A class of explicit one-step methods of order two for stiff problems

P. Novati
Università degli Studi dell'Aquila
Dipartimento di Matematica Pura ed Applicata
Via Vetoio, Coppito 67010 - L'Aquila - Italy
E-mail: novati@univaq.it

Abstract

In this paper we introduce a new class of explicit one-step methods of order 2 that can be used for solving stiff problems. This class constitutes a generalization of the two-stage explicit Runge-Kutta methods, with the property of having an A-stability region that varies during the integration in accordance with the accuracy requirements. Some numerical experiments on classical stiff problems are presented.

1 Introduction

Given a function $f : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N$ and a vector $y_0 \in \mathbb{R}^N$, in this paper we deal with the initial value problem (IVP)

$$\begin{cases} y'(t) = f(t, y(t)), & t > t_0, \\ y(t_0) = y_0, \end{cases} \tag{1}$$

As well known, if (1) is stiff, any classical explicit method is able to produce a good approximation of the solution only if the integration step is chosen very small, so that one is usually forced to employ an implicit scheme.

However, the computational complexity of implicit schemes represents a serious drawback if N is large, as could happen when (1) arises from spatial discretization of parabolic PDEs.

In order to overcome such problems, in recent years some authors have proposed various alternatives, such as the use of the so called Runge-Kutta-Chebyshev methods (see e.g . [1], [8], [18]) with the aim of creating explicit integrators with extended stability domains.

Other methods recently proposed are the so called exponential integrators, based on the approximation of the evolution operator $\exp(hA)$, where A is the Jacobian of f and h is the stepsize, by means of a Krylov projection method (see e.g. [3], [6], [7]), a series expansion method ([2], [11], [12], [17]), or an interpolation method ([13]). Such techniques generally performs better than classical explicit and implicit methods but other numerical problems are also introduced by these approaches. The improvement is substantially due to the fact that they are "problem dependent", in the sense that they provide approximations of $\exp(hA)$ (or other related matrix functions) that depends on the spectral property of A, and a good approximation of $\exp(hA)$ allows to solve the stiffness.

In [14] a new one-step explicit method of order 1 for stiff IVPs has been shown to be a promising alternative to the choice of implicit schemes. This method represents a modification of the explicit Euler method, described by the following recursion

$$y_{n+1} = y_n + h_n \phi(h_n, M_n) f(t_n, y_n), \tag{2}$$

where $M_n \in \mathbb{R}^{N \times N}$ is a diagonal matrix and $\phi : \mathbb{R} \times \mathbb{R}^{N \times N} \to \mathbb{R}^{N \times N}$ is defined by

$$\phi(h_n, M_n) := (1 + h_n)(1 + h_n M_n)^{-1}.$$

The main reason that led to the construction of the method (2) for stiff problems is that working with an A-stable method one has more than what is necessary. Indeed, the use of a stepsize arbitrarily large is feasible only from a theoretical viewpoint, because the accuracy requirements generally does not allow this choice. In this sense, the method (2) is clearly not A-stable, but its stability domain depends on h_n and M_n. As shown in [14], the matrix M_n is chosen in order to enlarge or also to restrict the stability domain dynamically during the integration in a close connection with the accuracy requirement.

Actually we must point out that in [14] and also in this paper the matrix M_n can be viewed as a sort of approximation of the Jacobian of f so that the method can be considered connected, in some sense, to the well known W-methods (see e.g. [10] for a wide survey). Anyway, as we shall see, the main difference is that the entries of M_n does not came from approximations of eigenvalues but they are computed by accuracy considerations only.

In this paper we want to extend the ideas of the method (2) in order to create a class of methods of order 2 obtained as modification of the two-stage explicit Runge-Kutta methods. As we shall see the main features of this new class of methods are the same of the method (2). Indeed these methods are able to solve efficiently stiff IVPs without inversions or other matrix function evaluations. Regarding the similarity with the Runge-Kutta-Chebyshev approach, that is the capability of working with extended stability domains without linear algebra difficulties, there is a basic difference: for the Runge-Kutta-Chebyshev the widening of such domains requires the increase of the number of stages, whereas the method here proposed is always two-stage but with changing parameters.

The paper is organized as follows. In Section 2 we define the scaled Runge-Kutta methods of order 2 and in Section 3 we study the linear stability. In Section 4 we describe the algorithm that defines the matrix sequence $\{M_n\}_{n \geq 0}$ and study the asymptotic stability corresponding to this choice. In Section 5 we present some numerical details and the final algorithm that has been used for the numerical experiments of Section 6. In a final section we discuss conclusion and open problems for the method proposed in this paper.

Based on the algorithm of Section 5 we have written a code `heunsc` which can be found at `http://univaq.it/~novati`.

2 The Scaled Runge-Kutta methods of order 2

An explicit two-stage Runge-Kutta method can be written in the following form

$$y_{n+1} = y_n + h\left(c_1 K_1 + c_2 K_2\right), \tag{3}$$

where $c_j \in \mathbb{R}$, $j = 1, 2$, and, as usual,

$$K_1 = f(x_n, y_n), \quad K_2 = f(x_n + a_2 h, y_n + h b_{21} K_1), \tag{4}$$

with $a_2, b_{21} \in \mathbb{R}$. In order to face stiff problems, generalizing the construction of the method (2), the idea is to define $c_j = c_j(h, M) \in \mathbb{R}^{N\times N}$, $j = 1, 2$, as suitable functions of h and M, where $M \in \mathbb{R}^{N\times N}$ is chosen to scale the problem. The coefficients a_2 and b_{21} can be maintained scalar.

Just to understand how to define the c_j's, assume for a moment to work with the scalar test IVP

$$\begin{cases} y'(x) = \lambda y(x), & x > 0, \\ y(0) = 1, \end{cases} \tag{5}$$

where $\lambda \in \mathbb{C}^- := \{z \in \mathbb{C} : \Re(z) < 0\}$, and let $M \in \mathbb{R}$, $M > 0$. We want to define the functions c_j such that the method is of order 2 and such that the stability domain becomes larger for $h \to \infty$. In particular we want the stability function of the corresponding method to be of the form

$$R(h, \lambda, M) := 1 + h\lambda\varphi + \frac{h^2\lambda^2}{2}\varphi^2$$

where $\varphi = \varphi(h, M)$ is a certain function of h and M. In order to have a method of order 2 it is necessary that $\varphi(h, M) \to 1$ for $h \to 0$. Moreover we require that $\varphi(h, M) \to 1/M$ for $h \to \infty$. This leads to the definition

$$\varphi = \varphi(h, M) := \frac{1 + h^2 M}{1 + h^2 M^2}.$$

So doing, as we shall see in the next section, the stability domain of the corresponding method is larger than the stability domain of an explicit Runge-Kutta method of order 2 if $M > 1$.

Going back to the general problem (1), and hence working with $N \geq 1$, we must define

$$\varphi(h, M) := \left(1 + h^2 M\right)\left(1 + h^2 M^2\right)^{-1}, \tag{6}$$

and it is not difficult to prove that the order conditions become

$$\begin{aligned} c_1 + c_2 &= \varphi, \\ a_2 c_2 &= \tfrac{1}{2}\varphi^2. \end{aligned}$$

Fixed $a_2 = \alpha \neq 0$, the general solution is given by the Butcher array

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \alpha & \alpha & 0 \\ \hline & \varphi\left(1 - \frac{1}{2\alpha}\varphi\right) & \frac{1}{2\alpha}\varphi^2 \end{array}$$

With these assumptions, the method takes now the form

$$y_{n+1} = y_n + h\left(\varphi\left(1 - \frac{1}{2\lambda}\varphi\right)K_1 + \frac{1}{2\lambda}\varphi^2 K_2\right) \tag{7}$$

where K_1 and K_2 are defined by (4). We define Scaled Runge-Kutta method any method of type (7).

Clearly, the definition (6) is only one among the possible choices for φ that ensure the preservation of the order and the extension of the A-stability region. Anyway, many numerical experiments revealed the good performance of this choice.

3 Linear stability

In order to understand the requirement $\varphi(h, M) \to 1/M$ for $h \to \infty$ that leads to (6), let us consider the linear stability properties of this method. Consider again the scalar test IVP (5). Applying the method (7) to (5) with $M \in \mathbb{R}$, $M > 0$, and $\varphi : \mathbb{R}^2 \to \mathbb{R}$, the stability function is given by

$$R(h, \lambda, M) := 1 + h\lambda \frac{1 + h^2 M}{1 + h^2 M^2} + \frac{h^2\lambda^2}{2}\left(\frac{1 + h^2 M}{1 + h^2 M^2}\right)^2.$$

Hence, the corresponding A-stability region

$$S(h, M) := \{h\lambda \in \mathbb{C} : |R(h, \lambda, M)| \leq 1\}$$

depends on h and M. Defining with S_2 the stability domain of a classic Runge-Kutta method of order 2, it is easy to see that $S(h, M)$ is an expansion of S_2 of the factor

$$\frac{1 + h^2 M^2}{1 + h^2 M} \geq 1,$$

i.e.,

$$S(h, M) := \left\{z \in \mathbb{C} : z = z' \frac{1 + h^2 M^2}{1 + h^2 M}, z' \in S_2\right\}.$$

In this way $S(h, M)$ tends to the stability domain of a standard Runge-Kutta method of order 2 for $h \to 0$, and, for $h \to \infty$, $S(h, M)$ tends to the region

$$S_M := \{z \in \mathbb{C} : z = z' M, z' \in S_2\}.$$

Just to give an example, if $\lambda = -r$, $r > 0$, setting $M = r$ it is possible to show that we get a method that is asymptotically stable for each $h \lessapprox 2$. Indeed, since $|R(h, -r, r)| \leq 1$ for $hr\varphi \leq 2$, where

$$\varphi = \frac{1 + h^2 r}{1 + h^2 r^2},$$

solving $hr\varphi = 2$ with respect to h, one gets

$$h(r) = q(r)^{1/3} - \frac{\frac{3-4r}{9r}}{q(r)^{1/3}} + \frac{2}{3}, \tag{8}$$

where

$$q(r) = \frac{8r^2 - 9r + 27 + 3\sqrt{3}\sqrt{15r^2 - 17r + 27}}{27r^2}.$$

Analyzing (8), we can see that

$$\begin{aligned} 1.87 &< h(r) < 1.95 \quad for \quad 1 < r < 10, \\ 1.95 &\leq h(r) < 2 \quad for \quad r \geq 10, \\ h(r) &\to 2 \quad for \quad r \to \infty. \end{aligned}$$

4 The definition of M_n

Up to now, the matrix M of the method (7) has been considered constant. Actually, in order to solve efficiently a stiff equation, such matrix must be updated during the integration. In other words, it is necessary to build a matrix sequence $\{M_n\}_{n\geq 0}$ with the properties of scaling the problem reducing in some sense the stiffness.

Among the possibilities, the most immediate consists in approximating the Jacobian of f (as for the Rosenbrock type methods or W-method), or approximating its eigenvalues. However, since we want to avoid such computations our idea is to define a sequence $\{M_n\}_{n\geq 0}$ of diagonal matrices (in this way the computation of $\varphi(h, M)$ does not produce a significant additional cost) without using any information on f. Starting from $M_0 = I_N$ (i.e., starting with an explicit two-stage Runge-Kutta method), we want to define dynamically M_n by monitoring the local error. So doing the procedure will be closely related with the stepsize control technique adopted.

Given y_n, h, M_n, in order to define M_{n+1}, let $\gamma, \beta \in \mathbb{R}$ be such that $\gamma > 1$, $0 < \beta < 1$ and define the temporary matrices $\underline{M}= \beta M_n$ and $\overline{M} = \gamma M_n$. Let now $e(h, \underline{M}) \in \mathbb{R}^N$ and $e(h, \overline{M}) \in \mathbb{R}^N$ be estimates for the local errors at $t_n + h$ produced by the method (7) with $\underline{M}$ and $\overline{M}$, that we denote by $\underline{y}$ and $\overline{y}$. Setting

$$d(h) := \min(\|e(h, \underline{M})\|_\infty, \left\|e(h, \overline{M})\right\|_\infty) \tag{9}$$

suppose that we are using a stepsize control technique such that

$$\|d(h)\|_\infty \approx \delta \tag{10}$$

where δ is a fixed tolerance. In this way we advance with y_{n+1} equal to $\underline{y}$ or $\overline{y}$ (depending on who gets the minimum in (9)).

For $i = 1, 2, ..., N$, let $M^{(i)}$ be the i-th element of the diagonal of a matrix $M \in \mathbb{R}^{N\times N}$, and let $v^{(i)}$ be the i-th element of a vector $v \in \mathbb{R}^N$. We define

$$M_{n+1}^{(i)} := \begin{cases} \max\left(1, \underline{M}^{(i)}\right) & \text{if} \quad \left|e^{(i)}(h, \underline{M})\right| < \left|e^{(i)}(h, \overline{M})\right| \\ \overline{M}^{(i)} \quad \text{if} & \left|e^{(i)}(h, \overline{M})\right| < \left|e^{(i)}(h, \underline{M})\right| \end{cases} \tag{11}$$

In doing so, we create an automatic procedure that can be applied to both linear and nonlinear case that does not require inversion nor eigenvalue approximations but only to apply the method (7) with $\underline{M}$ and $\overline{M}$. This represents the additional cost of the method.

In order to understand how the definition (11) reflects on the asymptotic stability of the method, examine again the scalar test equation (5). Under the hypothesis that h_n has been chosen such that the relation (10) holds, if we define M_{n+1} as stated in (11), we want to understand how large it is possible to choose h_{n+1} in order that

$$|R(h_{n+1}, \lambda, M_{n+1})| \leq 1. \tag{12}$$

Lemma 1 *Given $M > 0$, $a > 0$, there exists q^* such that*

$$\varphi(h, M) = \varphi(q^*h, aM)q^* \tag{13}$$

(φ defined by (6)) with

$$q^* \quad > \quad 1/a \quad \textit{for} \quad a > 1 \tag{14}$$

$$q^* \quad > \quad \frac{1 + h^2M}{1 + h^2M^2} \quad \textit{for} \quad a < 1 \tag{15}$$

Proof. Solving analytically with respect to q the equation $\varphi(h, M) = \varphi(qh, aM)q$ is a difficult task. Hence the idea is to prove that there exists a solution q^* satisfying the inequalities (14) and (15). Let us start with the case $a > 1$. We have

$$\begin{aligned} \frac{1 + (qh)^2 aM}{1 + (qh)^2 (aM)^2} q &< \frac{1 + (qh)^2 a^2M}{1 + (qh)^2 (aM)^2} qa \\ &= \frac{1 + h^2Mx^2}{1 + h^2M^2x^2} x \end{aligned}$$

where $x = qa$. Now

$$\frac{1 + h^2Mx^2}{1 + h^2M^2x^2} x \leq \frac{1 + h^2M}{1 + h^2M^2}$$

for $c < x \leq 1$ where $c < 1$ is a certain quantity depending on h and M. Hence, we have

$$\varphi(qh, aM)q < \varphi(h, M) \quad \text{for} \quad q \leq 1/a. \tag{16}$$

On the other side, it is clear that

$$\lim_{q\to\infty} \frac{1+(qh)^2\,aM}{1+(qh)^2\,(aM)^2}q = +\infty, \tag{17}$$

so that we can say that there exist $q' > 1/a$ such that

$$\varphi(qh, aM)q > \varphi(h, M) \quad \text{for} \quad q \geq q' > 1/a. \tag{18}$$

Since $\varphi(qh, aM)q$ is continuous with respect to q, the inequalities (16) and (18) prove that for $a > 1$ there exist $q^* > 1/a$ that solves (13).

Consider now the case $a < 1$ and assume $M > 1$ (for $M = 1$, as stated in (11) we cannot choose $a < 1$, so that we are in the previous case). If $aM > 1$ we get

$$\frac{1+(qh)^2\,aM}{1+(qh)^2\,(aM)^2}q < \frac{1+(qh)^2\,aM}{1+(qh)^2\,aM}q = q.$$

Hence, choosing

$$q \leq \frac{1+h^2M}{1+h^2M^2}$$

we get

$$\varphi(qh, aM)q < \varphi(h, M).$$

As before, by (17) we can state that for $a < 1$ there exist $q^* \geq \frac{1+h^2M}{1+h^2M^2}$ that solves (13). ■

Clearly, if (13) holds then we have $|R(h, \lambda, M)| = |R(q^*h, \lambda, aM)|$ for each $\lambda \in \mathbb{C}$.

Now suppose that at each step the local error is approximated via Richardson extrapolation by

$$e(h_k, M_k) := C_k \left|\eta(t_k + h_k, h_k, M_k) - \eta(t_k + h_k, h_k/2, M_k)\right|, \tag{19}$$

where $\eta(t_k+h_k, h_k, M_k)$ and $\eta(t_k+h_k, h_k/2, M_k)$ are approximations of $y(t_k+h_k)$ furnished by the method (7) with stepsize h_k and $h_k/2$. We are ready to prove the following.

Proposition 2 *For the scalar test equation (5) assume that at each step*

$$e(h_k, M_k) \leq \delta, \quad k \geq 0,$$

with

$$C_k \geq C > 0, \quad k \geq 0.$$

If

$$|R(h_{n-1}, \lambda, M_{n-1})| \leq 1, \quad n \geq 1, \tag{20}$$

then there exist h_n that satisfies

$$h_n > \frac{2M_{n-1}}{M_n}h_{n-1} = \frac{2}{\gamma}h_{n-1} \quad \textit{if} \quad M_n > M_{n-1}, \tag{21}$$

$$h_n > 2\frac{1+h_{n-1}^2M_{n-1}}{1+h_{n-1}^2M_{n-1}^2}h_{n-1} \quad \textit{if} \quad M_n < M_{n-1}, \tag{22}$$

and such that

$$|R(h_n, \lambda, M_n)| \leq 1.$$

Proof. Since h_n is such that

$$|e(h_n, M_n)| \leq \delta,$$

we have

$$\begin{aligned} |R(h_n, \lambda, M_n) y_n| &= |\eta(t_n + h_n, h_n, M_n)| \\ &\leq |\eta(t_n + h_n, h_n, M_n) - \eta(t_n + h_n, h_n/2, M_n)| + \\ &\quad |\eta(t_n + h_n, h_n/2, M_n)| \\ &\leq \frac{1}{C_n} |e(h_n, M_n)| + \left|R(h_n/2, \lambda, M_n)^2 y_n\right| \\ &\leq \frac{\delta}{C} + |R(h_n/2, \lambda, M_n)|^2 |y_n| . \end{aligned}$$

Now, if q_{n-1} is the solution of $|R(h_{n-1}, \lambda, M_{n-1})| = |R(q_{n-1}h_{n-1}, \lambda, M_n)|$, defining $h_n := 2q_{n-1}h_{n-1}$ we have $|R(h_n/2, \lambda, M_n)| = |R(h_{n-1}, \lambda, M_{n-1})|$ and hence

$$|R(h_n, \lambda, M_n) y_n| \leq \delta + |R(h_{n-1}, \lambda, M_{n-1})| |y_n|$$

Relations (21) and (22) arise from (14) and (15). Finally, since the above argumentations are independent of δ, the proposition is proved. ■

It is important to observe that working with a standard Runge-Kutta method of order 2 the factor C_k in (19) is equal to $1/3$. However, as explained in the next section, for our method such factor has to be taken greater than $1/3$ in order to get better approximation for the local error. In this way the above proposition always holds with $C = 1/3$.

Various numerical experiments (see Section 6) revealed that generally the diagonal components of M_n are increased during the smooth phases of the solution, whereas they are forced to remain closed to 1 during the transient phases (see also [14]). Hence, the typical situation of the stationary phases is represented by formula (21), that is, the asymptotic stability is maintained increasing the stepsize of a factor $2/\gamma$. Since γ must be chosen greater but closed to 1 (usually $1 < \gamma < 1.2$ to avoid numerical instability) the factor $2/\gamma$ allows an effective increase of the stepsize. On the other side, during the transient phase, the stepsize h_n is clearly small, so that

$$\frac{1 + h_n^2 M_n}{1 + h_n^2 M_n^2} \lesssim 1$$

and hence also formula (22) allows to increase considerably the stepsize without loosing the asymptotic stability.

5 Numerical implementation

Regarding the practical implementation of the method (7) with the definition (11), the stepsize control procedure we use is the Predictive Controller (PC) technique (see [4], [5] and [16] for a comprehensive survey), so that the stepsize selection fulfils the formula

$$h_{n+1} = s \left(\frac{\varepsilon}{e_n} \right)^{K_E} \left(\frac{e_{n-1}}{e_n} \right)^{K_P} \frac{h_n^2}{h_{n-1}}, \tag{23}$$

where $e_n = \|e(h_n, M_n)\|_\infty$ is an estimate for the local error, ε is the prescribed tolerance for the local error and s is a safety factor. The exponents K_E and K_P handle the stability of the selection ([16]).

Indeed, when applying an explicit method to a stiff problem, in order to avoid frequent step rejections and instability, it is important to work with a stepsize control procedure stable on the boundary of the stability domain (SC-stable methods [10]). Of course, these considerations apply also to the method here proposed in which the stability domain changes during the integration. Actually, for explicit method the the Proportional Integral (PI) contollers are generally used but since our method allows to enlarge the stepsize with no limitations imposed by the stiffness, the use of the PC controllers is recommended (see also [1]). Many numerical experiments confirmed this choice.

Regarding the local error estimates, in order to ensure the validity of Proposition 2 we use the Richardson extrapolation ([9]). Given h_n, as before let $\eta(t_n + h_n, h_n, M_n)$ and $\eta(t_n + h_n, h_n/2, M_n)$ be the approximations of $y(t_n + h_n)$ furnished by the method (7) with stepsize h_n and $h_n/2$ respectively. Since the method is of order 2 the absolute local error is usually approximated by

$$e_n^* := \frac{1}{3} \|\eta(t_n + h_n, h_n, M_n) - \eta(t_n + h_n, h_n/2, M_n)\|_\infty . \tag{24}$$

However, as mentioned in previous section, for our method the above approximation tends to became poor when the matrix $\left(1 + h_n^2 M_n^2\right)^{-1} \left(1 + h_n^2 M_n\right)$ is far from the identity. It is rather simple to observe this situation working with the scalar test equation (5), where, for a fixed h_n, $e_n^* \to 0$ for $M_n \to \infty$ even if the same is not true for the exact local error. These considerations suggest to modify slightly the formula (24). It is not a simple task to understand which is the best correction but the numerical experiments show that we can improve the error estimate with

$$e(h_n, M_n) := \frac{1}{3} \left| \left(1 + h_n^2 M_n\right)^{-1} \left(1 + h_n^2 M_n^2\right) \left[\eta(t_k + h_k, h_k, M_k) - \eta(t_k + h_k, h_k/2, M_k)\right] \right| \tag{25}$$

and so using $e_n := \|e(h_n, M_n)\|_\infty$ in (23) . This explains the use of the factor C_k in (19).

The following algorithm summarizes the practical implementation of the method for the integration of (1) in the interval $[t_0, t_f]$ with the adaptive construction of the matrix sequence $\{M_n\}_{n \geq 0}$.

Algorithm 3

```
1. given $\varepsilon$, $h_0$, $r$, $M_0 = I_N$, $n := 0$;
2. while $t_n \leq t_f$
    (a) compute $\underline{M}$= $\beta M_n$ and $\overline{M} = \gamma M_n$;
    (b) compute $e(h_n, \underline{M})$ and $e(h_n, \overline{M})$, using (25);
    (c) $e_n = \min(\|e(h_n, \underline{M})\|_\infty, \|e(h_n, \overline{M})\|_\infty)$;
    (d) compute $h_{n+1}$ using (23);
    (e) if $e_n \leq \varepsilon$, then $y_{n+1} := \eta(t_n + h_n, h_n/2, \underline{M})$ (or $y_{n+1} := \eta(t_n + h_n, h_n/2, \underline{M})$);
    (f) if $e_n > \varepsilon$, $h_n := h_{n+1}$ and go to 4;
    (g) define $M_{n+1}$ as in (11);
    (h) $t_{n+1} := t_n + h_n$, $n := n + 1$;
```

In the numerical experiments of the next section we want to test the method constructed with the above algorithm and based on the Butcher scheme

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \varphi\left(1 - \frac{1}{2}\varphi\right) & \frac{1}{2}\varphi^2 \end{array}$$

Such a method, that represents a modification of the well known two-stage explicit Heun method, has been implemented in the MATLAB code `heunsc` (SCaled HEUN method) available at `http://univaq.it/~novati`. The code is written following the format used in the MATLAB ODE suite [15].

The computational cost of `heunsc` is about of 7 function evaluations at each step (accepted or rejected). It is worth noting that with respect to any two-stage explicit Runge-Kutta method based on the Richardson extrapolation the cost is increased of only 2 function evaluations per step. So, the new approach introduce the factor 7/5 in the computational cost.

6 Numerical examples

In the numerical experiments of this section we want to compare the code `heunsc` with the MATLAB stiff solver `ode23s` (a Rosenbrock type method of order 2) and with the non-stiff solver `ode45` (the explicit Runge-Kutta (4,5) pair of Dormand and Prince of order 4). See [15] for details. In all test we adopt the same accuracy requirements.

Example 1. The beginning problems we consider are semilinear equations of the form

$$u' + Au = g(u) \tag{26}$$

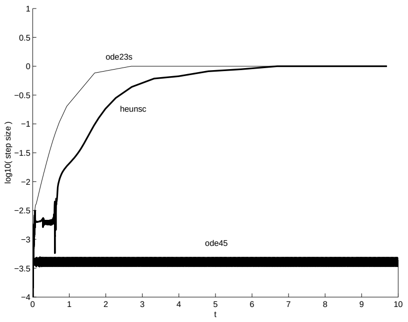

Figure 1: Stepsize curve for Example 1 with g defined by $g^{(i)}(y) = y^{(i)}(1-y^{(i)})$. For `heunsc`, $\gamma = 1.05$, $\beta = 0.95$ and PC.5.8.

where $g : \mathbb{R}^N \to \mathbb{R}^N$ and $A \in \mathbb{R}^{N\times N}$ is a positive definite matrix arising from the discretization of the second order differential operator

$$-\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right), \tag{27}$$

with central differences on a uniform meshgrid of meshsize $h = 1/(n+1)$ on the square $[0,1]\times[0,1]$, with Dirichelet boundary conditions. So doing A is of order $N = n^2$ with real spectrum.

In the numerical tests we are going to present the dimension of the problem is $N = 225$ so that the spectrum is contained in the interval $[-2029, -19]$. We integrate the corresponding (26) in the interval $[0, 10]$ starting with $u_0 = (1, ..., 1)^T$. The `heunsc` method works with $\gamma = 1.05$ and $\beta = 0.95$. Regarding the accuracy, we use $AbsTol = RelTol = 10^{-5}$ for each codes (see [15]) and the maximal admissible stepsize is fixed to $h_{\max} = 1$.

	`heunsc`	`ode23s`	`ode45`
successful steps	457	146	6115
failed attempts	1	0	403
function evaluations	3212	665	39109
partial derivatives	0	1	0
LU decompositions	0	146	0
solutions of linear systems	0	438	0

Tab.1 - Statistics for Example 1 with $g^{(i)}(y) = y^{(i)}(1 - y^{(i)})$

	`heunsc`	`ode23s`	`ode45`
successful steps	482	149	6118
failed attempts	0	0	411
function evaluations	3380	33974	39175
partial derivatives	0	149	0
LU decompositions	0	149	0
solutions of linear systems	0	447	0

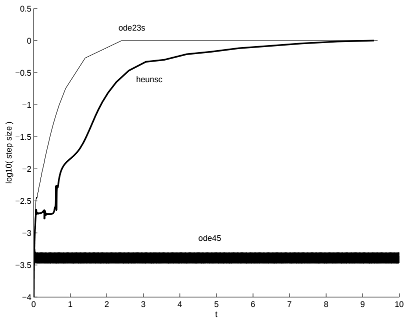

Figure 2: Stepsize curve for Example 1 with g defined by $g^{(i)}(y) = 10(y^{(i)})^4(1 - y^{(i)})$. For `heunsc`, $\gamma = 1.05$, $\beta = 0.95$ and PC.4.7.

Tab.2 - Statistics for Example 1 with $g^{(i)}(y) = 10\left(y^{(i)}\right)^4(1 - y^{(i)})$

Figs. 1 and 2 shows the curves of the stepsize of the three methods with g defined by $g^{(i)}(y) = y^{(i)}(1-y^{(i)})$, and $g^{(i)}(y) = 10\left(y^{(i)}\right)^4(1-y^{(i)})$, $i = 1, ..., N$, respectively. For these experiments the `heunsc` method works with the stepsize controller PC.5.8 (formula (23) with $K_E = 0.5$, $K_P = 0.8$) and PC.4.7 ($K_E = 0.4$, $K_P = 0.7$) respectively. Looking at the pictures we observe that the `heunsc` method actually behaves like a stiff solver, since it is able to detect where the solution becomes smooth allowing the increase of the stepsize. In Tabs. 1 and 2 we can also observe the computational statistics for this problem.

The comparison between `heunsc` and `ode45` needs no comments for both experiments even considering the computational cost. With respect to `ode23s` it is quite clear that even if `heunsc` requires more steps to complete the integration, it is actually cheaper. In the second experiment (Fig.2) it is clear because `ode23s` needs to update the Jacobian at each step. For the first one (Fig.1), it is sufficient to observe that since the bandwidth of A is $\sqrt{N}$, an LU factorization costs about N^2 scalar multiplications whereas an application of this matrix costs about $5N$ scalar multiplications because of its sparsity pattern. Of course, the difference of cost grows increasing the dimension of the problem.

Example 2. In order to observe the capabilities of the `heunsc` method on a classical stiff problem, we consider the Van der Pol equation

$$\begin{aligned} u' &= v \\ v' &= \mu(1-u^2)v - u \end{aligned}$$

with starting values $u(0) = 2$, $v(0) = 0$. Setting $\mu = 500$, we integrate this problem from 0 to 500 using $AbsTol = RelTol = 10^{-5}$. The `heunsc` method works with $\gamma = 1.15$, $\beta = 0.85$ and PC.3.6. In Fig.3 the stepsizes chosen by the three methods are plotted whereas the statistics are reported in Table 3.

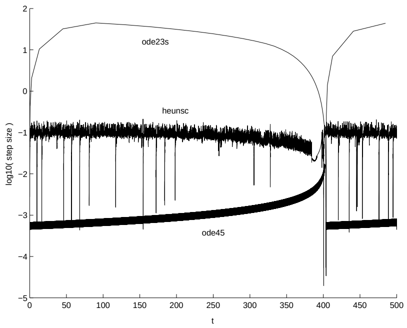

Figure 3: Stepsize curve for Example 2, the Van der Pol equation. For `heunsc`, $\gamma = 1.15$, $\beta = 0.85$ and PC.3.6.

	`heunsc`	`ode23s`	`ode45`
successful steps	7277	482	148520
failed attempts	118	2	9899
function evaluations	51771	2416	950515
partial derivatives	0	482	0
LU decompositions	0	484	0
solutions of linear systems	0	1452	0

Tab.3 - Statistics for Example 2, the Van der Pol equation.

The method is not competitive with `ode23s` because the solution is never so smooth to allow the increasing of the stepsize as in Example 1. Moreover, since this is a two-dimensional problem the additional cost of the stiff solver is without importance. However the comparison with the performance of the higher-order `ode45` shows the effectiveness of the scaled approach.

Example 3. In this final example we consider the two-dimensional Brusselator problem [7], whose equations are

$$\begin{aligned} \frac{\partial u}{\partial t} &= 1 + u^2 v - 4u + \alpha \Delta u, \\ \frac{\partial v}{\partial t} &= 3u - u^2 v + \alpha \Delta v, \end{aligned}$$

for $0 \leq x, y \leq 1$, with Neumann boundary conditions and initial condition given by

$$\begin{aligned} u(x,y,0) &= 3(1-x)^2 e^{-x^2-(y+1)^2} - 10\left(\frac{x}{5} - x^3 - y^5\right) e^{-x^2-y^2} - \frac{1}{3} e^{-(x+1)^2-y^2}, \\ v(x,y,0) &= 0, \end{aligned}$$

i.e., $u(x,y,0)$ is given by the MATLAB's `peaks` function.

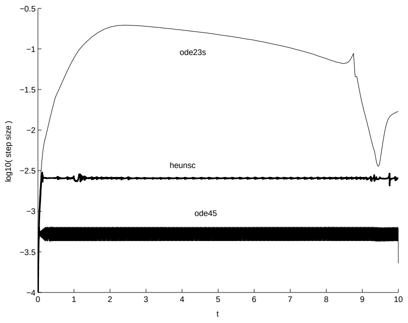

Figure 4: Stepsize curve for Example 3, the Brusselator problem. For `heunsc`, $\gamma = 1.05$, $\beta = 0.95$ and PC.4.7.

Using central differences for the Laplacian on a uniform 15 × 15 grid the dimension of the resulting IVP is $N = 450$ and we integrate it from 0 to 10. The parameter α controls the stiffness of the problem and we set $\alpha = 1$ (setting for instance $\alpha = 0.02$ we get a so called mildy-stiff problem as considered in [7]). In Fig.4 the curves of the accepted stepsizes are plotted, with accuracy $AbsTol = RelTol = 10^{-5}$. The `heunsc` method works with $\gamma = 1.05$, $\beta = 0.95$ and the stepsize selector PC.4.7. In Table 4 the statistics for this example are shown.

	`heunsc`	`ode23s`	`ode45`
successful steps	4071	375	4740
failed attempts	0	1	310
function evaluations	28503	169879	30301
partial derivatives	0	375	0
LU decompositions	0	376	0
solutions of linear systems	0	1128	0

Tab.4 - Statistics for Example 3, the Brusselator problem.

Although the stiff solver `ode23s` completes the integration in a relative small number of steps, its computational cost is much larger than the one of `heunsc`. This means that `heunsc` is again an effective alternative to the standard stiff solvers (further, we must observe that also `ode45` shows a cheap performance).

However, looking at Fig.4 we can observe that, contrary to Example 1, the stepsize selection for the `heunsc` does not allow to increase the stepsize over a certain threshold that for this example was about 0.0026. This phenomena can be explained looking again at the scalar test equation (5). Choosing as example $\lambda = -1000$ in Fig.5 we can observe the relation between the exact local error (starting from $y(0) = 1$)

$$LE = \left| e^{\lambda h} - \eta(h, h, M) \right|$$

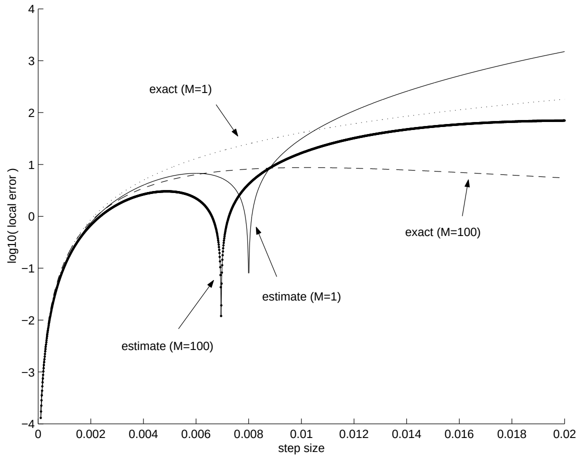

Figure 5: Local error, exact and estimated with $M = 1$ and $M = 100$ for the scalar test IVP.

and its estimate given by (25) with $M = 1$ and $M = 100$.

In Fig.5 it is interesting to observe a critical zone attained for a value of h greater than 0, i.e., an interval where the estimate is very poor because it presents a zero. For $M = 1$, that is, working with a standard two-stage explicit Runge-Kutta method of order 2, such interval is a neighbor of $h = -8/\lambda$ (it is not difficult to verify that), but such value is never reached because the boundary of the stability region is attained for $h = -2/\lambda$. On the contrary, for our method this interval is a very trap. In fact, the method is not able to increase the stepsize because of the bad local error estimate that causes instability in the stepsize selection. The only way for escaping from this situation appears when the solution is smooth enough that the local error is also very closed to 0 (as for Example 1). This explains the difference between the behavior of the method in Example 1 and Example 3. Indeed in Example 3 the solution is never so smooth to overcome this problem. Anyway such phenomena is present in all examples here considered. It is visible looking at the instability of the stepsize curve in Fig.1 for small value of t and in Fig.3 almost everywhere. A robust solution of this problem would be the use of an embedded approach but at the moment this is still not available.

7 Conclusions

The numerical experiments presented in the paper show that the `heunsc` method can be used to solve stiff problems especially when the solution is smooth after the transient phases, because it is automatically able to detect such situation allowing the growth of the stepsize. Indeed, the stability domain varies dynamically with the solution: it is large where the solution is smooth, and it collapses to the region of a classical explicit two-stage Runge-Kutta method during the transient phases, in a close connection with the accuracy requirements. Since the method is explicit, the most important feature with respect to other stiff solvers regards the computational cost together with the capability of exploiting the possible sparsity structure of the problem.

However, there are also some open problems for improving the capabilities of the methods here proposed.

1. The definition of γ and β for updating the matrix sequence $\{M_n\}_{n\geq 0}$ in our numerical experiments has not a solid theoretical base, in the sense that it must be possible to find out a more closed connection between such constants and the local error that would allow to get a more efficient scaling sequence $\{M_n\}_{n\geq 0}$.

2. The definition of the function φ is only one among the possible and it is still not clear if some other choices would allow better performances.

3. As we said at the end of previous section, we would like to dispose of an embedded method in order to avoid stepsize selection instability.

All these problems are now under study because we are convinced that these method can constitute an effective alternative to the classical stiff solvers, especially for large dimensional problems. The basic point is that when solving a certain problem the accuracy requirement introduces an upper bound for the feasible stepsize. Hence, using a classical implicit method, the possibility of having arbitrary large stepsize is not so essential, and the computational cost is very high.

References

[1] A. Abdulle, *Fourth order Chebyshev method with recurrence relation*, SIAM J. Sci. Comput. 23 (2002), pp. 2041-2054.

[2] L. Bergamaschi and M. Vianello, *Efficient computation of the exponential operator for large, sparse, symmetric matrices*, Numer. Linear Algebra Appl., 7 (2000), pp. 27-45 .

[3] E. Gallopoulos and Y. Saad, *Efficient solution of parabolic equations by Krylov approximation methods*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 1236-1264.

[4] K. Gustafsson, M. Lundh and G. Söderlind, *A PI stepsize control for the numerical solution of ordinary differential equations*, BIT 28 (1988), pp. 270-287.

[5] K. Gustafsson, *Control theoretic techniques for stepsize selection in explicit Runge-Kutta methods*, ACM TOMS 17 (1991), pp. 533-554.

[6] M. Hochbruck and C. Lubich, *On Krylov subspace approximations to the matrix exponential operator*, SIAM J. Numer. Anal., 34 (1997), pp. 1911-1925.

[7] M. Hochbruck, C. Lubich and H. Selhofer, *Exponential integrators for large systems of differential equations*, SIAM J. Sci. Comput., 19 (1998), pp. 1552-1574.

[8] P.J. van der Houwen and B.P. Sommeijer, *On the internal stability of explicit m-stage Runge-Kutta methods for large values of m*, Z. Angew. Math. Mech. 60 (1980), pp. 479-485.

[9] E. Hairer, S.P. Norsett and G. Wanner, *Solving ordinary differential equations. I,* 2nd ed., Springer-Verlag, Berlin, 1993.

[10] E. Hairer and G. Wanner, *Solving ordinary differential equations. II,* 2nd ed., Springer-Verlag, Berlin, 1996.

[11] I. Moret and P. Novati, *The computation of functions of matrices by truncated Faber series*, Numer. Func. Anal. and Optimiz., 22 (2001), pp. 697-719.

[12] P. Novati, *Solving linear initial valuue problems by Faber polynomials*, Numer. Linear Algebra Appl., 10 (2003), pp. 247-270.

[13] P. Novati, *A polynomial method based on Fejer points for the computation of functions of unsymmetric matrices*, Appl. Numer. Math., 44 (2003),.pp. 201-224.

[14] P. Novati,.*An explicit one-step method for stiff problems*, Computing 71 (2003), pp. 133-151.

[15] L. F. Shampine and M. W. Reichelt, *The MATLAB ODE Suite*, SIAM J. Sci. Comput., 18 (1997), pp. 1-22.

[16] G. Söderlind, *Automatic control and adaptive time-stepping*, Numerical Algorithms 31 (2002), pp.281-310.

[17] H. Tal-Ezer, *Spectral methods in time for parabolic problems*, SIAM J. Numer. Anal., 26 (1989), pp. 1-11.

[18] J.G. Verver, *Explicit Runge-Kutta methods for parabolic partial differential equations*, Appl. Numer. Math. 22 (1996), pp. 359-379.