Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

B.P. Sommeijer

Increasing the real stability boundary of explicit methods

Department of Numerical Mathematics Report NM-R8902 January

Increasing the Real Stability Boundary of Explicit Methods

B.P. Sommeijer
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

Based on the simplest well-known integration rules (such as the forward Euler scheme and the 'classical' Runge-Kutta method), an extension is proposed to enlarge the real stability boundary. The main characteristic of the resulting schemes is that the computational complexity is hardly increased.

1980 *Mathematics Subject Classification* : 65M10, 65M20.
Key Words and Phrases: numerical analysis, stiff differential equations, stability, explicit Runge-Kutta methods.

1. INTRODUCTION

In solving a parabolic partial differential equation (PDE), a widely used approach is to follow the 'method of lines' [8]. This means that first the spatial differential operators are replaced by e.g. finite difference or finite element approximations; this part of the solution process is usually termed the *semi-discretization.* Then, the time-continuous system of ordinary differential equations (ODEs) is *integrated in time* by some numerical integration method. Following this technique, several choices have to be made: for the semi-discretization as well as for the time integration one has to select a particular method and, for both discretizations, the size of the meh one is going to use. We will briefly discuss these choices:

(i) the particular way in which the *semi-discretization* is performed. Finite difference and finite element approximations are frequently used. As it does not fit in with the scope of this paper, we do not go into detail about these techniques (the reader is referred to [8] and [10] for a comprehensive discussion of these methods). Suffice to say that one could be led by the shape of the spatial domain (finite element approximations are more natural when the domain is irregular) and, of course, by one's personal preference and familiarity with (one of) the methods.

(ii) a second choice to be made, and which is of direct impact to the contents of this paper, is the *size of the spatial mesh* used in the semi-discretization. Usually, one will choose a mesh that is fine enough to represent adequately the spatial variation of the solution one is expected to find. This is, of course, a bit vague; firstly, because it is not known a priori what this spatial variation will be and secondly, it is hard to say what the relation is between the spatial meshsize and the accuracy of the final solution. Therefore, one has to make a guess about the mesh and, eventually, solve the problem once more on a different mesh. Then, a comparison of both solutions will give an indication about the

Report NM-R8902
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

accuracy. What we mean by 'represent adequately the spatial variation' depends, of course, on the specific problem. In a PDE context, a modest accuracy is usually sufficient for practical purposes. However, even for these low-accuracy demands (i.e., a relatively coarse mesh), the resulting system of ODEs will possess some *stiffness*; moreover, if the meshsize used in the semi-discretization becomes smaller, then the stiffness of the system increases. For a discussion on stiff systems of ODEs, we refer to [7]. It is this stiffness which has a serious effect on the third choice we have to make:

(iii) *the time integration method.* Typically, in almost all scientific subroutine libraries, like NAG and IMSL, parabolic equations (and stiff problems in general) are integrated in time by means of *implicit methods.* This approach is nowadays generally accepted as the most efficient one. The reason for the success of these methods is that they can efficiently cope with the stiffness of the ODE; or, more precisely, in applying a suitable implicit method, the time step can be chosen independent of the stiffness and therefore is merely determined by accuracy considerations. This is in sharp contrast with *explicit methods*, where the stiffness will impose a severe restriction on the time step. If this condition is disobeyed, then the resulting time integration process is *unstable*, which means that rounding errors will be amplified in each step. This amplification can be so drastic that already after a small number of steps the solution is completely destroyed by these amplified errors. Hence, in choosing an explicit method it is necessary to satisfy this stringent condition on the time step. This brings us to the fourth choice:

(iv) *the time integration stepsize.* As mentioned before, the choice of this parameter strongly depends on the integration method. In using an explicit method, we will encounter a stepsize restriction for stability reasons which forces us to take unrealistically small time steps. Implicit methods on the other hand, does not impose such a restrictive condition.

After the above discussion, one might ask : 'is there any room for explicit methods in integrating parabolic equations ?' Every day's experience says : 'there is', because people continue to use these methods. Most likely, the reason for their popularity is the *simplicity* of these methods. Evidently, this simplicity is of decisive importance if one has to program the time integrator oneself. In case no standard software is available, the reasoning will then be : 'in the time I need to program the implicit method, the computer has already solved my problem using a lot of explicit time steps'. Another point may be the storage requirements. Since explicit methods usually need considerably less storage than implicit methods, especially for higher-dimensional parabolic PDEs, this could be another motivation to choose an explicit integrator. However, one should realize that when standard software is available and can be called as a 'black box', and when storage is no problem, then the algorithms based on implicit methods are the best choice because they are more efficient in terms of computer time.

If however, one has reasons to choose for an explicit method, for example because of the abovementioned reasons, then the stepsize restriction will be encountered and very small time steps are necessary to obtain stability. A consequence of such small time steps is that, usually, the error in the time integration process will be much smaller than the error due to the semi-discretization. Recall that the spatial grid was chosen on the basis of modest accuracy requirements. Hence, the overall error is completely determined by the semi-discretization error, i.e., the difference between the ODE solution and the solution of the PDE. In other words, we cannot take profit, in terms of accuracy, from the small time steps. Therefore, one would like to have a mechanism by which accuracy and stability can be exchanged. That is, we would like to take larger time steps (to reduce the number of steps and thus the amount of computer time) and we are willing to sacrifice the accuracy of the time integration. Technically spoken, we need explicit methods with a larger stability boundary.

In the literature, several of such methods have been proposed. In [1], Du Fort and Frankel introduce a method for the *linear* 1-*dimensional* PDE $\partial u/\partial t = \partial^2 u/\partial x^2$, which can be regarded as being explicit but nevertheless unconditionally stable. However, this method has the disadvantage that, for reasons of convergence, the time step should tend faster to zero than the meshsize of the spatial grid. If this condition is violated, then the numerical solution converges to the solution of a different PDE (see also [9, p.176] for a discussion of this method). Zlatev and Østerby [11], and more generally, Jeltsch and Nevanlinna [5] discuss *explicit linear multistep methods* containing free parameters which can be utilized to enlarged the stability boundary. These methods possess the difficulty that the 'scaled error constant' (which determines to a large extent the accuracy) deteriorates if the parameters are chosen to generate a large stability boundary. However, similar to our aim, these methods offer the possibility to exchange accuracy and stability.

These deficiencies are not encountered in the stabilized Runge-Kutta methods described in van der Houwen [3] and in the Generalized Predictor-Corrector methods of van der Houwen and Sommeijer [4]. These schemes allow arbitrary time steps, but here the price is that many evaluations of the right-hand side of the ODE are required per step to maintain stability. Moreover, these methods are more complicated than the ones we are aiming at in this paper.

After this introduction we are able to formulate our aim: in this paper we will construct methods based on the simplest well-known integration rules (like the forward Euler method or 'the classical' Runge-Kutta method), and which allow for considerably larger time steps. The resulting methods, which are almost of the same simplicity as the basic methods, exhibit a time integration error that is in better accordance with the semi-discretization error. Similar to the methods described in [5] and [11], these schemes contain a parameter. The effect of increasing this parameter is that both the stability boundary as well as the 'inaccuracy' is enlarged. Finally, we remark that the schemes we will describe are mainly based on Runge-Kutta methods, and therefore do not fall in the class of (linear multistep) methods considered in [5] and [11].

In Section 2, the general idea is outlined and in four subsections methods of orders 1 up to 4 are constructed and analysed. Numerical results and comparisons are presented in Section 3 and finally, in Section 4, some conclusions are formulated.

2. GENERAL IDEA OF THE METHOD

As we have seen, the severe stability condition forces us to take time steps which are so small that, in most cases, the time integration error is negligible with respect to the semi-discretization error. Therefore, we propose the following modification to proceed the solution over a large time step:

> First, we 'bridge' a substantial part of this step by extrapolating approximations obtained in previous step points.
>
> Next, we apply a simple explicit integrator to perform the actual integration over the remaining (small) part of the step (see also Fig. 2.1).

Since a substantial part of this step is 'integrated' by a simple extrapolation, which is a very crude way to advance the solution of an ODE, we cannot expect the result in the end point of the step to be very accurate. Hence, in this part of the solution process we have sacrificed accuracy.

As a consequence of this extrapolation, the integrator is now applied to the remaining small part of the step; therefore, its 'effective' step is much smaller. This results in a less stringent stability condition. Hence, in this part of the solution process we have recovered stability and thus we have exactly obtained what we are aiming at. We will now give a definition of the scheme.

Suppose we are given the initial value problem

$$\frac{d}{dt}y(t) = f(t, y(t)), \qquad t \geq t_0, \qquad y(t_0) = y_0. \tag{2.1}$$

Let us define an equidistant grid on the t-axis by $t_n := t_0 + n\cdot h$, $n=1,2,\ldots$, where h is the integration step and let y_n denote a numerical approximation to the exact solution y(t) at $t=t_n$. Furthermore, let R(h;t,y) denote a one-step method to advance the solution over a step h, starting with the approximation y at time t.

Now, for n=k,k+1,... we consider the method

$$y_\mu^* = \sum_{j=0}^{k} a_j(\mu)\, y_{n-k}, \tag{2.2a}$$

$$y_{n+1} = R\,((1-\mu)h;\, t_n + \mu h,\, y_\mu^*), \tag{2.2b}$$

where μ is real number in [0,1). In (2.2a) we calculate y_μ^*, which is an approximation to $y(t_n+\mu h)$. The coefficients $a_j(\mu)$ are used to obtain a certain *order* of accuracy. In (2.2b) the integrator R starts in $t_n+\mu h$ with the approximation y_μ^* and integrates over the interval $[t_n+\mu h, t_{n+1}]$. Notice that a classical step-by-step application of R (i.e., μ=0) would result in the process $y_{n+1} = R(h;t_n,y_n)$, n=0,1,... .

In the sequel we will consider methods in which the extrapolation as well as the method R are both of order p, resulting in order p for the overall method (2.2). A straightforward Taylor-expansion reveals that the extrapolation is of order p=k if the coefficients $a_j(\mu)$, j=0,...,k satisfy the linear system

$$\begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 0 & 1 & 2 & 3 & \cdots & k \\ 0 & 1 & 2^2 & 3^2 & \cdots & k^2 \\ \cdot & \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & \cdot & & \cdot \\ 0 & 1 & 2^k & 3^k & \cdots & k^k \end{pmatrix} \begin{pmatrix} a_0(\mu) \\ a_1(\mu) \\ a_2(\mu) \\ \cdot \\ \cdot \\ a_k(\mu) \end{pmatrix} = \begin{pmatrix} 1 \\ -\mu \\ \mu^2 \\ \cdot \\ \cdot \\ (-1)^k\mu^k \end{pmatrix}. \tag{2.3}$$

In the following subsections we derive methods of increasing orders of accuracy. For the one-step method R we will throughout choose a pth-order, p-stage Runge-Kutta (RK) method.

2.1 FIRST-ORDER METHODS

Let us first consider the case k=1. The system (2.3) reduces to

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} a_0(\mu) \\ a_1(\mu) \end{pmatrix} = \begin{pmatrix} 1 \\ -\mu \end{pmatrix} \tag{2.3'}$$

and we find the first-order extrapolation

$$y_\mu^* = (1+\mu)\, y_n - \mu\, y_{n-1}. \tag{2.4a}$$

Let us combine this with the first-order forward Euler-method, which is the most simple Runge-Kutta scheme, to obtain

$$y_{n+1} = y_\mu^* + (1-\mu)\, h\, f(t_n+\mu h,\, y_\mu^*). \tag{2.4b}$$

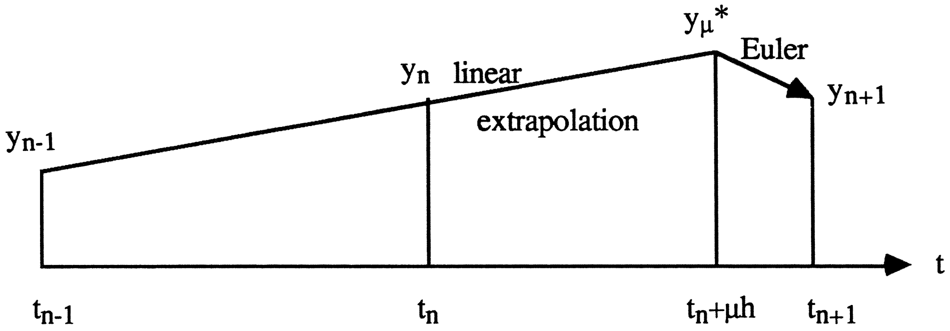

FIGURE 2.1. The method (2.4)

To start with, we will derive the local truncation error of the method (2.4), following the definition given by Lambert [6]. It is readily verified that this error is given by

$$y(t_{n+1}) - y_{n+1} = C_2\, h^2\, y''(t_n) + O(h^3), \qquad C_2 = [\tfrac{1}{2} - \tfrac{1}{2}\mu(1-2\mu)]. \tag{2.5}$$

From this expression we see that the method is first-order consistent. Furthermore, if $\mu \to 1$ the error constant C_2 tends to 1. However, for linear multistep methods, Henrici defines a 'scaled error constant' [2, p.223], by dividing the usual error constant by the sum of the coefficients in front of the derivative function f. Following Henrici's definition, we would obtain for the method (2.4) the scaled error constant $C_2/(1-\mu)$, which increases if $\mu \to 1$. This has, of course, consequences for the accuracy (see also the discussion on zero-stability, below). Since we are willing to accept a loss of accuracy in the time integration process as far as stability is recovered, this behaviour of the error constant does not need to be disastrous, as we shall see in the experiments.

Next, we will study the *stability*. Applying the scheme (2.4) to the linear test equation $y'=\lambda y$, $\lambda \in C$, we obtain the characteristic equation

$$\zeta^2 - P((1-\mu)z)\,(1+\mu)\,\zeta + P((1-\mu)z)\,\mu = 0, \tag{2.6}$$

where $z:=h\lambda$ and P denotes the stability polynomial of Euler's method. As usual (see [6]), the *stability region* is defined by the set of (complex) z-points for which the roots ζ of (2.6) are inside the unit circle and the method is said to be stable for a particular z-value if this z belongs to its stability region. The *real stability boundary* β is defined as the length of the largest interval $(-\beta,0)$ on the negative real axis which is still contained in the stability region. The following analysis is restricted to *real* values of z; hence, we will investigate what values can be obtained for the real stability boundary β.

Applying the Hurwitz-conditions (see [6, p.80]) to (2.6) reveals that the method is stable if

$$\frac{-1}{1+2\mu} < P((1-\mu)z) < 1. \tag{2.7}$$

Since for Euler's method $P(z)=1+z$, we easily find that z has to satisfy

$$-\beta(\mu) := \frac{-2(1+\mu)}{(1+2\mu)(1-\mu)} < z < 0. \tag{2.8}$$

From this expression we see that the stability boundary β can be made arbitrarily large by choosing μ sufficiently close to 1. For such μ-values, we have $\beta \approx \frac{4}{3(1-\mu)}$. For practical purposes this means that, given a certain problem (i.e., a given λ-value) and a certain stepsize h (determined on the basis of accuracy) we choose μ such that $z=h\lambda$ satisfies (2.8). For two values of μ we have plotted the stability *region* in the complex z-plane. These plots can be found in Appendix II.

Finally, since we are dealing with a *multistep* method, we discuss the concept of *zero-stability* (see [6]). Therefore we substitute P≡1 into (2.6) and require the roots to be on the unit disc, and those on the unit circle to be simple. We find

$$\zeta^2 - (1+\mu)\zeta + \mu = (\zeta - 1)(\zeta - \mu) = 0, \tag{2.6'}$$

and clearly the method is zero-stable for all μ in $[0,1)$. Here, we remark that the concept of zero-stability is closely related to the scaled error constant, mentioned above. We see from (2.6') that if $\mu \rightarrow 1$, the spurious root $\zeta=\mu$ converges to the so-called principal root $\zeta=1$. According to the definition, this means that the method tends to a zero-*un*stable method for $\mu \rightarrow 1$, which in turn results in an increasing scaled error constant.

We collect the above results in the following theorem:

Theorem 2.1. *The method* (2.4) *is first-order consistent and zero-stable for all* $\mu \in [0,1)$; *its real stability boundary is given by* $\beta(\mu) = \dfrac{2(1+\mu)}{(1+2\mu)(1-\mu)}$ □

2.2 SECOND-ORDER METHODS

Next we study what is possible for k=2. The system (2.3) reduces to

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} a_0(\mu) \\ a_1(\mu) \\ a_2(\mu) \end{pmatrix} = \begin{pmatrix} 1 \\ -\mu \\ \mu^2 \end{pmatrix}, \tag{2.3''}$$

and is solved by

$$a_0(\mu) = \tfrac{1}{2}(\mu+1)(\mu+2), \quad a_1(\mu) = -\mu(\mu+2), \quad a_2(\mu) = \tfrac{1}{2}\mu(\mu+1). \tag{2.9}$$

Hence we start with the second-order extrapolation

$$y_\mu{}^* = \tfrac{1}{2}(\mu+1)(\mu+2)\,y_n - \mu(\mu+2)\,y_{n-1} + \tfrac{1}{2}\mu(\mu+1)\,y_{n-2}. \tag{2.10a}$$

To integrate the remaining part of the step we select the second-order *improved Euler method* [6,p.119], resulting in

$$y_{n+1} = y_\mu{}^* + \tfrac{1}{2}(1-\mu)\,h\,[\,f_\mu{}^* + f(t_{n+1}, y_\mu{}^*+(1-\mu)hf_\mu{}^*)\,] \tag{2.10b}$$

with

$$f_\mu{}^* = f(t_n+\mu h, y_\mu{}^*).$$

Obviously, there are many possible choices for the integrator within the class of second-order, 2-stage RK methods. Any other choice is conceivable as well. Moreover, for $p \leq 4$, any pth-order, p-stage RK method has the same stability region; hence, its choice does not influence the linear stability analysis of the total scheme (2.10). On the other hand, the particular choice of the integrator does have influence on the local truncation error and it might be possible to make a better selection with respect to accuracy. However, as it is our intention to describe an algorithm based on simple and well-known RK methods, we selected the improved Euler method.

Again following Lambert's definition, a straightforward Taylor-expansion of the method (2.10) yields

$$y(t_{n+1}) - y_{n+1} = h^3 \left(C_{31}\, f_y{}^2 f + C_{32}\, f_{yy} f^2 \right) + O(h^4), \tag{2.11a}$$

where the error constants are given by

$$C_{31} = \frac{1}{6}(6\mu^2-\mu+1) \qquad \text{and} \qquad C_{32} = \frac{1}{12}(3\mu^3+3\mu^2+7\mu-1). \tag{2.11b}$$

Hence, the method is of second-order accuracy and we see that the local truncation error tends to $h^3\, y'''(t_n) + O(h^4)$ if $\mu \to 1$.

The characteristic equation associated with (2.10) is given by

$$\zeta^3 - P((1-\mu)z)\, a_0(\mu)\, \zeta^2 - P((1-\mu)z)\, a_1(\mu)\, \zeta - P((1-\mu)z)\, a_2(\mu) = 0, \tag{2.12}$$

where, again, P denotes the stability polynomial of the RK method, viz. $P(z)=1+z+\frac{1}{2}z^2$, and the coefficients $a_j(\mu)$ are given in (2.9). Application of the Hurwitz-conditions reveals that the method is stable if

$$\frac{-1}{2\mu^2+4\mu+1} < P < 1 \tag{2.13a}$$

and

$$\mu^3 + 2\mu^2 - 2 < 0. \tag{2.13b}$$

The first condition is fulfilled if

$$-\beta(\mu) := -\frac{2}{1-\mu} < z < 0\,; \tag{2.14a}$$

however, to satisfy the second constraint, we should restrict μ to the interval

$$0 \leq \mu < 0.839\ldots\,. \tag{2.14b}$$

Combining both results shows that the optimal real stability boundary is obtained by choosing μ as large as allowed. However, from our tests it appeared that it is advisable to choose μ slightly smaller than maximally allowed; therefore, in the experiments in Section 3 we will use $\mu=0.825$, resulting in $\beta\approx 11.43$. Since the stability boundary of the underlying improved Euler method equals 2, a gain factor of approximately 5.7 is possible. The stability *region* for $\mu=0.825$ is plotted in Fig. A3 (see Appendix II). It turned out that for smaller values of μ the typical shape of this stability region is preserved, although less far extended along the negative real axis.

Next we consider the zero-stability of (2.10). Substitution of $P\equiv 1$ into (2.12) gives

$$(\zeta - 1)\,(\zeta^2 - \tfrac{1}{2}\mu\,(\mu+3)\,\zeta + \tfrac{1}{2}\mu\,(\mu+1)) = 0, \qquad (2.12')$$

from which it can be deduced (for example, by applying the Hurwitz-conditions to the quadratic factor) that the method is zero-stable for all $\mu\in[0,1)$.

Finally we remark that, for $\mu=0.825$, the roots of (2.12') are given by $\zeta=1$ and $\zeta\approx 0.7889 \pm 0.3612\cdot i$; thus, at the origin, the spurious roots are approximately 0.8677 in modulus.

To complete this subsection, we summarize the results in:

Theorem 2.2. *The method* (2.10) *is second-order accurate and zero-stable for all* $\mu\in[0,1)$; *its real stability boundary is given by* $\beta(\mu) = \frac{2}{1-\mu}$, *where* μ *is restricted to the interval* [0,0.839...] []

2.3 THIRD-ORDER METHODS

Proceeding as in the previous subsections, we start with solving the system (2.3). For k=3 we obtain

$$\begin{aligned} a_0(\mu) &= \tfrac{1}{6}(\mu+1)(\mu+2)(\mu+3), & a_1(\mu) &= -\tfrac{1}{2}\mu(\mu+2)(\mu+3),\\ a_2(\mu) &= \tfrac{1}{2}\mu(\mu+1)(\mu+3), & a_3(\mu) &= -\tfrac{1}{6}\mu(\mu+1)(\mu+2). \end{aligned} \qquad (2.15a)$$

Hence, these coefficients define the third-order extrapolation, which will be combined with a third-order, 3-stage RK method. Among the many possible choices we select *Kutta's third-order method* (cf. [6, p.120]). Applied to the interval $[t_n+\mu h,\ t_{n+1}]$, this method reads

$$\begin{aligned} y_{n+1} &= y_\mu{}^* + \tfrac{1}{6}(1-\mu)\,h\,[\,k_1 + 4k_2 + k_3\,],\\ k_1 &= f(t_n+\mu h,\ y_\mu{}^*),\\ k_2 &= f(t_n+\tfrac{1}{2}(1+\mu)h,\ y_\mu{}^*+\tfrac{1}{2}(1-\mu)hk_1),\\ k_3 &= f(t_{n+1},\ y_\mu{}^*-(1-\mu)hk_1+2(1-\mu)hk_2). \end{aligned} \qquad (2.15b)$$

The characteristic equation associated with the method (2.15) reads

$$\zeta^4 - P((1-\mu)z)\, a_0(\mu)\, \zeta^3 - P((1-\mu)z)\, a_1(\mu)\, \zeta^2 - P((1-\mu)z)\, a_2(\mu)\, \zeta - P((1-\mu)z)\, a_3(\mu) = 0. \qquad (2.16)$$

Applying the Hurwitz-conditions to this equation would result in too complicated inequalities to be treated analytically. Therefore, we resorted to a numerical approach.

Considering P for the moment as an independent variable, we calculated for a large number of μ-values $\in [0,1)$ the bounds on P, such that the roots of (2.16) are within the unit circle. The upper bound is given by P=1 and the lower bound is a μ-dependent curve, $\gamma(\mu)$ say, which is plotted in Figure 2.2.

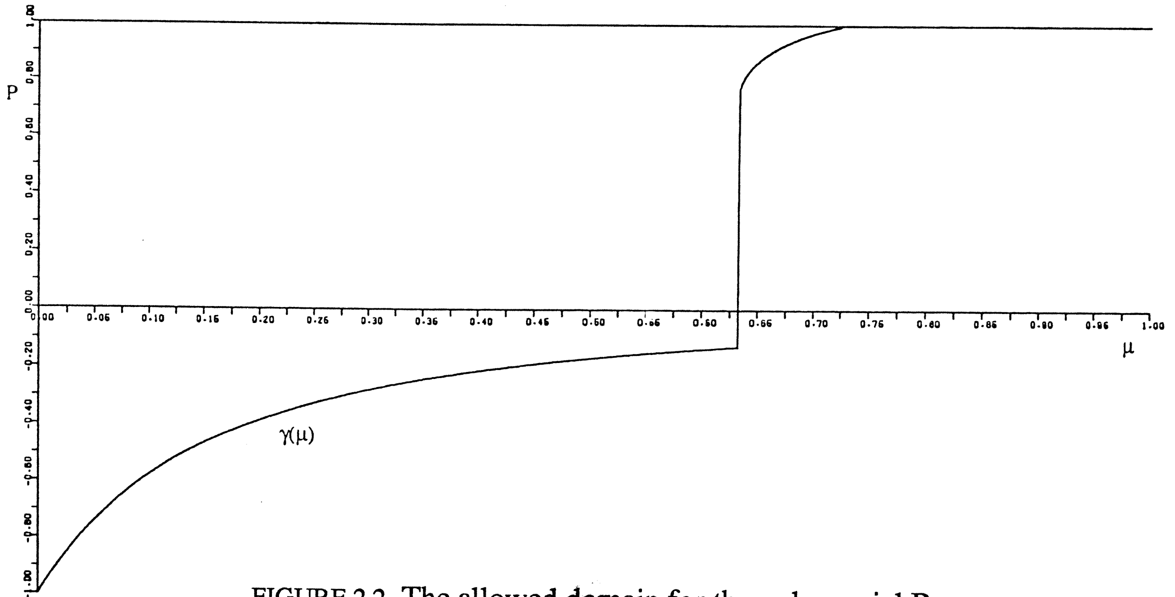

FIGURE 2.2 The allowed domain for the polynomial P

This plot shows that up to μ=0.632... there is a rather wide range for P to move in; however, beyond this critical μ-value, one of the nonlinear Hurwitz-conditions seems to become active, resulting in a small 'freedom' for P. This means that the polynomial P((1-μ)z) leaves the safe area for z-values close to the origin. Therefore, we have to choose $\mu \in [0,0.632]$ and, for each μ, the real stability boundary $\beta(\mu)$ is found by solving $P((1-\mu)z) := 1 + (1-\mu)z + (1-\mu)^2z^2/2 + (1-\mu)^3z^3/6 = \gamma(\mu)$ for z. For μ=0.632 we found $\beta(0.632) \approx 4.80$. In passing, we remark that the polynomial P associated with Kutta's method monotonically decreases if its argument runs from 0 along the negative axis. Furthermore, we checked the zero-stability of the method (2.15) as a function of μ. It turned out that the scheme is zero-stable for $0 \leq \mu \leq 0.73...$. In our experiments we take μ=0.625, which is a safe value with respect to (zero-)stability. The corresponding β-value equals 4.72, and the stability region is shown in Fig.A4 in Appendix II. Finally, we calculated the roots of (2.16) for $P \equiv 1$ and μ=0.625. These roots are given by $\zeta=1$, $\zeta \approx 0.5271$ and $\zeta \approx 0.5250 \pm 0.7532 \cdot i$; hence, the largest spurious roots have modulus 0.9181... at the origin.

In conclusion, we can formulate the following theorem:

Theorem 2.3. *The method* (2.15) *is third-order accurate for all* $\mu\in[0,1)$ *and zero-stable for* $\mu\in[0,0.73...]$. *The largest real stability boundary is obtained for* $\mu=0.632...$ and is given by $\beta = 4.80...$ []

2.4 FOURTH-ORDER METHODS

Finally, we consider the case k=4. The solution of (2.3) for this k-value is given by

$$a_0(\mu)=\frac{1}{24}(\mu+1)(\mu+2)(\mu+3)(\mu+4),\qquad a_1(\mu)=-\frac{1}{6}\mu(\mu+2)(\mu+3)(\mu+4),$$

$$a_2(\mu)=\frac{1}{4}\mu(\mu+1)(\mu+3)(\mu+4),\qquad a_3(\mu)=-\frac{1}{6}\mu(\mu+1)(\mu+2)(\mu+4),\tag{2.17a}$$

$$a_4(\mu)=\frac{1}{24}\mu(\mu+1)(\mu+2)(\mu+3).$$

For the integrator R we take the 'classical' fourth-order, 4-stage RK method (cf. [6, p.120]), resulting in

$$\begin{aligned}&y_{n+1}=y_\mu{}^* +\frac{1}{6}(1-\mu)\,h\,[\,k_1+2k_2+2k_3+k_4\,],\\&k_1=f(t_n+\mu h,\,y_\mu{}^*),\qquad k_2=f(t_n+\tfrac{1}{2}(1+\mu)h,\,y_\mu{}^*+\tfrac{1}{2}(1-\mu)hk_1),\\&k_3=f(t_n+\tfrac{1}{2}(1+\mu)h,\,y_\mu{}^*+\tfrac{1}{2}(1-\mu)hk_2),\qquad k_4=f(t_{n+1},\,y_\mu{}^*+(1-\mu)hk_3).\end{aligned}\tag{2.17b}$$

Similar to the previous subsection, the stability analysis of the fourth-order method (2.17) is performed numerically. Based on the characteristic equation, which is now of degree 5, we calculated the lower bound curve $\gamma(\mu)$ for the polynomial P. This curve is plotted in Figure 2.3, the analogue of Figure 2.2.

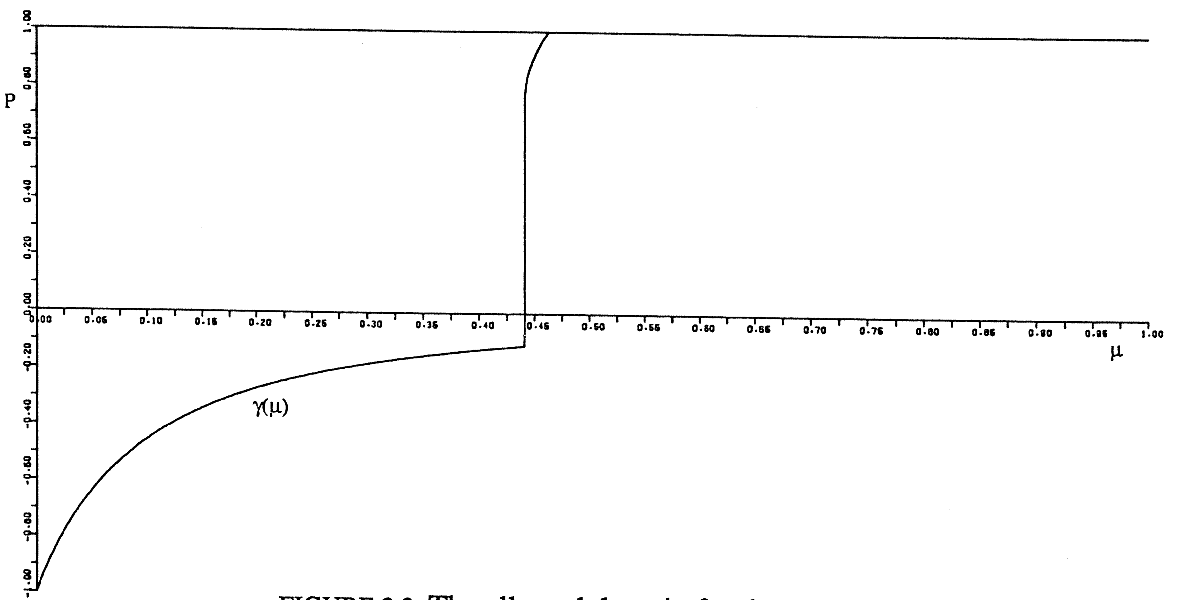

FIGURE 2.3 The allowed domain for the polynomial P

We see that up to $\mu=0.441...$, there is sufficient room for the polynomial P to yield a large real stability boundary (P assumes values in [0.27,1] for the relevant argument values) but if μ exceeds this value, then the polynomial P will quickly leave the safe area. Consequently, we have to require $\mu\in[0,0.441]$. Solving $P((1-\mu)z) := 1 + (1-\mu)z + (1-\mu)^2z^2/2 + (1-\mu)^3z^3/6 + (1-\mu)^4z^4/24 = \gamma(\mu)$ for the optimal μ-value, the real stability boundary is easily found to be $\beta(0.441)\approx4.98$. By a numerical search the method is verified to be zero-stable for $0\leq\mu\leq0.46...$. In testing this method, we will use $\mu=0.435$. For this μ-value the stability boundary equals $\beta\approx4.93$, and the stability region is plotted in Fig A5 (see Appendix II). The characteristic roots at the origin are given by $\zeta=1$, $\zeta\approx0.4392 \pm 0.1949{\cdot}i$ and $\zeta\approx0.1698 \pm 0.9557{\cdot}i$; hence, the spurious roots have modulus 0.4805 and 0.9707, respectively. We summarize these results in the following theorem:

Theorem 2.4. *The method* (2.17) *is fourth-order accurate for all* $\mu\in[0,1)$ *and zero-stable for* $\mu\in[0,0.46...]$. *The largest real stability boundary is obtained for* $\mu=0.441...$ and is given by $\beta = 4.98...$ []

3. NUMERICAL EXPERIMENTS

To investigate the effect on the accuracy of the final results, caused by an increase of the stability boundary, we will apply the methods described in the preceding subsections to a parabolic PDE. Furthermore, the second-order scheme (2.10) will be compared with second-order explicit methods described in the literature, which also possess an enlarged stability boundary (see also the Introduction).

As our test example, we consider the linear, 2-dimensional problem

$$\frac{\partial u}{\partial t} = \frac{1}{4}\left(\frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2}\right) - \frac{17}{16}u, \qquad t \geq 0, \tag{3.1a}$$

defined on the unit square in the (x_1,x_2)-plane and supplemented with initial- and Dirichlet boundary conditions taken from the exact solution

$$u(t, x_1, x_2) = \exp\left(-t + \tfrac{1}{2}(x_1 + x_2)\right). \tag{3.1b}$$

The semi-discrete system of ODEs is obtained by using symmetric, three-point finite difference approximations on a uniform mesh with meshsize $\Delta x_1 = \Delta x_2 = 1/20$. The spectral radius ρ of the Jacobian matrix of the resulting system is approximately given by

$$\rho \approx \frac{1}{4}\cdot\frac{8}{(\Delta x_1)^2} = 800\,. \tag{3.2}$$

The stepsizes are chosen maximal with respect to the stability condition which reads

$$h \cdot \rho \leq \beta(\mu) . \tag{3.3}$$

In the end point t=T of the integration interval, we compare the components of the discrete solution vector with the exact solution (3.1b) in the corresponding grid points and selected the component yielding the largest relative error. In the tables of results we present the quantity

$$\mathrm{sd} := - \log_{10} (\| \text{relative error of the fully discrete solution} \|_\infty) ; \tag{3.4}$$

hence, sd denotes the number of significant digits for the component with the largest (relative) error.

It turned out that the semi-discretization error for this problem and this spatial grid equals approximately $10^{-1.7}$. By this error we mean the (relative) difference between the *exact* solution of the ODE and the solution of the PDE (restricted to the grid points), measured in the maximum norm. Hence, an exact integration would result in an sd-value ≈ 1.7. Now, our hope is that by taking large time steps — and, consequently, introducing a time integration error — this accuracy is not destroyed (too much). In Table 3.1 the results for T=1 of the various time integration methods are presented for several values of the parameter μ. To see the effects of a long-term integration we also added the results for T=10 and T=20.

TABLE 3.1. sd-values for problem (3.1).

method	order	μ	β	h^{-1}	sd (T=1)	sd (T=10)	sd (T=20)
(2.4)	1	0	2	400	1.7	1.7	1.7
		0.5	3	267	1.8	1.8	1.8
		0.75	5.6	143	1.9	1.9	1.9
		0.90	13.6	59	1.6	1.6	1.6
		0.925	18.0	45	1.1	1.2	1.2
		0.95	26.9	30	0.5	0.4	1.2
(2.10)	2	0	2	400	1.7	1.7	1.7
		0.825	11.4	70	1.7	1.7	1.7
(2.15)	3	0	2.5	319	1.7	1.7	1.7
		0.625	4.7	170	1.7	1.7	1.7
(2.17)	4	0	2.8	288	1.7	1.7	1.7
		0.435	4.9	163	1.7	1.7	1.7

From this table it is clear that the methods of order p≥2 can take their maximal stable time step without influencing the *total* accuracy. However, for the first-order scheme, where the stability boundary can be enlarged without restriction, we see that a grow factor up to ≈ 6.5 (i.e., μ≤0.9) has no serious consequences for the accuracy. Increasing μ beyond this value, drastically decreases the accuracy.

For such μ-values, the scaled error constant blows up to such an extent that now the time integration error dominates the semi-discretization error.

In [5, p.81], Jeltsch and Nevanlinna prove that, for any $k \geq 2$, there exists an explicit linear k-step method of order k-1 with an arbitrary real stability boundary. This is of course a much stronger result than obtained in the present paper. However, if large stability boundaries are to be obtained, then the resulting methods quickly loose accuracy. Since such schemes are not explicitly given in [5], we constructed (following the proof of its existence) a 3-step second-order method. This method will be compared with the scheme (2.10), which has the same step number and order. Adopting the notation used in [5], this method is defined by its characteristic polynomials

$$\rho_{\delta,\varepsilon}(\zeta) = \delta^{-2}(\zeta-1)(\zeta-1+\delta^2)(A\zeta+B),$$
$$\sigma_{\delta,\varepsilon}(\zeta) = \delta^{-1}(\zeta-1+\delta)(\zeta-1+\varepsilon). \tag{3.5a}$$

The parameters A and B serve to make the method of second-order accuracy, resulting in

$$A = 1 - \varepsilon\,\delta^{-2}\left(\tfrac{1}{2}\delta^2 + \delta - 1\right), \qquad B = \varepsilon - A. \tag{3.5b}$$

Since this linear multistep method requires only one f-evaluation per step, a fair comparison with (2.10) is achieved if (3.5) is applied with half the stepsize. Or, in other words, the method (3.5) needs only half the stability boundary that is needed by (2.10). In [5] it is not explicitly stated how to choose the parameters δ and ε. By trial and error we determined several combinations of these parameters, yielding the required stability boundary. We did not make an attempt to find optimal parameter values. In Table 3.2 we give the results obtained by the method (3.5).

TABLE 3.2. sd-values for method (3.5) applied to problem (3.1).

method	order	δ	ε	h^{-1}	sd (T=1)	sd (T=10)	sd (T=20)
(3.5)	2	0.39	.0002	140	2.0	- 2.1	- 6.3
		0.38	.005	140	2.4	0.6	- 0.7
		0.37	.009	140	2.0	1.3	1.2

From this table we observe that the sd-values for T=1 are larger than 1.7, which would be obtained by exact integration, indicating that the time integration error and the semi-discretization error interfere in a favourable way. However, upon continuing the integration to the end points T=10 and T=20, the accuracy of the numerical solution steadily decreases. Apparently, this is due to the scaled error constant which is large and gets even larger if more stability is required. Therefore, the existence of arbitrarily large stability boundaries within this class of explicit linear multistep methods is more of theoretical interest than of practical value.

Next, we compare the scheme (2.10) with the method analysed by Zlatev and Østerby [11], which is also a 3-step second-order method, requiring one f-evaluation per step. It is given by

$$\begin{aligned} y_{n+1} &= a_2\, y_n + a_1\, y_{n-1} + (1-a_2-a_1)\, y_{n-2} + \\ &\quad h\,[b_2\, f(t_n,y_n) + (4.5-2a_2-0.5a_1-2b_2)\, f(t_{n-1},y_{n-1}) + (-1.5-0.5a_1+b_2)\, f(t_{n-2},y_{n-2})\,]\,, \end{aligned} \tag{3.6}$$

where

$$a_2 = 2.98, \quad a_1 = -2.961, \quad b_2 = 0.17,$$

and possesses the real stability boundary $\beta \approx 12.4$. We applied this method to problem (3.1) for several values of the stepsize. The results can be found in Table 3.3.

TABLE 3.3. sd-values for method (3.6) applied to problem (3.1).

method	order	h^{-1}	sd (T=1)	sd (T=10)	sd (T=20)
(3.6)	2	140	1.6	- 1.3	- 5.5
		70	0.8	- 2.7	- 7.1
		65(max.)	0.8	- 2.7	- 6.9

Again, the scaled error constant has blown up to such an extent that the obtained accuracies are substantially lower than for the scheme (2.10), especially if the integration is continued over a long interval.

Finally, we have implemented a straightforward generalization of the method of Du Fort and Frankel [1,9] for the inhomogeneous linear equation (3.1a). This generalization reads

$$\begin{aligned} \frac{U^{n+1}_{j,k} - U^{n-1}_{j,k}}{2h} &= \frac{1}{4}\left(\frac{U^{n}_{j+1,k} - U^{n+1}_{j,k} - U^{n-1}_{j,k} + U^{n}_{j-1,k}}{(\Delta x_1)^2} + \frac{U^{n}_{j,k+1} - U^{n+1}_{j,k} - U^{n-1}_{j,k} + U^{n}_{j,k-1}}{(\Delta x_2)^2}\right) \\ &\quad - \frac{17}{16}\left(\alpha\, U^{n+1}_{j,k} + (1-2\alpha)\, U^{n}_{j,k} + \alpha\, U^{n-1}_{j,k}\right), \end{aligned} \tag{3.7}$$

where j and k run through the grid points, $U^n_{j,k}$ represents an approximation to the exact solution $u(nh, j\Delta x_1, k\Delta x_2)$ and α is a parameter. This two-step method is second-order in time. For each (j,k) separately, the unknowns $U^{n+1}_{j,k}$ can be explicitly solved from (3.7). For several values of the stepsize h and parameter-values $\alpha \in [0,0.5]$, we applied this method to the test problem. The influence of the choice of α turned out be negligible. The results are given in Table 3.4.

TABLE 3.4. sd-values for method (3.7) applied to problem (3.1).

method	order	h^{-1}	sd (T=1)	sd (T=10)	sd (T=20)
(3.7)	2	10	0.2	- 2.4	- 6.2
		20	0.6	0.9	0.6
		35	1.5	1.5	1.5
		70	2.2	2.2	2.2
		140	1.8	1.8	1.8

From these results we conclude the the method (3.7) is able to take large time steps, but does not behave unconditionally stable. Clearly, for $h \leq 1/20$ the results are not satisfactorily. For $h=1/35$, a slight drop in accuracy is noticed and for the smaller time steps the method behaves well. As an advantage of this scheme we mention its lower storage requirements in comparison with the other explicit methods; a disadvantage is that for nonlinear problems a (scalar) equation has to be solved for each component. Moreover, for such problems the generalization is less straightforward.

4. CONCLUSIONS

In this paper we have proposed an extension to very simple integration rules, i.e., one-step explicit Runge-Kutta methods. This extension results in an increased real stability boundary, while the algorithmic and computational advantages of these simple schemes are maintained. Methods of orders 1,...,4 have been analysed. For the methods of order 3 and 4, the resulting stability boundary is found to be approximately twice as large. For the first- and second-order method, however, an increase with a factor 6 turns out to be possible in a realistic application. When properly used, e.g., in the context of semi-discrete parabolic equations, the enlarged (truncation) error of the new schemes will usually not influence the accuracy of the fully discrete solution of the PDE.

Hence, if one persists in using an explicit method, also in cases where the ODE possesses some stiffness, then these methods may help to relax the severe time step restriction.

As a disadvantage of the new methods we mention the fact that the algorithm has changed from a *one-step* method into a *multistep* method. This means that we have to calculate a few additional starting values prior to the actual application of the methods. However, with the help of the underlying simple (RK) method, these additional values can be easily obtained (see also Appendix I, where we describe an implementation).

Finally, it should be remarked that these schemes are not recommended to be used in case of hyperbolic PDEs or, in general, in case of ODEs where the Jacobian matrix has a more or less imaginary spectrum. This directly follows from an inspection of the stability regions plotted in Appendix II.

REFERENCES

[1] Du Fort, E.C. and S.P. Frankel, *Stability conditions in the numerical treatment of parabolic differential equations*, 1953, Math. tables and other aids to computing, Vol.7, p.135.

[2] Henrici, P., *Discrete variable methods in ordinary differential equations*, 1962, Wiley, New York.

[3] Houwen, P.J. van der, *Construction of integration formulas for initial value problems*, 1977, North-Holland, Amsterdam-New York-Oxford.

[4] Houwen, P.J. van der and B.P. Sommeijer, *Predictor-corrector methods with improved absolute stability regions*, 1983, IMA J. Numer. Anal., Vol. 3, pp. 417-437.

[5] Jeltsch, R. and O. Nevanlinna, *Stability of explicit time discretizations for solving initial value problems*, 1981, Numer. Math., Vol. 37, pp. 31-61.

[6] Lambert, J.D., *Computational methods in ordinary differential equations*, 1973, Wiley, London-New York-Sydney-Toronto.

[7] Lambert, J.D., *Stiffness*, 1980, Computational techniques for ordinary differential equations, I. Gladwell and D. K. Sayers (eds.), Academic Press, London.

[8] Mitchell, A.R. and D.F. Griffiths, *The finite difference method in partial differential equations*, 1980, Wiley, Chichester-New York-Brisbane-Toronto.

[9] Richtmyer, R.D. and K.W. Morton, *Difference methods for initial-value problems*, 1967, Wiley, New York.

[10] Wait, R. and A.R. Mitchell, *Finite element analysis and applications*, 1985, Wiley, Chichester-New York-Brisbane-Toronto-Singapore.

[11] Zlatev, Z. and O. Østerby, *Absolute stability properties of the explicit 3-step formulae*, 1980, Report DAIMI PB-124, Aarhus University, Denmark.

APPENDIX I

Here we describe a possible implementation (in a pseudo computer language) of the methods proposed in this paper. As an example, we use the second-order scheme (2.10), but the other methods are implemented quite similarly.

Suppose we have the ‘subroutine’:

RK2 (NEQN, T, H, Y_{IN}, Y_{OUT}), which performs one *single* integration step of size H, given the initial vector Y_{IN} (of dimension NEQN) at time T and yielding the result vector Y_{OUT}, i.e., the numerical result at T+H of Euler's improved method.

Now, the implementation of (2.10) could read (remarks between # # can be considered as comment):

```
begin
    declaration of the arrays Y, $Y_N$, $Y_{N-1}$, $Y_{N-2}$, Y*  (of size NEQN)
    T :=$t_0$;      $Y_{N-2}$:= $y_0$      # $t_0$, $y_0$ are the initial values #
    # determine the maximally stable stepsize $H_{EULER}$ for the RK2 method (i.e., for $\mu$=0) #
    # select a $\mu$-value (viz. $\mu$=0.625) and determine the corresponding stable stepsize H #
    FACTOR := $\lceil H / H_{EULER} \rceil$    # where $\lceil \cdot \rceil$ means rounded to the next larger integer #
    $H_{EULER}$ := H / FACTOR
    $Y_N$ := $y_0$
    to FACTOR do
        call RK2 (NEQN, T, $H_{EULER}$, $Y_N$, Y)
        $Y_N$ := Y;    T := T + $H_{EULER}$
    end do
    $Y_{N-1}$ := Y
    to FACTOR do
        call RK2 (NEQN, T, $H_{EULER}$, $Y_N$, Y)
        $Y_N$ := Y;    T := T + $H_{EULER}$
    end do
    NSTEPS := $\frac{T_{END} - t_0}{H}$ - 2    # $T_{END}$ denotes the end point of the integration interval, and it is
                                               assumed that the length of this interval is an integer multiple of
                                               the stepsize #
    to NSTEPS do
        Y* := $\frac{1}{2}(\mu+1)(\mu+2) Y_N - \mu(\mu+2) Y_{N-1} + \frac{1}{2}\mu(\mu+1) Y_{N-2}$
        call RK2 (NEQN, T+$\mu$·H, (1-$\mu$)·H, Y*, Y)
        $Y_{N-2}$ := $Y_{N-1}$;    $Y_{N-1}$ := $Y_N$;    $Y_N$ := Y;    T := T + H       # shift the date and increase T #
    end do
    # now, Y will contain the numerical solution at the end point $T_{END}$ #
end
```

APPENDIX II

Here we present plots of the stability regions (in the complex z-plane) for the various methods described in Section 2.

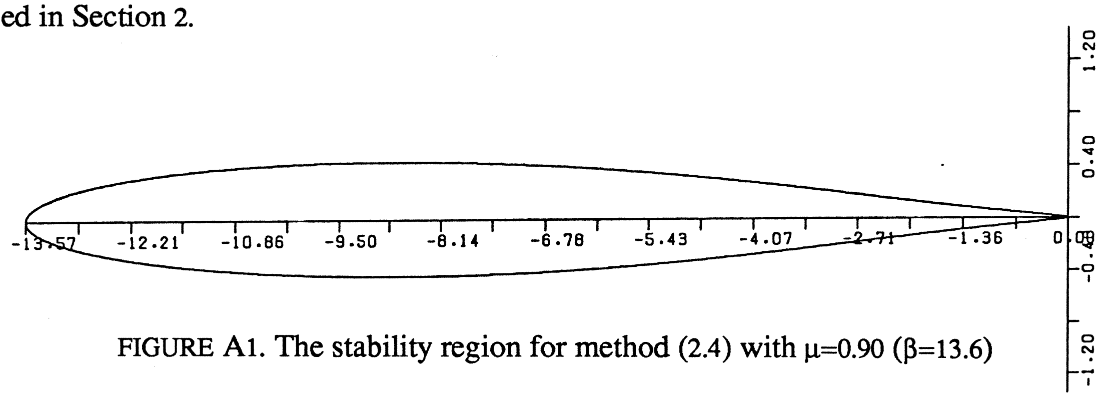

FIGURE A1. The stability region for method (2.4) with μ=0.90 (β=13.6)

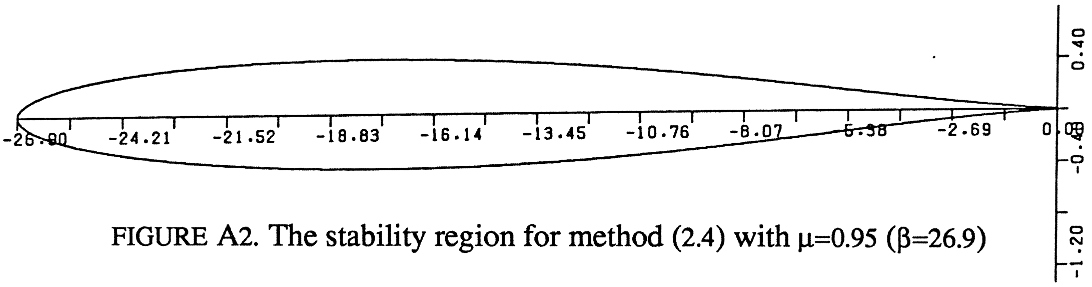

FIGURE A2. The stability region for method (2.4) with μ=0.95 (β=26.9)

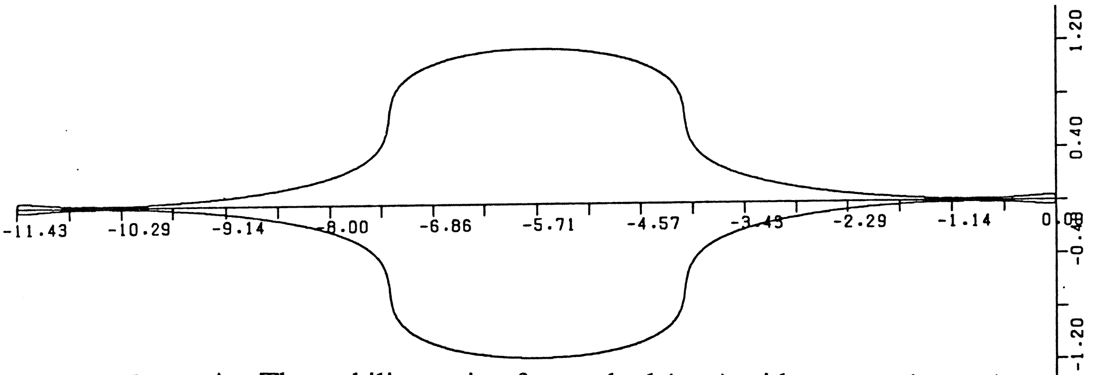

FIGURE A3. The stability region for method (2.10) with μ=0.825 (β=11.4)

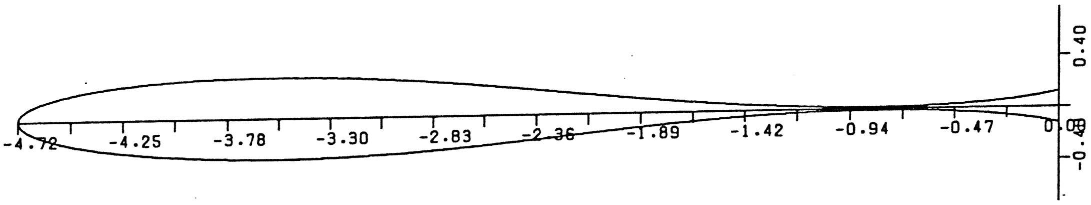

FIGURE A4. The stability region for method (2.15) with μ=0.625 (β=4.7)

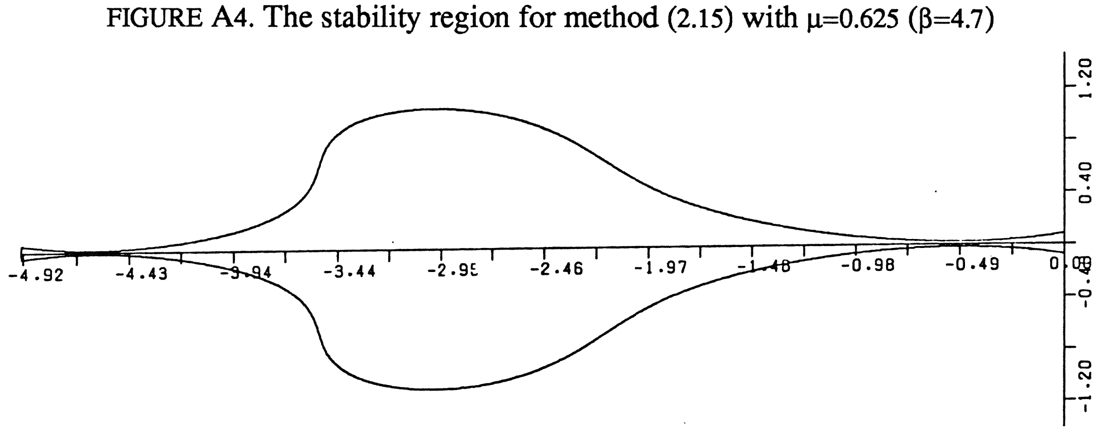

FIGURE A5. The stability region for method (2.17) with μ=0.435 (β=4.9)