Letter Section

Solving parabolic integro-differential equations by an explicit integration method

A.S. Vasudeva Murthy *
TIFR Centre, Indian Institute of Science Campus, Bangalore 560 012, India

J.G. Verwer
Department of Numerical Mathematics, CWI, Kruislaan 413, 1098 SJ Amsterdam, Netherlands

Received 26 April 1991

Abstract

Vasudeva Murthy, A.S. and J.G. Verwer, Solving parabolic integro-differential equations by an explicit integration method, Journal of Computational and Applied Mathematics 39 (1992) 121–132.

The subject of this note is the numerical integration in time of nonclassical parabolic initial-boundary value problems which involve nonlocal integral terms over the spatial domain. The integral terms may appear in the boundary conditions and/or in the governing partial differential equation itself. These terms generally complicate the application of standard methods. Our purpose here is to draw attention to an explicit method, the Runge–Kutta–Chebyshev method, whose application remains straightforward in many cases of practical interest. We discuss two numerical examples to illustrate this. A comparison is made, respectively, with the implicit BDF code DASSL and with the Keller-box scheme.

Keywords: Parabolic partial integro-differential equations, nonlocal boundary conditions, numerical integration, Runge–Kutta–Chebyshev method, BDF method, Keller-box scheme.

1. Introduction

The number of physical phenomena modelled by partial differential equations involving nonlocal integral terms is constantly increasing. These nonlocal terms may appear in the boundary conditions or in the governing equation itself. In the following we give two examples of such partial integro-differential equations (PIDEs) of parabolic type. The two examples arise, respectively, in biology and meteorology.

* This author's visit to CWI was supported by NUFFIC.

The Fisher equation

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2}+uf(u), \tag{1.1a}$$

with $f(u)=1-u$, and its variants has been a classical model for the evolution of the number of individuals u in a population, see, e.g., [6]. The diffusion term represents the random movements of the individuals while $f(u)$ denotes the growth rate of the individuals. Recently, Britton [2] has argued that the growth rate is valid only when the population is spatially uniform and hence biologically unrealistic. Therefore, Britton postulates that the growth rate should not only depend on the population at that point but also on the population at the neighbouring points as well. In particular, employing some kind of spatial averaging, his model for the growth rate reads

$$f(u)=1+\alpha u-(1+\alpha)\int_{-\infty}^{\infty} u(s,\,t)k(|x-s|)\,\mathrm{d}s, \tag{1.1b}$$

where $k(x)=\frac{1}{2}\exp(-x)$ and α is a positive parameter. Problem (1.1) now falls under the class of PIDEs.

The example from meteorology is a model for the evolution of the temperature distribution of air near the ground during calm clear nights. Let $T(z,\,t)$ denote the temperature of air at height z and time t. The associated PIDE, as proposed in [12], is given by

$$\frac{\partial T}{\partial t}=\frac{\partial}{\partial z}\left\{(K_{\mathrm{m}}+K_{\mathrm{t}})\frac{\partial T}{\partial z}\right\}-\frac{1}{\rho_{\mathrm{a}}c_{\mathrm{p}}}\frac{\partial F}{\partial z},\quad z>0, \tag{1.2a}$$

where K_{m} and K_{t} are the molecular and turbulent diffusion coefficients for air, ρ_{a} is the density of air, c_{p} is the specific heat of air at constant pressure, and F is the radiative flux. F is given by

$$\begin{aligned}
&F(z,T)=F^{\uparrow}(z,T)-F^{\downarrow}(z,T),\\
&F^{\uparrow}(z,T)=\epsilon\sigma T_{\mathrm{g}}^4(t)(1-k(z))-\int_0^z\sigma T^4(s,\,t)k'(|z-s|)\,\mathrm{d}s,\\
&F^{\downarrow}(z,T)=\int_z^{\infty}\sigma T^4(s,\,t)k'(|z-s|)\,\mathrm{d}s,
\end{aligned} \tag{1.2b}$$

where σ is the Stefan–Boltzmann constant, ϵ and T_{g} are respectively the emissivity and temperature of the ground, k is the derivative of the broad-band flux emissivity function for water vapour and k' denotes the derivative of k. Equation (1.2a) is obtained as an energy balance between diffusion (molecular plus turbulent) and infrared radiation in the presence of water vapour. For further details on parameters, coefficient functions and boundary conditions, we refer to [12]. In Section 4 we will consider a parabolic PIDE example having a nonlocal integral term in the boundary condition. We have taken this example from [5] in which an interesting collection of nonclassical parabolic and hyperbolic problems in one space dimension is discussed. For a two-space dimensional PIDE example, originating from combat modelling, we refer to [8].

Our subject is the numerical integration in time of PIDEs. For this purpose we adopt the numerical method-of-lines approach and thus concentrate on the numerical integration of semi-discrete PIDE systems obtained after spatial discretization. Like with PDEs without nonlocal terms, these systems take the form of ordinary differential equation (ODE) systems, usually stiff ones. It therefore seems natural to employ implicit integration methods from the stiff ODE field in order to efficiently cope with the numerical stability problem. However, an important point to notice is that for PIDEs like (1.1) and (1.2) the numerical algebra overhead required for solving the implicitly defined numerical solution is considerably larger than for PDEs. While for PDEs the matrices encountered are always sparse and of banded form, assuming a finite-difference or finite-element spatial discretization, for these PIDEs the matrices are not banded and can even be full, due to the nonlocal terms. It is clear that this complicates the solution of the matrix problem considerably and will generally require a much larger computational effort. Obviously, this complication will be most paramount if the PIDE problem is multi-dimensional.

The principal aim of this note is to point out, through some simple numerical illustrations, that for interesting classes of parabolic PIDEs there is no need to use an implicit method, since the numerical integration in time can often be accomplished efficiently by the considerably less involved explicit Runge–Kutta–Chebyshev (RKC) method, due to [10]. This stabilized method has been developed for the explicit numerical integration of stiff ODE systems originating from certain classes of multi-space dimensional parabolic PDEs and can be applied to related PIDEs without any modification. Also the convergence analysis presented in [13] remains valid for the related PIDE problems. Hence, the spatial nonlocal terms, while complicating the implicit time stepping, have no consequence for the application of the explicit RKC method. On the other hand, by their very nature, PIDEs are computationally very expensive. This means that special-purpose techniques are of interest here. We will therefore briefly discuss this for the RKC method and also for implicit methods.

In Section 2 we review the RKC method and properties of the ODE systems for which this method has been developed. Section 3 is devoted to a model PIDE problem which is related to (1.1) and (1.2) and which has been selected to illustrate the application of the RKC method. To support our findings, we here present results of a comparison between two existing user-oriented codes, one of which is based on the explicit RKC method and the other on the celebrated Gear or implicit backward differentiation (BDF) method. To show that the RKC method is not restricted to PIDEs with the nonlocal term in the governing equation, Section 4 is devoted to a numerical illustration of RKC when applied to the above-mentioned problem from [5]. We here also present a comparison with their Keller-box scheme. In Section 5 we briefly mention possibilities for future work, in particular concerning special-purpose techniques.

2. The Runge–Kutta–Chebyshev method

In this section we give a brief description of the second-order RKC method. For the standard ODE problem

$$\frac{\mathrm{d}U(t)}{\mathrm{d}t}=F(t,\,U(t)),\quad 0<t<t_{\mathrm{e}},\; U(0)=U_0, \tag{2.1}$$

the explicit RKC integration formula reads

$$\begin{aligned} &Y_0 = U_n, \qquad Y_1 = Y_0 + \tilde{\mu}_1 \tau F_0, \\ &Y_j = \mu_j Y_{j-1} + \nu_j Y_{j-2} + (1 - \mu_j - \nu_j) Y_0 + \tilde{\mu}_j \tau F_{j-1} + \gamma_j \tau F_0, \quad 2 \leqslant j \leqslant s, \\ &U_{n+1} = Y_s. \end{aligned} \tag{2.2}$$

U_n represents the approximation to the exact solution U at time $t = t_n$, $F_j = F(t_n + c_j \tau, Y_j)$ and c_j, μ_j, ν_j, $\tilde{\mu}_j$, γ_j are integration coefficients. Formula (2.2) represents one single integration step from t_n to t_{n+1}, using stepsize $\tau = t_{n+1} - t_n$ and s evaluations of (2.1). Like in Runge–Kutta formulas, Y_j represents an intermediate approximation at $t_n + c_j \tau$ with $0 \leqslant c_j \leqslant 1$. In fact, a brief inspection reveals that (2.2) can be rewritten to the standard s-stage, explicit Runge–Kutta form. The specific form of (2.2), together with the definition of the integration coefficients, can be explained from its field of application which is the time integration of parabolic PDEs and related problems.

To briefly explain this, suppose for the time being that F is a linear vector function of type

$$F(t, U) = MU + g(t), \quad M \text{ symmetric nonpositive definite.} \tag{2.3}$$

Among others, spatial discretization of linear parabolic initial-boundary value problems having time-independent coefficients in the elliptic operator frequently leads to grid functions F of this type. Let $\sigma(M)$ be the spectral radius of M. The integration coefficients of (2.2) now obey the following three principal design criteria.

(i) The method has order of convergence two uniform in s and $\sigma(M) \in [0, \infty)$.

(ii) Given $\sigma(M)$ and stepsize τ, the number of stages s required for step-by-step stability is close to minimal, for formulas of type (2.2), and approximately satisfies

$$s = \sqrt{\tfrac{3}{2} \tau \sigma(M)}. \tag{2.4}$$

(iii) The method has excellent internal stability properties which means that within a single integration step the number of stages s can take on extremely large values before damaging accuracy due to round-off error build-up. Note that (2.4) assumes that σ, or a safe upper estimate, is known. In practice it is always possible to safely estimate σ.

The step-by-step stability criterion (ii) is fulfilled by associating the stability function of (2.2) with a special shifted Chebyshev polynomial. The internal stability criterion (iii) is fulfilled by associating the intermediate-stage formula of (2.2) with the stable three-term Chebyshev recursion. This explains the name of the method. For details on (ii) the interested reader is referred to [10]. For details on (iii) and the uniform convergence criterion (i), see [13]. In the latter paper one also finds the expressions of the integration coefficients, all in analytic form for arbitrary values of s.

Although the theory behind the RKC method is a linear one, the method has proven to be very reliable as well for nonlinear problems (2.1), having a (close to) normal Jacobian matrix $\partial F(t, U)/\partial U$ with its eigenvalues lying in a long narrow strip along the real axis. Due to the one-step nature, it is also easy to apply the method with a variable stepsize τ_n. Then s is also to be taken dependent on n and, assuming the problem to be nonlinear, formula (2.4) is replaced by

$$s_n = \sqrt{\tfrac{3}{2} \tau_n \sigma(J_n)}, \qquad J_n = \frac{\partial F(t_n, U_n)}{\partial U}. \tag{2.5}$$

In Section 4 we will present numerical results obtained with a variable-stepsize code due to Sommeijer. This FORTRAN code is called RKC and is based on the second-order formula (2.2), the coefficients of which are found in [13]. The variable-stepsize mechanism of RKC underlies the usual strategy of local error control. Specifically, RKC first selects a stepsize and then adjusts s for stability. This means that the explicit code is applied as if the underlying method is unconditionally stable. Of course, if the problem becomes extremely stiff, then s becomes very large resulting in a very heavy workload per step. In that case one should consider to resort to a truly unconditionally stable implicit method. As outlined in the Introduction, we believe that for many parabolic PIDEs the RKC code can be used to advantage, since the necessity of solving large systems of nonlinear algebraic equations is avoided, while in addition the method is conceptually simple. The FORTRAN version of RKC amounts to only about 100 lines of code. Also the memory requirement is very modest. Only six arrays of storage of the length of the solution vector U of the ODE system are required. This is in sharp contrast with implicit methods storing the Jacobian matrix, since for PIDE problems like (1.1) and (1.2) this matrix is full. RKC is available from NETLIB (see [4]).

3. A model parabolic integro-differential equation

We consider the parabolic integro-differential equation

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2}+\int_0^1 f(s,t,u(s,t))k(|x-s|)\,\mathrm{d}s,\quad 0<x<1,\ 0\leqslant t\leqslant t_e=1, \tag{3.1a}$$

where f and k are smooth functions. In particular, we here put

$$f(x,t,u)=-u^4,\qquad k(y)=\frac{1}{(1+y)^2}. \tag{3.1b}$$

This choice is based on the model (1.2) for the temperature profile of air near the ground. We emphasize that this model is a highly idealized version of (1.2), merely formulated for the purpose of numerical illustration. We impose the initial condition

$$u(x,0)=2(\cos(\pi x)+1),\quad 0\leqslant x\leqslant 1, \tag{3.1c}$$

and the boundary conditions

$$u(0,t)=2(2-\sqrt{t}),\quad u_x(1,t)=0,\qquad 0\leqslant t\leqslant t_e. \tag{3.1d}$$

Model (3.1) is a rather straightforward extension of the linear parabolic model equation $u_t=u_{xx}$. Obviously, various other more complicated extensions are conceivable. In connection with the contrast explicit or implicit time-stepping, however, this model is already very meaningful since the integral term involves a full Jacobian matrix. Needless to say that in two or three space dimensions this contrast will be much sharper. For our purpose it suffices to consider a single-space dimension.

For the spatial discretization of (3.1) we have employed standard second-order finite differences on a uniform mesh with N mesh intervals. The integral term is approximated with the second-order repeated trapezoidal rule. Formulating the resulting semi-discrete system, we then arrive at a system of ODEs in $\mathbb{R}^N$, where $F(t, U)$ takes the semi-linear form

$$F(t, U) := MU + G(t, U). \tag{3.2}$$

M is the tridiagonal, constant-coefficient finite-difference matrix associated to $\partial^2/\partial x^2$, with the last row corrected for the Neumann boundary condition. The vector function $G(t, U)$ is the discrete representation of the integral term, while its first component also contains the Dirichlet boundary function $2(2-\sqrt{t})$.

The $N \times N$ Jacobian matrix $J(t) = M + \partial G(t, U)/\partial U$ is full, since $\partial G(t, U)/\partial U$ is a full matrix. For N large, that is, on a sufficiently fine mesh, the spectral properties are largely determined by the familiar tridiagonal matrix M, since the integral term is bounded, in terms of N. Hence, for a sufficiently fine mesh, the eigenvalues of $J(t)$ will lie in a long narrow strip around the negative axes and $J(t)$ is "sufficiently close" to normal. It is indeed our practical experience that the stability of RKC is dictated by the diffusion term. Hence formula (2.5) can be applied by substituting an appropriate upper estimate of $\partial f(t, x, u)/\partial u$. In the experiment described below we have used the spectral radius estimate $\sigma(J_n) = 4N^2 + 20$. Finally, note that if $\partial f(x, t, u)/\partial u$ is a constant, as a function of both x and t, then $\partial G(t, U)/\partial U$ is also constant in t and symmetric, due to the symmetry of the kernel function $k(y)$. The ODE system then falls into the constant-coefficient linear model class for which the internal stability and convergence properties of RKC have been analysed (see [13]).

We are now ready to describe the numerical experiment carried out for (3.1). The aim of this experiment was to compare Sommeijer's code RKC with the implicit BDF code DASSL, developed by Petzold (see [1,7]). DASSL, including the necessary linear algebra source which is based on LINPACK, is also available from NETLIB. DASSL is equipped with variable stepsize, local error and order control and has an option to compute the required Jacobian matrix numerically. We applied DASSL using all its standard default options, including a numerical Jacobian computation and solution of the $N \times N$ systems of linear algebraic equations by Gaussian elimination. For both RKC and DASSL the required local tolerance parameter TOL was chosen as 10^{-3}. We stress that DASSL varies its order between 1 and 5, so that the code can also be used efficiently for much higher tolerances, in contrast with RKC which has order 2. We consider the local tolerance value 10^{-3} to be reasonable, however, since we are solving a partial differential equation.

Table 1 lists the runtime statistics and the maximal error at $t = 1$, for a sequence of grid

Table 1
Comparison between RKC and DASSL for problem (3.1); the computer used is the ALLIANT FX/4

DASSL: rtol = atol = TOL = 10^{-3}						RKC: TOL = 10^{-3}			
N	Time steps	Function evaluations	Jacobian evaluations	CPU (secs)	Max. error	Time steps	Function evaluations	CPU (secs)	Max. error
10	53	122	18	1.3	$0.11\cdot 10^{-1}$	90	269	0.7	$0.11\cdot 10^{-1}$
20	50	115	16	4.5	$0.30\cdot 10^{-2}$	76	344	2.6	$0.28\cdot 10^{-2}$
40	50	113	16	24.1	$0.76\cdot 10^{-3}$	81	590	15.2	$0.81\cdot 10^{-3}$
80	48	109	15	148.4	$0.24\cdot 10^{-3}$	46	1034	98.3	$0.34\cdot 10^{-3}$
160	47	107	15	1081.0	$0.94\cdot 10^{-4}$	48	2046	751.4	$0.27\cdot 10^{-3}$

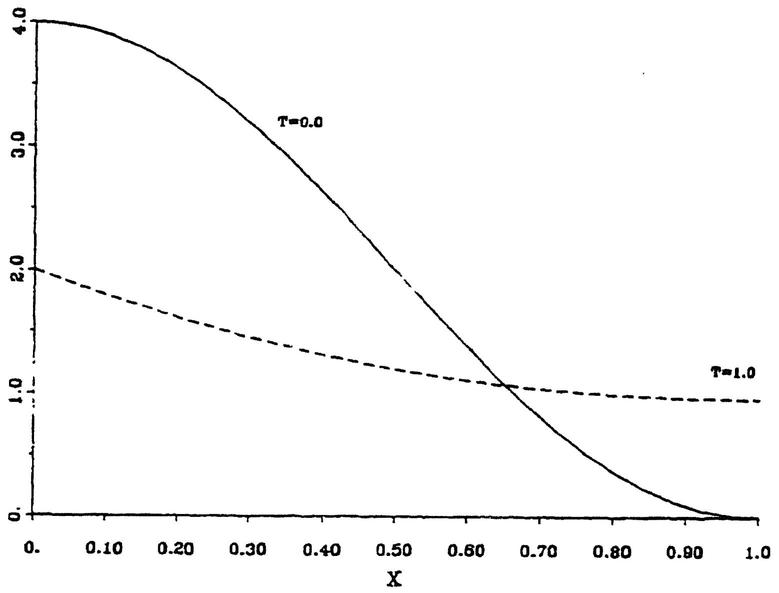

Fig. 1. Accurate reference solution for problem (3.1) at $t = 1$.

parameter values N. The numerical solution was compared with a very accurate reference solution computed with DASSL, using $N = 500$ and $\mathrm{TOL} = 10^{-5}$. This reference solution is plotted in Fig. 1. We see that the accuracy delivered by RKC and DASSL is very much the same. For low values of N one recovers the order 2 in space from the table. For the larger values of N the temporal error dominates. RKC takes considerably more time steps for N small than for N large. We owe this to the local error control which seems to be hindered by a somewhat nonsmooth numerical solution due to still too large errors. DASSL only slightly suffers from this phenomenon. We note that the CPU-time taken by DASSL is higher than that for RKC. While for RKC the costs are determined by the function evaluation, which is rather expensive here due to the integral term, for DASSL the costs are largely determined by the numerical algebra operations carried out in solving the implicit relation.

Although a numerical comparison like this is not conclusive, as it depends, for example, on the value of TOL, on how intelligently the ODE system is coded in connection with CPU-time, on the computer system used (vectorization, parallellization), it clearly indicates that in spite of using a fine mesh, the explicit code RKC can very well compete with the implicit code DASSL. Of course, it also shows that PIDE problems of the current type require a huge computational effort, for both integrators. For example, doubling the number of points will eventually increase the CPU-time for RKC by a factor of 8, assuming the number of time steps remains the same (see $N = 80$, 160). A factor of 2 is due to the increase in number of function evaluations, according to (2.5), while the remaining factor of 4 emanates from the integral term computation. For DASSL the factor of increase in CPU-time upon doubling the number of points will be not that large, assuming again that the number of time steps and other data will not change. However, DASSL requires a huge amount of memory since the Jacobian is a full matrix. These observations clearly indicate that for multi-space dimensional PIDE problems special-purpose techniques ought to be developed. We will return to this point in Section 5.

4. Parabolic equation with a nonlocal boundary condition

Following [5, Example 2.1], we next consider the model equation

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2},\quad 0<x<1,\ t>0, \tag{4.1a}$$

subjected to a given initial function at $t=0$ for $0 \leqslant x \leqslant 1$ and to the boundary conditions

$$\int_0^b u(x,t)\,\mathrm{d}x=\phi(t),\quad b\in(0,1),\ t\geqslant 0, \tag{4.1b}$$

$$u_x(1,t)=\psi(t),\quad t\geqslant 0, \tag{4.1c}$$

where ϕ and ψ are given functions and b is a given constant. Problem (4.1) serves, among others, as a model for diffusion subject to specification of mass, which is reflected by the nonlocal boundary condition (4.1b). Various finite-difference and finite-element Galerkin methods have been proposed for this problem. Fairweather and Saylor [5] discuss a finite-difference method based on the Keller-box scheme (see their Section 4). For this purpose they reformulate the problem into a system of three first-order differential equations, one of which serves to get rid of the nonlocal condition (4.1b). The problem is then in a form for which the Keller-box scheme gives an (almost) block diagonal coefficient matrix in the system of linear algebraic equations arising at each time step.

The purpose of this section is to show that the RKC method can also be applied to this problem, and in a straightforward manner. For this purpose we reformulate the nonlocal boundary condition to the separated Neumann type condition

$$u_x(0,t)=u_x(b,t)-\phi'(t), \tag{4.2}$$

by first differentiating with respect to t and then using (4.1a). Thus, (4.2) serves as the boundary condition at $x=0$. Equations (4.1a), (4.2) and (4.1c) can now be spatially differenced in the standard way, provided b is a meshpoint.

Introduce the uniform mesh $\{x_i\colon x_i=ih,\ 0\leqslant i\leqslant N,\ Nh=1\}$ such that $b\in\{x_i\}$. Let $u_i(t)$ represent the semi-discrete approximation to $u(x_i,t)$ based on second-order finite differences, both for the diffusion and boundary terms. Introduce the vector variable $U=[u_0,\ldots,u_N]^{\mathrm{T}}$. Then the semi-discrete system can be represented by the linear system of ordinary differential equations

$$\frac{\mathrm{d}U(t)}{\mathrm{d}t}=F(t,U):=MU+G(t), \tag{4.3}$$

where, as usual, the inhomogeneous term $G(t)$ contains the contribution from the boundary functions $\phi'(t)$ and $\psi(t)$. The $(N+1)\times(N+1)$ finite-difference matrix M is the same as that obtained for the standard Neumann problem, except that its first row differs due to the boundary condition (4.2). Hence, this row is given by

$$\frac{1}{h^2}(-2\ 2\ 0\ 0\ \ldots\ 0\ 1\ 0\ -1\ 0\ \ldots\ 0), \tag{4.4}$$

where the position of the three successive entries $1\ 0\ -1$ is determined by the location of the constant b in the mesh $\{x_i\}$.

Table 2
Results for [5, Problem (4.1), Case 4], comparison of relative errors at $x = 0.25$, $t = 0.1$ for (a) RKC method and (b) Keller-box scheme

h		$\tau = 0.05$	$\tau = 0.01$	$\tau = 0.005$	$\tau = 0.0025$	$\tau = 0.001$
0.25	(a)	$0.110 \cdot 10^{-0}$	$0.919 \cdot 10^{-1}$	$0.935 \cdot 10^{-1}$	$0.939 \cdot 10^{-1}$	$0.949 \cdot 10^{-1}$
	(b)	$0.275 \cdot 10^{-0}$	$0.361 \cdot 10^{-0}$	$0.627 \cdot 10^{-0}$	$0.116 \cdot 10^{+1}$	$0.277 \cdot 10^{+1}$
0.05	(a)	$0.250 \cdot 10^{-1}$	$0.251 \cdot 10^{-2}$	$0.376 \cdot 10^{-2}$	$0.389 \cdot 10^{-2}$	$0.397 \cdot 10^{-2}$
	(b)	$0.187 \cdot 10^{-1}$	$0.424 \cdot 10^{-2}$	$0.432 \cdot 10^{-2}$	$0.783 \cdot 10^{-2}$	$0.937 \cdot 10^{-2}$
0.01	(a)	$0.297 \cdot 10^{-1}$	$0.967 \cdot 10^{-3}$	$0.121 \cdot 10^{-3}$	$0.880 \cdot 10^{-4}$	$0.148 \cdot 10^{-3}$
	(b)	$0.130 \cdot 10^{-1}$	$0.523 \cdot 10^{-3}$	$0.243 \cdot 10^{-3}$	$0.172 \cdot 10^{-3}$	$0.153 \cdot 10^{-3}$
0.005	(a)	$0.299 \cdot 10^{-1}$	$0.107 \cdot 10^{-2}$	$0.240 \cdot 10^{-3}$	$0.300 \cdot 10^{-4}$	$0.287 \cdot 10^{-4}$
	(b)	$0.128 \cdot 10^{-1}$	$0.410 \cdot 10^{-3}$	$0.130 \cdot 10^{-3}$	$0.605 \cdot 10^{-4}$	$0.409 \cdot 10^{-4}$
0.0025	(a)	$0.299 \cdot 10^{-1}$	$0.110 \cdot 10^{-2}$	$0.270 \cdot 10^{-3}$	$0.595 \cdot 10^{-4}$	$0.114 \cdot 10^{-5}$
	(b)	$0.128 \cdot 10^{-1}$	$0.381 \cdot 10^{-3}$	$0.102 \cdot 10^{-3}$	$0.325 \cdot 10^{-4}$	$0.130 \cdot 10^{-4}$
0.001	(a)	$0.299 \cdot 10^{-1}$	$0.110 \cdot 10^{-2}$	$0.278 \cdot 10^{-3}$	$0.677 \cdot 10^{-4}$	$0.949 \cdot 10^{-5}$
	(b)	$0.128 \cdot 10^{-1}$	$0.373 \cdot 10^{-3}$	$0.942 \cdot 10^{-4}$	$0.247 \cdot 10^{-4}$	$0.520 \cdot 10^{-5}$

To compare the accuracy of the RKC finite-difference method with that of the Keller-box scheme proposed in [5], we have applied the method with constant stepsize τ to one of their example problems. The problem considered is their Case 4 which corresponds to the intermediate point $b = 0.75$ and exact solution $u(x, t) = \exp(-\pi^2 t)\sin(\pi x)$. The problem data are adjusted to this exact solution. In Table 2 we present the relative error at $(x, t) = (0.25, 0.1)$ for both methods for a selection of (τ, h)-values (the errors of the box scheme have been copied from [5, Table 4]). The errors of the two schemes appear to be very much comparable if τ is close to h. However, if $\tau \ll h$, then RKC is more accurate, while for $h \ll \tau$ it is the other way round. Note that the accuracy of the box scheme deteriorates if $\tau \ll h$. Noteworthy is that the RKC finite-difference method nicely shows its space/time second order and that there is no such deterioration.

The number of stages s was determined by the formula $s = 1 + \text{entier}[(1 + \tau\sigma/0.65)^{1/2}]$ with 2 as a minimum. Note that this formula yields almost the same values for s as (2.4), for all $\tau\sigma > 0$. We refer to [13] for its explanation. Noteworthy is that the RKC method has been applied using the common spectral radius estimate $\sigma(M) = 4/h^2$, thereby ignoring the first row entries 1, -1 of the nonsymmetric matrix M. Hence, as far as stability is concerned, the separated Neumann condition appears to have no influence at all. We recall that this also holds for the pure Neumann problem, i.e., problem (4.1) with (4.1b) replaced by a condition like (4.1c). For the pure Neumann problem this is easy to prove by applying Gershgorin's theorem to the nonsymmetric matrix M and by transforming M to a symmetric, nonpositive definite form, using the diagonal similarity transformation DMD^{-1} ($D = \text{diag}(1/\sqrt{2}, 1, \dots, 1, 1/\sqrt{2})$). This way one proves that for the pure Neumann problem the RKC method has the same stability properties as for the associated Dirichlet problem, whose discretization directly leads to a symmetric, negative definite M. We have also tried to prove this result for the present problem with the separated Neumann condition, by attempting to also transform M to a

Table 3
Function evaluations per time step: $s = 1 + \text{entier}[(1 + \tau\sigma/0.65)^{1/2}]$, $\sigma = 4/h^2$

h	$\tau = 0.05$	$\tau = 0.01$	$\tau = 0.005$	$\tau = 0.0025$	$\tau = 0.001$
0.25	3	2	2	2	2
0.05	12	6	4	3	2
0.01	56	25	18	13	8
0.005	111	50	36	25	16
0.0025	222	100	71	50	32
0.001	555	249	176	125	79

symmetric, nonpositive definite form. However, such a proof appears to be much more difficult to find and so far our attempts have been in vain.

Finally, no efficiency comparison has been carried out for the RKC method and the Keller-box scheme, since we have not implemented the box scheme ourselves (see [5] for details). Table 3 shows the number of function evaluations per time step for the above used selection of (τ, h)-values. Apart from the very cheap inhomogeneous term (only two nonzero components), each function evaluation amounts to a matrix–vector multiplication with an almost tridiagonal matrix, which can be coded very efficiently on super/parallel computers. On the other hand, when the stiffness really becomes large, i.e., when $\tau\sigma$ is very large, the effort per time step becomes rather substantial and the Keller-box scheme will require less CPU-time. It is clear that as far as efficiency is concerned, while the stiffness is very large, the explicit RKC method is a better candidate for the PIDE problem of the previous section and, in particular, for problems like (4.1) in more space dimensions.

5. Special-purpose techniques

Nonclassical problems can sometimes be transformed to a form such that available numerical methods can be efficiently applied in the standard way (cf. [5]). In this respect, the diffusion problem of Section 4 provides a nice illustration, both for the RKC and the Keller-box method. For PIDE problems of the type mentioned in the Introduction and in Section 3, the situation is different. Application of the two codes RKC and DASSL does result here in a huge computational effort, especially if such a PIDE problem has more than one space dimension. The illustration presented in Section 3 is evidential. For such problems it can be most advantageous to consider the use of special-purpose techniques, that is, to modify the methods so as to reduce the effort required in their standard use. In this final section we will briefly point out some possible, fruitful modifications.

For RKC the costs are determined by the integral term which is much more expensive to compute than the diffusion term. To reduce the computing time for RKC, it therefore may be useful to implement an idea suggested in [9]. Loosely speaking, the idea discussed in this paper is to compute the true derivative only in the first few stages, and to replace in all remaining stages the ODE system by a very cheap approximation. This replacement should maintain stability, which in the present application is determined mainly by the diffusion operator. The idea underlies their hypothesis that only the first few stages determine the accuracy and that the remaining ones mainly serve for providing good stability properties. Because much can be gained in terms of CPU-time, it is of interest to investigate whether this special-purpose technique indeed is of use for PIDEs of the type of Section 3. A second possibility is to apply RKC in an operator-splitting setting (see also [11]). In large-scale computing in numerical time-dependent PDEs, operator splitting is frequently used to advantage. For the current PIDE problem, operator-splitting methods to be investigated should of course treat the diffusion and integral term apart.

For the implicit code DASSL the CPU-time is largely determined by the overhead costs for solving at each time step the system of nonlinear algebraic equations. The use of direct solvers requiring the Jacobian matrix be given in explicit form is expensive here, since this Jacobian is a full matrix. This also points at a second difficulty, that of the memory requirement. Suppose we have a PIDE problem like in Section 3 in two space dimensions, and suppose that we wish to solve it on a uniform 100×100 mesh. The Jacobian then has 10^8 entries, the storage of which is too large for an efficient processing, even on modern super mainframes. To remedy both difficulties, that of the overhead and of the memory requirement, an attractive alternative is provided by the so-called matrix-free iteration methods. A matrix-free iteration method does not require the Jacobian matrix be given explicitly, as it is based on matrix–vector product operations, like in RKC when the problem is linear. Matrix-free iteration methods are, in theory, also directly applicable to nonlinear problems (inexact Newton methods). These methods are accelerated using some form of (problem-dependent) preconditioning, which for parabolic PIDEs can be based on the sparsely differenced diffusion operator. Recently quite some research is in progress on using these special-purpose methods in implicit method-of-lines schemes (see [3]) and we believe they offer excellent opportunities for efficiently applying implicit integrators to PIDE problems.

Acknowledgement

The authors wish to thank Joke Blom for her programming assistance.

References

[1] K.E. Brenan, S.L. Campbell and L.R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations* (Elsevier, New York, 1989).
[2] N.F. Britton, Aggregation and the competitive exclusion principle, *J. Theoret. Biol.* **136** (1989) 57–66.
[3] P.N. Brown and A.C. Hindmarsh, Reduced storage matrix methods in stiff ODE systems, *Appl. Math. Comput.* **31** (1989) 40–91.
[4] J.J. Dongarra and E. Grosse, Distribution of mathematical software via electronic mail, *Comm. ACM* **30** (1987) 403–407.
[5] G. Fairweather and R.D. Saylor, The reformulation and numerical solution of certain non-classical initial-boundary value problems, *SIAM J. Statist. Comput.* **12** (1991) 127–144.
[6] P.C. Fife, *Mathematical Aspects of Reacting and Diffusing Systems*, Lecture Notes in Biomath. **28** (Springer, Berlin, 1979).
[7] L.R. Petzold, A description of DASSL, a differential-algebraic system solver, in: R.S. Stepleman, Ed., *IMACS Trans. Sci. Comput.* (North-Holland, Amsterdam, 1983) 65–68.

[8] V. Protopopescu, R.T. Santoro, R.L. Cox and P. Rusu, Combat modelling with partial differential equations, the bidimensional case, Report TM-11343, Oak Ridge National Laboratory, 1990.
[9] B.P. Sommeijer and P.J. van der Houwen, On the economization of stabilized Runge–Kutta methods with applications to parabolic initial value problems, *Z. Angew. Math. Mech.* **61** (1981) 105–114.
[10] P.J. van der Houwen and B.P. Sommeijer, On the internal stability of explicit m-stage Runge–Kutta methods for large m-values, *Z. Angew. Math. Mech.* **60** (1980) 479–485.
[11] P.J. van der Houwen and B.P. Sommeijer, Parallel solution of the Burgers equation, Report NM-R9104, Dept. Numer. Math., CWI, 1991.
[12] A.S. Vasudeva Murthy, J. Srinivasan and R. Narasimha, Modelling the lifted minimum phenomenon in the atmosphere, Report 91 AS 1, Centre for Atmospheric Sci., Indian Institute of Sci., 1990.
[13] J.G. Verwer, W.H. Hundsdorfer and B.P. Sommeijer, Convergence properties of the Runge–Kutta–Chebyshev method, *Numer. Math.* **57** (1990) 157–178.