ELSEVIER Journal of Computational and Applied Mathematics 84 (1997) 277–280

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

Letter

A simple ODE solver based on 2-stage Radau IIA

Jacques J.B. de Swart*,[1]

CWI, Cluster MAS, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

Received 27 June 1997; received in revised form 14 July 1997; accepted 15 July 1997

Abstract

By simplifying the Newton process needed to solve the nonlinear equations associated with the 2-stage Radau IIA method, we come up with an efficient solver that needs only one LU-decomposition of the dimension of the problem per time step.

Keywords: Numerical analysis; Runge–Kutta methods; Newton iteration

AMS classification: 65L05; 65L06

1. Introduction

Suppose that we want to solve the ODE initial value problem

$$y' = f(t, y), \qquad y(t_0) = y_0, \qquad y, f \in \mathbb{R}^d \tag{1}$$

with the third-order, 2-stage Radau IIA method, which we write in the form

$$R(Y_n) = 0, \tag{2}$$

where $Y_n := (g_n^{\mathrm{T}}, y_n^{\mathrm{T}})^{\mathrm{T}}$ contains approximations to the solution of (1) at time point $t_{n-1} + \frac{1}{3}h$ and t_n, respectively, and where

$$R(Y_n) := Y_n - (1,1)^{\mathrm{T}} \otimes y_{n-1} - h(A \otimes I)F(Y_n),$$

$$A := \begin{bmatrix} \frac{5}{12} & -\frac{1}{12} \\ \frac{3}{4} & \frac{1}{4} \end{bmatrix}, \qquad F(Y_n) := \begin{pmatrix} f(t_{n-1} + \frac{1}{3}h, g_n) \\ f(t_n, y_n) \end{pmatrix}.$$

Furthermore, h is the timestep, $\otimes$ denotes the Kronecker product, and I is the identity matrix. Here and in the sequel, the dimension of I will always be clear from the context. Normally, we solve the

* E-mail: jacques@cwi.nl.

[1] Supported by Dutch Technology Foundation STW, Grant No. CWI22.2703.

PII S0377-0427(97)00141-6

system of nonlinear equations (2) by a modified Newton process of the form

$$(I - hA \otimes J)\Delta Y_n^{(j+1)} = -R(Y_n^{(j)}), \quad j = 0, 1, 2, \ldots, \tag{3}$$

where $\Delta Y_n^{(j+1)}$ is shorthand for $Y_n^{(j+1)} - Y_n^{(j)}$ and J is the Jacobian of f at the approximated solution in t_{n-1}. If we use Gaussian elimination to solve the linear systems in (3), then the LU-costs are $\frac{16}{3}d^3$.

Several attempts have been made to reduce these costs. By means of Butcher transformations we can rewrite (3) as

$$(I - hB \otimes J)(Q \otimes I)\Delta Y_n^{(j+1)} = -(Q \otimes I)R(Y_n^{(j)}), \tag{4}$$

where $B = QAQ^{-1}$ is such that (4) is easier to solve than (3). In [2] we find the strategy to choose Q such that

$$B = \begin{bmatrix} \frac{1}{3} & \frac{1}{6}\sqrt{2} \\ -\frac{1}{6}\sqrt{2} & \frac{1}{3} \end{bmatrix}.$$

Due to this special form, (4) can be written as one complex system of dimension d, so that the LU-costs are reduced to $\frac{8}{3}d^3$.

The paper [4] proposes to replace the matrix A in (3) by its lower Crout factor L. To make the method suitable for parallel implementation, the matrix L is then transformed by a matrix Q such that QLQ^{-1} is a diagonal matrix. The resulting scheme again has the form (4), where B is now a diagonal matrix. This means that two processors can compute an LU-decomposition of dimension d concurrently. The LU-costs are now $\frac{2}{3}d^3$ on two processors, and $\frac{4}{3}d^3$, if only one processor is available. The price we have to pay for substituting A by L is a slower convergence of the Newton process. For the linear test equation $y' = \lambda y$, the propagation of the iteration errors of the scheme is described by the matrix

$$Z(z) = z(I - zL)^{-1}(A - L), \qquad z := h\lambda.$$

The reason for choosing the Crout decomposition of A is that the spectral radius of $Z(z)$ is zero for $z \to \infty$.

In [3] this approach is refined by approximating QAQ^{-1} by a matrix M with real eigenvalues and then applying a second transformation that diagonalizes M. The resulting scheme has the same form as in [4], but its convergence behavior is better. In fact, it is shown that Q and M can be chosen such that the maximum of $\rho(Z(z))$ is minimized in the left-half complex plane. Here, $\rho(\cdot)$ denotes the spectral radius function.

Recently, Amodio and Brugnano [1] suggested to select an upper triangular matrix Q such that QAQ^{-1} has a lower Crout factor with identical diagonal entries, thereby reducing the LU-costs to $\frac{2}{3}d^3$. This work inspired us to the approach described in the next section.

2. A new approach

We advocate a mixture of [1] and [3] by choosing Q *lower* triangular such that (i) the lower Crout factor L of QAQ^{-1} has identical diagonal entries and (ii) the minimization property of [3] is

Table 1
Several measures of the matrix $Z(z)$

Method	$\max_{z\in\mathbb{C}^-} \|Z(z)\|_\infty$	$\max_{z\in\mathbb{C}^-} \rho(Z(z))$	$\lim_{z\to\infty} \|Z(z)\|_\infty$	$\lim_{z\to 0} \|\frac{1}{z}Z(z)\|_\infty$
[4]	0.20	0.18	0.20	0.15
[1]	0.20	0.18	0.20	0.15
Presented here	0.22	0.18	0.18	0.17

retained. The advantage of the *lower* triangular form of Q is that QLQ^{-1} is still lower triangular, so that the transformation can be omitted.

The matrix Q can be found by carrying out the recipe in [3, p. 171] with $\gamma=1$, $\beta=0$ and $\delta=0$, thus leading to

$$B=\begin{bmatrix} \frac{1}{6}\sqrt{6} & 0 \\ 4-\frac{4}{3}\sqrt{6} & \frac{1}{6}\sqrt{6} \end{bmatrix}.$$

Table 1 compares the convergence behaviour of this approach with that of [4] and [1]. From this table it can be seen that the measures of $Z(z)$ do not differ much for the three methods, whereas the method presented here is the cheapest to implement.

The reason that we restrict our considerations to two stages, is that the two aforementioned objectives cannot be achieved for more than two stages. However, many applications do not require an order higher than 3.

3. Implementation

Since the lower-left element of B is nonzero, the value $y_n^{(j+1)}$ depends on $g_n^{(j+1)}$ by the formula

$$y_n^{(j+1)} = y_n^{(j)} + (I-\tfrac{1}{6}\sqrt{6}hJ)^{-1}((4-\tfrac{4}{3}\sqrt{6})hJ\,\Delta g - [0\,I]R(Y_n^{(j)})),$$

where $\Delta g = g_n^{(j+1)} - g_n^{(j)}$. If J is nonsparse, the matrix–vector multiplication $hJ\,\Delta g$ can be expensive. Therefore, we rewrite $(I-\frac{1}{6}\sqrt{6}hJ)^{-1}(4-\frac{4}{3}\sqrt{6})hJ\,\Delta g$ as

$$(8-4\sqrt{6})\,\Delta g + (I-\tfrac{1}{6}\sqrt{6}hJ)^{-1}(4\sqrt{6}-8)\,\Delta g.$$

Using first-order Taylor approximations around t_{n-1} as predictor, the resulting iteration scheme, which computes y_n given y_{n-1} and $f_y = f(t_{n-1}, y_{n-1})$, simply reads:

$g_n = y_{n-1} + \frac{1}{3}hf_y$
$y_n = y_{n-1} + hf_y$
compute $LU = I - \frac{1}{6}\sqrt{6}hJ$

until convergence do

$$f_g = f(t_{n-1} + \tfrac{1}{3}h, g_n)$$
$$f_y = f(t_n, y_n)$$
$$\Delta g = (LU)^{-1}(y_{n-1} - g_n + \tfrac{5}{12}hf_g - \tfrac{1}{12}hf_y)$$
$$\Delta y = (LU)^{-1}((4\sqrt{6} - 8)\Delta g + y_{n-1} - y_n + \tfrac{3}{4}hf_g + \tfrac{1}{4}hf_y)$$
$$g_n = g_n + \Delta g$$
$$y_n = y_n + (8 - 4\sqrt{6})\Delta g + \Delta y$$

end

References

[1] P. Amodio, L. Brugnano, A note on the efficient implementation of implicit methods for ODEs, submitted for publication, 1997.

[2] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, 2nd revised edn., Springer, Berlin, 1996.

[3] P.J. van der Houwen, J.J.B. de Swart, Parallel linear system solvers for Runge–Kutta methods, Adv. Comput. Math. 7 (1997) 157–181.

[4] P.J. van der Houwen, J.J.B. de Swart, Triangularly implicit iteration methods for ODE-IVP solvers, SIAM J. Sci. Comput. 18 (1) (1997) 41–55.