Journal of Computational and Applied Mathematics 25 (1989) 1–13
North-Holland

A composite integration scheme for the numerical solution of systems of ordinary differential equations

John CARROLL
School of Mathematical Sciences, N.I.H.E., Dublin 9, Ireland

Received 25 July 1986
Revised 15 August 1988

Abstract: A generalization of a composite linear multistep method [2] is developed and applied to the approximate integration of systems of ordinary differential equations. The proposed scheme is second-order accurate and L-stable. An algorithm, based on the integration formula derived in this paper, is applied to approximate the solutions of a number of standard test problems. The numerical results indicate that the method is competitive with other fixed-order methods particularly in terms of computational overhead and could provide the basis for efficient temporal integration in the semidiscretization of time dependent partial differential equations.

Keywords: Numerical analysis, initial-value problems, composite integration, L-stability.

1. Introduction

In [2], a composite linear multistep formula was introduced as the time integration scheme in the numerical solution of the coupled system of nonlinear partial differential equations that model the transient behaviour of Silicon VLSI devices. In this paper, we can generalize the formula to a composite integration scheme which retains the important features of second order accuracy and L-stability of the original method. This investigation was prompted by the difficulties associated with the numerical solution of the type of problem discussed in [2] where the Jacobian matrix $J = \partial f / \partial y$ is large and consequently its frequent evaluation is expensive in terms of computational overhead and cpu time. In particular we examine the application of a composite integration scheme to approximate the solution $y : [a, b] \to R^N$ of the initial-value problem

$$y' = f(x, y), \quad y(a) \text{ given}, \tag{1.1}$$

where it is assumed that f is Lipschitz continuous on $[a, b]$, that is, for some vector norm, there exists a constant L such that

$$\| f(x, y) - f(x, z) \| \leqslant L \| y - z \|$$

for all $x \in [a, b]$ and all $y, z \in R^N$. More specifically, if we require that the first partial derivatives of f are bounded by a constant K.

$$|\partial f_i/\partial y_j| \leqslant K, \quad 1 \leqslant i, j \leqslant N,$$

for all $x \in [a, b]$ and all $y \in R^N$, the conditions (1.2) and (1.3) guarantee a unique solution to the problem (1.1) [3]. In Section 2, we present an outline derivation of the numerical method where the choice of parameters will be constrained by requiring that the composite scheme is 2nd-order accurate and a common iteration matrix is employed to solve the resulting algebraic equations. An estimate of the local truncation error is obtained using a linear combination of available function values and provides the basis for automatic stepsize control. Implementation details are considered in Section 3 including criteria as to when a step is accepted and the control of Jacobian evaluations required for the iterative algorithm. We report details of some numerical experiments in Section 4 when the proposed method is applied to the approximate integration of some standard test problems including the test suite of [9]. We will conclude that, because of the economy in Jacobian evaluations, this generalized integration scheme could provide a basis for an efficient procedure in the time integration of large systems of differential equations resulting, for example, from the semidiscretization of systems of coupled nonlinear partial differential equations reported in [2].

2. A composite integration scheme

In the numerical integration of the problem (1.1) we consider a formula pair which enables stepping from x_n to x_{n+1} to be split into two separate tasks. We apply the one-step theta scheme to step from $x = nh$ to $x = (n + \gamma)h$, for some $0 < \gamma < 1$,

$$y_{n+\gamma} = y_n + \gamma h\left[(1-\theta)f_n + \theta f_{n+\gamma}\right], \quad 0 < \theta \leqslant 1, \tag{2.1}$$

and the solution is used in a 2-step backward-differentiation type formula which steps from $x = nh$ to $x = (n + 1)h$ of the form

$$\alpha_0 y_n + \alpha_1 y_{n+\gamma} + \alpha_2 y_{n+1} = hf_{n+1}. \tag{2.2}$$

To generate the coefficients γ, α_0, α_1 and α_2, we impose two conditions on the composite formula namely (a) it is second-order accurate and (b) for computational efficiency, both schemes share a common iteration matrix. To consider accuracy requirements, we combine (2.1) and (2.2) to obtain

$$\alpha_2 y_{n+1} = -(\alpha_0 + \alpha_1)y_n - \alpha_1\gamma h\left[(1+\theta)f_n + \theta f_{n+\gamma}\right] + hf_{n+1},$$

and, by Taylor expansion, we compare the leading terms with the corresponding Taylor series polynomial for the exact solution $y(x_{n+1})$:

$$y(x_{n+1}) = y(x_n) + hy'(x_n) + h^2y'(x_n)/2! + h^3y^{(3)}(x_n)/3! + \cdots.$$

Requiring agreement in terms up to $O(h^2)$ yields the following relations:

$$\alpha_2 = -(\alpha_0 + \alpha_1), \qquad \alpha_2 = 1 - \gamma\alpha_1, \qquad \alpha_2 = 2(1 - \gamma^2\theta\alpha_1), \tag{2.3a, b, c}$$

while both methods have a common iteration matrix if

$$\alpha_2\gamma\theta = 1. \tag{2.3d}$$

Combining (2.3a) to (2.3d) finally gives the scheme

$$\begin{aligned} &y_{n+\gamma}=y_n+\gamma h\left[(1-\theta)f_n+\theta f_{n+\gamma}\right] \quad 0<\theta\leqslant 1,\\ &\alpha_0 y_n+\alpha_1 y_{n+\gamma}+\alpha_2 y_{n+1}=hf_{n+1},\\ &\gamma\theta=1-1/\sqrt{2}, \qquad \alpha_2=2(1-\gamma\theta)/(1-2\gamma\theta),\\ &\alpha_1=(1-\alpha_2)/\gamma, \qquad \alpha_0=-\alpha_1-\alpha_2. \end{aligned} \tag{2.4}$$

In addition, the common iteration matrix takes the form

$$B=I-\theta\gamma hJ, \qquad J=\partial f/\partial y,$$

and the scheme is second-order accurate with an associated error estimate:

$$\text{errest}=\left(\frac{3\gamma^2\theta-4\gamma\theta+1}{12(1-\gamma\theta)}\right)h^3 y_n^{(3)}(\xi). \tag{2.5}$$

Using a Taylor expansion, it is easily verified that we can approximate $y^{(3)}(\xi)$ using the following combination of (available) function values:

$$Y_n^{(3)}(\xi)\simeq\frac{2}{h^2}\left[\frac{1}{\gamma}f_n-\frac{1}{\gamma(1-\gamma)}f_{n+\gamma}+\frac{1}{1-\gamma}f_{n+1}\right]. \tag{2.6}$$

To consider the stability properties of the composite scheme (2.4), we apply it to approximate the solution of the autonomous scalar problem

$$y'=\lambda y, \quad \operatorname{Re}(\lambda)<0, \tag{2.7}$$

Using the notation $q=h\lambda$, for the one-step theta scheme we obtain

$$y_{n+\gamma}=\frac{1+\gamma(1-\theta)q}{1-\gamma\theta q}y_n,$$

and, substituting for $y_{n+\gamma}$ in the two-step scheme (2.2), we have

$$y_{n+1}=\frac{2(1-\gamma\theta)+[1-2\gamma\theta(1-\gamma\theta)]q}{2(1-\gamma\theta)(1-\gamma\theta q)^2}y_n,$$

which can be rewritten as

$$y_{n+1}\equiv R(q,\theta)y_n=\left(\frac{1+\dfrac{1-2\gamma\theta(1-\gamma\theta)}{2(1-\gamma\theta)}q}{(1-\gamma\theta q)^2}\right)y_n. \tag{2.8}$$

Inspection of (2.8) shows that the composite integration scheme is both A and L stable. Recalling that $\gamma\theta=1-\frac{1}{2}\sqrt{2}$, it is interesting to observe that

$$|R(q,\theta)-e^q|=\frac{3\sqrt{2}-4}{6}q^3+o(q^4)\simeq 0.04044lq^3+o(q^4)$$

where the coefficient of q^3 compares directly with the coefficient of $h^3y^{(3)}(\xi)$ in the error estimate (2.5).

3. Implementation

In the spirit of [9], we summarise the essential ingredients of the proposed scheme:

(a) A formula to compute the next approximation,
(b) An estimate for the local truncation error,
(c) A strategy for accepting or rejecting the approximation,
(d) A strategy for choosing the next stepsize,
(e) A strategy for Jacobian updating.

To implement (2.4), we consider the iterative application of Newton's method to the the theta scheme (2.1), employing a fixed number of iterations $i = 1, 2, \ldots, i\max$, as follows:

$$\left[I - \gamma h \theta J_{n+\gamma}^{(i)}\right] \Delta y_{n+\gamma}^{(i)} = y_n - y_{n+\gamma}^{(i)} + \gamma h\left[(1-\theta) f_n + \theta f_{n+\gamma}^{(i)}\right] \tag{3.1}$$

where

$$\Delta y_{n+\gamma}^{(i)} = y_{n+\gamma}^{(i+1)} - y_{n+\gamma}^{(i)}, \qquad y_{n+\gamma}^{(0)} = y_n.$$

Equivalently the iterative application of the two-step method (2.2), where it is now assumed that $y_{n+\gamma}$ is determined from (3.1), may be written as

$$\left[I - \frac{h}{\alpha_2} J_{n+1}^{(i)}\right] \Delta y_{n+1}^{(i)} = \frac{1}{\alpha_2}\left[h f_{n+1}^{(i)} - \left(\alpha_0 y_n + \alpha_1 y_{n+\gamma} + \alpha_2 y_{n+1}^{(i)}\right)\right] \tag{3.2}$$

using the same notation as before where $i = 1, 2, \ldots, i\max$ is the iteration counter. To save computational expense, we employ the standard modified Newton iteration scheme (noting that $h/\alpha_2 = h\gamma\theta$) resulting from setting

$$I - h\gamma\theta J_{n+\gamma}^{(i)} = I - \frac{h}{\alpha} J_{n+1}^{(i)} = I - h\gamma\theta J_m \equiv B_m$$

where J_m is some piecewise constant approximation to the Jacobian matrices of both formulae for some $0 \leqslant m \leqslant n$. The iterative application of (2.4) is therefore implemented by the following recursions:

$$\begin{aligned} &B_m \, \Delta y_{n+\gamma}^{(i)} = y_n - y_{n+\gamma}^{(i)} + \gamma h\left[(1-\theta) f_n + \theta f_{n+\gamma}^{(i)}\right], (3.3) \\ &y_{n+\gamma}^{(0)} = y_n, \quad f_{n+\gamma}^{(0)} = f_n, \qquad i = 1, 2, \ldots, i\max, \end{aligned} \tag{3.3}$$

$$\begin{aligned} &B_m \, \Delta y_{n+1}^{(i)} = \frac{1}{\alpha_2}\left[h f_{n+1}^{(i)} - \left(\alpha_0 y_n + \alpha_1 y_{n+\gamma} + \alpha_2 y_{n+1}^{(i)}\right)\right], \\ &y_{n+1}^{(0)} = y_{n+\gamma}, \quad f_{n+1}^{(0)} = f_{n+\gamma}, \qquad i = 1, 2, \ldots, i\max. \end{aligned} \tag{3.4}$$

In implementing (3.3), (3.4) to approximate the solution of a nonlinear initial-value problem, the resulting nonlinear algebraic equations for each of the required approximations $y_{n+\gamma}$ and y_{n+1} are solved using the described modified iterative algorithm for a fixed number of iterations (typically $i\max = 4, 5$). The algorithm may be summarised as follows:

```
Given absolute ATOL and/or relative RTOL tolerances:
Initially: $n = m = 0$: Factorize $B_0 = I - h\gamma\theta J_0$
WHILE (final time is not reached) DO
    Define $e_n = \mathrm{RTOL} * |y_n| + \mathrm{ATOL}$
    Update $B_m$ if necessary
```

```
WHILE $|\Delta y_{n+\gamma}| > 0.1 * e_{n+\gamma}$ and $i < i\max$
Compute $y_{n+\gamma}$ from (3.3):
WHILE $|\Delta y_{n+1}| > 0.1 * e_{n+1}$ and $i < i\max$
Compute $y_{n+1}$ using (3.4):
Using (2.5) compute
$\tau_{n+1} = \left| \frac{3\gamma^2\theta - 4\gamma\theta + 1}{12(1-\gamma\theta)} h^3 y_n^{(3)}(\xi) \right|$
and hence the quantity:
$r^2 = \frac{1}{N} \sum_{i=1}^{N} \left( \frac{\tau_{n+1,i}}{e_{n+1,i}} \right)^2$
IF
IF $r \geqslant 1$ THEN reject $y_{n+1}$ and $h_{new} = 0.5 * h_{old}$
        ELSE accept $y_{n+1}$ and
            IF $r > 0.5$ THEN $h_{new} = h_{old}$
                     ELSE $h_{new} = r^{1/3}\ h_{old}$
                          provided $h_{old}$ was unchanged for the previous 3 steps
END IF
```

The decision to update the Jacobian (and hence the iteration) matrix is based on a combination of the following:

(i) $h_{\text{new}} \geqslant \beta h_{\text{old}}$,

(ii) $r > 0.85$,

(iii) the Jacobian has not been updated for the past k steps.

The choice of β and k are arbitrarily chosen to be 2 and 15 [7] respectively. It will be useful in later applications to measure the rate of convergence:

$$\lambda = \| \Delta y_L^{(i+1)} / \Delta y_L^{(i)} \|$$

(where L is $n+\gamma$ or $n+1$) and, following [7], decide to update J whenever $\lambda > 0.5$. For the systems of ordinary differential equations considered in this paper, it proves more expedient to reduce the stepsize before deciding to update J. We remark that, while many codes iterate to convergence, we impose an iteration limit (typically $i\max \leqslant 5$). As noted in [11], in the former case, the number of iterations, and hence the number of function evaluations, is open-ended and typically much greater than for the latter approach. For their PECE algorithms [11], when iterating to convergence, they noted limitations on the convergence of the iterative process itself. In [6], the iteration matrix is held fixed over as many steps is possible. If, after 5 iterations, the correction vector is still "large", they re-evaluate the Jacobian matrix at the most recently converged solution whereupon the iterative cycle is retried with a step-size halving if this is not successful.

4. Numerical experiments

Taking the specific case $\theta = 0.55$ [7], we apply the composite integration scheme to a number of test problems. The first set comprise Class A to E of [9]. At different (absolute) tolerances, we measure

Table 1

Class A with TOL $=10^{-2}$

	A1	A2	A3	A4
NSTEP	22	25	33	26
NFE	62	68	122	90
NJE	8	9	14	15

Table 2

Class B with TOL $=10^{-2}$

	B1	B2	B3	B4	B5
NSTEP	98	22	21	28	49
NFE	464	68	64	138	261
NJE	19	8	8	7	9

Table 3

Class C with TOL $=10^{-2}$

	C1	C2	C3	C4	C5
NSTEP	22	34	23	65	234
NFE	73	111	83	488	1792
NJE	8	11	8	16	19

Table 4

Class D with TOL $=10^{-2}$

	D1	D2	D3	D4	D5	D6
NSTEP	130	130	30	22	27	14
NFE	806	622	155	62	82	42
NJE	23	22	15	8	8	6

(a) the number of composite integration steps NSTEP,

(b) the total number of functions evaluations NFE, and

(c) the total number of Jacobian evaluations NJE. The method of testing is analogous to that in [9] but for each problem the initial stepsize was chosen arbitrarily to be TOL/10.

The results are summarized in Tables 1–10.

In [9], results are summarized for 5 different methods: two variable order schemes GEAR and SDBASIC and three fixed order schemes TRAPEX, GENRK and IMPRK. The authors conclude "tentatively" that GEAR, SDBASIC and TRAPEX are "quite good" with certain limitations:

(i) GEAR is less efficient when the eigenvalues are close to the imaginary axis;

(ii) SDBASIC becomes less efficient as the problem becomes more highly nonlinear and

Table 5

Class E with TOL $=10^{-2}$

	E1	E2	E3	E4	E5
NSTEP	13	18	129	63	29
NFE	13	52	407	296	29
NJE	5	6	31	14	10

Table 6

Class A with TOL $=10^{-4}$

	A1	A2	A3	A4
NSTEP	60	60	87	80
NFE	264	256	386	431
NJE	10	13	12	14

Table 7

Class B with TOL $=10^{-4}$

	B1	B2	B3	B4	B5
NSTEP	409	56	60	85	199
NFE	1814	255	297	388	852
NJE	67	10	10	12	20

Table 8

Class C with TOL $=10^{-4}$

	C1	C2	C3	C4	C5
NSTEP	68	70	69	259	1029
NFE	336	317	358	1591	5471
NJE	11	13	15	36	164

Table 9

Class D with TOL $=10^{-4}$

	D1	D2	D3	D4	D5	D6
NSTEP	567	561	63	32	63	35
NFE	3287	2692	313	101	278	144
NJE	99	80	11	12	16	10

Table 10

Class E with TOL $=10^{-4}$

	E1	E2	E3	E4	E5
NSTEP	25	43	396	193	39
NFE	44	174	1412	953	39
NJE	9	8	130	37	14

Table 11

All Problems, TOL $= 10^{-2}$

	GEAR	SDBASIC	TRAPEX	(2.4)
NSTEP	4059	593	465	1307
NFE	10798	3366	7802	6450
NJE	445	2571	705	307

(iii) TRAPEX becomes relatively less efficient at more stringent tolerances and also when the problem exhibits strong nonlinear coupling between smooth and transient components.
In Tables 11 and 12, we present a summary of the total number of integration steps, function and Jacobian evaluations for the various methods at two tolerances.

Note that GEAR is not included in Table 12 since this method failed to solve the chemical kinetics problem E5 due to a round-off error (which introduced a small negative concentration whereupon the method attempted to follow the resulting unstable solution). Excluding E5, the statistics for GEAR at TOL $= 10^{-4}$ are NSTEP = 7172, NFE = 18539 and NJE = 541.

It is evident from Tables 1–10 that the proposed method (2.4) experiences considerable difficulty when solving the four problems C5, D1, D2 and E3 and the statistics collected from this subset tend to dominate the overall figures in Tables 11 and 12. This is reflected in the figures presented below:

Method (2.4) with TOL $= 10^{-2}$:

	{C5, D1, D2, E3}	All Problems
NSTEP	623	1307
NFE	3627	6450
NJE	95	307

Method (2.4) with TOL $= 10^{-4}$:

	{C5, D1, D2, E3}	All Problems
NSTEP	2350	4608
NFE	12866	22457
NJE	473	833

In the latter case, namely when TOL $= 10^{-4}$, it can be seen that the number of both function and Jacobian evaluations for the 4 problems C5, D1, D2 and E3 comprise approximately 57% of the corresponding total figures. The behaviour of (2.4) when applied to C5 is similar to that reported in [9] for the three fixed order methods tested. Note that, in terms of function and

Table 12

All Problems, TOL $= 10^{-4}$

	SDBASIC	TRAPEX	(2.4)
NSTEP	1215	1662	4608
NFE	8242	37095	22457
NJE	6898	2091	833

Jacobian evaluations, the composite scheme is more competitive than TRAPEX at both tolerances tested. The scheme is designed to be economical with respect to Jacobian evaluations and this is clearly evident when $\mathrm{TOL} = 10^{-4}$.

We remark that the proposed method (2.4) is a fixed-order method and, as noted in [4], the advantages of using variable-order methods include

(i) the ease with which they can adjust order and steplength;

(ii) the availability of higher-order stable schemes and not least;

(iii) their efficiency with respect to computational effort.

However, we believe that (2.4) is efficient especially at less stringent tolerances. A particular application of the method (which will be reported in a future paper) where stringent tolerances are typically unnecessary lies in the simulation of parabolic partial differential equations.

We next consider the application of the composite integration scheme to the approximate solution of a number of linear and nonlinear test problems taken from the literature. The problems are detailed below:

(P1) $$y_1' = -6y_1 + 5y_2 + 2\sin(x), \qquad y_2' = 94y_1 - 95y_2;$$
$$y_1(0) = 0, \qquad y_2(0) = 0, \qquad 0 \leqslant x \leqslant 100;$$

with exact solution [1]:

$$y_1(x) = \gamma\, e^{-x} + \frac{1}{\lambda}\left[\frac{10}{99}\, e^{-1000x} - 9496\cos(x) + 9506\sin(x)\right],$$
$$y_2(x) = \gamma\, e^{-x} + \frac{1}{\lambda}\left[-\frac{108}{99}\, e^{-1000x} - 9494\cos(x) + 9306\sin(x)\right],$$

where $\gamma = 94/99$ and $\lambda = 10001$.

(P2) $$y_i' = -\alpha y_1 - \beta y_2 + (\alpha + \beta - 1)\, e^{-x},$$
$$y_2' = \beta y_1 - \alpha y_2 + (\alpha - \beta - 1)\, e^{-x} \qquad \alpha = 1, \quad \beta = 15;$$
$$y_1(0) = 1, \quad y_2(0) = 1; \qquad 0 \leqslant x \leqslant 20;$$

with exact solution [5]:

$$y_1(x) = y_2(x) = e^{-x}.$$

(P3) $$y_1' = -4498y_1 - 5996y_2 + 0.006 - x,$$
$$y_2' = 2248.5y_1 + 2997y_2 - 0.503 + 3x;$$
$$y_1(0) = \frac{25498}{1500}, \qquad y_2(0) = -\frac{16499}{1500}, \qquad 0 \leqslant x \leqslant 25;$$

with exact solution [8]

$$y_1(x) = -2\, e^{-x} + 7\, e^{-1500x} + \frac{1}{1500}(17998 - 14991x),$$
$$y_2(x) = 1.5\, e^{-x} - 3.5\, e^{-1500x} - \frac{1}{1500}(13499 - 11245.5x).$$

(P4) $$y_1' = -0.2[(4b + g)y_1 + (2b - 2g)y_2] - \frac{2\mu}{25}\, e^{bt}(2y_1 + y_2)^2,$$
$$y_2' = -0.2[(2b - 2g)y_1 + (b + 4g)y_2] - \frac{\mu}{25}\, e^{bt}(2y_1 + y_2)^2;$$
$$b = 0.2, \qquad g = 200, \qquad \mu = 10^{-5},$$
$$y_1(0) = 2, \qquad y_2(0) = 1, \qquad 0 \leqslant x \leqslant 20,$$

with exact solution [13]:

$$y_1(x) = F(x), \qquad y_2(x) = 2F(x), \qquad F(x) = \mathrm{e}^{-bt}(1 + \mu x)^{-1}.$$

(P5) $y_i' = -\beta_i y_i - y_i^2, \quad 1 \leqslant i \leqslant 4;$

$$\beta_1 = 1000, \qquad \beta_2 = 800, \qquad \beta_3 = -10, \qquad \beta_4 = 0.001,$$

$$y_i(0) = -1, \qquad 1 \leqslant i \leqslant 4; \qquad 0 \leqslant x \leqslant 20;$$

with exact solution [19]:

$$y_i(x) = \frac{\beta_i}{1 - (1 + \beta_i)\,\mathrm{e}^{\beta_i x}}, \quad 1 \leqslant i \leqslant 4.$$

(P6) $y_1' = -0.04y_1 + 10^4 y_2 y_3, \qquad y_1(0) = 1;$

$$y_2' = 0.04y_1 - 10^4 y_2 y_3 - 3 \times 10^7 y_2^2, \qquad y_2(0) = 0;$$

$$y_3' = 3 \times 10^7 y_2^2, \qquad y_3(0) = 0.$$

This problem was considered by many authors including [5], [6], and [10]. In the numerical experiments which follow, the exact solution was approximated by the NAG routine D02EBF with a tolerance of 10^{-7} with the range of integration from 0 to 40.

Using the value $\theta = 0.55$, we applied (2.4) to the approximate integration of the test problems (P1) to (P6) for three tolerance values 10^{-m}, $m = 2$, 3, and 4, using relative error control for (P3) and absolute error control for the remaining problems. The statistics measured include NSTEP, NFE and NJE (which were introduced earlier) together with ERRGLO, the largest absolute component of error encountered over the integration interval. The initial stepsize for all problems was arbitrarily chosen to be TOL/20. The results are summarised in Tables 13, 14 and 15.

In the case of (P6), we can assess the accuracy of the final solution values (at XEND = 40) by comparing the numerical approximations obtained by (2.4) with the approximation reported in [10] (where a tolerance of 10^{-9} was employed). They are given in Table 16.

In addition to variable-stepsize integration, a number of problems were solved using fixed-stepsize sequences. With some of the problems given above, we also considered the following cases:

(P7) $y_i' = 0.2(y_2 - y_1), \qquad y_1(0) = 0;$

$$y_2' = 10y_1 - (60 - .125x)y_2 + 0.125x, \quad y_2(0) = 0.$$

Table 13

Problems (P1)–(P6), TOL $= 10^{-2}$

	(P1)	(P2)	(P3)	(P4)	(P5)	(P6)
NSTEP	49	50	37	29	37	35
NFE	246	167	143	73	174	99
NJE	9	18	12	7	10	8
ERRGLO	0.40 E−2	0.33 E−2	0.22 E−2	0.33 E−2	0.16 E−1	0.36 E−2

Table 14

Problems (P1)–(P6), TOL $=10^{-3}$

	(P1)	(P2)	(P3)	(P4)	(P5)	(P6)
NSTEP	81	68	48	38	64	39
NFE	444	377	218	119	355	116
NJE	13	19	12	8	11	10
ERRGLO	0.93 E−3	0.18 E−3	0.13 E−2	0.11 E−2	0.20 E−2	0.41 E−3

Table 15

Problems (P1) − (P6), TOL $=10^{-4}$

	(P1)	(P2)	(P3)	(P4)	(P5)	(P6)
NSTEP	155	91	84	57	115	54
NFE	1084	491	401	189	672	230
NJE	19	18	14	10	13	12
ERRGLO	0.22 E−3	0.57 E−4	0.39 E−3	0.29 E−3	0.44 E−3	0.11 E−3

Table 16

(P6)—Comparison with EPISODE [10]

	FORMULA (2.4)			EPISODE [10]
	TOL $=10^{-2}$	TOL $=10^{-3}$	TOL $=10^{-4}$	TOL $=10^{-9}$
$Y_1(40)$	0.713303 E0	0.715426 E0	0.715722 E0	0.71583 E0
$Y_2(40)$	0.907555 E−5	0.917102 E−5	0.917167 E−5	0.91855 E−5
$Y_3(40)$	0.286688 E0	0.284565 E0	0.284269 E0	0.28416 E0

Taken from [15], where the interval of integration is [0, 400], this is the nonautonomous form of D1 in [9] and the exact solution will be approximated by the NAG routine D02EBF using a tolerance of 10^{-7}.

(P8) $$y_1' = -y_2 + \left(1 - y_1^2 - y_2^2\right), \quad y_1(0) = 1;$$
$$y_2' = y_1 + \left(1 - y_1^2 - y_2^2\right), y_2(0) = 0.$$

The interval of integration was taken as $0 \leqslant x \leqslant 20$ and the exact solution is given by:

$$y_1(x) = \cos(x), \qquad y_2(x) = \sin(x).$$

(P9) $$y_1' = -10004 y_1 + 10^5 y_2^4, \qquad 0 \leqslant x \leqslant 5,$$
$$y_2' = y_1 - y_2 - y_2^4,$$
$$y_1(0) = 10000/10004, \qquad y_2(0) = 1.$$

Table 17

Max errors with uniform step

h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
(P1)	0.72 E−3	0.18 E−3	0.44 E−4	0.15 E−4
(P2)	0.23 E−3	0.54 E−4	0.13 E−4	0.32 E−5
(P4)	0.39 E−4	0.26 E−4	0.10 E−4	0.13 E−4
(P7)	0.22 E−4	0.30 E−4	0.74 E−5	0.24 E−5
(P8)	0.30 E−2	0.74 E−3	0.16 E−3	0.21 E−4
(P9)	0.24 E−3	0.61 E−4	0.16 E−4	0.98 E−5
(P10)	0.24 E−1	0.59 E−2	0.15 E−2	0.37 E−3

This problem was taken from [14] where an approximation to the exact solution was provided. In experiment however, the exact solution was approximated by the NAG routine D02EBF with a tolerance of 10^{-7}.

(P10) $y_1' = -y_1, \quad y_1(0) = 5;$

$y_2' = y_1^2 - 2y_2, \quad y_2(0) = 5.$

This problem was solved on the interval [0,20] and has the exact solution

$$y_1(x) = 5\,\mathrm{e}^{-x}, \qquad y_2(x) = 5\,\mathrm{e}^{-2x}(1 + 5x).$$

Using the uniform step size sequences $h = 2^{-m}$, $3 \leqslant m \leqslant 6$, we applied the composite integration scheme (2.4) to a number of the above test problems and, in each case, we measure the maximum absolute component of error over the integration interval. The results are summarised in Table 17. In the case of (P7), we compare the numerical approximations obtained using (2.4) for various values of h with the analytical solution given in [15] at $x = 400$. The results are presented in Table 18.

5. Conclusions

The numerical results illustrate some of the properties of the proposed approach. The algorithm is competitive with alternative fixed order methods especially with respect to Jacobian evaluations. However, unlike variable order formulae, it is clearly becomes inefficient at the more

Table 18

(P7)—Reference solution, $x = 400$

h	y_1	y_2
$\frac{1}{8}$	22.2422490237	27.1107399846
$\frac{1}{16}$	22.2422273401	27.1107199744
$\frac{1}{32}$	22.2422219152	27.1107149984
$\frac{1}{64}$	22.2422205585	27.1107137577
Exact [15]	22.242220	27.110713

stringent tolerances. We conclude nonetheless that low order formulae are still useful in many applications, for example in the simulation of parabolic partial differential equations where the space variable is discretized giving a stiff system of ordinary differential equations in the time variable. In such applications, it is unnecessary to specify temporal accuracy constraints significantly less than the errors resulting from the spatial discretization. Such applications will be reported in a future paper. We remark finally that the choice of θ was taken arbitrarily to be 0.55 throughout the numerical experiments in this paper. This value was prompted by [7] and some limited numerical testing of other values of θ, notably $\theta = 0.5$ which was considered in [2], suggested that it is close to optimal for the problems selected. Further investigations into the choice of an optimal value of θ, for certain problem classes, will be considered in a future study.

References

[1] J.F. Andrus, Numerical solution of systems of ordinary differential equations separated into subsystems, *SIAM J. Numer. Anal.* **16** (1979) 605–611.

[2] R.E. Bank, W.M. Coughran, W. Fichtner, E.H. Grosse, D.J. Rose and R.K. Smith, Transient simulation of silicon devices and circuits, *IEEE Trans. Electron Devices* **ED-32** (1985) 1992–2007.

[3] G.D. Byrne and A.C. Hindmarsh, A polyalgorithm for the numerical solution of ordinary differential equations, *ACM Trans. Math. Software* **1** (1975) 71–96.

[4] J.R. Cash, On the integration of stiff systems of O.D.E.s using extended backward differentiation formulae, *Numer. Math.* **34** (1980) 235–246.

[5] J.R. Cash, Second derivative extended backward differentiation formulas for the numerical integration of stiff systems, *SIAM J. Numer. Anal.* **18** (1981) 21–36.

[6] J.R. Cash and A. Singhal, Mono-implicit Runge–Kutta formulae for the numerical integration of stiff differential systems, *IMA J. Numer. Anal.* **2** (1982) 211–227.

[7] T.S. Chua and P.M. Dew, The design of a variable-step integrator for the simulation of gas transmission networks, Rep. No. 157, Dept. Computer Studies, Leeds, 1982.

[8] B.L. Ehle and J.D. Lawson, Generalized Runge–Kutta processes for stiff initial value problems, *J. Inst. Math. Appl.* **16** (1975) 11–21.

[9] W.H. Enright, T.E. Hull and B. Lindberg, Comparing numrical methods for stiff systems of ODE's, *BIT* **15** (1975) 10–48.

[10] A.C. Hindmarsh and G.D. Byrne, Applications of EPISODE: an experimental package for the integration of systems of ordinary differential equations, in: L. Lapidus and W. Schliesser, Eds., *Numerical Methods for Differential Systems* (Academic Press, New York, 1976).

[11] R.W. Klopfenstein and C.B. Davis, PECE algorithms for the solution of stiff systems of ordinary differential equations, *Math. Comp.* **25** (1971) 457–473.

[12] F.T. Krogh, On testing a subroutine for the numerical integration of ordinary differential equations, *J. ACM* **20** (1973) 545–562.

[13] W. Liniger, High order A-stable averaging algorithms for stiff differential systems, in: L. Lapidus and W. Schliesser, Eds., *Numerical Methods for Differential Systems* (Academic Press, New York, 1976).

[14] Oey Li-Yauw, Implicit schemes for differential equations, *J. Comp. Physics* **45** (1982) 443–468.

[15] J.M. Sanz-Serna, Linearly implicit variable coefficient methods of Lambert–Sigurdsson type, *IMA J. Numer. Anal.* **1** (1981) 39–45.

[16] L.F. Shampine, Evaluation of implicit formulas for the solution of ODEs, *BIT* **19** (1979) 495–502.

[17] L.F. Shampine, Implementation of Rosenbrock methods, Rep. SAND80-2367J, Sandia Laboratories, Albuquerque, 1980.

[18] L.F. Shampine, Evaluation of a test set for stiff ode solvers, *ACM Trans. Math. Software* **7** (1981) 409–420.

[19] H. Tian-Min, Numerical small parameter method for stiff ode's, *BIT* **23** (1983) 118–131.