ELSEVIER Journal of Computational and Applied Mathematics 71 (1996) 225–236

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

Highly stable parallel Volterra Runge–Kutta methods

Antonia Vecchio

Istituto per Applicazioni della Matematica, Consiglio Nazionale delle Ricerche, Via P. Castellino, 111, 80131 Napoli, Italy

Received 28 July 1994; revised 11 October 1995

Abstract

A new family of parallel V-stable methods for second kind Volterra integral equations are proposed. The methods belong to the class of Volterra Runge–Kutta methods and can be applied provided that some not very restrictive conditions on the integral equation kernel are satisfied.

Keywords: Volterra Runge–Kutta methods; Stability; Parallelism

AMS classification: Primary 65R20; secondary 45D05, 45L10, 65Y05

1. Introduction

One of the main problems in constructing second kind Volterra integral equations (VIEs) solvers is to get methods with good stability properties. The literature contains a number of contributions to stabiity analysis of numerical methods for VIEs which vary according to the choice of the Volterra equation adopted as a test equation. A test equation that has, so far, received particular emphasis is the well known *linear convolution equation*:

$$y(t) = 1 + \int_{t_0}^{t} [\lambda + \mu(t-s)]\, y(s)\, \mathrm{d}s \tag{1.1}$$

with λ and μ real parameters satisfying

$$\lambda < 0, \qquad \mu \leqslant 0. \tag{1.1'}$$

Related to (1.1), the best stability property one can require of a numerical method is V_0-stability (see e.g. [2, p. 438] or [3]), but until 1993 very few V_0-stable methods have been proposed in the literature [7, 9, 12, 13, 14] and it was in [1] that the first V_0-stable method of order exceeding

SSDI 0377-0427(95)00235-9

one was obtained. In 1993 in [5] we have proposed a test equation which is a slight generalization of (1.1) where λ is allowed to assume complex values:

$$y(t) = 1 + \int_{t_0}^{t} [\lambda + \mu(t-s)]\, y(s)\, ds, \qquad \mathrm{Re}(\lambda) \leqslant 0, \quad \mu \leqslant 0. \tag{1.2}$$

For this test equation we have given the following definitions:

Definition 1.1. A numerical method is said to be *stable* (strongly) in the point $(h\lambda, h^2\mu)$ of the set $Q := \{(h\lambda, h^2\mu)$: $\mathrm{Re}(\lambda) \leqslant 0,\ \mu \leqslant 0\}$ if for such values the numerical solution y_n is bounded $(\lim_{n\to\infty} y_n = 0)$. Then the (strong) *stability region* of the method is defined by the set $\boldsymbol{S} \subseteq Q$ where the method is (strongly) stable.

Definition 1.2. The method will be called (strongly) *V-stable* if its (strong) stability region $\boldsymbol{S}$ coincides with Q

Of course in the case $\mathrm{Im}(h\lambda) = 0$ the strong V-stability coincides with V_0-stability.

The aim of this paper is to exploit the potential of parallel computers to construct methods with a reasonable sequential computational cost, which are highly stable with respect to (1.2).

In [4] we introduced the parallel iterated Voltera Runge–Kutta (PIVRK) methods with the goal of taking advantage of parallel architectures in solving the implicit relations for the stage values of a classical Volterra Runge–Kutta methods. This was done by solving such set of equations by an iteration scheme. By means of an appropriate choice of the iteration parameters the stage vector equations can be uncoupled and solved in parallel. Moreover in [2, 4] we found that the stability regions of such parallel methods, though large, have the drawback to present a gap near the vertical axis (see Fig. 1 for example).

In [5], together with the definition of V-stability, we have given a technique for constructing V-stable methods, called *transformed* methods, of any order, which can be applied to any second kind Volterra integral equation, whose kernel satisfies some mild conditions. In particular we have constructed V-stable transformed Volterra Runge–Kutta (TVRK) methods whose computational cost is at most doubled with respect to the traditional Pouzet Volterra Runge–Kutta methods.

Now the natural purpose of this paper is to merge the results of [4, 5] and to construct parallel methods having better stability characteristics than the above-mentioned PIVRK and cost less than the TVRK methods.

In particular in Section 2, we derive the method satisfying these requisites, simply by constructing the transform of some PIVRK methods and in Section 3 we give their stability functions and prove the V-stability of a large number of methods belonging to this family whose order ranges from 2 to 8. Finally in Section 4 we report the results of some numerical experiments in order to compare the performance of the proposed methods with the PIVRK ones.

2. The methods

Let us consider the second kind Volterra integral equation

$$y(t) = g(t) + \int_{t_0}^{t} k(t, s, y(s))\,\mathrm{d}s, \quad t \in [t_0, T] \tag{2.1}$$

where g and k are given continuous functions on $[t_0, T]$ and $S \times R$, respectively, with $S := \{(t, s): t_0 \leqslant s \leqslant t \leqslant T\}$. Moreover let us assume, as in [5], that kernels $k(t, s, y)$ can be split according to

$$k(t, s, y) = k_1(t, s, y) + k_2(t, s, y) - k_2(s, s, y) \tag{2.2}$$

where the derivative of k_2 with respect to t exists. We observe that the hypothesis (2.2) is not very restrictive since for example for any kernel $k(t, s, y)$ being differentiable w.r.t. t, we may define $k_1 := k(s, s, y)$ and $k_2 := k(t, s, y)$. Another example are kernels of the form $A(t, s, y) + B(s, y)[f(t) - f(s)]$ with f differentiable (note that in this case k itself does not need to be differentiable).

The *parallel transformed Volterra Runge–Kutta* (PTVRK) methods arise from the s-stage parallel iterated Pouzet Runge–Kutta (PIPVRK) methods (*original methods*):

$$y_{n+1} = F_n(t_{n+1}) + h \sum_{j=1}^{s} b_j k(t_{n+1}, t_n + c_j h, Y_{nj}^m) \tag{2.3a'}$$

$$Y_{ni}^{v+1} - h d_i k(t_n + c_i h, t_n + c_i h, Y_{ni}^{v+1}) = F(t_n + c_i h) + h \sum_{j=1}^{s} (a_{ij} - \delta_{ij} d_i) k(t_n + c_i h, t_n + c_j h, Y_{nj}^{v}) \quad v = 0, \ldots, m, \quad i = 1, \ldots, s, \tag{2.3b}$$

$$F_n(t) = g(t) + h \sum_{r=0}^{n-1} \sum_{j=1}^{s} b_j k(t, t_r + c_j h, Y_{rj}^m). \tag{2.3c}$$

We recall that such methods, that we introduced in [4], are called PIPVRKI when (2.3a') is used and PIPVRK2 if the underlying VRK method $\frac{c|A}{|b}$ is stiffly accurate (i.e., $c_s = 1$, $b_i = a_{si}$, $i = 1, \ldots, s$) and y_{n+1} is simply determined by

$$y_{n+1} = Y_{ns}^{ms}. \tag{2.3a''}$$

By applying the transformation introduced in [5] to (2.3) we get

$$y_{n+1} = F_n^1(t_{m+1}) + h \sum_{j=1}^{s} b_j [k_1(t_{n+1}, t_n + c_j h, Y_{nj}^m) + U_{nj}^m], \tag{2.4'}$$

$$Y_{nj}^{v+1} - h d_j [k_1(t_n + c_j h, t_n + c_j h, Y_{nj}^{v+1}) + U_{nj}^{v+1}] = F_n^1(t_n + c_j h) + h \sum_{k=1}^{s} (a_{jk} - \delta_{jk} d_k)[k_1(t_n + c_j h, t_n + c_k h, Y_{nk}^{v}) + U_{nk}^{v}], \quad v \geqslant 0, \; j = 1, \ldots, s, \tag{2.4b}$$

$$U_{nj}^{v+1} - hd_j \frac{\partial k_2}{\partial t}(t_n + c_j h, t_n + c_j h, Y_{nj}^{v+1})$$

$$= F_n^2(t_n + c_j h) + h \sum_{k=1}^{s} (a_{jk} - \delta_{jk} d_k) \frac{\partial k_2}{\partial t}(t_n + c_j h, t_n + c_k h, Y_{nk}^{v}), \quad v \geqslant 0,\ j = 1, \ldots, s,$$

$$Y_{nj}^{0} = F_n^1(t_n + c_j h), \quad U_{nj}^{0} = F_n^2(t_n + c_j h),\ j = 1, \ldots, s,$$

$$F_n^1(t) = g(t) + h \sum_{l=0}^{n-1} \sum_{k=1}^{s} b_k [k_1(t, t_l + c_k h, Y_{lk}^{m}) + U_{lk}^{m}] \quad F_n^2(t) = h \sum_{l=0}^{n-1} \sum_{k=1}^{s} b_k \frac{\partial k_2}{\partial t}(t, t_l + c_k h, Y_{lk}^{m}).$$

Then by eliminating U_{nj} from (2.4b) we get the following formulation of the PTVRK method:

$$y_{n+1} = F_n^1(t_{n+1}) + h \sum_{j=1}^{s} b_j [k_1(t_{n+1}, t_n + c_j h, Y_{nj}^{m}) + U_{nj}^{m}], \tag{2.5a'}$$

$$U_{nj}^{m} = hd_j \frac{\partial k_2}{\partial t}(t_n + c_j h, t_n + c_j h, Y_{nj}^{m}) + F_n^2(t_n + c_j h) + h \sum_{k=1}^{s} (a_{jk} - \delta_{jk} d_k)$$

$$\times \frac{\partial k_2}{\partial t}(t_n + c_j h, t_n + c_k h, Y_{nk}^{m-1}), \quad j = 1, \ldots, s,$$

$$Y_{nj}^{v+1} - hd_j k_1(t_n + c_j h, t_n + c_j h, Y_{nj}^{v+1}) - h^2 d_j^2 \frac{\partial k_2}{\partial t}(t_n + c_j h, t_n + c_j h, Y_{nj}^{v+1}) = \Phi(Y_{nj}^{v}, Y_{nj}^{v-1}),$$

$$\Phi(Y_{nj}^{v}, Y_{nj}^{v-1})$$

$$= F_n^1(t_n + c_j h) + h \sum_{k=1}^{s} a_{jk} F_n^2(t_n + c_k h) + h \sum_{k=1}^{s} (a_{jk} - \delta_{jk} d_k) k_1(t_n + c_j h, t_n + c_k h, Y_{nk}^{v})$$

$$+ h^2 \sum_{k=1}^{s} d_k (a_{jk} - \delta_{jk} d_k) \frac{\partial k_2}{\partial t}(t_n + c_k h, t_n + c_k h, Y_{nk}^{v})$$

$$+ h^2 d_j \sum_{k=1}^{s} (a_{jk} - \delta_{jk} d_k) \frac{\partial k_2}{\partial t}(t_n + c_j h, t_n + c_k h, Y_{nk}^{v})$$

$$+ h^2 \sum_{k=1}^{s} \sum_{r=1}^{s} (a_{jk} - \delta_{jk} d_k)(a_{kr} - \delta_{kr} d_r) \frac{\partial k_2}{\partial t}(t_n + c_k h, t_n + c_r h, Y_{nr}^{v-1}), \quad v \geqslant 1,\ j = 1, \ldots, s,$$

$$\Phi(Y_{nj}^{0}, Y_{nj}^{-1}) = F_n^1(t_n + c_j h) + h \sum_{k=1}^{s} (a_{jk} - \delta_{jk} d_k)[k_1(t_n + c_j h, t_n + c_k h, Y_{nk}^{0}) + F_n^2(t_n + c_k h)],$$

$$Y_{nj}^{0} = F_n^1(t_n + c_j h), \quad j = 1, \ldots, s,$$

$$F_n^1(t) = g(t) + h \sum_{l=0}^{n-1} \sum_{k=1}^{s} b_k [k_1(t, t_l + c_k h, Y_{lk}^{m}) + U_{lk}^{m}],$$

$$F_n^2(t) = h \sum_{l=0}^{n-1} \sum_{k=1}^{s} b_k \frac{\partial k_2}{\partial t}(t, t_l + c_k h, Y_{lk}^{m}).$$

Also in this case

$$y_{n+1} = Y_{ns}^{m} \tag{2.5a''}$$

if the VRK method $\frac{c|A}{|b}$ is stiffly accurate. Analogous to the PIPVRK methods, if (2.5a') is used, then the method will be referred to as PTVRK1, otherwise as PTVRK2.

As we already proved in [6], the order p of the PTVRK methods, for every $D = \mathrm{diag}(d_i)$, is given by

$$p = \min\{p^*, m+1\} \quad \text{in the PTVRK1 case,}$$

$$p = \min\{p^*, m\} \quad \text{in the PTVRK2 case}$$

where p^* is the order of the $\frac{c|A}{|b}$ VRK method.

We observe that, in spite of its involved form, the PTVRK method is of course suitable for implementation on multi-processor machines, since at each iteration the components Y_{nj}^{ν} can be computed in parallel. Therefore, if we compare the PTVRK methods (2.5) with the V-stable TVRK methods [5], we note that their computational cost is considerably reduced because, instead of solving a system of s implicit equation at each step, we have to solve, m times, one implicit equation. Moreover it is worth to observe (see also [11]) that in the case of a VIEs system, the jth processor has to solve a sequence of m implicit system in each of which the decomposition of the matrix

$$I - hd_j\frac{\partial k_1}{\partial y} - h^2 d_j^2 \frac{\partial^2 k_2}{\partial t \partial y}$$

is required and therefore, if the diagonal matrix D has equal entries, then the processors need the same LU decomposed matrix.

On the other hand, compared with the original PIPVRK methods (2.3), the PTVRK methods require much more effort per step, but they present nicer stability region as will be shown in the following section. Since the stability regions are full along the vertical axis, the methods are also suitable for solving oscillating problems.

3. V-stable PTVRK methods

In the previous section we have constructed the PTVRK methods as the transform of the PTVRK. Such methods arise as the Volterra analogues of some parallel methods for ordinary differential equations, which are called PDIRK methods [5, 6]. In this section we show how the stability properties of the PTVRK methods are strictly related to the ones of the ODE PDIRK methods. To this purpose we give the following definition.

Definition 3.1. A PTVRK and a PDIRK method are said to be *correspondent* if they are characterized by the same coefficients c, b, A, D.

Moreover the following notations are useful in the next theorem:

$$R = hx(I - hxD)^{-1}(A - D) \in C^{s\times s}, \qquad l = (1,1,\ldots,1)^{\mathrm{T}} \in \mathbb{R}^s, \qquad e^* = (0,0,\ldots,1)^{\mathrm{T}} \in \mathbb{R}^s,$$

$$f_1(z) = 1 + zb^{\mathrm{T}}[R^m + (I - R^m)(I - zA)^{-1}]e \in C,$$

$$f_2(z) = e^{*\mathrm{T}}[R^m + (I - R^m)(I - zA)^{-1}]e \in C, \tag{3.1}$$

$$\Lambda = \begin{pmatrix} h\lambda & 1 \\ h^2\mu & 0 \end{pmatrix}.$$

Now, we derive the following theorem.

Theorem 3.2. *The (strong) stability region of the PTVRK1 and PTVRK2 methods is the set* $S := \{(h\lambda, h^2\mu)$: $\mathrm{Re}(\lambda) \leqslant 0,\ \mu \leqslant 0\}$ *such that* $|f_1(z)| \leqslant 1$ *and* $|f_2(z)| \leqslant 1$ $(|f_1(z)| < 1,\ |f_2(z)| < 1)$ *respectively, where* $z(h\lambda, h^2\mu)$ *runs through the spectrum of the matrix* $\Lambda(h\lambda, h^2\mu)$.

Proof. Let us consider the method in the form (2.4) and apply it to the test equation (1.2). It can be easily seen that this is equivalent to applying (2.3) to the system

$$y(t) = 1 + \int_0^t [\lambda y(s) + u(s)]\,\mathrm{d}s,$$

$$u(t) = \int_0^t \mu y(s)\,\mathrm{d}s.$$

Thus the PTVRK method is (strongly) stable for the value (λ, μ) whenever the eigenvalues x of the matrix

$$\begin{pmatrix} \lambda & 1 \\ \mu & 0 \end{pmatrix}$$

belong to the (strong) stability region of the method (2.3) w.r.t. the basic test equation

$$y(t) = 1 + x\int_0^t y(s)\,\mathrm{d}s, \quad \mathrm{Re}(x) \leqslant 0. \tag{3.2}$$

Therefore, in order to study the stability of the PTVRK methods we have to apply (2.3) to (3.2). We obtain

$$y_{n+1} = y_n + hxb^{\mathrm{T}}Y_n^m \quad \text{or} \quad y_{n+1} = e^{*\mathrm{T}}Y_n^m,$$

$$Y_n^m = R^m Y_n^0 + (I - R^m)(I - R)^{-1}R_1 y_n e, \qquad Y_n^0 = y_n e$$

with $R_1 = (I - hxD)^{-1}$. Then, since $(I - R)^{-1}R_1 = (I - hxA)^{-1}$ and setting $z = hx$, there results

$$y_{n+1} = f_1(z)y_n \quad \text{or} \quad y_{n+1} = f_2(z)y_n$$

where z runs through the set of eigenvalues of Λ. □

From this theorem we immediately obtain

Corollary 3.3. *If the corresponding PDIRK method is A-stable then the PTVRK method is V-stable.*

We observe that the last result could be directly deduced from Theorem 2.1 in [5]. By means of Corollary 3.3 we can assure, for example, the V-stability of the following PTVRK methods of order

p ranging from 2 to 8. In fact the corresponding ODE PDIRK methods are proved to be A-stable in [10, 11].

We report the coefficients of the matrix D of such V-stable methods, but we refrain to report the coefficients A, b and c since they are standard and can be found in literature or in [10]. In the following I_s is the s by s identity matrix.

PTVRK1–Gauss–$s = 2$–$p = 4$–$m = 3$ $D = (1.0685790213)I_2$ [11].

PTVRK1–Radau IIA–$s = 2$–$p = 3$–$m = 2$ $D = ((3 + \sqrt{3})/6)I_2$ [11].

PTVRK2–Radau IIA–$s = 2$–$p = 3$–$m = 3$ $D = (0.43586650)I_2$ [11].

PTVRK2–Radau IIA–$s = 3$–$p = 5$–$m = 5$

$$D = \operatorname{diag}\begin{pmatrix} 0.32039049 \\ 0.13997017 \\ 0.37167618 \end{pmatrix} \quad [10]. \tag{3.3}$$

PTVRK2–Radau IIA–$s = 4$–$p = 7$–$m = 7$

$$D = \operatorname{diag}\begin{pmatrix} 0.32049937 \\ 0.08915379 \\ 0.18173957 \\ 0.23336280 \end{pmatrix} \quad [10].$$

PTVRK1–Lobatto IIIA–$s = 2$–$p = 2$–$m = 1$ $D = (1/2)I_2$ [11].

PTVRK2–Lobatto IIIA–$s = 2$–$p = 2$–$m = 2$ $D = (1 + \sqrt{\frac{1}{2}})I_2$ [11].

PTVRK1–Lobatto IIIA–$s = 3$–$p = 4$–$m = 3$ $D = (1.0685790213)I_3$ [11].

PTVRK2–Lobatto IIIA–$s = 3$–$p = 4$–$m = 4$ $D = (0.5728160625)I_3$ [11].

PTVRK2–Lobatto IIIA–$s = 4$–$p = 6$–$m = 6$ $D = (0.3341423671)I_4$ [11].

PTVRK2–Lobatto IIIA–$s = 5$–$p = 8$–$m = 8$ $D = (0.2343731596)I_5$ [11].

We underline that most of the matrix D, above reported, are computed in [11] with the aim of obtaining A-stable PDIRK methods with constant diagonal matrix. Such characteristic, as we mentioned in the previous section, is convenient for the LU decomposition of the matrix connected to the implicit relation appearing in the method. The matrices D which are not constant are determined with the purpose of minimizing the spectral radius of a matrix related to the iteration process of the ODE PDIRK method (see [10] for details).

In order to better compare the stability properties of the PTVRK and the original PIVRK methods we report the plot of the stability region in the plane $\{\mathrm{Re}(h\lambda) \leqslant 0\ h^2\mu \leqslant 0\}$ of one of the previous methods. Of course the plot of the PTVRK is full, since the method is V-stable (see Fig. 1).

Moreover we consider the PTVRK method based on the 3-stage Gauss coefficients and we construct a matrix D in order to have an highly stable methods with order $p = 6$ and requiring only $m = 5$ iteration per step. We observe that, even if the corresponding PDIRK method cannot be

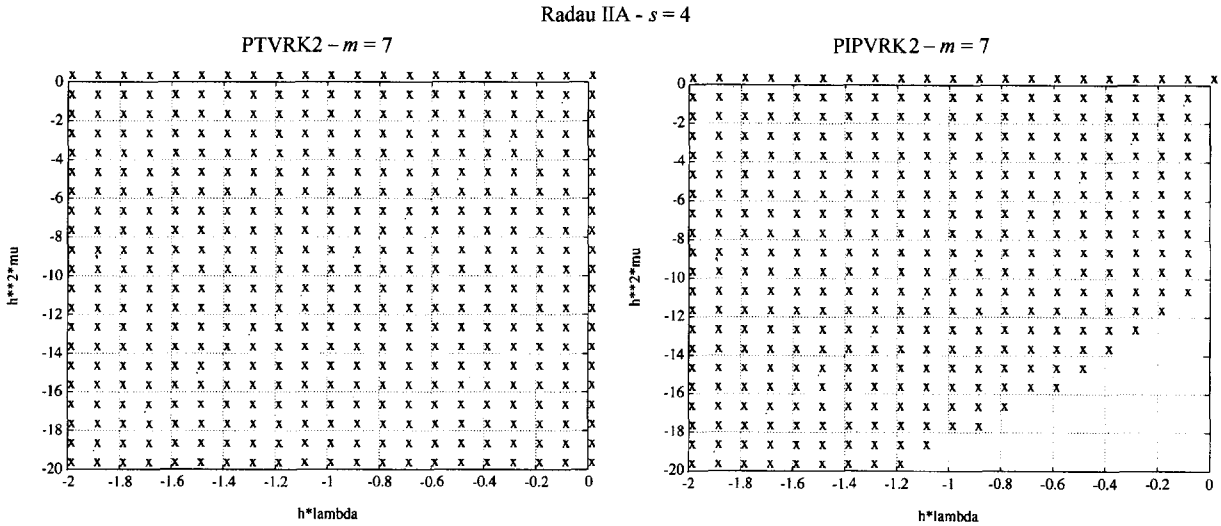

Fig. 1

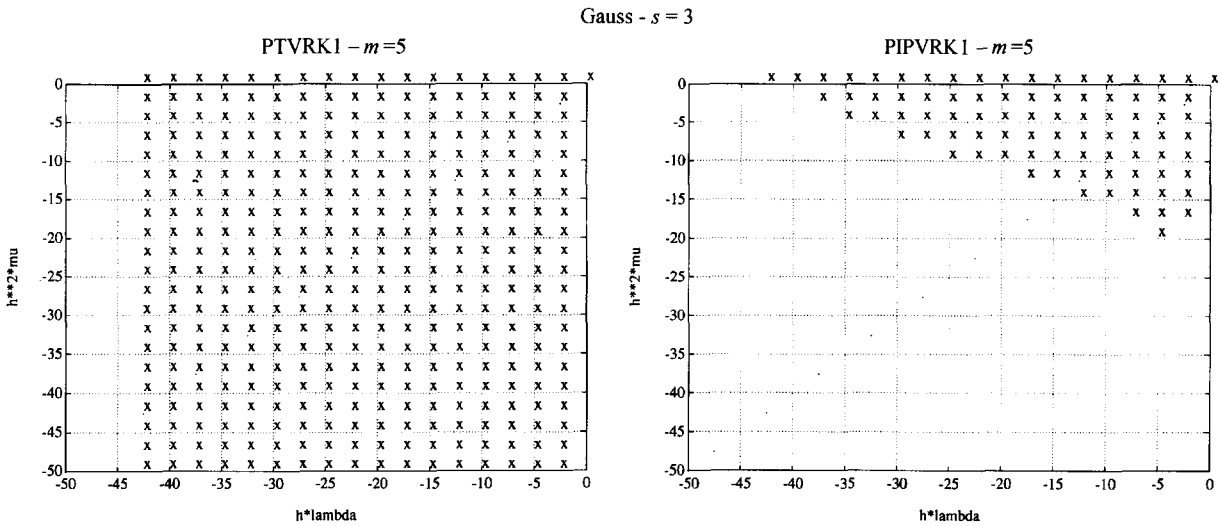

Fig. 2

A-stable for any choice of D, as can be verified by the expression of the stability function $f_1(z)$ in (3.1), the following PTVRK method has a satisfactory stability region. Then we report the matrix D of this method together with its stability plot. For the sake of comparison we also report the plot of the stability region of the original PIPVRK (see Fig. 2).

PTVRK1–Gauss–$s = 3$–$p = 6$–$m = 5$,

$$D = \operatorname{diag}\begin{pmatrix} 0.23572124 \\ 0.12554113 \\ 0.37296599 \end{pmatrix},$$

We can observe that the stability region of the PTVRK method is larger than the other one and this suggests us to apply this method also in the case where the corresponding PDIRK method is not A-stable.

This fact can be explained by considering the PTVRK method as an "*iterated version*" of a TVRK method instead of the "*transformed version*" of an iterated method. In other words, let us consider the TVRK method [5], which in the case of the Gauss coefficients is V-stable.

$$y_{n+1} = F_1^*(t_{n+1}, t_n) + h\sum_{i=1}^{s} b_i[k_1(t_n + c_ih, t_n + c_ih, Y_{ni}) + F_2^*(t_n + c_ih, t_n)]$$
$$+ h^2\sum_{i=1}^{s} b_i \sum_{j=1}^{s} a_{ij}\frac{\partial}{\partial t}k_2(t_n + c_ih, t_n + c_jh, Y_{nj}), \tag{3.4}$$

$$Y_{ni} = F_1^*(t_n + c_ih, t_n) + h\sum_{j=1}^{s} a_{ij}[k_1(t_n + c_ih, t_n + c_jh, Y_{nj}) + F_2^*(t_n + c_jh, t_n)]$$
$$+ h^2\sum_{j=1}^{s} a_{ij} \sum_{k=1}^{s} a_{jk}\frac{\partial}{\partial t}k_2(t_n + c_jh, t_n + c_kh, Y_{nk}), \quad i = 1, \ldots, s, \tag{3.4'}$$

$$F_1^*(t, t_n) = g(t) + h\sum_{l=0}^{n-1}\sum_{i=1}^{s} b_i[k_1(t, t_l + c_ih, Y_{li}) + F_2^*(t_l + c_ih, t_l)]$$
$$+ h\sum_{j=1}^{s} a_{ij}\frac{\partial}{\partial t}k_2(t_l + c_ih, t_l + c_jh, Y_{lj}),$$

$$F_2^*(t, t_n) = h\sum_{l=0}^{n-1}\sum_{i=1}^{s} b_i\frac{\partial}{\partial t}k_2(t, t_l + c_ih, Y_{li}).$$

The PTVRK method (2.5) can be obtained from (3.4) by solving (3.4') by a special diagonal iteration scheme. According to this point of view we observe that in the case of the PTVRK 3-stage Gauss method, we are dealing with an "iterated version" of a V-stable method and so good stability properties can be expected.

4. Numerical experiments

In order to illustrate the advantages and disadvantages of PTVRK methods we have applied it to a system of VIEs obtained by discretizing in space a partial integral equation of the following type:

$$v(x, t) = v(x, 0) + p(x, t) + \int_0^t k(t, s)v_{xx}(x, s)\,ds, \quad 0 \leqslant x \leqslant 1, \ t \in [0, T],$$
$$v(x, 0) = v_0(x), \quad 0 \leqslant x < 1, \ v(0, t) = v(1, t) = 0, \tag{4.1}$$

with

$$k(t, s) = \int_s^t G(\tau - s)\,\mathrm{d}\tau, \qquad p(x, t) = \int_0^t f(x, s)\,\mathrm{d}s$$

and G, f and v_0 are given.

Equations of this type, although they do not arise directly from practical problems, are obtained by the usual approach of integrating with respect to time some Volterra integro-differential equations which are common in mathematical viscoelasticity pertaining materials with memory ([8, p. 10, Example 2.7]). Physically, indeed, $v(x, t)$ is the displacement at time t of point x on a one-dimensional rod located at $0 \leqslant x \leqslant 1$. The function G is the linear stress relaxation modulus and the function p is related to the initial external force.

Since in the most common practical case $G(t, s)$ is a decreasing exponential function of $(t - s)$, (see, e.g., [15]), we have assumed

$$G(t, s) = \alpha \mathrm{e}^{-\beta(t-s)}$$

then we have chosen v_0 and p so that the true solution is

$$v(x, t) = \sin(\pi x)\mathrm{e}^{-1}.$$

With these choices our test partial VIE has the following expression:

$$v(x, t) = p^*(x, t) + \frac{\alpha}{\beta}\int_0^t [1 - \mathrm{e}^{-\beta(t-s)}]v_{xx}(x, s)\,\mathrm{d}s, \quad 0 \leqslant x \leqslant 1, \ t \in [0, T], \tag{4.2}$$

with

$$p^*(x, t) = \sin(\pi x)\left[\mathrm{e}^{-t} + \pi^2\alpha\left(\frac{\mathrm{e}^{-\beta t}}{\beta(\beta - 1)} - \frac{\mathrm{e}^{-t}}{\beta - 1} + \frac{1}{\beta}\right)\right].$$

Thus we semidiscretize with respect to x by defining the equidistant grid $x_i = i\Delta$, $i = 0, \ldots, N$ with $\Delta = 1/N$ and by choosing the second order approximation

$$v_{xx}(x_i, t) \approx \frac{v(x_{i+1}, t) - 2v(x_i, t) + v(x_{i-1}, t)}{\Delta^2}. \tag{4.3}$$

Finally by setting $v(x_i, t) = v_i(t)$, $p^*(x_i, t) = p_i^*(t)$ our test system becomes

$$V(t) = P(t) + \int_0^t k(t, s)\Gamma V(s)\,\mathrm{d}s,$$

where

$$V(t) = [v_1(t), \ldots, v_{N-1}(t)]^{\mathrm{T}}, \qquad P(t) = [p_1^*(t), \ldots, p_{N-1}^*(t)]^{\mathrm{T}}, \qquad k(t, s) = \frac{\alpha}{\beta}[1 - \mathrm{e}^{-\beta(t-s)}] \in \mathbb{R},$$

$$\Gamma = \frac{1}{\Delta^2}\begin{pmatrix} -2 & 1 & 0 & 0 & \cdots & 0 \\ 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -2 \end{pmatrix} \in \mathbb{R}^{N-1 \times N-1}.$$

In the following experiments we have fixed $N = 10$, $\beta = 2$ and we have applied the PTVRK method (3.3) and the related original PIVRK with different values of α by using the splitting we mentioned in Section 2:

$$k_1(t, s) = k(s, s) = 0, \qquad k_2(t, s) = k(t, s) = \frac{\alpha}{2}[1 - e^{-2(t-s)}]. \tag{4.4}$$

The numerical experiments have been performed on a small transputer network (3 transputers with 2MBytes of memory).

In the following table the computational time required by the methods for obtaining the same accuracy are listed. Since the methods are applied with fixed stepsize, the h value reported in the table is about the largest one for obtaining the absolute error of size 10^{-17} in the final point of the integration range [0, 40].

Table 1

α	PIVRK		PTVRK	
	h	time	h	time
0.1	0.25	$\approx$ 3 min	0.25	$\approx$ 6 min
1	0.1	$\approx$ 20 min	0.25	$\approx$ 6 min
10	0.03	$\approx$ 210 min	0.1	$\approx$ 40 min

Of course it is to be noted that both the methods are very expensive, compared with the size of the system ($N = 10$), but it is due to the "smallness" of the computer at our disposal.

From these results we can easily observe that in the cases of VIEs system which does not present stability problems ($\alpha = 0.1$) the transformed method is not convenient since it requires double computational labour compared with the PIVRK method. On the other hand for $\alpha = 1$ and $\alpha = 10$ such method requires a severe stepsize restriction due to stability problem (in both the cases the choice $h = 0.25$ causes an overflow in the numerical solution) while the PTVRK method allows a large stepsize and therefore, in spite of its theoretical higher computational cost, it results to be more convenient.

Of course the splitting (4.4) is not unique, but from our experience the results obtained by choosing different splitting leads to the same conclusions. Obviously a "dummy" splitting must be avoided, i.e., $k_2(s, s, y) \equiv 0$ since in this case the PTVRK methods in some sense "includes" the original PIVRK method itself.

5. Concluding remarks

By applying a technique we proposed in [5] to the class of PIVRK methods that we introduced in [4] we construct a new family of s-stage V-stable iterated methods. The price we pay for this improvement in the stability characteristics of the method is of course an increased computational effort per iteration (about s^2 function evaluation more) and the requirement of the splitting (2.2) for the kernel of the integral equation.

References

[1] A. Bellen, Z. Jackiewicz, R. Vermiglio and M. Zennaro, Stability analysis of Runge–Kutta methods for Volterra integral equations of the second kind, *IMA J. Numer. Anal.* **10** (1990) 103–118.
[2] H. Brunner, S.P. Nørsett and P.H.M. Wolkenfelt, On V_0-stability of numerical methods for Volterra integral equations of the second kind, Report NW 84/80, Math. Centrum Amsterdam, 1980.
[3] H. Brunner and P.J. van der Houwen, *The Numerical Solution of Volterra Equations* (North Holland, Amsterdam, 1986).
[4] M.R. Crisci, P.J. van der Houwen, E. Russo and A. Vecchio, Stability of parallel Volterra Runge–Kutta methods, *J. Comput. Appl. Math.* **45** (1993) 169–180.
[5] M.R. Crisci, P.J. van der Houwen and A. Vecchio, V-stable methods for second kind Volterra integral equations, *Rapp. Tecn. IAM 98/93*, 1993.
[6] M.R. Crisci and A. Vecchio, Stability plots of parallel Volterra Runge–Kutta methods for Volterra integral equations, Rapp. Tecn. IAM 87/91, 1991.
[7] N. Ferraro, A class of parallel V_0-stable methods for Volterra integral equations, *Ricerche Mat.* **44** (1995) 221–230.
[8] G. Gripenberg, S.O. Londen and O. Staffans, *Volterra Integral and Functional Equations* (Cambridge University Press, Cambridge, 1990).
[9] L. Lopez, Metodi ad un passo fortemente stabili per equazioni integrali di Volterra di seconda specie di tipo stiff, *Calcolo* **XXIII** (1986) 249–263.
[10] R. Peluso and G. Piazza, Numerical methods for second kind Volterra equations (Italian). *Bollettino Un. Mat. Ital.* **4-b** (1985) 155–165.
[11] G. Piazza, Runge–Kutta type methods for solving Volterra equations, (Italian), *Calcolo* **XXI** (1984) 127–149.
[12] A.C. Pipkin, *Lectures on Viscoelasticity Theory*, Applied Math. Sciences, Vol. 7 (Springer, Berlin, 1986).
[13] P.J. van der Houwen and B.P. Sommeijer, Iterated Runge–Kutta on parallel computers, *SIAM J. Sci. Statist. Comput.* **12** (1991) 1000–1028.
[14] P.J. van der Houwen, B.P. Sommeijer and W. Couzy, Embedded diagonally implicit Runge–Kutta algorithms on parallel computers, *Math. Comp.* **58** (1992) 135–159.
[15] P.J. van der Houwen and H.J.J. te Riele, Backward differentiation type formulas for Volterra equations of the second kind, *Numer. Math.* **37** (1981) 205–217.