Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

J.G. Blom, H. Brunner

The numerical solution of nonlinear Volterra integral equations of the second kind by collocation and iterated collocation methods

Department of Numerical Mathematics Report NM-R8522 October

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

The Numerical Solution of Nonlinear Volterra Integral Equations of the Second Kind by Collocation and Iterated Collocation Methods

J.G. Blom
Centre for Mathematics and Computer Science,
Kruislaan 413, 1098 SJ Amsterdam, The Netherlands

H. Brunner
Institut de Mathématiques, Université de Fribourg,
CH-1700 Fribourg, Switzerland

The subject of this paper is a variable-stepsize one-step method of collocation type for solving general nonlinear second kind Volterra integral equations. We extend the iterated collocation method corresponding to polynomial spline collocation to nonlinear Volterra integral equations of the second kind. The resulting superconvergence properties of either the collocation approximation or the iterated collocation approximation are used to obtain (local and global) error estimates which in turn form the basis of a variable stepsize code. The performance of this code is illustrated by means of numerous test problems.

1980 Mathematics subject classification: Primary:65R20. Secondary:45D05, 45L10
Key Words & Phrases: nonlinear Volterra integral equation, polynomial splines, collocation, iterated collocation, superconvergence, error estimates, variable stepsize

Note: This paper is submitted for publication

0. INTRODUCTION

Most of the work on the numerical solution of Volterra integral equations of the second kind done to date deals with fixed-step methods. Recently, some automatic codes have been published . Some of these employ fixed stepsizes and make use of extrapolation tableaux to control the global error (VOLCON[13] , IVRKX(C)[19]). Others use variable stepsizes based on local error controlling techniques (VOLTEX[14] , RKVIEP[12] , INTSOL[16] , ORION[3] ,[17]).

The variable stepsize method we propose offers the user the choice to control the global or the local error, with the possibility to control also the uniform error, as suggested by Arndt[2] .

It is believed by the authors that iterated collocation is a very robust and not too expensive method to control the global error. Since there exists a class of problems to which this method cannot be applied, some other combinations of collocation methods are investigated. The performance of these methods was considerably improved by using uniform error control.

The variable-stepsize collocation methods described below have been implemented in a FORTRAN code COLVI2 (see [4]).

Report NM-R8522
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

1. POLYNOMIAL SPLINE COLLOCATION

1.1. Exact collocation equations

Consider the nonlinear second-kind Volterra integral equation,

$$y(t) = g(t) + \int_0^t k(t,s,y(s))ds, \quad t \in I := [0,T], \tag{1.1.1}$$

where g and k are given continuous functions on I and on $S \times \mathbb{R}$ (with $S := \{(t,s): 0 \leqslant s \leqslant t \leqslant T\}$, respectively, and where k is such that (1.1.1) possesses a unique solution $y \in C(I)$ (compare, e.g., Miller[18] , or Brunner and van der Houwen[10]).

In order to discretize (1.1.1) let

$$\Pi_N : 0 = t_0 < t_1 < \cdots < t_N = T \quad (N \geqslant 1)$$

denote a partition (or: mesh) for the given interval I, and set

$$\begin{aligned} h_n &:= t_{n+1} - t_n \quad (n = 0,...,N-1), \\ \sigma_0 &:= [t_0,t_1], \quad \sigma_n := (t_n,t_{n+1}] \quad (n = 1,...,N-1), \\ Z_N &:= \{t_n: n = 1,...,N-1\} \quad \text{(the set of interior mesh points)}, \end{aligned}$$

and

$$\bar{Z}_N := Z_N \cup \{T\}.$$

In the following analysis it will always be assumed that the mesh sequence $\{\Pi_N\}$ $(N \in \mathbb{N})$ is quasi-uniform; i.e.,

$$\frac{\max\limits_{(n)}(h_n)}{\min\limits_{(n)}(h_n)} \leqslant \gamma < \infty$$

uniformly for $N \in \mathbb{N}$ (for ease of notation we have suppressed the superscript N in $t_n^{(N)}$, $h_n^{(N)}$, indicating the dependence of these quantities on N). In the following $h := \max\limits_{(n)}(h_n)$ denotes the diameter of the mesh Π_N.

The exact solution y of (1.1.1) will be approximated in the polynomial spline space

$$S_{m-1}^{(-1)}(Z_N) := \{u: u|_{\sigma_n} =: u_n \in \pi_{m-1},\ n = 0,...,N-1\ (m \geqslant 1)\}, \tag{1.1.2}$$

whose elements reduce, on each subinterval σ_n, to (real) polynomials of degree not exceeding $m-1$ and which, in general, possess (finite) discontinuities at their knots Z_N. Here, the value $m = 1$ yields, of course, the space of step functions having knots Z_N.

Let $\{c_j\}$, with $0 \leqslant c_1 < \cdots < c_m \leqslant 1$, be a given set of parameters, and define the sets

$$X_n := \{t_{n,j} := t_n + c_j h_n: j = 1,...,m\} \quad (n = 0,...,N-1), \tag{1.1.3a}$$

and

$$X(N) := \bigcup_{n=0}^{N-1} X_n. \tag{1.1.3b}$$

The desired approximation $u \in S_{m-1}^{(-1)}(Z_N)$ will be determined by requiring that u satisfy the integral equation (1.1.1) on the finite set $X(N)$ ('collocation' on $X(N)$):

$$u(t) = g(t) + \int_0^t k(t,s,u(s))ds \quad \text{for all } t \in X(N).$$

This *collocation equation* may also be written as

$$u_n(t) = g(t) + \int_{t_n}^{t} k(t,s,u_n(s))ds + \sum_{i=0}^{n-1} \int_{t_i}^{t_{i+1}} k(t,s,u_i(s))ds, \quad t \in X_n \quad (n=0,\ldots,N-1), \tag{1.1.4}$$

thus explicitly exhibiting its recursive character. Note that if the collocation parameters $\{c_j\}$ are such that $c_1=0$ and $c_m=1$, then the approximation defined by (1.1.4) will be continuous on I; i.e., we then have

$$u \in S_{m-1}^{(0)}(Z_N) := S_{m-1}^{(-1)}(Z_N) \cap C(I).$$

For the subsequent analysis we introduce the integrals

$$\Phi_{n,i}^{(j)}[u_i] := \begin{cases} \int_0^1 k(t_{n,j}, t_i + \tau h_i, u_i(t_i+\tau h_i))d\tau, & \text{if } 0 \leqslant i \leqslant n-1, \\ \int_0^{c_j} k(t_{n,j}, t_n + \tau h_n, u_n(t_n+\tau h_n))d\tau, & \text{if } i = n \end{cases} \quad (j=1,\ldots,m). \tag{1.1.5}$$

With this notation the collocation equation (1.1.4) becomes

$$u_n(t_{n,j}) = g(t_{n,j}) + h_n \Phi_{n,n}^{(j)}[u_n] + \sum_{i=0}^{n-1} h_i \Phi_{n,i}^{(j)}[u_i], \quad j=1,\ldots,m \ (n=0,\ldots,N-1). \tag{1.1.6}$$

Setting

$$u_n(t_n+\tau h_n) = \sum_{l=1}^{m} L_l(\tau) Y_{n,l}, \quad t_n + \tau h_n \in \sigma_n, \tag{1.1.7}$$

with $Y_{n,l} := u_n(t_{n,l})$, and where the polynomials

$$L_l(\tau) := \prod_{\substack{\kappa=1 \\ \kappa \neq l}}^{m} \frac{(\tau - c_\kappa)}{(c_l - c_\kappa)}$$

represent the Lagrange canonical polynomials for the collocation parameters $\{c_j\}$, we see that (1.1.6) represents, for each $n=0,\ldots,N-1$, a nonlinear system in $\mathbb{R}^m$ for the vector $Y_n := (Y_{n,1}, \ldots, Y_{n,m})^T$. Once Y_n has been determined, the approximation u on the subinterval σ_n is given by (1.1.7).

1.2. Discretization of the collocation equations

In most applications the integrals (1.1.5) occurring in the collocation equation (1.1.6) cannot be evaluated analytically, and one is forced to resort to employing suitable quadrature formulas for their approximation. In the following we shall use *m-point interpolatory quadrature formulas* of the form

$$\hat{\Phi}_{n,i}^{(j)}[u_i] := \sum_{l=1}^{m} w_l k(t_{n,j}, t_{i,l}, u_i(t_{i,l})), \quad \text{if } 0 \leqslant i \leqslant n-1; \tag{1.2.1}$$

for the approximation of $\Phi_{n,n}^{(j)}[u_n]$ we shall employ either

$$\tilde{\Phi}_{n,n}^{(j)}[u_n] := \sum_{l=1}^{m} w_l k(t_{n,j}, t_{n,l}, u_n(t_{n,l})), \tag{1.2.2a}$$

or

$$\hat{\Phi}_{n,n}^{(j)}[u_n] := \sum_{l=1}^{m} w_{j,l} k(t_{n,j}, t_n + c_j c_l h_n, u_n(t_n + c_j c_l h_n)). \tag{1.2.2b}$$

Here, the quadrature weights are given by

$$w_l := \int_0^1 L_l(\tau)d\tau,$$

and by

$$w_{j,l} := c_j w_l \quad (j,l=1,...,m).$$

Note that when using the quadrature approximation (1.2.2a) we need kernel values $k(t,s,.)$ at points (t,s) no longer contained in S (since $t_{n,l}>t_{n,j}$ for $l>j$), while (1.2.2b) is based on the abscissas $t_n+c_jc_lh_n\leqslant t_{n,j}$ (i.e., the affine images of $t_{n,l}$ in $[t_n,t_{n,j}]$).

The resulting fully discretized collocation equations are thus given by, respectively,

$$\tilde{u}_n(t_{n,j}) = g(t_{n,j}) + h_n\tilde{\Phi}^{(j)}_{n,n}[\tilde{u}_n] + \sum_{i=0}^{n-1} h_i\hat{\Phi}^{(j)}_{n,i}[\tilde{u}_i], \quad j=1,...,m \ (n=0,...,N-1); \tag{1.2.3a}$$

and

$$\hat{u}_n(t_{n,j}) = g(t_{n,j}) + h_n\hat{\Phi}^{(j)}_{n,n}[\hat{u}_n] + \sum_{i=0}^{n-1} h_i\hat{\Phi}^{(j)}_{n,i}[\hat{u}_i], \quad j=1,...,m \ (n=0,...,N-1); \tag{1.2.3b}$$

they yield approximations $\tilde{u}$ and $\hat{u}$ in $S^{(-1)}_{m-1}(Z_N)$ (whose restrictions to the subinterval σ_n are given by interpolation formulas analogous to (1.1.7)). These approximations will, in general, be different from the 'exact' collocation approximation u defined by (1.1.6); due to the differing quadrature errors associated with (1.2.2a) and (1.2.2b) we have also $\tilde{u}\neq\hat{u}$. However, the approximations u, $\tilde{u}$, and $\hat{u}$ all exhibit the same order of (global and local) convergence; in particular, we have the following results (see, e.g., Brunner and van der Houwen[10] ; compare also de Hoog and Weiss[15] , Brunner[7] , and Brunner and Nørsett[11]).

THEOREM 1.1. *Let g and k be smooth functions, such that (1.1.1) has a unique solution y which is sufficiently many times continuously differentiable on I, and let u, $\tilde{u}$, $\hat{u}\in S^{(-1)}_{m-1}(Z_N)$ denote, respectively, the collocation approximations determined by (1.1.6), (1.2.3a), and (1.2.3b).*
Moreover, let p, $\tilde{p}$, and $\hat{p}$ be the largest integers satisfying

$$\max_{t_n\in\bar{Z}_N}|y(t_n)-u(t_n)| = O(N^{-p}),$$

$$\max_{t_n\in\bar{Z}_N}|y(t_n)-\tilde{u}(t_n)| = O(N^{-\tilde{p}}),$$

and

$$\max_{t_n\in\bar{Z}_N}|y(t_n)-\hat{u}(t_n)| = O(N^{-\hat{p}}) \quad (\text{as } N\to\infty, \text{ with } Nh\leqslant\gamma T \text{ fixed}).$$

(a) If the collocation parameters $\{c_j\}$ are given by the zeros of the Legendre polynomials $P_m(2s-1)$ (i.e., the Gauss points for (0,1)), then

$$p = \tilde{p} = \hat{p} = m. \tag{1.2.4a}$$

(b) If the $\{c_j\}$ are the zeros of $P_{m-1}(2s-1)-P_m(2s-1)$ (i.e., the Radau II points for (0,1]), then

$$p = \tilde{p} = \hat{p} = 2m-1. \tag{1.2.4b}$$

(c) If the $\{c_j\}$ are the zeros of $s(s-1)P'_{m-1}(2s-1)$ (i.e., the Lobatto points for [0,1]), then

$$p = \tilde{p} = \hat{p} = 2m-2. \tag{1.2.4c}$$

Note that $\tilde{p}$ attains the indicated values only if the kernel $k(t,s,.)$ is smooth not only for $(t,s)\in S$ but also for $(t,s)\in S':=\{(t,s)\colon 0\leqslant s\leqslant t+\delta\}\cap I\times I$, for some $\delta>0$. This is, of course, a consequence of the special form of the quadrature approximation (1.2.2a).

We also recall that the order of global convergence on I of these three spline approximations equals m. Hence, while collocation at the Radau II points or at the Lobatto points leads to a higher order of convergence on $\overline{Z}_N$ ('local superconvergence'), collocation using the Gauss points does not have this local superconvergence property (in contrast to polynomial spline collocation applied to initial-value problems for ordinary differential equations).

In order to prepare the ground for the convergence analysis for iterated collocation (Section 2) we consider a somewhat different discretization of the collocation equation (1.1.6). To be precise, let $z\in S_m^{(-1)}(Z_N)$ be determined by the (exact) collocation equation

$$z_n(t_{n,j}) = g(t_{n,j}) + h_n\Phi_{n,n}^{(j)}[z_n] + \sum_{i=0}^{n-1} h_i\Phi_{n,i}^{(j)}[z_i], \tag{1.2.5}$$

where now $j=1,...,m+1$, $t_{n,j}:=t_n+c_jh_n$, with $0\leqslant c_1<\cdots<c_m<c_{m+1}=1$. Here, the integrals $\Phi_{n,i}^{(j)}$ are as in (1.1.5) (except that now j varies between 1 and $m+1$). These integrals are approximated by *m-point* (instead of $(m+1)$-point!) *interpolatory quadrature formulas* using abscissas based on the first m of the $m+1$ collocation parameters; i.e., we employ again the quadrature approximations (1.2.1), (1.2.2a), (1.2.2b), with z replacing u, and with j ranging from 1 to $m+1$. Accordingly, the two possible discretizations of (1.2.5) are given by

$$\tilde{z}_n(t_{n,j}) = g(t_{n,j}) + h_n\tilde{\Phi}_{n,n}^{(j)}[\tilde{z}_n] + \sum_{i=0}^{n-1} h_i\hat{\Phi}_{n,i}^{(j)}[\tilde{z}_i], \quad j=1,...,m+1\ (n=0,...,N-1), \tag{1.2.6a}$$

and

$$\hat{z}_n(t_{n,j}) = g(t_{n,j}) + h_n\hat{\Phi}_{n,n}^{(j)}[\hat{z}_n] + \sum_{i=0}^{n-1} h_i\hat{\Phi}_{n,i}^{(j)}[\hat{z}_i], \quad j=1,...,m+1\ (n=0,...,N-1); \tag{1.2.6b}$$

these equations yield, respectively, approximations $\tilde{z}$ and $\hat{z}$ in the space $S_m^{(-1)}(Z_N)$. In analogy to (1.1.7) we set, e.g.,

$$\tilde{z}_n(t_n+\tau h_n) = \sum_{l=1}^{m+1} L_l^*(\tau)\tilde{Z}_{n,l}, \quad t_n+\tau h_n\in\sigma_n, \tag{1.2.7}$$

with $\tilde{Z}_{n,l}:=\tilde{z}_n(t_{n,l})$, and with $L_l^*(\tau)$ denoting the l-th Lagrange canonical polynomial (of degree m) with respect to $c_1,\ldots,c_{m+1}$.

Consider first (1.2.6a): it follows from the quadrature approximation (1.2.2a) and from (1.2.7) that

$$\tilde{\Phi}_{n,n}^{(j)}[\tilde{z}_n] = \sum_{l=1}^{m} w_l k(t_{n,j},t_{n,l},\tilde{Z}_{n,l}), \quad j=1,...,m+1$$

(note that $L_{m+1}^*(c_l)=0$ for $l=1,...,m$). Hence, the first m equations of (1.2.6a) (corresponding to $j=1,...,m$) constitute a nonlinear system in $\mathbb{R}^m$ for $\tilde{Z}_{n,1},\ldots,\tilde{Z}_{n,m}$; the last equation ($j=m+1$) then determines, in an explicit way, the approximation $\tilde{Z}_{n,m+1}$ which, by $c_{m+1}=1$, equals $\tilde{z}_n(t_{n+1})$ and thus approximates $y(t_{n+1})$.

Turning to the discretization (1.2.6b), we observe that now, by (1.2.2b),

$$\hat{\Phi}_{n,n}^{(j)}[\hat{z}_n] = \sum_{l=1}^{m} w_{j,l} k(t_{n,j},t_n+c_jc_lh_n, \sum_{s=1}^{m+1} L_s^*(c_jc_l)\hat{Z}_{n,s}), \quad j=1,...,m+1,$$

depends not only on $\hat{Z}_{n,1},\ldots,\hat{Z}_{n,m}$, but also on $\hat{Z}_{n,m+1}$; hence, (1.2.6b) represents a nonlinear system in $\mathbb{R}^{m+1}$, with $\hat{Z}_{n,m+1}$ approximating $y(t_{n+1})$.

The following theorem shows that the approximations generated by the discretizations (1.2.6a) and (1.2.6b) exhibit the same order of convergence.

THEOREM 1.2. *Let g and k be smooth functions, such that the integral equation (1.1.1) possesses a unique solution y which is sufficiently many times continuously differentiable on I, and let z, $\tilde{z}$, $\hat{z} \in S_m^{(-1)}(Z_N)$ denote, respectively, the collocation approximations determined by (1.2.5), (1.2.6a), and (1.2.6b). If p, $\tilde{p}$ and $\hat{p}$ are the largest integers satisfying*

$$\max_{t_n \in \overline{Z}_N} |y(t_n) - z(t_n)| = O(N^{-p}),$$

$$\max_{t_n \in \overline{Z}_N} |y(t_n) - \tilde{z}(t_n)| = O(N^{-\tilde{p}}),$$

and

$$\max_{t_n \in \overline{Z}_N} |y(t_n) - \hat{z}(t_n)| = O(N^{-\hat{p}}),$$

and if the first m collocation parameters $c_1, \ldots, c_m$ are the zeros of $P_m(2s-1)$ (Gauss points for (0,1)), while $c_{m+1} = 1$, then

$$p = \tilde{p} = \hat{p} = 2m. \tag{1.2.8}$$

The assertion $\tilde{p} = 2m$ will hold only if the kernel $k(t,s,.)$ can be extended smoothly to the domain $S' := \{(t,s)\colon 0 \leqslant s \leqslant t+\delta\} \cap I \times I$, where $\delta > 0$ is suitably chosen.

A proof of this result may be found in Brunner[7] , or in Brunner and van der Houwen[10] .

We mention in passing that the discretized collocation equation (1.2.6a) represents an *m-stage implicit Volterra-Runge-Kutta method of POUZET type* (cf. Brunner, Hairer and Nørsett[9]). It will be shown in the next section (Theorem 2.1) that this method is equivalent to a certain discretized version of an *iterated collocation method* , in the sense that they both produce identical approximations to y on $\overline{Z}_N$.

2. ITERATED COLLOCATION APPROXIMATIONS

2.1. Exact iterated collocation

Suppose that the collocation approximation $u \in S_{m-1}^{(-1)}(Z_N)$ (defined by (1.1.6)) has been computed. The (exact) *iterated collocation approximation* u^I corresponding to u is then defined by

$$u^I(t) := g(t) + \int_0^t k(t,s,u(s))ds, \quad t \in I. \tag{2.1.1}$$

(Compare also Brunner[8] and the references cited there for some background and history of iterated approximations for (linear) Fredholm and Volterra integral equations.) It is readily verified that u^I has the following properties:

(i) $u^I \in C(I)$ (while u is, in general, not continuous on I);
(ii) $u^I(t) = u(t)$ for all $t \in X(N)$.

This second property implies, in particular, that if $c_m = 1$ (i.e., if $\overline{Z}_N$ is a subset of the set $X(N)$ of collocation points), then $u^I(t_n) = u(t_n)$ for all $t_n \in \overline{Z}_N$.

In the following we shall be interested in the values $u^I(t_n)$, $t_n \in \overline{Z}_N$, in the case when $c_m < 1$. Setting

$$\Phi_{n,i}[u_i] := \int_0^1 k(t_n, t_i + \tau h_i, u_i(t_i + \tau h_i))d\tau, \quad 0 \leqslant i \leqslant n-1 \leqslant N-1, \tag{2.1.2}$$

we may write (2.1.1) in the form

$$u^I(t_n) = g(t_n) + \sum_{i=0}^{n-1} h_i \Phi_{n,i}[u_i], \quad t_n \in \overline{Z}_N. \tag{2.1.3}$$

2.2. Discretized iterated collocation

The integrals (2.1.2) will, in general, have to be approximated by appropriate numerical quadrature; as in Section 1.2 we choose m-point interpolatory quadrature formulas of the form

$$\hat{\Phi}_{n,i}[u_i] := \sum_{l=1}^{m} w_l k(t_n, t_{i,l}, u_i(t_{i,l})), \tag{2.2.1}$$

with quadrature weights $\{w_l\}$ as in (1.2.1). Moreover, as indicated in Section 1.2, we shall usually not be able to compute u itself but only either $\tilde{u} \in S_{m-1}^{(-1)}(Z_N)$, or $\hat{u} \in S_{m-1}^{(-1)}(Z_N)$ (i.e., the solutions of the discretized collocation equations (1.2.3a), (1.2.3b)). Accordingly, the *discretization* of (2.1.3) will be given either by

$$\tilde{u}^I(t_n) := g(t_n) + \sum_{i=0}^{n-1} h_i \hat{\Phi}_{n,i}[\tilde{u}_i], \quad t_n \in \overline{Z}_N, \tag{2.2.2a}$$

with $\tilde{u} \in S_{m-1}^{(-1)}(Z_N)$ defined by (1.2.3a), or by

$$\hat{u}^I(t_n) := g(t_n) + \sum_{i=0}^{n-1} h_i \hat{\Phi}_{n,i}[\hat{u}_i], \quad t_n \in \overline{Z}_N, \tag{2.2.2b}$$

with $\hat{u} \in S_{m-1}^{(-1)}(Z_N)$ given by (1.2.3b).

In the convergence analysis for $y(t_n) - \tilde{u}^I(t_n)$ and $y(t_n) - \hat{u}^I(t_n)$ the following result will play an important rôle.

LEMMA 2.1. *Let the iterated approximations $\tilde{u}^I(t_n)$ and $\hat{u}^I(t_n)$ be given by (2.2.2a) and (2.2.2b), and assume that g and k in (1.1.1) satisfy the customary smoothness hypotheses. If the collocation parameters $\{c_j\}$ are the zeros of $P_m(2s-1)$ (Gauss points for (0,1)), then*

$$\max_{t_n \in \overline{Z}_N} |\tilde{u}^I(t_n) - \hat{u}^I(t_n)| = O(N^{-2m}) \quad (\text{as } N \to \infty,\ Nh \leqslant \gamma T). \tag{2.2.3}$$

PROOF Assuming that $k(t,s,y)$ has a bounded partial derivative $\partial k / \partial y$, with L denoting an upper bound for its absolute value, we obtain, using (2.2.2a) and (2.2.2b) together with (2.2.1),

$$\begin{aligned} |\tilde{u}^I(t_n) - \hat{u}^I(t_n)| &\leqslant \sum_{i=0}^{n-1} h_i |\hat{\Phi}_{n,i}[\tilde{u}_i] - \hat{\Phi}_{n,i}[\hat{u}_i]| \\ &\leqslant h \sum_{i=0}^{n-1} \sum_{l=1}^{m} |k(t_n, t_{i,l}, \tilde{u}_i(t_{i,l})) - k(t_n, t_{i,l}, \hat{u}_i(t_{i,l}))| \\ &\leqslant hL \sum_{i=0}^{n-1} \sum_{l=1}^{m} |\tilde{u}_i(t_{i,l}) - \hat{u}_i(t_{i,l})|, \quad t_n \in \overline{Z}_N. \end{aligned}$$

Here, we have used the fact that weights $\{w_l\}$ of the Gaussian quadrature formula are all positive and hence bounded by one (since $\sum_{l=1}^{m} w_l = 1$). An argument analogous to the one used in Brunner[8] (pp. 1138-1139), involving the Gauss quadrature errors and a discrete Gronwall inequality, yields, observing (1.2.3a) and (1.2.3b),

$$|\tilde{u}_i(t_{i,l}) - \hat{u}_i(t_{i,l})| \leqslant Ch^{2m}, \quad l = 1,...,m \ (i = 0,...,N-1),$$

with some constant C not depending on N. The estimate (2.2.3) now follows since $Nh \leqslant \gamma T$ for all $N \in \mathbb{N}$. □

THEOREM 2.1. *Let* $0 \leqslant c_1 < \cdots < c_m < 1$. *Assume that* $\tilde{u} \in S_{m-1}^{(-1)}(Z_N)$ *is the solution of the discretized collocation equation (1.2.3a), and let* $\tilde{z} \in S_m^{(-1)}(Z_N)$ *be determined by the discretized collocation equation (1.2.6a) (m-stage implicit Volterra-Runge-Kutta method of Pouzet type), where the collocation parameters are* $0 \leqslant c_1 < \cdots < c_m < c_{m+1} = 1$. *It then follows that*

$$\tilde{u}^I(t_n) = \tilde{z}(t_n), \quad t_n \in \bar{Z}_N; \tag{2.2.4}$$

i.e., the values of the discretized iterated collocation approximation $\tilde{u}^I$ *determined by (2.2.2a) are identical with the approximations* $\tilde{z}(t_n)$ *furnished by the m-stage implicit VRK method of Pouzet type (1.2.6a).*

PROOF Using

$$\tilde{u}_i(t_i + \tau h_i) = \sum_{l=1}^{m} L_l(\tau) \tilde{Y}_{i,l}, \quad t_i + \tau h_i \in \sigma_i,$$

with $\tilde{Y}_{i,l} := \tilde{u}_i(t_{i,l})$, we may write (1.2.3a) as

$$\begin{aligned} \tilde{Y}_{n,j} &= g(t_{n,j}) + \sum_{i=0}^{n} h_i \sum_{l=1}^{m} w_l k(t_{n,j}, t_{i,l}, \tilde{Y}_{i,l}), \\ & \qquad j = 1, \ldots m \ (n = 0, \ldots, N-1), \end{aligned} \tag{2.2.5a}$$

and (2.2.2a) becomes

$$\tilde{u}^I(t_n) = g(t_n) + \sum_{i=0}^{n-1} h_i \sum_{l=1}^{m} w_l k(t_n, t_{i,l}, \tilde{Y}_{i,l}), \quad t_n \in \bar{Z}_N. \tag{2.2.5b}$$

On the other hand, it follows from (1.2.7) that the first m equations of (1.2.6a) ($j = 1, \ldots, m$) assume the form

$$\begin{aligned} \tilde{Z}_{n,j} &= g(t_{n,j}) + \sum_{i=0}^{n} h_i \sum_{l=1}^{m} w_l k(t_{n,j}, t_{i,l}, \tilde{Z}_{i,l}), \\ & \qquad j = 1, \ldots, m \ (n = 0, \ldots, N-1). \end{aligned} \tag{2.2.6a}$$

This nonlinear system (2.2.6a) is identical with the nonlinear system (2.2.5a); hence, they possess, for all sufficiently small values of the mesh diameter h, the same solution: $\tilde{Y}_{n,j} = \tilde{Z}_{n,j}$, $j = 1, \ldots, m$ $(n = 0, \ldots, N-1)$.

For $j = m+1$ in (1.2.6a) we find, using $c_{m+1} = 1$ (and hence $t_{n,m+1} = t_{n+1}$),

$$\begin{aligned} \tilde{z}(t_{n+1}) &= \tilde{Z}_{n,m+1} = g(t_{n+1}) + \sum_{i=0}^{n} h_i \sum_{l=1}^{m} w_l k(t_{n+1}, t_{i,l}, \tilde{Z}_{i,l}) \\ & \qquad (n = 0, \ldots, N-1). \end{aligned} \tag{2.2.6b}$$

Since $\tilde{Z}_{i,l} = \tilde{Y}_{i,l}$ $(l = 1, \ldots, m;\ i = 0, \ldots, N-1)$, (2.2.6b) coincides with (2.2.5b) (with n replaced by $n+1$), thus verifying (2.2.4). □

COROLLARY 2.1. *Let g and k in (1.1.1) be subject to the smoothness hypotheses of Theorem 1.2, and let* $c_1, \ldots, c_m$ *be the zeros of* $P_m(2s-1)$. *If* $\tilde{u}^I(t_n)$ *is defined by (2.2.2a), with* $\tilde{u} \in S_{m-1}^{(-1)}(Z_N)$ *denoting the solution of the discretized collocation equation (1.2.3a), then we have*

$$\max_{t_n \in \bar{Z}_N} |y(t_n) - \tilde{u}^I(t_n)| = O(N^{-2m}) \tag{2.2.7}$$

(as $N \to \infty$, *with* $Nh \leqslant \gamma T$*).*

PROOF The above local superconvergence result follows readily from Theorem 1.2 and from (2.2.4). □

In most applications one will not work with the discretized collocation equation (1.2.3a) and the

corresponding discretized iterated collocation approximation (2.2.2a), since it may not be possible to extend the given kernel smoothly to the domain S'. Instead, the pair of discretized equations (1.2.3b), (2.2.2b) will be used: as (1.2.3b) employs only kernel values $k(t,s,.)$ with $(t,s)\in S$, (1.2.3b) and (2.2.2b) represent the natural discretizations of exact collocation and iterated collocation for (1.1.1). The following local superconvergence result holds.

THEOREM 2.2. *Let g and k in (1.1.1) be subject to the smoothness hypotheses of Theorem 1.2, and let $c_1,\ldots,c_m$ be the zeros of $P_m(2s-1)$. If $\hat{u}^I(t_n)$ is determined by (2.2.2b), where $\hat{u}\in S_{m-1}^{(-1)}(Z_N)$ is the solution of the discretized collocation equation (1.2.3b), then we have*

$$\max_{t_n\in\bar{Z}_N}|y(t_n)-\hat{u}^I(t_n)| = O(N^{-2m}) \tag{2.2.8}$$

(as $N\to\infty$, $Nh\leqslant\gamma T$).

PROOF The result of Theorem 2.2 can be established in complete analogy to (2.2.7) and (2.2.4), using (1.2.6b), (1.2.3b), and (2.2.2b).

An alternative way to verify (2.2.8) consists in writing

$$|y(t_n)-\hat{u}^I(t_n)| \leqslant |y(t_n)-\tilde{u}^I(t_n)| + |\tilde{u}^I(t_n)-\hat{u}^I(t_n)|, \quad t_n\in\bar{Z}_N;$$

Since the collocation parameters $c_1,\ldots,c_m$ are the Gauss points for (0,1), (2.2.8) follows by Corollary 2.1 and by Lemma 2.1. □

We note in passing that the 'exact' iterated collocation approximation u^I defined by (2.1.1), with $u\in S_{m-1}^{(-1)}(Z_N)$ being the solution of the exact collocation equation (1.1.6), satisfies also

$$\max_{t_n\in\bar{Z}_N}|y(t_n)-u^I(t_n)| = O(N^{-2m}) \tag{2.2.9}$$

(as $N\to\infty$, $Nh\leqslant\gamma T$).
This is an immediate consequence of

$$|y(t_n)-u^I(t_n)| \leqslant |y(t_n)-\hat{u}^I(t_n)| + |\hat{u}^I(t_n)-u^I(t_n)|$$

and of the form of the Gauss quadrature errors determining the order of $\hat{u}^I(t_n)-u^I(t_n)$. Hence, (2.2.9) and (2.2.8) generalize the results of Brunner[8] to nonlinear Volterra integral equations.

As an illustration for the above local superconvergence results, we mention the case of *linear* polynomial spline collocation ($m=2$): if $\hat{u}\in S_1^{(-1)}(Z_N)$ is determined by (1.2.3b), with $c_1=(3-\sqrt{3})/6$, $c_2=(3+\sqrt{3})/6$, then the corresponding values $\hat{u}^I(t_n)$ given by (2.2.2b) are convergent of *order* $\hat{p}=4$: $\max_{t_n\in\bar{Z}_N}|y(t_n)-\hat{u}^I(t_n)| = O(N^{-4})$.

2.3. Polynomial solutions to (1.1.1)

Suppose that the exact solution to (1.1.1) is a polynomial, $y\in\pi_{m-1}$. It is then clear that the (exact) collocation approximation $u\in S_{m-1}^{(-1)}(Z_N)$ defined by (1.1.6) satisfies $u(t)=y(t)$ for all $t\in I$, since π_{m-1} is a subspace of $S_{m-1}^{(-1)}(Z_N)$. This does no longer hold, however, for the collocation approximation $\hat{u}\in S_{m-1}^{(-1)}(Z_N)$ determined by the *discretized* collocation equation (1.2.3b). To see this, note first that

$$\begin{aligned} y(t)-\hat{u}(t) &= (y(t)-u(t)) + (u(t)-\hat{u}(t)) \\ &= u(t)-\hat{u}(t) = \sum_{l=1}^{m} L_l(\tau)(Y_{n,l}-\hat{Y}_{n,l}), \quad t=t_n+\tau h_n\in\sigma_n. \end{aligned}$$

Let

$$E_{n,i}^{(j)}[u_i] := \Phi_{n,i}^{(j)}[u_i] - \hat{\Phi}_{n,i}^{(j)}[u_i] \quad (i\leqslant n)$$

denote the quadrature errors induced by the approximations (1.2.1), (1.2.2b) to the integrals (1.1.5). It follows from (1.1.6) and (1.2.3b), invoking a standard argument based on a discrete Gronwall inequality, that the order of the differences $Y_{n,l}-\hat{Y}_{n,l}$ is given by the order of the quadrature errors $E_{n,i}^{(j)}[u_i]$. According to the above we thus obtain, e.g.,

$$\max_{t_n\in\bar{Z}_N}|y(t_n)-\hat{u}(t_n)| = \begin{cases} O(N^{-2m}), & \text{if } \{c_j\} \text{ are the Gauss points;} \\ O(N^{-(2m-1)}), & \text{if } \{c_j\} \text{ are the Radau II points;} \\ O(N^{-(2m-2)}), & \text{if } \{c_j\} \text{ are the Lobatto points.} \end{cases}$$

Note in particular that if the collocation parameters $\{c_j\}$ are the zeros of $P_m(2s-1)$ (Gauss points), then the errors $y(t_n)-\hat{u}(t_n)$ and $y(t_n)-\hat{u}^I(t_n)$ (cf. (2.2.8)) exhibit the same order of local superconvergence:

$$\max_{t_n\in\bar{Z}_N}|y(t_n)-\hat{u}(t_n)| = O(N^{-2m}), \quad \max_{t_n\in\bar{Z}_N}|y(t_n)-\hat{u}^I(t_n)| = O(N^{-2m})$$

(as $N\to\infty$, $Nh\leqslant\gamma T$).
This result will have certain implications in connection with global error estimation based on collocation at the Gauss points and corresponding iterated collocation (see Section 3.2.1).

3. Variable stepsize method

A one-step method with variable stepsize can be separated into three major components:
- a basic fixed-step method
- an error estimator
- a stepsize strategy

In this chapter these three will be discussed.

3.1. Fixed-step method

The underlying fixed-step method will be a collocation method as described in the previous chapters. The integrals will be approximated by interpolatory quadrature formulas. For the sake of convenience we give a summary of the formulas used to approximate the solution at the point t_{n+1} (cf. 1.2.3b):

$$\begin{aligned} \hat{Y}_{n,j} &= g(t_{n,j}) + \hat{F}_n(t_{n,j}) + \\ &\quad h_n c_j \sum_{l=1}^{s} w_l k(t_{n,j}, t_n + c_j c_l h_n, \sum_{\kappa=1}^{m} L_\kappa(c_j c_l)\hat{Y}_{n,\kappa}) \\ \hat{F}_n(t) &:= \sum_{i=0}^{n-1} h_i \sum_{l=1}^{s} w_l k(t, t_{i,l}, \hat{Y}_{i,l}) \\ &\quad j=1,\ldots,m;\ s=m \text{ or } m-1. \end{aligned} \tag{3.1.1}$$

If $c_m=1$, the approximate solution in t_{n+1}, $\hat{u}_{n+1}$ equals $\hat{Y}_{n,m}$. Otherwise, $\hat{u}_{n+1}$ is either computed by Lagrange interpolation:

$$\hat{u}_{n+1} = \sum_{\kappa=1}^{m} L_\kappa(1)\ \hat{Y}_{n,\kappa}, \tag{3.1.2}$$

which is the discrete analogue of (1.1.7), or by iterated collocation (cf. 2.2.2b):

$$\hat{u}^I_{n+1} = g(t_n+h_n)+\hat{F}_n(t_n+h_n)+h_n\sum_{l=1}^{s} w_l k(t_n+h_n, t_{n,l}, \hat{Y}_{n,l}). \tag{3.1.3}$$

The free collocation parameters can be used to get a method that combines high accuracy with a minimum amount of work involved. The order p of the method is assumed to give an indication of the accuracy of the solution. The amount of computational work involved is normally expressed by

the number of kernel evaluations needed. For various choices of collocation parameters these values are (see Theorem 1.1, Theorem 1.2 and Theorem 2.2):

(a) Gauss-Legendre points $c_1, \ldots, c_m$; $s = m$
$p = m$
$n_{kev} = \frac{N(N-1)}{2}m^2 + Nn_I m^2$

(b) Gauss-Legendre points $c_1, \ldots, c_{m-1}, c_m = 1$; $s = m-1$
$p = 2m-2$
$n_{kev} = \frac{N(N-1)}{2}(m^2-m) + Nn_I(m^2-m)$

(c) Gauss-Legendre points $c_1, \ldots, c_m$ + Iterated collocation; $s = m$
$p = 2m$
$n_{kev} = \frac{N(N-1)}{2}(m^2+m) + Nm + Nn_I m^2$

(d) Lobatto points $c_1, \ldots, c_m$; $s = m$
$p = 2m-2$
$n_{kev} = \frac{N(N-1)}{2}(m-1)^2 + 2N(m-1) + Nn_I(m-1)^2$

(e) Radau II points $c_1, \ldots, c_m$; $s = m$
$p = 2m-1$
$n_{kev} = \frac{N(N-1)}{2}m^2 + Nn_I m^2$

Here, N denotes the number of steps taken and n_I the number of iterations needed to solve the (non)-linear system for $\hat{Y}_{n,j}$; $n_I = 1$ if the integral equation has a linear kernel, and n_I is bounded by a constant for nonlinear kernels.

Notice the low number of kernel evaluations in the case of Lobatto points as compared to the amount needed when using (the same number of) Radau II points. This decrease in the number of kernel evaluations is caused by the fact that one can make use of the coincidence of the points $t_{n-1}+c_m h_{n-1}$ and $t_n + c_1 h_n$ and reformulate (3.1.1) into :

$$\begin{aligned}
\hat{Y}_{n,1} &= \hat{u}_n \\
\hat{Y}_{n,j} &= g(t_{n,j}) + \hat{F}_n(t_{n,j}) + h_n c_j w_1 k(t_{n,j}, t_n, \hat{u}_n) + \\
&\quad h_n c_j \sum_{l=2}^{m} w_l k(t_{n,j}, t_n + c_j c_l h_n, \sum_{\kappa=1}^{m} L_\kappa(c_j c_l)\hat{Y}_{n,\kappa}), \\
&\quad\quad j = 2, \ldots, m, \qquad (3.1.1')
\end{aligned}$$

$$\begin{aligned}
\hat{F}_n(t) &:= h_0 w_1 k(t, t_0, y(t_0)) + \\
&\quad \sum_{i=0}^{n-2} \left[h_i \sum_{l=2}^{m-1} w_l k(t, t_{i,l}, \hat{Y}_{i,l}) + (h_i + h_{i+1}) w_m k(t, t_i + h_i, \hat{Y}_{i,m}) \right] + \\
&\quad h_{n-1} \sum_{l=2}^{m} w_l k(t, t_{n-1,l}, \hat{Y}_{n-1,l}).
\end{aligned}$$

From the above, it seems that the method with the best accuracy / computational work ratio will be obtained by using the Lobatto points.

3.2. Error estimation

To estimate the error in the approximate solution $\hat{u}_{n+1}$ in the point t_{n+1} one has the choice between global and local error estimation. Both will be treated below.

3.2.1. Global error estimation.

The global error in the point t_{n+1} is defined by $|y(t_{n+1})-\hat{u}_{n+1}|$. It is normally approximated by the difference between $\hat{u}_{n+1}$ and a reference solution that is assumed to have a higher accuracy. We will compute the reference solution with a higher order method using the same stepsize.

Here, two combinations of methods are preferable with respect to reliability and efficiency: Gauss-Legendre for the computation of $\hat{u}_{n+1}$ combined with iterated collocation, using the same collocation points, for the reference solution, and Lobatto combined with a Lobatto method of the next higher order.

If the error in $\hat{u}_{n+1}$ is $O(N^{-2\kappa})$, then the number of kernel evaluations for these two variants is approximately:

$$\text{Gauss}_{2\kappa} + \text{iterated collocation}: \quad (2\kappa^2+\kappa)\,N^2$$

$$\text{Lobatto}_{\kappa+1} + \text{Lobatto}_{\kappa+2}: \quad \frac{2\kappa^2+2\kappa+1}{2}\,N^2$$

For each method the index denotes the number of collocation points needed in order to obtain the desired order.

Note, that Gauss+($c_m=1$), respectively Radau II combinations, need roughly the same number of kernel evaluations as the Lobatto method, and can be expected to be equally robust.

Although the Gauss method seems obviously less efficient, it has some additional advantages. Its implementation is straightforward and it needs considerably less overhead. Besides, the order of accuracy not only of the reference solution but also of the quadrature part of the approximation doubles the order of $\hat{u}(t)$ in the steppoints, which makes it more robust for a variable stepsize method (see also Section 3.3). However, as mentioned in Section 2.3, it has one drawback. If the solution of the integral equation is a polynomial of degree $<m$, $\hat{u}(t)$ has an error of $O(N^{-2m})$ not only in the collocation points but also in the steppoints. Since the reference solution is equally accurate this results in a very unreliable error estimate. It is however possible to detect this failure automatically and switch to another method whenever this undesired superconvergence in the steppoints occurs (see Section 4.2).

3.2.2. Local error estimation.

In analogy to what is common practice in the theory of initial-value problems for ordinary differential equations, we define the local error as the error resulting from a single step of the approximating method, not taking into account the errors inherited from previous steps. So, the local error in the point t_{n+1} is defined by:

$$LE_{n+1} := \left| \int_{t_n}^{t_{n+1}} k(t_{n+1},s,y_n(s))ds \; - \; h_n \sum_{l=1}^{m} w_l k(t_{n+1},t_{n,l},\hat{Y}_{n,l}) \right|, \tag{3.2.2.1}$$

with

$$y_n(t) := g(t) + \hat{F}_n(t) + \int_{t_n}^{t} k(t,s,y_n(s))ds \tag{3.2.2.2}$$

(cf. Arndt[1]).

An estimation of the local error is obtained by approximating the integral by a higher-order method using the same stepsize; i.e.

$$\int_{t_n}^{t_{n+1}} k(t_{n+1},s,y_n(s))ds \approx h_n \sum_{l=1}^{\tilde{m}} \tilde{w}_l k(t_{n+1},t_n+\tilde{c}_l h_n,\tilde{Y}_{n,l}), \tag{3.2.2.3}$$

where

$$\tilde{Y}_{n,j} = g(t_n+\tilde{c}_j h_n) + \hat{F}_n(t_n+\tilde{c}_j h_n) + h_n\tilde{c}_j \sum_{l=1}^{\tilde{m}} \tilde{w}_l k(t_n+\tilde{c}_j h_n, t_n+\tilde{c}_j\tilde{c}_l h_n, \sum_{\kappa=1}^{\tilde{m}} \tilde{L}_\kappa(\tilde{c}_j\tilde{c}_l)\tilde{Y}_{n,\kappa}). \tag{3.2.2.4}$$

Note that it is assumed that $\tilde{c}_{\tilde{m}}=1$.

3.3. Stepsize strategy

Having an estimation of the error, we will now discuss how to determine the stepsize to limit this error to a user specified tolerance.

The stepsize strategy consists of advancing the solution with a trial stepsize, then accept or reject the result and repeating the process with as trial stepsize a modification of the previous stepsize. This modification should account for the (lack of) accuracy in the accepted or rejected trial solution. It should result in as large a stepsize as possible while still providing an accuracy that satisfies the specified tolerance.

The control of the global error in a steppoint, say t_{n+1}, is a straightforward process if a global error estimation is available. A trial solution will then be accepted if the global error estimation is less than the specified error tolerance, i.e.

$$GEE_{n+1} \leqslant TOL. \tag{3.3.1}$$

The modification of the stepsize reads:

$$h_{new} = h_n \left[\frac{TOL}{GEE_{n+1}}\right]^{\frac{1}{p}}. \tag{3.3.2}$$

Here, p is the order of the underlying method, and h_{new} stands for h_{n+1} if the trial solution has been accepted, and otherwise for h_n.

Note that, once a certain point t_{n+1} has been reached, we cannot control the error in the lag term part any more, since the approximating solution for that part of the interval has already been accepted. This means that it is defensible to control only the local error and assume that the global error in the endpoint will be more or less equal to the sum of the local errors.
In this case one accepts a trial solution only if

$$LEE_{n+1} \leqslant \frac{TOL.h_n}{t_{n+1}}, \tag{3.3.3}$$

and modifies the stepsize by:

$$h_{new} = h_n \left[\frac{TOL.h_n}{t_{n+1}.LEE_{n+1}}\right]^{\frac{1}{q}}, \tag{3.3.4}$$

where q is the order of the quadrature formula to approximate the integral.

So far, the stepsize strategy is completely analogous to the strategy used in variable stepsize methods for ordinary differential equations. However, the intention is to limit the global error over the whole interval to the user-defined tolerance, i.e. to have

$$|y(t) - \hat{u}(t)| \leqslant TOL, \qquad \text{for } 0 \leqslant t \leqslant T.$$

The disadvantage of proceeding in the same way as in the case of ODEs is, as Arndt ([1] ,[2]) points out, that the consequence of accepting the approximated solution to future quadrature computations is not taken into account. This means that the assumption that the sum of the local errors equals the global error in the endpoint is not necessarily valid.

To obviate the problem of the possibly large errors in future lag term computations, once t_{n+1} has

been reached, one has to demand that:

$$|y(t) - \hat{u}(t)| \leqslant TOL, \qquad \text{for all } t \in [t_{n+1}, T].$$

This reads in discretized form:

$$\left| \sum_{i=0}^{l-1} \left[\int_{t_i}^{t_{i+1}} k(t_l, s, y(s))ds - h_i \sum_{j=1}^{m} w_j k(t_l, t_{i,j}, \hat{Y}_{i,j}) \right] \right| \leqslant TOL, \tag{3.3.5}$$

$$\text{for } t_l = T, T-H, \ldots, t_{n+1}; \quad \text{with } H \text{ some fixed number.}$$

Let

$$EE_{i+1}^{l} \approx \int_{t_i}^{t_{i+1}} k(t_l, s, y(s))ds - h_i \sum_{j=1}^{m} w_j k(t_l, t_{i,j}, \hat{Y}_{i,j}). \tag{3.3.6}$$

Then (3.3.5) is satisfied when:

$$\sum_{i=n}^{l-1} |EE_{i+1}^{l}| + \left| \sum_{i=0}^{n-1} EE_{i+1}^{l} \right| \leqslant TOL.$$

In this way one gets, as Arndt suggests, a modified stepsize strategy that strives for a uniform control of the error, .i.e.:

Let

$$EE_{n+1} = \max_{t_{n+1} \leqslant t_l \leqslant T} \left[(t_l - t_n) \cdot \frac{|EE_{n+1}^{l}|}{h_n} + \left| \sum_{i=0}^{n-1} EE_{i+1}^{l} \right| \right]. \tag{3.3.7}$$

Accept the trial solution in t_{n+1} if

$$EE_{n+1} \leqslant TOL. \tag{3.3.8}$$

The modification of the stepsize is:

$$h_{new} = h_n \left[\frac{TOL}{EE_{n+1}} \right]^{\frac{1}{q}}, \tag{3.3.9}$$

with q the order of the quadrature formula.

Note that although global error control does not lead to an erroneous error estimate in the endpoint, it is certainly possible that by allowing too large a stepsize at the start, it will be impossible to meet the tolerance as the integration advances, due to an unacceptably large error in the lag term. Therefore, to avoid problems later on, it seems advisable to use the global error control without the uniform error control only if $p \ll q$, which is the case for the Gauss collocation method combined with iterated collocation to estimate the global error.

1. Implementation

For the miscellaneous programming matters and heuristics, e.g. stepsize constraints, the solution of nonlinear systems, etc., we refer to Blom and Brunner[4] . Here, we consider two more general topics.

1.1. Approximation of the lag term

The lion's share of the computational work in all methods is the approximation of the lag term by some quadrature rule. Moreover, for all methods except the iterated collocation method the additional storage space needed for all $\hat{Y}_{n,j}$ is substantial.

However, the number of kernel evaluations needed to compute the reference solution can be significantly reduced by approximating the lag term by Lagrangian interpolation instead of using a

quadrature formula. Interpolation over m points yields:

$$\hat{F}_n^{(1)}(t_n+\tau h_n) = \sum_{\kappa=1}^{m} \hat{F}_n(t_{n,\kappa}) \tag{4.1.1}$$

and adds an additional interpolation error of $O(h_n^m)$, unless the kernel satisfies $k \in \pi_{m-2}$. So, in general, this interpolation gives rise to a reference solution of $O(h_n^m)$, which is not adequate.
However, one can also interpolate over $2m$ points:

$$\hat{F}_n^{(2)}(t_n+\tau h_n) = \sum_{\kappa=1}^{\bar{m}} \bar{L}_\kappa\left(\frac{h_{n-1}+\tau h_n}{h_{n-1}+h_n}\right)\hat{F}_n(t_{n-1}+\bar{c}_\kappa(h_{n-1}+h_n)), \tag{4.1.2}$$

where

$$\bar{m} = 2m$$

and

$$\bar{c}_j = c_j \frac{h_{n-1}}{h_{n-1}+h_n}, \qquad j=1,\ldots,m,$$

$$\bar{c}_j = \frac{h_{n-1}+c_l h_n}{h_{n-1}+h_n}, \qquad j=\bar{m}-m+1,\ldots,\bar{m};\ \ l=j+m-\bar{m},$$

and where the $\bar{L}_\kappa$ denote the Lagrange canonical polynomials corresponding to the $\bar{c}_j$. This interpolation introduces an error of $O((h_{n-1}+h_n)^{\bar{m}})$, unless $k \in \pi_{\bar{m}-2}$, which is small enough to preserve the legitimacy of the reference solution in case the approximating method is based on Gauss collocation.

For all other methods that are mentioned in Section 3.1 this kind of approximation of the lag term cannot be used when computing the global error estimation, since the error in the $\hat{F}_n$ will always be a spoiling factor, no matter how accurate the interpolation will be. However, since local error estimation needs only the values of $\hat{F}_n$ in some intermediate points, one can use the interpolation formula (4.1.2) which is sufficiently accurate for all methods described.

Note that for Lobatto collocation the value of $\bar{m}$ is $2m-1$ instead of $2m$, due to the coincidence of $t_{n-1}+c_m h_{n-1}$ and $t_n+c_1 h_n$.

4.2. Detection of a polynomial solution when using Gauss collocation

As mentioned in Section 2.3 the Gauss method yields an error of $O(N^{-2m})$ in the steppoints if the solution is an element of π_{m-1}. If the reference solution is equally or less accurate, then this results in an unreliable global error estimate. One can detect this error by computing the approximations $\hat{\bar{Y}}_{n,j}$ to $y(t_n+c_j h_n/2)$ and use Lagrangian interpolation to compute the solution at some point of the interval $[t_n, t_n+h_n/2]$, say t_n, both with these approximations and with the $\hat{Y}_{n,j}$. If $y \notin \pi_{m-1}$, then $e_1 = |y(t_n)-\hat{u}_n| \approx O(h_n^m)$ and $e_2 = |y(t_n)-\hat{\bar{u}}_n| \approx O((h_n/2)^m)$. So, if $e_1/e_2 \approx 2^m$, the order of the Gauss method is as expected.

To avoid the computation of the quadrature of the lag term, we use Lagrange interpolation over the points 0.0, c_j, 1.0 . No additional kernel evaluations are needed, since the $\hat{F}_n(t_n)$ and the $\hat{F}_n(t_{n+1})$ already have been calculated during the computation of the reference solution in t_n, respectively t_{n+1}. Now, if $e_1/e_2 \ll 2^m$, then it is almost certain that $y \in \pi_{m-1}$.

Note that if both $y \in \pi_{m-1}$ and $\int_0^{t_n} k(t,s,y(s))ds \in \pi_{m+1}$, then it is not clear what the value of the ratio e_1/e_2 will be. But in this case the approximation of both the lag term and the solution is 'exact', which means that the only problem that can arise is an unjustifiable rejection of the current combination of methods for approximating the solution and the reference solution.

5. NUMERICAL PERFORMANCE

The ideas presented in the preceding sections have resulted in a FORTRAN code named COLVI2; compare Blom and Brunner[4] , where a description of this code and its usage is given.

To show some properties of the code, we tested it on the six problems that were used by Schlichte[19] to compare his method, IVRKX, with the codes ORION of Bader and Kunkel[3] , VOLTEX of Hock[14] , and VOLCON of Hairer, Lubich and Schlichte[13] . The performance data of these codes for the six problems are listed in the Appendix. We also added the results of VE1 of Bownds and Appelbaum[6] ,[5] and of INTSOL of Jones and McKee[16] , in so far as we could find these results in the literature.

The tests were performed on a CDC CYBER-750, in single precision (a machine precision of approximately 14 digits).

The problems are:

1 (problem 4 from Schlichte)

$$y(t) = \tfrac{1}{2}t^2e^{-t} + \int_0^t \frac{(t-s)^2}{2}e^{s-t}.y(s)\,ds, \qquad \text{for } t\in[0,5]$$

with solution:

$$y(t) = \tfrac{1}{3}\left[1 - e^{-\frac{3}{2}t}[\cos(\tfrac{1}{2}t\sqrt{3}) + \sqrt{3}\sin(\tfrac{1}{2}t\sqrt{3})]\right]$$

2 (problem 15 from Schlichte)

$$y(t) = 1 + (\sin(t))^2 - \int_0^t 3\sin(t-s).y^2(s)\,ds, \qquad \text{for } t\in[0,5]$$

with solution:

$$y(t) = \cos(t)$$

3 (problem 19 from Schlichte)

$$y(t) = \cos(t) - \int_0^t \frac{2}{(t-s+2)^2}.(y(s)+y^3(s))\,ds, \qquad \text{for } t\in[0,40]$$

with solution:

$$y(40) \approx -0.65013110133344$$

4 (problem 20 from Schlichte)

$$y(t) = 1 + \int_0^t (t-s)^3(4-t+s)e^{s-t}.\frac{y^4(s)}{1+2y^2(s)+2y^4(s)}\,ds, \qquad \text{for } t\in[0,10]$$

with solution:

$$y(10) = 1.2599558233723$$

5 (problem 16 from Schlichte)

$$y(t) = e^{-t} + \int_0^t e^{s-t}[y(s)+e^{-y(s)}]\,ds, \qquad \text{for } t\in[0,40]$$

with solution:

$$y(t) = \ln(t+e)$$

6 (problem 21 from Schlichte)

$$y(t) = t - 1 + (1+t^2)e^{-t^2} + \int_0^t t^2 e^{-ts}.y(s)\,ds, \qquad \text{for } t \in [0,5]$$

with solution:

$$y(t) = t$$

The following combinations of methods for the approximation of the solution, respectively the reference solution, have been used:

Gm + IG	m points Gauss + iterated collocation
1Gm + 1G(m+1)	$m-1$ Gauss points + ($c_m = 1$), together with m Gauss points + ($c_{m+1} = 1$)
Lm + L(m+1)	m, combined with $m+1$ points Lobatto collocation
Rm + R(m+1)	m, together with $m+1$ points Radau collocation

As stepsize strategies we used global error estimate / global error control (G), and, if applicable, lōcal error estimate / local error control (L). In the latter the lag term needed for the reference solution has been approximated by interpolation (cf. 4.1.2).

The meaning of the headers in the tables is:

sd	number of significant digits : $-\frac{^{10}\log\lvert y(T)-\hat{u}(T)\rvert}{\max(1.0,\lvert y(T)\rvert)}$
EE/GE	ratio between the global error estimate in T and the real global error Note: if the local error is uniformly controlled, the global error in T is estimated by: $\sum_{i=0}^{N-1} EE'_{i+1}$, with $t_l = T$ (cf. 3.3.6)
N	number of intervals used
#F tot	number of failed steps in total
#F 0	number of failed steps at the start in $t = 0.0$
#kev	total number of kernel evaluations needed.

Table 5.1 gives the results that are obtained when no special care has been taken. E.g. for the (iterated) Gauss method, no testing on polynomial solutions has been done, and for the other methods, the error is not uniformly controlled.

For all experiments the initial stepsize was 1.0; the maximal stepsize allowed was 5.0, which was actually used in problem 5.

Table 5.1 Results for method combinations, without special features

Prob. 1		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG	5.2	1	8	2	2	760	9.3	1	33	5	3	12100
G	1G4+1G5	4.6	1	4	0	0	320	7.1	1	12	2	1	2592
G	L4+L5	4.5	1	4	0	0	278	7.1	1	12	2	1	2123
G	R3+R4	4.5	1	5	1	0	425	---	-	--	-	-	----
G	G8+IG	7.4	1	3	0	0	432	9.8	1	4	0	0	720
G	1G6+1G7	8.2	1	3	0	0	432	8.2	1	3	0	0	432
G	L6+L7	8.5	1	3	0	0	399	8.1	1	4	0	0	654
G	R5+R6	6.8	1	3	0	0	366	8.0	1	4	0	0	610
L	1G4+1G5	4.6		4	0	0	276	7.1		11	2	1	1311
L	L4+L5	4.5		4	0	0	282	7.1		11	2	1	1213
L	R3+R4	5.6		6	2	0	472	4.9		20	2	0	2612
L	1G6+1G7	8.2		3	0	0	429	8.2		3	0	0	429
L	L6+L7	8.5		3	0	0	438	8.1		4	0	0	634
L	R5+R6	6.8		3	0	0	366	8.0		4	0	0	538
Prob. 2		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG	6.0	5E-1	9	3	1	1596	8.4	1	43	5	2	21988
G	1G4+1G5	---	-	--	-	-	----	---	-	--	-	-	----
G	L4+L5	---	-	--	-	-	----	---	-	--	-	-	----
G	R3+R4	---	-	--	-	-	----	---	-	--	-	-	----
G	G8+IG	6.2	1	3	0	0	1136	8.4	1E-1	4	1	0	2152
G	1G6+1G7	5.2	1E-1	3	0	0	1224	---	-	--	-	-	----
G	L6+L7	4.1	1	3	0	0	1070	---	-	--	-	-	----
G	R5+R6	---	-	--	-	-	----	---	-	--	-	-	----
L	1G4+1G5	1.5		7	1	0	1396	4.9		22	5	1	6429
L	L4+L5	1.3		8	2	0	1535	4.6		23	4	1	5425
L	R3+R4	2.8		12	4	1	2251	5.6		45	8	2	14730
L	1G6+1G7	5.2		3	0	0	1221	6.3		5	1	0	2490
L	L6+L7	4.1		3	0	0	1109	5.9		5	1	0	2246
L	R5+R6	2.9		4	1	0	1672	5.4		7	1	1	2740

Table 5.1 (cont.) Results for method combinations, without special features

Prob. 3		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG	4.2	1	70	31	2	75696	8.7	1	366	72	4	1615392
G	1G4+1G5	5.6	2	34	14	0	31552	7.9	1	98	41	2	231828
G	L4+L5	4.7	1	43	20	0	38350	8.6	1	104	50	2	214553
G	R3+R4	4.5	1	44	18	0	39759	7.3	1	175	52	2	513900
G	G8+IG	6.2	1	27	13	0	48824	7.1	1	54	19	2	154616
G	1G6+1G7	5.3	1	15	2	0	14832	7.2	1	33	9	0	61242
G	L6+L7	5.3	1	17	1	0	13679	7.9	1	39	12	1	73395
G	R5+R6	5.0	1	28	7	0	37235	7.8	1	45	10	1	88511
L	1G4+1G5	6.1		54	21	0	34093	9.5		176	45	2	262811
L	L4+L5	7.2		57	13	0	26104	9.9		190	47	2	226241
L	R3+R4	6.3		90	30	0	58567	9.2		358	68	2	714682
L	1G6+1G7	7.3		26	5	0	21167	9.2		50	18	0	72508
L	L6+L7	6.8		28	5	0	21113	9.3		50	11	1	55681
L	R5+R6	6.8		28	3	0	18805	9.8		61	24	1	87408
Prob. 4		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG	4.5	1	11	5	1	2224	7.7	1	50	9	3	29588
G	1G4+1G5	4.9	1	10	0	0	2368	7.5	1	32	3	2	19168
G	L4+L5	4.8	1	10	0	0	1920	7.4	1	33	3	2	16102
G	R3+R4	5.2	1	15	2	1	3841	7.3	1	60	2	2	48541
G	G8+IG	7.5	1	4	0	0	1360	8.8	1	7	1	1	3240
G	1G6+1G7	5.7	1	5	0	0	1872	7.5	1	7	0	0	3240
G	L6+L7	5.6	1	5	0	0	1641	7.1	1	7	0	0	2761
G	R5+R6	6.0	1	5	0	0	1525	8.0	1	9	0	0	3843
L	1G4+1G5	6.1		12	1	0	2255	8.1		42	3	2	15303
L	L4+L5	6.3		13	1	0	2168	8.1		44	3	2	13338
L	R3+R4	4.7		21	2	1	3749	7.6		90	3	2	42831
L	1G6+1G7	5.7		5	0	0	1657	8.3		8	0	0	2896
L	L6+L7	5.6		5	0	0	1526	8.4		8	1	0	3001
L	R5+R6	5.5		6	0	0	1689	7.8		10	1	0	3709

Table 5.1 (cont.) Results for method combinations, without special features

Prob. 5		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG	3.2	-5E-3	11	0	0	1672	5.8	-6E-3	39	2	2	16984
G	1G4+1G5	4.3	1	29	0	0	15776	7.3	1	93	1	1	146048
G	L4+L5	4.3	1	30	0	0	13335	7.3	1	98	1	1	127043
G	R3+R4	4.2	1	51	1	0	35325	7.3	1	227	3	2	658650
G	G8+IG	9.2	5E-5	10	0	0	5432	9.3	4E-4	11	0	0	6672
G	1G6+1G7	5.0	1	10	0	0	5616	7.5	1	17	0	0	13968
G	L6+L7	4.9	1	10	0	0	4868	7.5	1	18	0	0	13252
G	R5+R6	4.3	1	11	0	0	5551	7.5	1	24	0	0	21594
L	1G4+1G5	4.2		27	0	0	7349	7.3		92	1	1	61043
L	L4+L5	4.6		33	1	0	8367	7.4		99	1	1	54443
L	R3+R4	4.1		47	0	0	13581	7.2		208	2	2	212869
L	1G6+1G7	5.3		11	0	0	4693	7.0		15	3	0	8856
L	L6+L7	5.5		11	1	0	4538	8.0		19	0	0	9258
L	R5+R6	4.9		12	1	0	4791	7.0		20	1	0	9826
Prob. 6		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG	3.9	-7E-3	3	0	0	120	3.9	-3E-5	5	2	0	400
G	1G4+1G5	---	-	--	-	-	----	---	-	--	-	-	----
G	L4+L5	---	-	--	-	-	----	---	-	--	-	-	----
G	R3+R4	---	-	--	-	-	----	---	-	--	-	-	----
G	G8+IG	10.4	-4E-4	3	0	0	432	10.4	-4E-4	3	0	0	432
G	1G6+1G7	5.5	1	3	0	0	432	---	-	--	-	-	----
G	L6+L7	5.4	1	3	0	0	399	---	-	--	-	-	----
G	R5+R6	4.7	1	3	0	0	366	---	-	--	-	-	----
L	1G4+1G5	2.5		4	1	0	327	3.8		9	3	1	1044
L	L4+L5	2.4		4	1	0	332	3.8		9	3	1	987
L	R3+R4	1.9		5	1	0	336	3.3		14	5	1	1679
L	1G6+1G7	5.5		3	0	0	429	5.5		4	1	0	745
L	L6+L7	5.4		3	0	0	438	5.5		4	1	0	744
L	R5+R6	4.7		3	0	0	366	4.7		4	1	0	635

TABLE 5.1. Results for method combinations, without special features

All failures in Table 5.1 were due to a rejection of a step with the minimum allowed stepsize, which was 5 E-3.

As was expected, problem 6, with solution $y = t$, resulted in a bad performance for the Iterated Gauss method; but problem 5, with solution $y = \ln(t+e)$, also seemed to suffer from a bad error estimate. Both problems were rerun with the polynomial test on and with an automatic escape to a local error estimate and uniform local error control Gauss_m, $\text{Gauss}_{m+1}+(c_{m+2}=1)$ combination. The escape was made after two consecutive occurrences of polynomial behaviour, and the last three steps were discarded. As can be seen in Table 5.2, in most cases an escape was needed.

Prob. 5		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG		escape	at	t	=	2.3		escape	at	t	=	3.2
L	G4+1G6	4.5	6E-2	14	6	0	8604	6.2	1E-3	33	9	2	21541
G	G8+IG		escape	at	t	=	21.0		escape	at	t	=	2.7
L	G8+1G10	9.2	-1E-1	10	3	0	11537	9.3	-3E-2	11	2	0	16360
Prob. 6		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	G4+IG		escape	at	t	=	2.8		escape	at	t	=	1.8
L	G4+1G6	7.3	1	7	4	2	1417	9.6	1	11	5	2	2436
G	G8+IG	10.4	-4E-4	3	0	0	624	10.4	-4E-4	3	0	0	624
L	G8+1G10												

TABLE 5.2. Results for Iterated Gauss; esc. to Gauss_m + $\text{Gauss}_{m+1}+(c_{m+2}=1)$, local error control

In Table 5.3 the results are listed that were obtained by using a uniform error control, or rather an error control in discrete points spaced 1.0. Here, if a step was rejected which already had a minimum size, the tolerance was relaxed by a factor 4 and the computation was resumed.
Recall that when using iterated Gauss collocation to compute the reference solution there is no need to employ a uniform error control (see Section 3.3).

Table 5.3		Results with uniform error control and tolerance relaxing											
Prob. 1		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	1G4+1G5	4.8	1	4	0	0	390	7.2	1	10	1	1	2016
G	L4+L5	4.6	1	4	0	0	368	7.2	1	11	1	1	2056
G	R3+R4	4.6	1	5	1	0	548	7.2	1	18	2	2	4752
G	1G6+1G7	8.2	1	3	0	0	531	9.0	1	4	0	0	830
G	L6+L7	8.5	1	3	0	0	516	8.9	1	4	0	0	784
G	R5+R6	6.8	1	3	0	0	465	8.3	1	4	0	0	731
L	1G4+1G5	4.8	5E-1	4	0	0	346	7.3	3E-1	11	1	1	1498
L	L4+L5	4.7	1	4	0	0	372	7.3	4E-1	11	1	1	1478
L	R3+R4	5.7	-1	6	1	0	536	8.2	2E-1	22	3	2	3631
L	1G6+1G7	8.2	1	3	0	0	528	9.0	1	4	0	0	742
L	L6+L7	8.5	-2	3	0	0	555	8.9	1	4	0	0	764
L	R5+R6	6.8	2	3	0	0	465	8.3	1	4	0	0	659

Table 5.3 (cont.) Results with uniform error control and tolerance relaxing

Prob. 2 [1]		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	1G4+1G5	3.2	1	53	14	1	55521	6.6	1	24	16	2	20762
G	L4+L5	3.1	1	19	16	1	9849	7.4	1	24	16	2	16953
G	R3+R4	4.3	1	70	4	2	65979	6.0	1	54	17	3	50337
G	1G6+1G7	4.8	1	4	0	0	1766	6.4	1	115	13	1	506569
G	L6+L7	4.1	1	4	0	0	1577	---	-	--	-	-	----
G	R5+R6	---	-	--	-	-	----	6.1	1	8	18	1	12305
L	1G4+1G5	3.5	-1E-1	9	2	1	2204	6.3	-5E-2	26	3	2	8058
L	L4+L5	3.8	-1E-1	9	2	1	1943	6.3	-5E-2	27	3	2	7345
L	R3+R4	3.1	2E-2	15	5	2	3446	5.7	8E-3	52	4	3	18608
L	1G6+1G7	4.8	2E-1	4	0	0	1678	6.4	-4E-2	6	1	1	3026
L	L6+L7	4.1	5E-2	4	0	0	1557	6.3	-3E-2	6	1	1	2776
L	R5+R6	3.0	-5E-3	4	0	0	1452	6.3	-1E-2	8	1	1	3481
Prob. 3		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	1G4+1G5	5.0	1	33	6	0	30327	7.4	1	105	26	2	252649
G	L4+L5	4.5	1	38	12	0	37036	8.3	1	113	32	2	242699
G	R3+R4	4.6	1	51	24	0	63912	7.5	1	186	23	2	526699
G	1G6+1G7	4.8	1	16	0	0	17587	8.3	1	37	8	0	82764
G	L6+L7	6.1	1	18	2	0	21046	8.9	1	40	12	1	90095
G	R5+R6	5.3	1	27	5	0	40615	7.8	1	44	7	1	91051
L	1G4+1G5	6.1	1	55	25	0	47213	9.6	1	177	51	2	297461
L	L4+L5	6.5	3E-1	60	20	0	44197	9.9	2	189	44	2	263067
L	R3+R4	6.3	1	89	30	0	73244	9.2	1	359	68	2	769421
L	1G6+1G7	7.3	-1	26	5	0	28064	9.1	1	48	16	0	80400
L	L6+L7	7.7	6	28	4	0	28346	9.2	1	50	12	1	71930
L	R5+R6	6.9	-6	28	3	0	25616	10.0	2	63	22	1	106765

1. The tolerance has been relaxed two times in all global methods except for the R3+R4, the 1G6+1G7, and the L6+L7 combinations with tolerance 1 E-4

Table 5.3 (cont.) Results with uniform error control and tolerance relaxing

Prob. 4		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	1G4+1G5	4.9	1	10	0	0	2697	7.4	1	32	2	2	20422
G	L4+L5	4.8	1	10	0	0	2370	7.5	1	34	2	2	18776
G	R3+R4	5.3	1	15	1	1	4320	7.3	1	60	2	2	51008
G	1G6+1G7	5.8	1	5	0	0	2158	8.8	1	8	1	0	4904
G	L6+L7	5.7	1	5	0	0	1979	8.6	1	8	1	0	4401
G	R5+R6	6.0	1	5	0	0	1822	8.8	1	10	1	0	5818
L	1G4+1G5	6.1	-1	12	1	0	2724	8.1	-3E-1	42	2	2	16896
L	L4+L5	6.3	-5	13	1	0	2816	8.1	-2E-1	44	2	2	15506
L	R3+R4	4.8	-8E-2	21	2	1	4603	7.8	-2E-1	90	2	2	46162
L	1G6+1G7	5.7	-2E+1	5	0	0	1901	8.5	2	8	1	0	3837
L	L6+L7	5.6	-5	5	1	0	2124	8.2	-1	8	0	0	3177
L	R5+R6	6.7	-2E+1	6	1	0	2310	7.9	1	11	1	0	4816
Prob. 5		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	1G4+1G5	4.3	1	29	0	0	19535	7.3	1	92	1	1	155629
G	L4+L5	4.3	1	30	0	0	18375	7.3	1	97	1	1	141564
G	R3+R4	4.2	1	51	1	0	42108	7.3	1	225	3	2	678032
G	1G6+1G7	5.0	1	10	0	0	7948	7.5	1	17	0	0	17543
G	L6+L7	4.9	1	10	0	0	7624	7.5	1	18	0	0	17555
G	R5+R6	4.3	1	11	0	0	7960	7.5	1	24	0	0	26478
L	1G4+1G5	4.2	3E-2	27	0	0	10898	7.3	2E-2	92	1	1	73391
L	L4+L5	4.6	3E-2	33	1	0	14145	7.4	3E-2	100	1	1	72694
L	R3+R4	4.1	2E-2	47	0	0	19580	7.2	2E-2	208	2	2	240743
L	1G6+1G7	5.3	-1E-2	11	0	0	7102	7.0	-6E-4	15	3	0	13036
L	L6+L7	5.5	1	11	1	0	7853	8.0	6E-2	19	0	0	14016
L	R5+R6	4.9	-3E-1	12	1	0	7838	7.0	-6E-3	20	1	0	14468

Table 5.3 (cont.)		Results with uniform error control and tolerance relaxing											
Prob. 6		requested tolerance 1 E-4						requested tolerance 1 E-7					
method		sd	EE/GE	N	#F tot	#F 0	#kev	sd	EE/GE	N	#F tot	#F 0	#kev
G	1G4+1G5	7.7	1	9	2	2	1805	8.7	1	21	3	3	8174
G	L4+L5	7.1	1	9	2	2	1648	8.6	1	21	3	3	6918
G	R3+R4	6.3	1	12	2	2	2392	8.4	1	38	4	4	19892
G	1G6+1G7	5.6	1	5	0	0	1223	10.1	1	8	2	2	3154
G	L6+L7	5.5	1	5	0	0	1152	10.1	1	8	2	2	2922
G	R5+R6	5.7	1	6	1	1	1573	10.2	1	9	2	2	3340
L	1G4+1G5	6.8	1	9	2	2	1277	9.2	1	21	3	3	4401
L	L4+L5	6.9	1	9	2	2	1315	9.1	1	22	3	3	4383
L	R3+R4	6.9	1	12	2	2	1549	8.7	1	40	4	4	10090
L	1G6+1G7	5.6	1	5	0	0	1019	10.0	1	8	2	2	2339
L	L6+L7	5.5	1	5	0	0	1050	9.9	1	8	2	2	2345
L	R5+R6	5.7	1	6	1	1	1260	9.7	1	9	2	2	2401

TABLE 5.3. Results with uniform error control and tolerance relaxing

The failures in Table 5.3 were caused by a lack of working storage, which occurred, depending of the method used, after about 200 - 350 successful steps.

Furthermore, we tried for one problem what the influence is if only the stepsize is adjusted according to the estimated error, but if no steps are rejected. Although in most cases this results in fewer kernel evaluations (if the code reaches the endpoint), it is by far not as reliable.

To present a survey of the performance of the code on these six problems a weighted sum has been computed over all (available) 'sd'- and '#kev'-values. The weights are inversely proportional to the square of the length of the interval on which the solution has been computed.
Table 5.4 contains the figures of the average performance of the different methods. For the iterated collocation method the results have been taken from Table 5.1 and 5.2 for problems 1-4, respectively 5 and 6. For the other methods the 'sd'- and '#kev'-values of Table 5.3 have been used. The results in the first column are the average results computed over all problems. The second contains the average values when the 'bad ones' are left out, i.e. for the Gauss + Iterated collocation method problems 5 and 6, and for all other methods problem 2.

method		All problems				Selected problems			
		tol. 1 E-4		tol. 1 E-7		tol. 1 E-4		tol. 1 E-7	
		sd	#kev	sd	#kev	sd	#kev	sd	#kev
G	G4+IG	5.3	941	8.3	11583	5.0	1023	8.5	16681
G	G8+IG	7.8	579	9.0	1162	6.8	667	8.5	1524
G	1G4+1G5	5.0	9861	7.4	7072	5.3	729	7.6	4334
G	L4+L5	4.7	2220	7.7	6104	5.1	694	7.8	3934
G	R3+R4	4.9	11942	7.3	17759	5.0	1135	7.5	11243
G	1G6+1G7	5.7	743	8.4	85557	5.9	538	8.7	1355
G	L6+L7	5.8	697	8.8	1297	6.1	522	8.8	1297
G	R5+R6	5.6	650	8.1	3277	5.6	650	8.5	1472
L	1G4+1G5	5.3	902	8.0	3995	5.6	642	8.3	3183
L	L4+L5	5.5	874	8.0	3721	5.8	660	8.4	2996
L	R3+R4	5.2	1355	7.8	9942	5.6	937	8.2	8209
L	1G6+1G7	6.2	708	8.3	1421	6.4	514	8.7	1100
L	L6+L7	6.2	709	8.4	1337	6.6	540	8.8	1049
L	R5+R6	5.7	712	8.2	1606	6.2	565	8.6	1231

TABLE 5.4. Average performance

6. CONCLUSIONS

It is clear that, in general, higher order results in a better performance.
In most cases local error estimation used in the higher order methods does not seem to yield a significant decrease in the total number of kernel evaluations. Since it is less reliable than global error estimate / control, which moreover supplies a global error estimate in the endpoint for free, it seems advisable to use global error control in the stepsize strategy.

If the solution is not polynomial, then the Gauss + Iterated collocation method is the most efficient when both computational time and storage are taken into account, especially for tolerances that are not too tight.
The uniform error control was of great benefit to all other methods. The approximations were more dependable with in most cases only a slight increase in computational work.

REFERENCES

1. H. ARNDT (1983). On step size control for Volterra integral equations, in *Numerical Methods of Approximation Theory, ISNM 67*, 9-17, ed. L. Collatz, G. Meinardus and H. Werner, Birkhäuser Verlag, Basel-Boston-Stuttgart.
2. H. ARNDT (1985). An adaptive step size control for Volterra integral equations, in *Constructive Methods for the Practical Treatment of Integral Equations, ISNM 73*, 44-52, ed. G. Hämmerlin and K.-H. Hoffmann, Birkhäuser Verlag, Basel-Boston-Stuttgart.
3. G. BADER and P. KUNKEL. An adaptive multistep method for the solution of second kind Volterra integral equations, *(in preparation)*.
4. J.G. BLOM and H. BRUNNER. Algorithm XXX: Discretized collocation and iterated collocation for nonlinear Volterra integral equations of the second kind, *(to be submitted to ACM Trans. Math. Software)*.
5. J.M. BOWNDS (1982). Theory and performance of a subroutine for solving Volterra integral equations, *Computing*, 28, 317-332.
6. J.M. BOWNDS and L. APPELBAUM (1985). Algorithm 627: A FORTRAN subroutine for solving Volterra integral equations, *ACM Trans. Math. Software*, 11, 58-65.

7. H. BRUNNER (1980). Superconvergence in collocation and implicit Runge-Kutta methods for Volterra-type integral equations of the second kind, in *Numerical Treatment of Integral Equations, ISNM 53*, 54-72, ed. J. Albrecht and L. Collatz, Birkhäuser Verlag, Basel-Boston-Stuttgart.
8. H. BRUNNER (1984). Iterated collocation methods and their discretization for Volterra integral equations, *SIAM J. Numer. Anal.*, 21, 1132-1145.
9. H. BRUNNER, E. HAIRER, and S.P. NØRSETT (1982). Runge-Kutta theory for Volterra integral equations of the second kind, *Math. Comp.*, 39, 147-163.
10. H. BRUNNER and P.J. VAN DER HOUWEN (1985 (to appear)). *The Numerical Solution of Volterra Equations*, North-Holland, Amsterdam.
11. H. BRUNNER and S.P. NØRSETT (1981). Superconvergence of collocation methods for Volterra and Abel integral equations of the second kind, *Numer. Math.*, 36, 347-358.
12. R.P. DUNCAN (1982). *A Runge-Kutta method using variable stepsizes for Volterra integral equations of the 2nd kind,* Master's Thesis, Technical Report 157/82, Department of Computer Science, University of Toronto.
13. E. HAIRER, CH. LUBICH, and M. SCHLICHTE (1985). Fast numerical solution of nonlinear Volterra convolution equations, *SIAM J. Sci. Statist. Comput.*, 6, 532-541.
14. W. HOCK (1981). An extrapolation method with step size control for nonlinear Volterra integral equations, *Numer. Math.*, 38, 155-178.
15. F. DE HOOG and R. WEISS (1975). Implicit Runge-Kutta methods for second kind Volterra integral equations, *Numer. Math.*, 23, 199-213.
16. H.M. JONES and S. MCKEE (1985). Variable step-size predictor-corrector schemes for second kind Volterra integral equations, *Math. Comp.*, 44, 391-404.
17. P. KUNKEL (1982). *Ein adaptives Verfahren zur Lösung von Volterra'schen Integralgleichungen zweiter Art,* Diplomarbeit, Institut für Angewandte Mathematik, Universität Heidelberg.
18. R.K. MILLER (1971). *Nonlinear Volterra Integral Equations*, Benjamin, Menlo Park (Ca).
19. M. SCHLICHTE (1984). *Anwendung eines impliziten Runge-Kutta-Verfahrens auf Volterra'sche Integralgleichungen zweiter Art mit Faltungskern,* Diplomarbeit, Institut für Angewandte Mathematik, Universität Heidelberg.

Appendix

prob	method	tol. 1 E-4		tol. 1 E-7		prob	method	tol. 1 E-4		tol. 1 E-7	
		sd	#kev	sd	#kev			sd	#kev	sd	#kev
1	IVRKX	7.2	510	9.0	1712	2	IVRKX	5.4	2124	7.7	5208
	ORION	3.5	923	6.5	1624		ORION	2.0	1442	4.4	2998
	VOLTEX	5.6	1124	8.7	4110		VOLTEX	2.0	3562	5.6	15092
	INTSOL	4.0	2722	4.8	127964		INTSOL	1.1	64808		
	VE1ca	4.1	1537	7.6	5684		VE1ca	5.2	2527	7.6	7550
	VE1ex	5.4	414	8.1	630		VE1ex	1.8	300	4.1	488
	IVRKXC	7.2	359	9.0	868		IVRKXC	5.4	1041	7.7	1980
	VOLCON	4.7	238	9.6	762		VOLCON	4.5	1262	8.3	2662
3	IVRKX	5.0	23763	7.6	54320	4	IVRKX	4.9	1044	7.4	2704
	ORION	6.3	42325	7.8	182344		ORION	4.7	1805	6.1	6411
	VOLTEX	5.4	76882	7.6	417014		VOLTEX	5.3	5626	8.2	22918
	VE1ca	2.4	19943	---	----		VE1ca	4.5	4157	8.5	12111
							VE1ex	4.9	830	6.8	1450
	IVRKXC	5.0	4237	7.6	6358		IVRKXC	4.9	582	7.4	1052
	VOLCON	6.6	2230	8.5	5370		VOLCON	4.8	366	7.3	1382
5	IVRKX	5.2	8625	8.9	21528	6	IVRKX	4.4	1029	9.7	4512
	ORION	5.6	2647	8.1	7114		ORION	6.0	402	8.7	675
	VOLTEX	5.2	120752	8.1	160910		VOLTEX	2.5	634	6.3	2648
	INTSOL	3.6	337332				INTSOL	1.7	4312	2.0	9749
	VE1ca	3.7	9225	---	----						
	VE1ex	4.3	342	6.8	730						
	IVRKXC	5.2	2167	8.9	3864						
	VOLCON	6.5	2578	8.2	5170						

TABLE A.1. Results of other published codes

The results of INTSOL were derived from Kunkel[17], all other results, except those of VE1, from Schlichte[19]. VE1ca gives the results of VE1 when the kernel has been approximated by Chebyshev series. The number of terms, m, has been chosen as the minimal number such that the estimated error, as given by VE1, is less than the tolerance. VE1ex gives the results when an exact decomposition was provided. In this case no global error estimate can be calculated.

Note that for VOLCON and for IVRKXC, '#kev' stands for the total number of K- plus f-evaluations where $k(t,s,y)$ is written as $K(t-s).f(s,y)$. For VE1ex, '#kev' represents the number of evaluations of the right-hand side of the system of ODE's, '#Fev', times the dimension of the system, 'DIM', times two. In the case of VE1ca, #kev = #Kev + (DIM+2).#Fev, where $DIM = 2m + 1$.

method	tol. 1 E-4		tol. 1 E-7	
	sd	#kev	sd	#kev
IVRKX	5.4	738	8.4	2216
ORION	4.7	653	6.9	1643
VOLTEX	4.3	1636	7.4	6101
INTSOL	2.6	19278	3.4	68857
VE1ca	4.0	1112	7.9	5420
VE1ex	4.1	232	6.5	373
IVRKXC	5.5	329	8.1	654
VOLCON	5.4	333	8.4	787

TABLE A.2. Average performance