{1}{2}r & \ldots & 0 & 0 \\ & \ddots & \ddots & \ddots & & \\ \vdots & & & & & \vdots \\ & & & \ddots & \ddots & \ddots \\ 0 & 0 & 0 & \ldots & 1+r & -\frac{1}{2}r \\ 0 & 0 & 0 & \ldots & -r & 1+r \end{bmatrix} \begin{pmatrix} \bar{S}_{0j} \\ \bar{S}_{1j} \\ \vdots \\ \bar{S}_{(M-1)j} \\ \bar{S}_{Mj} \end{pmatrix} = \left(1 + \frac{r}{2}\delta_x^2\right) \begin{pmatrix} S_{0j}^{1/2} \\ S_{1j}^{1/2} \\ \vdots \\ S_{(M-1)j}^{1/2} \\ S_{Mj}^{1/2} \end{pmatrix}. \tag{2.10}$$

The coefficient matrix is the same for all these systems and the same comments as after the first step of the ADI scheme apply. Step (1.2) of the algorithm is now completed.

If the initial condition is smooth then the Peaceman–Rachford ADI method generates very good solutions and can be used with large time steps. However, it was observed experimentally that the method requires smaller and smaller time steps in order to obtain an acceptable solution as the initial data becomes less and less smooth. The same is reported for the Crank–Nicholson method in [17]. There, it is advocated that the solution process should start with some steps of the implicit Euler method to smooth the data and then to continue with Crank–Nicholson. This means, in the present context, that we should use the Backward Time Centred Space (BTCS) scheme to execute step (1.2) of the algorithm

$$\bar{S}_{ij} - r\delta_x^2 \bar{S}_{ij} - r\delta_y^2 \bar{S}_{ij} = \bar{C}_{ij}. \tag{2.11}$$

A simple approximate factorization (in order to reduce computational costs) of this is given by

$$\left(1 - r\delta_x^2\right) S_{ij}^{1/2} = \bar{C}_{ij} \tag{2.12}$$

$$\left(1 - r\delta_y^2\right) \bar{S}_{ij} = S_{ij}^{1/2}. \tag{2.13}$$

This scheme is first order accurate in time [9, p310] and second order in space (because second order accurate approximations to the second order derivatives are used). It is also unconditionally stable [9, p313] and we refer to it as the implicit Euler ADI scheme in the following.

The tridiagonal systems (2.12, 2.13) split up in the same way as the stage equations of the Peaceman–Rachford scheme but the right hand sides are much simpler. The coefficient matrices of the small systems are the matrices in (2.5) and (2.10), respectively, when r is replaced by $2r$. There are no difficulties in constructing the right hand sides of (2.12) as these are just the values at the old time level plus the boundary terms $\mathbf{b}_i$ (replace r by $2r$) from the left hand side and it is not necessary to distinguish between $i = 0, M$

and other values of i. The Dirichlet boundary values of $S_{ij}^{1/2}$ for $j = 0$ and $j = M$ are set to $c_l(y_i)e^{\lambda\tau}$ and $c_r(y_i)e^{\lambda\tau}$, respectively. Fictitious grid points are introduced for the solution of (2.13) in the same manner as for the solution of (2.4b).

Experiments established that the implicit Euler ADI method is much less sensitive to non-smooth initial data than the Peaceman–Rachford ADI method and larger time steps are possible while maintaining smooth solutions.

B. Advancing the Solution of the Hyperbolic Equation

We consider the numerical solution of the equation (1.1a) from time t_k to t_{k+1} with a time step τ. The main ingredients for a solution of this problem will be MacCormack's method and, again, splitting schemes. A comparison was made in [8] between several different approaches for the solution of a one-dimensional analogue of our problem (1.1a-1.1b). It was found that a combination of MacCormack's method and splitting schemes worked well and was cost-effective and therefore we will use an adaptation of this approach in two dimensions.

MacCormack's method is based on the second order accurate improved Euler method (Heun's method) [18, p51] defined by the Butcher array in Figure 1. It is best explained

$$\begin{array}{c|cc} & 0 & \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

FIG. 1. Butcher array of the improved Euler method.

in the one dimensional case (see [8]). Consider the ODE $u'(t) = s(u(t))$ with given initial data U^k (e.g. an approximation to $u(t_k)$). A step τ from t_k to t_{k+1} of the improved Euler method for this ODE can be written as

$$Z = U^k + \tau s(U^k) \tag{2.14a}$$
$$W = U^k + \tau s(Z) \tag{2.14b}$$
$$U^{k+1} = \frac{1}{2}(Z + W)\,. \tag{2.14c}$$

For a one dimensional hyperbolic PDE (with velocity function $v(x,t)$)

$$\frac{\partial u(x,t)}{\partial t} = -(v(x,t)u(x,t))_x, \tag{2.15}$$

we substitute $s = -(vu)_x$ in the above scheme and approximate the spatial derivative in (2.14a) by first order forward differences $\frac{\Delta_x}{h}$ and the spatial derivative in (2.14b) by first order backward differences $\frac{\nabla_x}{h}$ (the application of forward and backward differences can be reversed). The resulting scheme is MacCormack's method for equation (2.15) and is second order accurate in time and space [8, p10]. The method is equivalent to the Lax–Wendroff scheme [16, p167] for the constant coefficient case $v \equiv \alpha = const$ and hence stable in the von Neumann sense for Courant numbers $\mathcal{C} = \frac{|\alpha|\tau}{h}$ satisfying $\mathcal{C} \leq 1$. An additional source term on the right of the hyperbolic equation (2.15) can be included by an averaging procedure without sacrificing the order of accuracy or stability [8, p10]. The values of Z are a first order approximation (in time) to grid values of $u(x, t_{k+1})$ and therefore the boundary conditions of u at t_{k+1} apply. Furthermore, the values of v at time t_{k+1} must be used in equation (2.14b) (otherwise the accuracy of the method drops to order one).

The equation which we have to solve as part of step (2) in the FD algorithm is a little more complicated and reads

$$\frac{\partial n}{\partial t} = -(nG_1(c)c_x)_x, \tag{2.16}$$

and we also have to approximate the spatial derivative c_x. This is done by forward differences if the MacCormack stage equation contains a backward difference and *vice versa*. This strategy guarantees an almost centred stencil of the method and results in the following scheme (in two-dimensional notation)

$$\begin{aligned} Z_{ij} &= N_{ij}^k - \frac{\tau}{h}\left(N_{i,j+1}^k G_1\left(C_{i,j+1}^k\right)(C_{\nabla_x})_{i,j+1}^k - N_{ij}^k G_1\left(C_{ij}^k\right)(C_{\nabla_x})_{ij}^k\right) && \text{(2.17a)} \\ W_{ij} &= N_{ij}^k - \frac{\tau}{h}\left(Z_{ij} G_1\left(C_{ij}^{k+1}\right)(C_{\Delta_x})_{ij}^{k+1} - Z_{i,j-1}^k G_1\left(C_{i,j-1}^{k+1}\right)(C_{\Delta_x})_{i,j-1}^{k+1}\right) && \text{(2.17b)} \\ N_{ij}^{k+1} &= \frac{1}{2}\left(Z_{ij} + W_{ij}\right), && \text{(2.17c)} \end{aligned}$$

where we use the notation

$$(C_{\nabla_x})_{ij}^k = \frac{1}{h}\left(C_{ij}^k - C_{i,j-1}^k\right) \tag{2.18}$$

$$(C_{\Delta_x})_{ij}^k = \frac{1}{h}\left(C_{i,j+1}^k - C_{ij}^k\right) \tag{2.19}$$

for backward and forward difference approximations to $c_x(x_j, y_i, t_k)$.

The algorithm presented by (2.17a-2.17c) has a straight-forward extension to two dimensions by including the approximations of the convection in the y-direction. Unfortunately, this approach results in an even greater restriction of the allowable time step size τ for reasons of stability and leads to an unacceptable increase in computational cost. We circumvent this difficulty by using a three term Strang type splitting of (1.1a) which uses one dimensional hyperbolic equations as subproblems.

With MacCormack's method in one dimension at hand we are now ready to state the algorithm for step (2) of the FD algorithm. Assuming that we are given N_{ij}^k, C_{ij}^k and C_{ij}^{k+1} as approximations of $n(x_j, y_i, t_k)$, $c(x_j, y_i, t_k)$ and $c(x_j, y_i, t_k + \tau)$, respectively, we perform a time step τ to obtain N_{ij}^{k+1} as an approximation to $n(x_j, y_i, t_k + \tau)$, the solution of equation (1.1a) at time $t_k + \tau$.

FD Subalgorithm (2)

(2.1) Solve $\frac{\partial}{\partial t} v_{ij}^{(1)} = g_1(v_{ij}^{(1)}, C_{ij}^k)$, $v_{ij}^{(1)}(0) = N_{ij}^k$ to obtain $v_{ij}^{(2)}(0)$ as an approximation to $v_{ij}^{(1)}(\tau/2)$ for $i, j = 0, 1, \ldots, M$;

(2.2) Solve $v_t^{(2)} = -(v^{(2)} G_1(C_{ij}^k)(c_x)_{ij}^k)_x$ with initial data $v_{ij}^{(2)}(0)$ from step (2.1) and constant boundary values at $x = 1$ to obtain $v_{ij}^{(3)}(0)$ as an approximation to $v_{ij}^{(2)}(\tau/2)$ ($i = 0, 1, \ldots, M$ and $j = 0, 1, \ldots, M-1$) employing MacCormack's method as described above;

(2.3) Solve $v_t^{(3)} = -(v^{(3)} G_1(C_{ij}^{\{k,k+1\}}(c_y)_{ij}^{\{k,k+1\}})_y$ with initial data $v_{ij}^{(3)}(0)$

from step (2.2) to obtain $v_{ij}^{(4)}(0)$ as an approximation to $v_{ij}^{(3)}(\tau)$ $(i = 0, 1, \ldots, M$ and $j = 0, 1, \ldots, M-1)$ using MacCormack's method as described above but adjusted for the y-direction. The first stage of MacCormack's method uses approximations of c and c_y at the kth time level (C_{ij}^k) whereas the second step uses those at time level $k+1$ (C_{ij}^{k+1});

(2.4) Solve $v_t^{(4)} = -(v^{(4)} G_1(C_{ij}^{k+1})(c_x)_{ij}^{k+1})_x$ with initial data $v_{ij}^{(4)}(0)$ from step (2.3) to obtain $v_{ij}^{(5)}(0)$ as an approximation to $v_{ij}^{(4)}(\tau/2)$ $(i = 0, 1, \ldots, M$ and $j = 0, 1, \ldots, M-1)$ employing MacCormack's method as described above. Constant boundary values at $x = 1$ are used and the correct choice is discussed below;

(2.5) Solve $\frac{\partial}{\partial t} v_{ij}^{(5)} = g_1(v_{ij}^{(5)}, C_{ij}^{k+1})$ with initial data $v_{ij}^{(5)}(0)$ from step (2.4) to obtain N_{ij}^{k+1} as an approximation to $v_{ij}^{(5)}(\tau/2)$ and $n(x_j, y_i, t_k + \tau)$ for $i = 0, 1, \ldots, M$ and $j = 0, 1, \ldots, M-1$;

End of FD Subalgorithm (2).

We cannot use the boundary values of n at $x = 1$ in the intermediate steps of the splitting algorithm because the intermediate solutions are not related to these values. An approach to correct intermediate boundary values is given in [15, 14]. We require boundary values in steps (2.2) and (2.4) only because steps (2.1) and (2.5) are initial value problems and step (2.3) is pure convection in the y-direction and the y-boundaries are outflow boundaries and no condition is imposed there. On $x = 1$ we will use constant boundary conditions in steps (2.2) and (2.4) because the conditions for n are time-independent. In step (2.2) we use the result of step (2.1) at $x = 1$ as boundary condition because the source term g_1 has an impact also on the boundary and this is just approximated by step (2.1). Similarly, g_1 has an impact on the values along $x = 1$ in step (2.5) and the result of step (2.5) for $x = 1$ should be the correct boundary data of n. Therefore correct initial data for step (2.5) is required along $x = 1$. Our aim is $v^{(5)}(1, y, \tau/2) = n_r(y)$ and we prescribe initial values $v^{(5)}(1, y, 0)$ such that we arrive at the desired result. We write the initial condition in terms of the final result and obtain

$$
\begin{aligned}
v^{(5)}(1, y, 0) &= v^{(5)}(1, y, \tau/2) - \frac{\tau}{2} v_t^{(5)}(1, y, \tau/2) + \mathcal{O}(\tau^2) && (2.20) \\
&= n_r(y) - \frac{\tau}{2} g_1(v^{(5)}(1, y, \tau/2), C_{iM}^{k+1}) + \mathcal{O}(\tau^2) && (2.21) \\
&\approx n_r(y) - \frac{\tau}{2} g_1(n_r(y), C_{iM}^{k+1}), && (2.22)
\end{aligned}
$$

where we have used a Taylor expansion of the solution, the boundary condition, the differential equation from step (2.5) and finally truncated the series to obtain an approximation for $v^{(5)}(1, y, 0)$. This approximation of the initial condition of step (2.5) along $x = 1$ is, in fact, also the boundary condition at the final time of step (2.4) and, because we chose constant boundary conditions, also at initial time (hence defining the initial condition of step (2.4) along $x = 1$).

The decision of whether to solve in the x- or y-convection with full or half time steps in the splitting algorithm is important if we are to obtain the largest allowable time step size subject to stability, as we will see in the next section. If the order of x- and y-convection is reversed, then the boundary values at $x = 1$ have to be adjusted accordingly.

C. Time Step Size Control Based on Courant Numbers

When MacCormack's method is applied to solve the advection equation $u_t + \alpha u_x = 0$ with constant velocity α, then a von Neumann stability analysis reveals that it is stable for Courant numbers satisfying $\mathcal{C} \leq 1$, where

$$\mathcal{C} = \frac{|\alpha|\tau}{h}. \tag{2.23}$$

It is well known (and observed in [8]) that the method works best in the sense of accuracy, for Courant numbers $\mathcal{C} \approx 1$.

The velocity function used in steps (2.2) – (2.4) is not constant, but is space dependent. We therefore define the Courant number of our problem to be

$$\mathcal{C} = \frac{\max_{i,j} |G_1(C_{ij}^k) \max\{1/2|(C_{\nabla_x})_{ij}^k|, |(C_{\nabla_y})_{ij}^k|\}|\tau}{h} \tag{2.24}$$

and select the step size τ so that

$$\mathcal{C} \leq \mathcal{C}_{max}, \text{ where } \mathcal{C}_{max} \in (0,1) \tag{2.25}$$

is a safety factor. The values of $(C_{\nabla_x})_{ij}^k$ and $(C_{\nabla_y})_{ij}^k$ are required for the next step in the FD algorithm and consequently there is almost no computational overhead for the time step selection. If the more active direction of convection is treated with half time steps in step (2) of the FD algorithm (this is the x-direction in our description) then we obtain smaller values of $\mathcal{C}$ for a given pair h, τ. This, in turn, means that we can choose a larger τ without violating condition (2.25) and therefore to favour overall time steps.

D. Observations on the Implementation of the Scheme

We implemented the method derived in this section and applied it to Example I. The results can be found in [8]. We used the implicit Euler ADI method in the solution of the parabolic equation (1.1b) because the source term in (1.1b) introduces a non-smoothness in the solution and the Peaceman–Rachford ADI method requires too small time steps. As reported in [8], a problem of small oscillations in the solution of n was observed in regions where the gradient of n was steep. This led to negative values of n at some grid points, which in turn caused the solution to become unbounded at later times. It was found that this problem could be avoided by using sufficiently small time steps but this increased considerably the overall computational effort.

In order to overcome the difficulty of generating spurious negative values of n, in the next section we turn our attention to a method which preserves the positivity of the solution. Furthermore, we will aim for a second order accurate method in time.

III. THE METHOD OF LINES APPROACH

The method of lines (MOL) is a standard approach used for the solution of time-dependent PDEs. The PDE is replaced by a large, in general, stiff system of ODEs by approximating the spatial derivatives through finite differences. The solution of the ODE is a time-continuous function for each spatial grid point and hence defines an approximation to the solution of the PDE restricted to the grid.

Subsection A. describes the spatial approximation of (1.1a-1.1b). We assume in the following that n and c describe non-negative quantities and we want this property to be preserved in the solution of the ODE system. Space discretizations may destroy this property and must therefore be carried out carefully. Our approach leads to a second order accurate semidiscretization away from extremal points of the solution.

In Subsection B., a second order splitting scheme for the numerical solution of the ODE system is developed. It is specially designed with the ODE system obtained in Subsection A. in mind and we address issues related to computational efficiency and positive solutions.

Finally, Subsection C. presents numerical results of the MOL approach applied to Example 1.

A. Approximation of the Spatial Derivatives

As stated in the introduction, we consider the case that the solutions n and c of our PDE system (1.1a-1.1b) are non-negative for all times $t \geq 0$. This property should carry over to the ODE system obtained by spatially discretizing the PDE and we define

Definition. (Positive ODE system, positive semidiscretization)
An ODE system $u'(t) = f(u(t)), t \in [0, t_{final}]$ is called positive *if its solution satisfies $u(t) \geq 0$ for all $t \in [0, t_{final}]$. A semidiscretization of a given PDE (with non-negative solution) is called* positive *if it leads to a positive ODE system.*

Lemma 3.1. *The ODE system $u'(t) = f(u(t)), t \in [0, t_{final}]$ with a continuously differentiable function $f = (f_1, f_2, \ldots f_m)$ is positive if and only if, for any vector $v \in \mathcal{R}^m$ and all $i = 1, 2, \ldots, m$,*

$$v_i = 0, \quad v_j \geq 0 \text{ for all } j \neq i \quad \Rightarrow \quad f_i(v) \geq 0. \tag{3.1}$$

Proof. See [11, p26] ■

In the following, we look at the discretization of the parabolic and hyperbolic equation separately.

The Parabolic Equation (1.1b)

The semidiscretization of a linear diffusion equation (with periodic boundary conditions and without source term) is considered in [11, p28] with respect to positivity. The positivity requirement leads to an order barrier two and it is shown that the standard second order central difference operator leads to a minimal error coefficient. Guided by these results, Equation (1.1b) is discretized in space as

$$\frac{\mathrm{d}\, C_{ij}(t)}{\mathrm{d}\, t} = \frac{1}{h^2}\left(\delta_x^2 C_{ij}(t) + \delta_y^2 C_{ij}(t)\right) - \lambda C_{ij}(t) + g_2(N_{ij}(t), C_{ij}(t)) \tag{3.2}$$

for $i = 0, 1, \ldots, M$ and $j = 1, 2, \ldots, (M-1)$. The given boundary values of c are used on the left and right boundary and rows of fictitious grid points are introduced at $y = -h$ and $y = 1 + h$ in order to approximate the homogeneous Neumann boundary conditions at $y = 0$ and $y = 1$ with second order accuracy by central differences. This process leads to $C_{-1,j} = C_{1j}$ and $C_{M+1,j} = C_{M-1,j}$ for all $j = 1, 2, \ldots, (M-1)$. The second order accurate spatial discretization of (1.1b) by (3.2) is now fully defined.

Lemma 3.2. *The ODE system defined by (3.2) is positive if the prescribed Dirichlet boundary values are non-negative, the source function g_2 is continuously differentiable and if $g_2(n, 0) \geq 0$ for all possible values of n.*

Proof. This result follows from Lemma 3.1. ∎

Remark. *The condition required by Lemma 3.2 is satisfied for the source function g_2 of Example 1 and for the given boundary conditions.*

The Hyperbolic Equation (1.1a)

For the semidiscretization of Equation (1.1a), we will again focus on positivity of the resulting ODE system. A first order upwind space discretization would give us such a system but would also introduce a large amount of numerical diffusion which non-physically damps any travelling wave solution. This diffusive effect is less pronounced on finer spatial meshes but this leads to even larger ODE systems and hence to greater computational cost. On the other hand, higher order (linear) space discretizations are more accurate but prone to introducing "wiggles" in the solution, and possibly negative values, which we would like to avoid. A widely used approach for combining high order methods and oscillation–free solutions is that of *flux limiters*. We will follow the construction of a positive semidiscretization given in [10].

The approximation of two-dimensional convection is built from the one-dimensional setting

$$\frac{\partial w(x,y,t)}{\partial t} = -(v(x,y,t)w(x,y,t))_x, \quad t \geq 0, \quad (x,y) \in (0,1)^2, \tag{3.3}$$

$$w(x,y,0) = w_0(x,y) \geq 0, \tag{3.4}$$

$$w(1,y,t) = w_1(y,t), \tag{3.5}$$

where the quantity w is convected in the x-direction only by the space and time dependent velocity field v. The function $v(x,y,t)$ is assumed to be known for the following discussion. We suppose an inflow boundary at $x = 1$ and therefore require $v(1,y,t) \leq 0$.

The function

$$\tilde{f}(x,y,t) = v(x,y,t)w(x,y,t) \tag{3.6}$$

denotes the flux function of (3.3) and $W_{ij}(t)$ denotes a time-continuous approximation of $w(x_j, y_i, t)$. We define the semidiscrete flux

$$f_{ij} = v(x_j, y_i, t)W_{ij}(t) \tag{3.7}$$

and let $F_{i,j+1/2}$ be a consistent approximation of $\tilde{f}(x_{j+1/2}, y_i, t)$ depending on the neighbouring (semidiscrete) fluxes $f_{il},\ l = 0, 1, \ldots, M$. Here consistency means that $F_{i,j+1/2} = \tilde{f}(x_{j+1/2}, y_i, t)$ if $f_{il} = \tilde{f}(x_{j+1/2}, y_i, t)$ for $l = 0, 1, \ldots, M$, [19].

Equation (3.3) is now approximated in space by the semidiscrete, conservative expression

$$\frac{\mathrm{d}\, W_{ij}(t)}{\mathrm{d}\, t} = -\frac{1}{h}\left(F_{i,j+1/2} - F_{i,j-1/2}\right), \tag{3.8}$$

where we have yet to define the terms $F_{i,j+1/2}$ for $j = -1, 0, 1, \ldots, M-1$. This is our next concern.

The general flux expression

$$F_{i,j+1/2} = f_{ij} + \frac{1}{2}\Phi(r_{i,j+1/2})(f_{ij} - f_{i,j-1}) \tag{3.9}$$

is considered for positive velocities $v(x_j, y_i, t)$ in [10]. The function Φ is called the limiter function, and the ratio r monitors the smoothness of the flux and is defined as

$$r_{i,j+1/2} = \frac{f_{i,j+1} - f_{i,j}}{f_{i,j} - f_{i,j-1}}. \tag{3.10}$$

Remark. *In actual computations we use*

$$r_{i,j+1/2} = \frac{f_{i,j+1} - f_{i,j} + \varepsilon}{f_{i,j} - f_{i,j-1} + \varepsilon}$$

with ε some very small number (e.g. 10^{-30}) in order to avoid division by zero in uniform flow regions [12].

The expression corresponding to (3.9) for negative velocity $v(x_j, y_i, t)$ is obtained by *reflecting* all indices that appear in $f_{i,\cdot}$ about $j + 1/2$ yielding

$$F_{i,j+1/2} = f_{i,j+1} + \frac{1}{2}\Phi(r^{-1}_{i,j+3/2})(f_{i,j+1} - f_{i,j+2}). \tag{3.11}$$

As can be seen, we favour the flux interpolation approach in contrast to the state interpolation approach used in [10]. The reason is that, in our objective equation (1.1a), the velocity depends on c and values thereof are known only at grid points.

Inserting (3.9) and (3.11) in Equation (3.8) gives

$$\frac{\mathrm{d}\,W_{ij}(t)}{\mathrm{d}\,t} = -\frac{1}{h}\left[\left(1 + \frac{1}{2}\Phi(r_{i,j+1/2})\right) - \frac{\Phi(r_{i,j-1/2})}{2r_{i,j-1/2}}\right](f_{ij} - f_{i,j-1}) \tag{3.12}$$

and

$$\frac{\mathrm{d}\,W_{ij}(t)}{\mathrm{d}\,t} = -\frac{1}{h}\left[\left(1 + \frac{1}{2}\Phi(r^{-1}_{i,j+1/2})\right) - \frac{\Phi(r^{-1}_{i,j+3/2})}{2r^{-1}_{i,j+3/2}}\right](f_{i,j+1} - f_{ij}), \tag{3.13}$$

respectively.

Following [10], we require the bracketed terms to be non-negative, that is $1 + \frac{1}{2}\Phi(a) - \frac{\Phi(b)}{2b} \geq 0$ for all $a, b \in \mathcal{R}$. This will lead to a positive semidiscretization (see below). Sufficient conditions on Φ are

$$\Phi(r) = 0 \text{ if } r \leq 0, \quad \text{and} \quad 0 \leq \Phi(r) \leq \delta, \quad \Phi(r) \leq 2r \;\text{ if }\; r > 0. \tag{3.14}$$

Here, $\delta > 0$ serves as a free parameter which can be used to obtain higher accuracy near peaks [10].

With regard to the conditions in Lemma 3.1, the limiter Φ should be chosen as a continuously differentiable function. The limiters proposed in [10] lead, whenever allowed by the conditions (3.14), to the so called κ-methods (including the second order central, second order upwind and third order upwind biased approximation) but are not differentiable for $r \geq 0$. Therefore, we propose using the limiter function due to van Leer, see e.g. [19],

$$\Phi_{VL}(r) = \frac{|r| + r}{1 + |r|}. \tag{3.15}$$

This limiter satisfies the conditions (3.14) for $\delta = 2$ and is continuously differentiable for all $r \neq 0$; it is not differentiable for $r = 0$.

Remark. *Numerical experiments with both limiters indicate the superiority of van Leer's limiter with respect to positivity. However, if positivity of the numerical solution is less important than accuracy then the limiter proposed in [10] should be used.*

Lemma 3.3. *Define $F_{i,j\pm1/2}$ by (3.9) or (3.11) depending on the sign of $v(x_j, y_i, t)$ using van Leer's limiter (3.15). If, depending on the sign of $v(x_j, y_i, t)$, $r_{i,j\pm1/2} > 0$ or $r_{i,j+1\pm1/2} > 0$ (that is away from local extrema of w if v has the same sign locally around the point (x_j, y_i)) then the approximation (3.8) is second order accurate in space for sufficiently smooth functions w and v, i.e*

$$\frac{1}{h}(F_{i,j+1/2} - F_{i,j-1/2}) - (v(x_j, y_i, t)w(x_j, y_i, t))_x = \mathcal{O}(h^2). \tag{3.16}$$

Proof. Taylor expansion of the expressions.

■

The statement of the lemma is that the approximation is second order in smooth, monotone regions of the solution w.

Next we describe exactly which of the flux approximations (3.9, 3.11) to apply for the semidiscretization at a specific point (x_j, y_i) in order to obtain a positive semidiscretization.

1. $W_{ij}(t) > 0$. In this case Lemma 3.1 does not impose any conditions and we compute $F_{i,j\pm1/2}$ by applying (3.9) if $v(x_j, y_i, t) \geq 0$ and (3.11) if $v(x_j, y_i, t) < 0$.
2. $W_{ij}(t) = 0$. Here Lemma 3.1 requires $\frac{\partial W_{ij}(t)}{\partial t} \geq 0$ for the right hand side of Equation (3.8). From $W_{ij}(t) = 0$ it follows that $f_{ij} = 0$ and we decide upon the approximation depending on the sign of v at neighbouring points.

   a. $v(x_{j-1}, y_i, t) \geq 0$: Use (3.9) to compute $F_{i,j\pm1/2}$ giving Equation (3.12). The bracketed expression is non-negative and hence so is the right hand side.

   b. Else if $v(x_{j+1}, y_i, t) \leq 0$: Use (3.11) to compute $F_{i,j\pm1/2}$ giving Equation (3.13). The bracketed expression is non-negative and hence so is the right hand side.

   c. $v(x_{j-1}, y_i, t) < 0$ and $v(x_{j+1}, y_i, t) > 0$. This exceptional case can only occur if the PDE's solution has negative values (see the remark below), or because of too coarse a spatial grid, or through rounding errors. In this case we simply use central differences to approximate the spatial derivative at the point (x_j, y_i).

Remark. *Suppose, at time t and point (x_j, y_i), that case 2c occurs and that $w(x, y, t) \geq 0$ for all times t, $w(x_j, y_i, t) = 0$ but $w \neq 0$ locally around (x_j, y_i) for time t and that the function w and v are sufficiently smooth. If the spatial grid is sufficiently fine then it follows from the conditions of case 2c that $(v(x_j, y_i, t)w(x_j, y_i, t))_x > 0$. The PDE (3.3) then implies that $w_t(x_j, y_i, t) < 0$. That is, for small $\varepsilon > 0$, we have $w(x_j, y_i, t + \varepsilon) < 0$, a contradiction to the non-negativity assumption on w.*

We now consider the semidiscretization near the boundary. Figure 2 shows the stencils of the semidiscretization for both positive and negative velocities for grid points away from the boundary.

FIG. 2. Stencils of the semidiscretization at (x_j, y_i) using (3.9) on the left and (3.11) on the right.

We first consider the inflow boundary at $x = 1$ and the discretization at the point (x_{M-1}, y_i) (the values of $w(x_M, y_i, t)$ are given by the Dirichlet boundary condition). Points further to the left constitute no difficulties. If (3.9) is applied for the discretization of the convection term then all required fluxes $f_{il},\ l = M-3, \ldots, M$ are known. The approximation (3.11) is used if $W_{i,M-1} > 0$ and $v(x_{M-1}, y_i, t) \le 0$ or if $W_{i,M-1} = 0$ and $v(x_M, y_i, t) \le 0$ (the velocity is always non-positive on an inflow boundary at $x = 1$ and the exceptional case 2c cannot occur). The application of (3.11) requires the flux $f_{i,M+1}$ which we define by second order extrapolation, namely

$$f_{i,M+1} := 3f_{iM} - 3f_{i,M-1} + f_{i,M-2}. \tag{3.17}$$

In any case, the semidiscretization does not contradict Lemma 3.1 regarding positivity.

We now consider the outflow boundary at $x = 0$ for which we have $v(0, y_i, t) \le 0$. We define $f_{i,-1}$ by second order extrapolation

$$f_{i,-1} = 3f_{i,0} - 3f_{i,1} + f_{i,2}. \tag{3.18}$$

This allows us to apply approximation (3.11) at the grid point $(0, y_i)$ if $W_{i,0} > 0$ or if $W_{i,0} = 0$ and $v(x_1, y_i, t) \le 0$ and leads to a positive semidiscretization. In the exceptional case $W_{i,0} = 0$ and $v(x_1, y_i, t) > 0$ we apply the 2nd order central difference discretization using the defined value of $f_{i,-1}$. At the grid point (x_1, y_i) we apply (3.11) if $W_{i,1} > 0$ and $v(x_1, y_i, t) < 0$ or if $W_{i,1} = 0$ and $v(x_2, y_i, t) \le 0$. If $W_{i,1} > 0$ and $v(x_1, y_i, t) \ge 0$ or if $W_{i,1} = 0$ and $v(x_0, y_i, t) \ge 0$ (that is $= 0$ because of the outflow boundary) then we apply (3.9) using the defined value of $f_{i,-1}$. All of these possibilities lead to a positive semidiscretization. In the remaining exceptional case we again use central differences.

Finally, the scheme we have developed is applied to the hyperbolic equation (1.1a) of the PDE system. Again we consider the discretization of the convection in the x-direction only. The corresponding y-part is computed in a similar manner and added, with the source term $g_1(N_{ij}, C_{ij})$, to the right hand side of the semidiscretization.

The velocity function v is now defined as

$$v(x, y, t) = G_1(c(x, y, t))c_x(x, y, t) \tag{3.19}$$

and the function n is the counterpart of the convected quantity w above. We apply the scheme just described to this new situation and all that remains is to approximate the velocity v at spatial grid points. We define approximations of $c_x(x_j, y_i, t)$ by

$$(C_{ij})_x = \frac{1}{2h}(C_{i,j+1} - C_{i,j-1}), \quad i = 0, 1, \ldots, M, \ \ j = 1, 2, \ldots, M-1, \tag{3.20}$$

$$(C_{i,0})_x = \frac{1}{h}(C_{i,0} - 3C_{i,1} + 2C_{i,2}), \quad i = 0, 1, \ldots, M, \tag{3.21}$$

$$(C_{i,M})_x = -\frac{1}{h}(C_{i,M} - 3C_{i,M-1} + 2C_{i,M-2}), \quad i = 0, 1, \ldots, M, \tag{3.22}$$

where the boundary values of c are used if necessary. This approximation is second order accurate. The approximation for $v(x_j, y_i, t)$ is then given by

$$v(x_j, y_i, t) \approx G_1(C_{ij})(C_{ij})_x. \tag{3.23}$$

Remark. *This semidiscretization of Equation (1.1a) is second order accurate in smooth monotone regions of the solution n. The resulting ODE system is not positive (by Lemma 3.1) because even Van Leer's limiter is not smooth enough and some exceptional cases in the convection discretization might lead to positivity violation (boundary treatment and case 2c). Despite this complication we expect the scheme to perform satisfactorily and the numerical results presented in Subsection C. support this.*

The ODEs for $N_{ij}(t)$ and $C_{ij}(t)$ constitute a large autonomous system of ODEs with initial conditions $N_{ij}(0) = n_0(x_j, y_i)$ and $C_{ij}(0) = c_0(x_j, y_i)$ and an appropriate numerical scheme for the solution of this system is developed in the next section.

B. Time Split Scheme for the ODE System

The discretization of the spatial derivatives in the previous section results in a large m-dimensional system of ODEs in autonomous form

$$u'(t) = f_1(u(t)) + f_2(u(t)), \quad t \in (0, t_{final}], \quad u(0) = u_0. \tag{3.24}$$

The vector $u(t)$ contains all unknown time continuous functions $N_{ij}(t)$ and $C_{ij}(t)$ in an appropriate order. The function $f_2(u(t))$ represents the approximation of the diffusion and reaction terms in (1.1a, 1.1b) and $f_1(u(t))$ the discretized convection. We regard f_2 as the stiff part and f_1 as the non-stiff part of the ODEs.

In this section, we will develop a method for the solution of (3.24) which is both computationally efficient and positive in the sense of the following definition.

The last requirement is clarified by

Definition. (Positive ODE solver)
An ODE solver is called positive *if it generates non-negative approximations to the solution of a positive ODE system.*

An implicit method should be used for the solution of $u' = f_2(u)$ because of the stiffness of this equation, otherwise stability of the scheme would unnecessarily restrict the time step.

We could also use an implicit solver for the ODE $u' = f_1(u)$ but here the second requirement comes into play. Even if the step size in the algorithm is not restricted by stability, it will be restricted by positivity (except when we use the first order implicit Euler method) and one of the major advantages of implicitness is lost. Furthermore, experiments indicate that methods that require the Jacobian of the right hand side are less appropriate for the solution of ODE systems resulting from a (non-smooth) flux-limited spatial discretization of the advection equation. There is a very large number of rejected steps which is not the case if unlimited discretizations are used (but then we have negative values in the solution). The non-smoothness of the ODEs' right hand side, introduced by the limiter, is the cause of this behaviour here and we therefore apply an explicit method with good positivity properties for the solution of this ODE.

With these two competing concepts (implicitness and stability versus explicitness and positivity) in mind, it seems natural to use a splitting approach for the solution of (3.24).

The solution of (3.24) at time $t_k + \tau$ with initial data u_k at time t_k can be written as

$$u(t_k+\tau) = u_k + \int_{t_k}^{t_k+\tau} [f_1(u(\xi)) + f_2(u(\xi))]\,\mathrm{d}\xi \tag{3.25}$$

$$= \underbrace{u_k + \int_{t_k}^{t_k+\frac{\tau}{2}} f_1(u(\xi))\mathrm{d}\xi}_{=z_1\left(t_k+\frac{\tau}{2}\right)} + \int_{t_k}^{t_k+\tau} f_2(u(\xi))\mathrm{d}\xi + \int_{t_k+\frac{\tau}{2}}^{t_k+\tau} f_1(u(\xi))\mathrm{d}\xi, \tag{3.26}$$

where $z_1\left(t_k+\frac{\tau}{2}\right)$ is the solution of $z_1'(t) = f_1(z_1(t)),\ z_1(t_k) = u_k$ at $t = t_k + \frac{\tau}{2}$. Therefore we propose the following (Strang) splitting step for (3.24)

$$\begin{array}{lrcll} z_1'(t) = f_1(z_1(t)), & z_1(t_k) & = & u_k & \Rightarrow z_1\left(t_k+\frac{\tau}{2}\right), \\ z_2'(t) = f_2(z_2(t)), & z_2(t_k) & = & z_1\left(t_k+\frac{\tau}{2}\right) & \Rightarrow z_2\left(t_k+\tau\right), \\ z_3'(t) = f_1(z_3(t)), & z_3\left(t_k+\frac{\tau}{2}\right) & = & z_2(t_k+\tau) & \Rightarrow z_3\left(t_k+\tau\right). \end{array} \tag{3.27}$$

Theorem 3.4. *If each ODE system in the splitting scheme (3.27) is solved with a method of consistency order two or higher and the approximation to $z_3(t_k+\tau)$ is used as an approximation for $u(t_k+\tau)$, then the order of consistency of the splitting scheme (3.27) for the ODE (3.24) is two.*

Proof. Straightforward by Taylor expansion. ∎

Remark. *If the right hand side of an ODE system is split into more than two parts then the theorem, applied recursively to an appropriately extended splitting scheme, also gives a consistency order two. This can be useful for multidimensional diffusion discretization or for separate treatment of source terms.*

As stated at the beginning of this section, we will treat ODEs with right hand side f_1 explicitly and those with f_2 implicitly. We can therefore regard this splitting scheme as an implicit-explicit (IMEX) scheme. Such schemes have appeared frequently in recent literature, e.g. [3, 2, 7]. An important question in these investigations is the linear stability of IMEX schemes. The scalar test equation

$$u'(t) = \mu u(t) + \lambda u(t), \quad \mu, \lambda \in \mathcal{C} \tag{3.28}$$

is considered in [7] and the following question is investigated:

> Is the IMEX method stable for given eigenvalues μ, λ if $\tau\mu$ is inside the stability domain of the underlying explicit scheme and $\tau\lambda$ is inside the stability domain of the underlying implicit scheme?

The answer for the methods considered in [7] is negative, except for the IMEX Euler method of order one. For the splitting algorithm proposed here we obtain the following result.

Lemma 3.5. *The stability function of the splitting algorithm (3.27) applied to the test equation (3.28) is given by*

$$R(\tau\mu, \tau\lambda) = R^{(1)}\left(\frac{\tau\mu}{2}\right) R^{(2)}\left(\tau\lambda\right) R^{(1)}\left(\frac{\tau\mu}{2}\right), \tag{3.29}$$

where $R^{(i)}$ is the stability function of the method used to solve ODEs with right hand side f_i, $(i = 1, 2)$.

We can now answer the question raised earlier.

Lemma 3.6. *The splitting algorithm (3.27) is stable with respect to the test equation (3.28) if $\frac{\tau\mu}{2}$ is inside the stability domain of the underlying explicit scheme and $\tau\lambda$ is inside the stability domain of the underlying implicit scheme.*

Remark. *Note that the explicit method with stability function $R^{(1)}$ is applied with time steps $\tau/2$ which, in fact, means that its stability domain is enlarged by a factor two.*

The specific choice of methods for the solution of the ODEs in (3.27) is considered next.

The Explicit Method

A positive, non-stiff problem of the form

$$u'(t) = f_1(u(t)), \quad u(t_k) = u_k \tag{3.30}$$

has to be solved in steps one and three of the splitting scheme (3.27). We will use an explicit RKM to advance the initial condition over a time step τ. Theorem 3.4 states that we require only a second order accurate method in order to obtain overall second order consistency of the splitting scheme. We therefore restrict ourselves to such methods. A more important question is how to restrict the time step τ so that the solution of the RKM is positive.

We will look at positivity of the method applied to the constant coefficient problem with $f_1(u) = Pu$. Constant coefficient problems are positive if and only if the off-diagonal elements of the matrix P are non-negative [13]. The question of positivity of a method reduces to the question of absolute monotonicity of its stability function $R(x)$.

Definition. *[13] A rational function $R(x)$ is called* absolutely monotonic *at a point $x \in \mathcal{R}$ if and only if R and all its derivatives are non-negative in x.*

The threshold factor *of R, $T(R)$, is defined as*

$$T(R) = \sup\{r | r = 0 \textit{ or } (r > 0 \textit{ and } R \textit{ is absolutely monotonic at } x = -r)\}. \tag{3.31}$$

Let $\alpha > 0$. Define the (linear) positivity threshold *$\tau_{max}(\alpha)$ of a method with stability function R as the largest $\tau_0 \in [0, \infty]$ such that the method is positive for all step sizes $\tau \in (0, \tau_0)$ on all positive systems (3.30) with $f_1(u) = Pu$ and diagonal elements $p_{ii} \geq -\alpha$.*

With these definitions at hand we state the following result due to Bolley and Crouzeix [4], see also [13],

Theorem 3.7. *It holds*

$$\tau_{max}(\alpha) = T(R)\frac{1}{\alpha}\,. \tag{3.32}$$

Absolute monotonicity of polynomials is investigated in [13], and for s-stage explicit RKMs of order s, it is shown that $T(R) = 1$. The optimal stability polynomial with regard to absolute monotonicity for three stage, second order methods is $R(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{12}$ with $T(R) = 2$, in other words, the largest allowable step size for positivity of the method is doubled at the cost of only one more evaluation of the right hand side of the ODE. An examination of the domain of linear stability of this function shows that it is also larger than the corresponding domain of a 2-stage, second order method. The solution of the order conditions for order two and the optimality conditions on the stability function for a 3-stage explicit RKM are given in form of a Butcher array in Figure 3 (left). We use the free parameters (a_{32}, b_2, b_3) in the method to satisfy one of the two third-order conditions. The Butcher array on the right of Figure 3 gives one possible choice of these parameters and this is the method that will be used for the solution of the ODEs in steps one and three of the splitting scheme (3.27). We note that we cannot satisfy both third-order conditions without reducing the threshold factor of the method to $T(R) = 1$.

$$\begin{array}{c|ccc} & 0 & & \\ & \frac{1}{12 b_3 a_{32}} & 0 & \\ & -\frac{b_2 + 13 b_3^2 a_{32}^2 - 6 b_3 a_{32}}{12 b_3^2 a_{32}} & a_{32} & 0 \\ \hline & 1 - b_1 - b_2 & b_2 & b_3 \end{array} \qquad \begin{array}{c|ccc} & 0 & & \\ & \frac{2}{9} & 0 & \\ & \frac{1}{6} & \frac{1}{2} & 0 \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array}$$

FIG. 3. Butcher array of a general 3-stage, second order explicit RKM with optimal linear positivity (left) and a specific choice which satisfies one of the two third-order conditions (right).

The Implicit Method

The ODE system

$$z_2'(t) = f_2(z_2(t)), \quad z_2(t_k) = \tilde{z}_1 \tag{3.33}$$

has to be solved from t_k to $t_k + \tau$ in step two of the splitting algorithm (3.27). It is the semidiscretization of a heat conduction equation and, because of its stiffness, we solve it with a linearly implicit scheme. The only implicit relations in these methods are linear equation systems involving the Jacobian of the right hand side of the ODE system. The Jacobian J_2 of f_2 is broadly banded in our case (i.e. a bandwidth $\mathcal{O}(M)$ on an $M \times M$ grid, whatever smart arrangement of the components of z_2 is used) and therefore the application of direct solvers for the linear systems would ruin the efficiency of the splitting scheme (3.27). Iterative methods could be a way out of this but we favour another approach. The right hand side of the ODE can be written as $f_2 = f_{2a} + f_{2b}$, where f_{2a} contains the terms arising from the discretization of c_{yy} and f_{2b} contains all the remaining terms. We now note that the Jacobian J_{2a} of f_{2a} is a tridiagonal matrix if the components of z_2 are appropriately arranged and that the Jacobian J_{2b} of f_{2b} is a pentadiagonal matrix for a different arrangement of z_2. Therefore, from a computational efficiency point of view, it seems worthwhile to consider linearly implicit splitting methods. We employ the linearly implicit trapezoidal splitting method [6]. This method involves only the solution of two linear systems (one with J_{2a}, the other with J_{2b}) per time step. A time step τ of this method to obtain an approximation $\tilde{z}_2$ of $z_2(t_k + \tau)$

is given by

$$\tilde{z}_2^{(1)} = \tilde{z}_1 + \frac{\tau}{2} f_{2a}(\tilde{z}_1) \tag{3.34}$$

$$\tilde{z}_2^{(2)} = \tilde{z}_2^{(1)} + \frac{\tau}{2} f_{2b}(\tilde{z}_2^{(1)}) \tag{3.35}$$

$$\tilde{z}_2^{(3)} = \tilde{z}_2^{(2)} + \frac{\tau}{2} \left(I - \frac{\tau}{2} T_{2b}\right)^{-1} f_{2b}(\tilde{z}_2^{(2)}) \tag{3.36}$$

$$\tilde{z}_2 = \tilde{z}_2^{(3)} + \frac{\tau}{2} \left(I - \frac{\tau}{2} T_{2a}\right)^{-1} f_{2a}(\tilde{z}_2^{(3)}), \tag{3.37}$$

where

$$T_{2a} = \frac{\partial f_{2a}}{\partial u}(\tilde{z}_2^{(3)}) + \mathcal{O}(\tau) = \frac{\partial f_{2a}}{\partial u}(u_k) + \mathcal{O}(\tau) \tag{3.38}$$

$$T_{2b} = \frac{\partial f_{2b}}{\partial u}(\tilde{z}_2^{(2)}) + \mathcal{O}(\tau) = \frac{\partial f_{2b}}{\partial u}(u_k) + \mathcal{O}(\tau) \tag{3.39}$$

are first order approximations to the Jacobians J_{2a} and J_{2b}, respectively. This method has a consistency order two and, applied to the test equation $u'(t) = A_1 u(t) + A_2 u(t)$ with real matrices A_1 and A_2, we obtain the amplification matrix $R(\tau A_1, \tau A_2) = (I - \frac{\tau}{2}A_1)^{-1}(I - \frac{\tau}{2}A_2)^{-1}(I + \frac{\tau}{2}A_2)(I + \frac{\tau}{2}A_1)$. If the matrices A_1 and A_2 commute and if they have a non-positive logarithmic matrix norm then the method is A-stable, see [6].

Automatic Time Step Size Control

An automatic time step size control is advisable for the practical application of the method in order to avoid time steps that are too large (which could lead to instability of the method or inaccurate solutions) on one hand and time steps that are too small (which lead to unnecessary costs in the computation of the numerical solution) on the other. The aim is to select a sequence of time step sizes such that the local error from one step to the next of the method is bounded by some given tolerance *tol*. We use the approach via Richardson extrapolation for the time step size selection, see e.g. [18, pp 55]. (Embedding worked equally well for a similar splitting method applied to semidiscretizations of a spatially one-dimensional version of (1.1a-1.1b) but we expect Richardson extrapolation to give more reliable results in general.)

Let p be the order of the ODE solver ($p = 2$ in our case), m be the dimension of the ODE system (3.24), τ be the current time step size, and rtol_i and atol_i are given relative and absolute tolerances for each component of the solution. We compute a time step τ with the splitting scheme (3.27) with initial values u_k and obtain u_τ as an approximation of $u(t_k + \tau)$. Next we compute the solution of the splitting scheme (3.27) and initial condition u_k with two time steps $\tau/2$ which gives an approximation $u_{2\times\frac{\tau}{2}}$ of $u(t_k + \tau)$. A scaled error measure is then defined by

$$\rho = \frac{1}{2^p - 1} \sqrt{\frac{1}{m} \sum_{i=1}^{m} \left(\frac{(u_{2\times\frac{\tau}{2}})_i - (u_\tau)_i}{\mathrm{atol}_i + \mathrm{rtol}_i \max\{|(u_k)_i|, |(u_{2\times\frac{\tau}{2}})_i|\}} \right)^2}. \tag{3.40}$$

The step is rejected if $\rho > 1$ and accepted otherwise. The new time step size τ_{new} (for the following step or for the re-computation of the rejected step) is computed by

$$\tau_{new} = \tau \cdot \min\{2, \max\{0.8\rho^{-\frac{1}{2^p-1}}, 0.25\}\}. \tag{3.41}$$

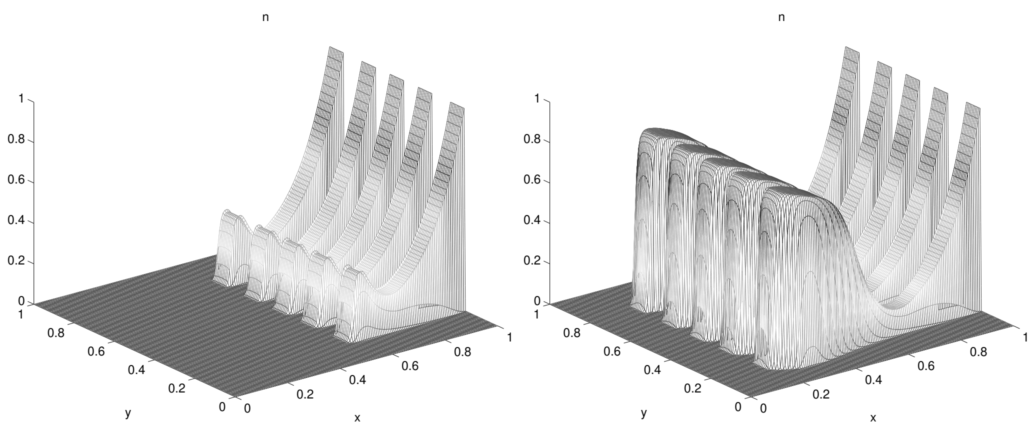

FIG. 4. Solution n of Example I. at times $t = 0.5$ (left) and $t = 0.9$ (right) obtained with the splitting algorithm (MOL).

If the step is accepted then we set $u_{k+1} = u_{2\times\frac{\tau}{2}}$, the more accurate pth order approximation. Extrapolation to order $p + 1$ would be possible but one would have to assess the stability (and positivity) properties of the resulting method anew. Numerical experiments indicated a poorer performance of the method if used with extrapolation and therefore we do not use it.

The Jacobians J_{2a} and J_{2b} are computed with respect to u_k (as indicated in the section on the implicit method) at the beginning of a splitting step. They therefore can be reused in rejected steps which makes these less expensive.

C. Numerical Experiments

We solve the equations of Example I. on a 150×150 spatial grid. The PDEs are discretized in space as described in this section (using van Leer's limiter (3.15)) and give rise to an ODE system with 45149 equations. This system is solved up to final times $t = 0.5$ and $t = 0.9$. For the convection in the x-direction we assume *a priori* a negative velocity. Numerical tests on the actual numerical velocities proved this assumption to be valid. Our reference solution is computed with the explicit RK code `DOPRI5` applied with $tol = 10^{-12}$ to the ODE system and we therefore study the temporal error (including the time splitting error) only. We use the splitting algorithm based on (3.27) for the solution of the ODE system and compare the results with those obtained by the Krylov–ROW code `ROWMAP` [21] applied to the non-split system. We tried to solve the ODE system with the linearly implicit code `RADAU5` but the system was too large (the large bandwidth of the Jacobian requires too much memory; an approximation of the Jacobian with fewer minor diagonals did not give results in reasonable times either). For comparison, we include test results obtained with the explicit code `DOPRI5`.

Figure 4 shows the solution n at times $t = 0.5$ (left) and $t = 0.9$ (right) obtained with our splitting method. The development of n in time and space is more important than that of c from a biological point of view. Therefore we do not include plots of c. The results obtained are in coincidence with experimental observations.

Tables I, II and III, respectively, summarize the results obtained for the three codes under consideration for a final time 0.9. The splitting algorithm (Table I) performs best from the point of view of positivity. The solution is positive (to within rounding errors)

TABLE I. Results for Example I. We solve the ODE system up to $t = 0.9$ with the splitting algorithm.

$-\log(tol)$	steps + rej. steps	$\log(l_2\text{-error})$	$\min_i\{U_i\}$	$\max_i\{U_i\}$	cpu time (s)
2	26 + 2	−1.63	−.659E-03	.990E+00	77.10
3	43 + 2	−1.80	−.196E-04	.990E+00	122.93
4	83 + 0	−2.10	−.121E-19	.990E+00	221.37
5	177 + 1	−2.60	−.794E-20	.990E+00	465.29
6	385 + 0	−3.31	−.697E-20	.990E+00	1010.86
7	899 + 36	−4.08	−.469E-20	.990E+00	2457.38
8	2376 + 484	−4.87	−.402E-20	.990E+00	7355.57

TABLE II. Results for Example I. The ODE system is solved up to $t = 0.9$ with ROWMAP. The code does not return solutions for tolerances larger than 10^{-5} because the time steps selected are too small.

$-\log(tol)$	steps + rej. steps	$\log(l_2\text{-error})$	$\min_i\{U_i\}$	$\max_i\{U_i\}$	cpu time (s)
5	260 + 26	−2.88	−.118E-02	.990E+00	1476.83
6	933 + 196	−3.82	−.208E-03	.990E+00	2750.62
7	2063 + 549	−4.58	−.343E-09	.990E+00	4026.03
8	5484 + 1833	−5.52	−.142E-14	.990E+00	7035.15

TABLE III. Results for Example I. The ODE system is solved up to $t = 0.9$ with DOPRI5. The code does not return solutions for tolerances larger than 10^{-3} because the time steps selected are too small.

$-\log(tol)$	steps + rej. steps	$\log(l_2\text{-error})$	$\min_i\{U_i\}$	$\max_i\{U_i\}$	cpu time (s)
3	48985 + 2	−2.45	−.536E-03	.990E+00	18606.87
4	48987 + 1	−3.32	−.487E-04	.990E+00	18721.90
5	48989 + 1	−5.04	−.653E-06	.990E+00	19833.17
6	48990 + 1	−6.09	−.327E-10	.990E+00	19827.00
7	48990 + 0	−7.18	−.109E-11	.990E+00	19835.27
8	48991 + 1	−7.18	−.149E-14	.990E+00	19901.20

for low tolerance requirements already (10^{-4}). The `ROWMAP` code (Table II) cannot achieve this. However, the accuracy of this code is better for more stringent tolerances. Figure 5 shows that the splitting scheme performs better than the `ROWMAP` code for a final time $t = 0.9$ and an l_2-error up to about 10^{-4}, (results for final time 0.5 are also included in this figure). The explicit code `DOPRI5` is, of course, restricted in the time step size by stability but also its positivity results are not very good for large tolerances (Table III).

If the less smooth maximum limiter [10] is used instead of van Leer's limiter then the positivity of the solution is violated more significantly (for both the splitting method and `ROWMAP`). Furthermore, `ROWMAP` requires many more steps (and has many more rejected steps) in order to compute a solution at a given time point and the required tolerance

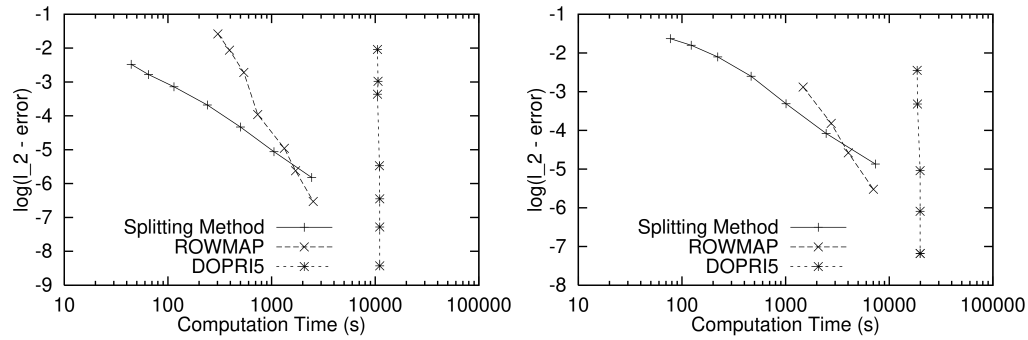

FIG. 5. Accuracy Achieved vs Computation Time diagrams for Example I. at $t = 0.5$ (left) and $t = 0.9$ (right).

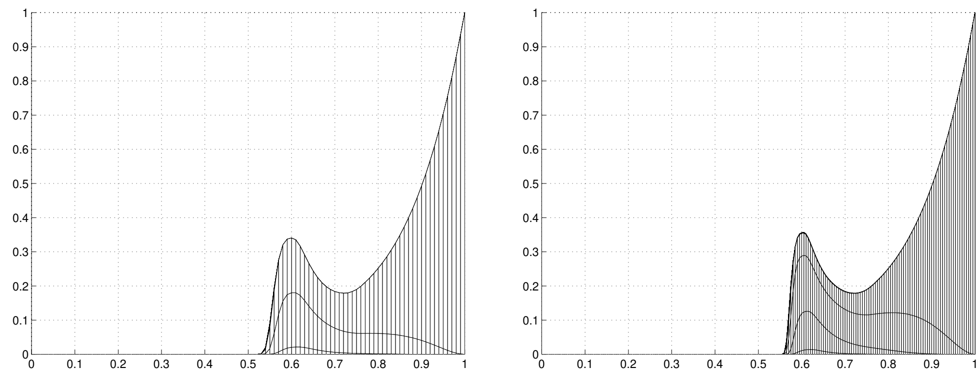

FIG. 6. Solution n of Example 1 at $t = 0.5$ with 100 grid points in each spatial direction (left) and 200 grid points (right).

tol must be no larger than 10^{-6} in order to be able to compute a solution up to time 0.9 at all.

Finally, we look at the numerical wave speed of the approximation to n in the discretization with van Leer's limiter and the splitting algorithm on different grids. Figure 6 shows that the wave speed of the solution does not depend significantly on the spatial resolution. In both cases (100 and 200 grid points in each spatial direction), the wave peak is at the same place after the same time ($t = 0.5$). However, it is observed that the solution on 100 grid points is more "smeared" than that on the finer grid.

With these observations in mind we arrive at the following conclusions.

- Discretization with van Leer's limiter appears to be best with regard to both positivity and efficiency. The maximum limiter might be more accurate but the resulting ODE system is not sufficiently positive and this leads to difficulties in the numerical solution of the ODE system (oscillations, rejected steps).
- Methods employing directly the Jacobian are not suitable for the solution of ODE systems where the right hand side is generated from a non-smooth limiter when only low accuracy is required. Better results are obtained for smaller tolerances but this also leads to long computation times (for all codes).

- The splitting scheme developed here works better than standard codes in the accuracy range 10^{-2} to 10^{-4}. This accuracy range is usually sufficient in simulations of biological processes. The code is here considerably faster than `DOPRI5` and `ROWMAP`.

IV. DISCUSSION AND CONCLUSIONS

In this paper we have presented two different numerical methods for the solution of a system of coupled parabolic–hyperbolic partial differential equations. The system of equations arose from a mathematical model of a biological problem in which cells migrate in response to the gradient field of a chemical. The first numerical method employed a finite difference scheme based on a splitting of the PDE system. Although this method is computationally inexpensive, it is prone to generating spurious oscillations in the solution in certain regions of the spatial domain. These lead to incorrect negative solution values. The second numerical scheme used the method of lines technique and special attention was given to preserving the positivity of the solution. In order to achieve this we employed a flux limiter in the discretization of the hyperbolic equation. The resulting system of stiff ODEs was then solved using appropriate splitting techniques - an explicit scheme for those terms arising from the convection terms of the hyperbolic PDE and a linearly implicit splitting method for those terms arising from the reaction-diffusion terms of the PDE system.

Concerning the application of these methods to particular problems of type (1.1a, 1.1b) we conclude that the first approach works well where the solution to the system does not contain steep gradients. In the absence of any steep gradients, the second order discretization of the convective terms (MacCormack's method) is likely not to generate large, spurious oscillations which would affect the positivity of the solution i.e. it is likely, therefore, that one encounters no difficulties resulting from negative values of the numerical solution. However, if, as is the case in Example I., steep gradients are present in the solution, then MacCormack's method leads to spurious oscillations and it is more appropriate to use the second method, employing a flux limiting procedure for the discretization of the convective terms.

The MOL approach appears to be a good option for the class of PDE problems under consideration here. The developed splitting method is of order 2 in time and possesses good stability and positivity properties. A further advantage is that we can easily use a time step size control which keeps the local temporal error below a user defined tolerance. Furthermore, the computational costs for the solution of the large ODE system can be reduced significantly by the time splitting procedure described. Numerical test have shown the efficiency and reliability for the problem considered.

REFERENCES

1. A. R. A. Anderson and M. A. J. Chaplain. Continuous and discrete mathematical models of tumor-induced angiogenesis. *Bull. Math. Biol.*, 60:555–597, 1998.
2. U. M. Asher, S. J. Ruuth, and R. J. Spiteri. Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. *Appl. Numer. Math.*, 25:151–167, 1997.
3. U. M. Asher, S. J. Ruuth, and B. T. R. Wetton. Implicit-explicit methods for time-dependent partial differential equations. *SIAM J. Numer. Anal.*, 32:797–823, 1995.

4. C. Bolley and M. Crouzeix. Conservation de la positivité lors de la discrétisation des problèmes d'évolution paraboliques. *RAIRO Anal. Numer.*, 12:237–245, 1978.
5. M. A. J. Chaplain and A. M. Stuart. A model mechanism for the chemotactic response of endothelial cells to tumour angiogenesis factor. *IMA J. Math. Appl. Med. Biol.*, 10:149–168, 1993.
6. C. Eichler-Liebenow, N. H. Cong, R. Weiner, and K. Strehmel. Linearly implicit splitting methods for higher space-dimensional parabolic differential equations. *Appl. Numer. Math.*, 28:259–274, 1998.
7. J. Frank, W. Hundsdorfer, and J. G. Verwer. On the stability of implicit-explicit linear multistep methods. *Appl. Numer. Math.*, 25:193–205, 1997.
8. A. Gerisch. Finite difference methods for coupled nonlinear hyperbolic and parabolic partial differential equations in one and two dimensions. M.Sc., University of Dundee, 1997.
9. S. K. Godunov and V. S. Ryabenkii. *Difference Schemes*, volume 19 of *Studies in Mathematics and its Applications*. North–Holland Amsterdam, 1987.
10. W. Hundsdorfer, B. Koren, M. van Loan, and J. G. Verwer. A positive finite-difference advection scheme. *J. Comput. Phys.*, 117:35–46, 1995.
11. W. H. Hundsdorfer. Numerical solution of advection-diffusion-reaction equations. Note NM-N9603, Department of Numerical Mathematics, CWI, 1996.
12. B. Koren. A robust upwind discretization method for advection, diffusion and source terms. In C. B. Vreugdenhill and B. Koren, editors, *Numerical Methods for Advection–Diffusion Problems*, volume 45 of *Notes on Numerical Fluid Mechanics*, chapter 5, pages 117–138. Vieweg, Braunschweig, 1993.
13. J. F. B. M. Kraaijevanger. Absolute monotonicity of polynomials occuring in the numerical solution of initial value problems. *Numer. Math.*, 48:303–322, 1986.
14. R. J. LeVeque. Intermediate boundary conditions for time-split methods applied to hyperbolic partial differential equations. *Math. Comp.*, 47:37–54, 1986.
15. R. J. LeVeque and J. Oliger. Numerical methods based on additive splittings for hyperbolic partial differential equations. *Math. Comp.*, 40:469–497, 1983.
16. A. R. Mitchell and D. F. Griffiths. *The Finite Difference Method in Partial Differential Equations*. John Wiley & Sons, Chichester, 1980.
17. R. Rannacher. Discretization of the heat equation with singular initial data. *Z. Angew. Math. Mech.*, 62:T346–T348, 1982.
18. K. Strehmel and R. Weiner. *Numerik gewöhnlicher Differentialgleichungen*. B. G. Teubner, Stuttgart, 1995.
19. P. K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM J. Numer. Anal.*, 21:995–1011, 1984.
20. J. W. Thomas. *Numerical Partial Differential Equations, Finite Difference Methods*. Number 22 in Texts in Applied Mathematics. Springer-Verlag, New York, 1995.
21. R. Weiner, B. A. Schmitt, and H. Podhaisky. ROWMAP—a ROW-code with Krylov techniques for large stiff ODEs. *Appl. Numer. Math.*, 25:303–319, 1997.