Efficient Implementation of Stable Richardson Extrapolation Algorithms

István Faragó [1)] , Ágnes Havasi [2)] and Zahari Zlatev [3*)]

[1)] Department of Applied Analysis and Computational Mathematics, Eötvös Loránd University, Budapest, Hungary
e-mail: faragois@cs.elte.hu

[2)] Department of Meteorology, Eötvös Loránd University, Budapest, Hungary
e-mail: hagi@nimbus.elte.hu

[3)] National Environmental Research Institute, Aarhus University, Roskilde, Denmark
e-mail: zz@dmu.dk

Abstract

Richardson Extrapolation is a powerful computational tool which can successfully be used in the efforts to improve the accuracy of the approximate solutions of systems of ordinary differential equations (ODEs) obtained by different numerical methods (including here combined numerical methods consisting of appropriately chosen splitting procedures and classical numerical methods). Some stability results related to two implementations of the Richardson Extrapolation (Active Richardson Extrapolation and Passive Richardson Extrapolation) are formulated and proved in this paper. An advanced atmospheric chemistry scheme, which is commonly used in many well-known operational environmental models, is applied in a long sequence of experiments in order to demonstrate the fact that

(a) it is indeed possible to improve the accuracy of the numerical results when the Richardson Extrapolation is used (also when very difficult, badly scaled and stiff non-linear systems of ODEs are to be treated),

(b) the computations can become unstable when the combination of the Trapezoidal Rule and the Active Richardson Extrapolation is used,

(c) the application of the Active Richardson Extrapolation with the Backward Euler Formula is leading to a stable computational process,

(d) experiments with different algorithms for solving linear systems of algebraic equations are very useful in the efforts to select the most suitable approach for the particular problems solved and

(e) the computational cost of the Richardson Extrapolation is much less than that of the underlying numerical method when a prescribed accuracy has to be achieved.

Key words: Richardson Extrapolation, Stability, Backward Euler Formula, Trapezoidal Rule, Sparse Matrix Technique, Atmospheric Chemistry Scheme.

1980 AMS Subject Classification: Primary: 65L05, Secondary: 65L07, 65L20, 65L07

*) Corresponding Author: Phone: +45 39671978, e-mail: zz@dmu.dk

1. Richardson Extrapolation

Consider the classical initial value problem for systems of s ($s \geq 1$) ordinary differential equations (ODEs):

$$\textbf{(1)} \qquad \frac{dy}{dt} = f(t,y), \qquad t \in [a,b], \qquad a < b, \qquad y(a) = y_0,$$

where the unknown function $y : [a,b] \to \Re^s$ is continuously differentiable, while right-hand-side function $f(t,y)$ is continuous. However, it is often necessary to introduce much more restrictive assumptions when numerical methods of order p are used in the treatment of (1). In such a case it is necessary to assume that the two functions y and f are continuously differentiable up to orders $p+1$ and p respectively. It is also worthwhile to emphasize the fact that many mathematical models arising in different fields of science and engineering can be represented (after discretization of the spatial derivatives) by systems of ODEs of type (1); see, for example, Hundsdorfer and Verwer (2003), Lambert (1991), Zlatev (1995) or Zlatev and Dimov (2006).

Richardson Extrapolation is sometimes used either (a) in an attempt to improve the accuracy of the calculated approximations or (b) to control automatically the accuracy that is achieved by the selected numerical method.

1.1. Using Richardson Extrapolation to improve the accuracy of the approximate solution

Richardson Extrapolation can be introduced in the following way. Assume that $t_n \in [a,b]$ is a given time-point and that $y(t_n)$ is the value of the exact solution of (1) at $t = t_n$. Assume also that two approximations of $y(t_n)$ have been obtained by applying a numerical method of order p and by using two time-stepsizes h and $0.5h$. More precisely, starting from a time-point $t = t_{n-1}$, where $t_{n-1} = t_n - h$, the two approximations are calculated by using first one large time-step and, after that, two small time-steps. Denoting these two approximations with z_n and w_n respectively, we can write:

$$\textbf{(2)} \qquad y(t_n) = z_n + h^p K + O\left(h^{p+1}\right)$$

and

$$\textbf{(3)} \qquad y(t_n) = w_n + (0.5h)^p K + O\left(h^{p+1}\right),$$

where K is some quantity depending on the numerical method used to calculate z_n and w_n. Eliminating the terms containing K from (2) and (3) gives:

$$\textbf{(4)} \qquad y(t_n) = \frac{2^p w_n - z_n}{2^p - 1} + O(h^{p+1}).$$

Denote

$$\mathbf{(5)} \qquad y_n = \frac{2^p w_n - z_n}{2^p - 1}.$$

It is clear that the approximation y_n, being of order $p+1$, will in general be more accurate than both w_n and z_n (at least when the stepsize h is sufficiently small). Thus, Richardson Extrapolation can be used in the efforts to improve the accuracy of the approximate solutions.

1.2. Using Richardson Extrapolation to control the stepsize

Richardson Extrapolation can also be used in an attempt to evaluate the leading term of the global truncation error made in the calculation of the approximation w_n. Neglect the terms $O(h^{p+1})$ in (2) and (3). Subtract (3) from (2). The result is:

$$\mathbf{(6)} \qquad K = \frac{2^p (w_n - z_n)}{h^p (2^p - 1)}.$$

Substitute K from (6) in (3):

$$\mathbf{(7)} \qquad y(t_n) - w_n = \frac{w_n - z_n}{2^p - 1} + O(h^{p+1}),$$

which means that the quantity $ERROR_n$:

$$\mathbf{(8)} \qquad ERROR_n = \left| \frac{w_n - z_n}{2^p - 1} \right|$$

can be used as an evaluation of the leading term of the global truncation error of the approximation w_n when the stepsize h is sufficiently small. Assume that the desired accuracy of the approximate solution of (1) is determined by some prescribed in advance error tolerance parameter TOL. If the evaluation of the global error computed by using (8) differs substantially from TOL, then $ERROR_n$ can also be used to determine a new stepsize h_{new}, which will hopefully give an error closer to TOL. Such an automatic control of the stepsize is usually carried out by applying the following formula:

$$\mathbf{(9)} \qquad h_{new} = \omega \sqrt[p]{\frac{ERROR_n}{TOL}}\ h,$$

where ω is some precaution parameter ($\omega = 0.9$ is used in many well-known codes for solving systems of ODEs by automatic control of the stepsize; see, for example Shampine and Gordon, 1975). Thus, the Richardson extrapolation can be applied in codes for solving systems of ODEs with automatic stepsize control.

1.3. Some remarks about the Richardson Extrapolation

The following three remarks provide some additional information about the Richardson Extrapolation.

Remark 1: The device sketched above was first extensively been used by L. F. Richardson, Richardson (1927), who called it *"the deferred approach to the limit"*. It is often used in different areas of numerical analysis and applied mathematics (see, for example, Henrici (1968), Hundsdorfer and Verwer (2003) and Lambert (1991). ■

Remark 2: The Richardson extrapolation does not depend too much of the particular numerical method. It can be used both when classical numerical algorithms are applied in the solution of differential equations and when more advanced numerical methods which are combinations of splitting procedures and classical numerical algorithms are devised and used. Two issues are important: (a) the large time-step and the two small time-steps must be handled by using the same numerical method and (b) the order $\mathbf{p}$ of the selected method should be known. ■

Remark 3: The version of the Richardson Extrapolation described in this section is perhaps the simplest one. Some more complicated versions of this device can be found in Faragó (2008). ■

2. Two ways of implementing the Richardson Extrapolation

The device, which was described in the previous section, can be implemented in two different ways. The first implementation will be called Active Richardson Extrapolation, while the name Passive Richardson Extrapolation will be used for the second one. In the Active Richardson Extrapolation the hopefully improved approximation $\mathbf{y_n}$ is used in the calculation of $\mathbf{z_{n+1}}$ and $\mathbf{w_{n+1}}$ at every time-step $\mathbf{n}$ ($\mathbf{n = 1, 2,}\ldots$). In the Passive Richardson Extrapolation, the values of $\mathbf{z_n}$ and $\mathbf{w_n}$ are used to calculate $\mathbf{z_{n+1}}$ and $\mathbf{w_{n+1}}$ (again at every time-step $\mathbf{n}$, $\mathbf{n = 1, 2,}\ldots$). This means that the calculated at a given time-step approximation $\mathbf{y_n}$ is never used in the further computations when the Passive Richardson Extrapolation is selected. The two implementations of the Richardson Extrapolation are depicted in Fig. 1 and Fig. 2.

It is intuitively clear that the incorporation of the improved values $\mathbf{y_1, y_2,}\ldots$ in the computations may lead to more accurate results. Extensive experiments indicate that very often this is not the case. However, if the problem is very stiff and if some of the components of the solution vector $\mathbf{y(t)}$ are quickly varying in some parts of the time-interval, then the Active Richardson Extrapolation may sometimes produce better results.

On the other hand, while it is nearly clear that the Passive Richardson Extrapolation will produce stable numerical results when the underlying algorithm is stable (a rigorous proof is given in Theorem 2), the same conclusion cannot be drawn when the Active Richardson Extrapolation is used. In other words, the Active Richardson Extrapolation may be unstable also when the underlying numerical algorithm is stable. It is proved in Section 5 (Theorem 3) that the Active Richardson Extrapolation may produce unstable computations when it is combined with the well-

known Trapezoidal Rule (the Trapezoidal Rule is described in many text books treating the topic of the numerical solution of systems of ODEs; see, for example, Lambert, 1991).

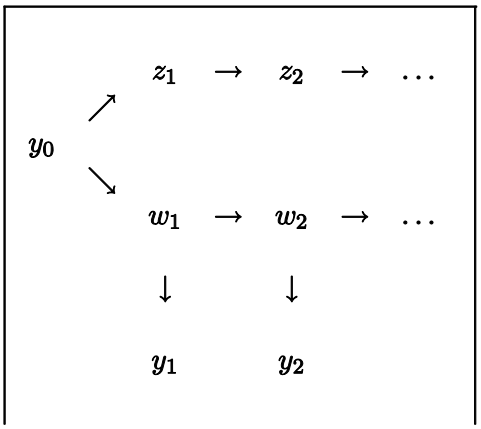

Figure 1: Schematic diagram of the Passive Richardson Extrapolation.

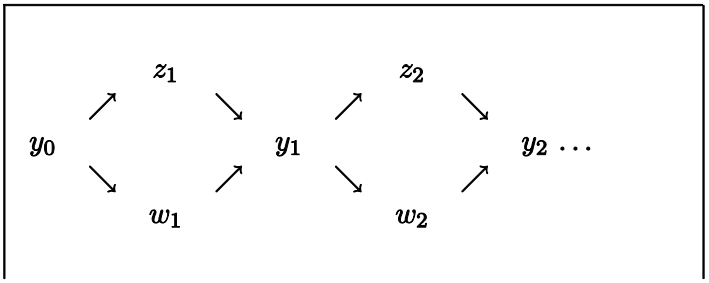

Figure 2: Schematic diagram of the Active Richardson Extrapolation.

3. Dahlquist's test-problem and stability functions

It is desirable to preserve the stability properties of the selected numerical method for solving systems of ODEs when this method is combined with the Richardson Extrapolation. The preservation of the stability properties will be discussed both in this section and in the following three sections.

The stability studies related to the numerical methods for solving systems of ODEs are usually based on the application of the famous test-problem:

$$\textbf{(10)} \qquad \frac{dy}{dt} = \lambda y, \qquad t \in [0,\infty), \qquad y \in C, \qquad \lambda \in C, \qquad \mathrm{Re}(\lambda) \le 0.$$

This test-problem was introduced by G. Dahlquist in 1963 (Dahlquist, 1963) and used in several thousand papers after that. The importance of (10) is emphasized in many publications. For example, Hundsdorfer and Verwer declare that *"in spite of its simplicity this test-equation is of major importance for predicting the stability behavior of numerical ODE methods"* (Hundsdorfer and Verwer, 2003, p. 144).

Consider the class of the one-step numerical methods for solving systems of ODEs (see, for example, Henrici, 1968). The approximation $\mathbf{y_n}$ of $\mathbf{y(t_n)}$ calculated by an arbitrary one-step method can be expressed as a function of $\mathbf{y_{n-1}}$ for any value of $\mathbf{n}$. The Runge-Kutta methods (these methods are discussed in detail, for example, in Butcher, 2003) belong to the class of one-step methods.

The following recurrent relation can be obtained (Hundsdorfer and Verwer, 2003) when many one-step methods are used in the solution of (10):

(11) $$\mathbf{y_n = R(\mu)\, y_{n-1}},$$

where $\mu = \lambda \mathbf{h}$ and $\mathbf{R(\mu)}$ is a polynomial if an explicit numerical method is used and a rational function if the method is implicit. Very often $\mathbf{R(\mu)}$ is called the stability function of the method (Hundsdorfer and Verwer, 2003, p. 37). Since $\mathbf{y_n = [R(\mu)]^n y_0}$, it is clear that the computations will be stable when (10) is solved if

(12) $$\mathbf{|R(\mu)| \le 1.}$$

The stability functions of three well-known and commonly used numerical methods for solving systems of ODEs are given below. If the Trapezoidal Rule:

(13) $$\mathbf{y_n = y_{n-1} + 0.5h\left[f(t_{n-1}, y_{n-1}) + f(t_n, y_n)\right]}$$

is applied in the solution of (10), then the stability function is given by

(14) $$\mathbf{R_{TR}(\mu) = \frac{1 + 0.5\mu}{1 - 0.5\mu}.}$$

If the Backward Euler Formula:

(15) $$\mathbf{y_n = y_{n-1} + h f(t_n, y_n)}$$

is applied in the solution of (10), then the stability function is given by

(16) $$\mathbf{R_{BE}(\mu) = \frac{1}{1 - \mu}.}$$

If the θ-method:

(17) $$\mathbf{y_n = y_{n-1} + h\left[(1-\theta) f(t_{n-1}, y_{n-1}) + \theta f(t_n, y_n)\right]}$$

is applied in the solution of (10), then the stability function is given by

(18) $$R_\theta(\mu)=\frac{1+(1-\theta)\mu}{1-\theta\mu}.$$

The Trapezoidal Rule and the Backward Euler Formula are special cases of (17) obtained by using $\theta = 0.5$ and $\theta = 1$ respectively.

It should be mentioned here that the stability function from (12) can very often be represented as a ratio of two polynomials:

(19) $$R(\mu)=\frac{P(\mu)}{Q(\mu)}.$$

This representation will be used in the following sections.

4. Two Stability Definitions

Several well-known stability definitions are relevant when stiff systems of ODEs are solved numerically (see Burrage, 1992, Butcher, 2003, Hairer and Wanner, 1991, or Lambert, 1991). Two of them will be used in the remaining part of this paper.

Definition 1: Consider the set S containing all values of $\mu = \alpha + i\beta$ with $\alpha \le 0$ for which (12) is satisfied. If $S \supset C^- = \{\mu = \alpha + i\beta, \alpha \le 0\}$, then the method with stability function $R(\mu)$ is called A-stable. ■

It can be proved by using the maximum modulus theorem (about the maximum modulus theorem see, for example, Wylie, 1975) that Definition 1 is equivalent to the following statement (Hairer and Wanner, 1991).

Theorem 1: A numerical method with stability function $R(\mu)$ is A-stable if and only if

(20) $|R(i\beta)| \le 1$ **for all real values of** β

and

(21) $R(\mu)$ **is analytic function when** $\alpha \le 0$. ■

Definition 2: A numerical method with a stability function $R(\mu)$ is called L-stable if it is A-stable and the relationship:

(22) $$\lim_{\mu \to \infty}(R(\mu)) = 0$$

holds. ■

5. Is the Richardson Extrapolation Stable if the Underlying Method is Stable?

The answer to the above important question depends on the implementation of the Richardson Extrapolation (see Section 2). As mentioned above, for the Passive Richardson Extrapolation the answer is always positive, while the Active Richardson Extrapolation might become unstable. More precisely, the following two theorems can be formulated and proved:

Theorem 2: Assume that the underlying method is either A-stable or L-stable. Assume that the Passive Richardson Extrapolation is used. Then the combined method will be stable when (10) is solved.

Proof: It is clear, see Fig. 1, that the calculation of the sequences $\{z_n\}$ and $\{w_n\}$, $n = 1, 2, \ldots$, is a stable process when (10) is solved (because the underlying numerical method is assumed to be stable). It is also clear that the calculation of $\{y_n\}$ by using (5) is stable, because at every time-step n only a simple linear combination of the two values z_n and w_n is used to calculate y_n. Moreover, the calculation of any $y_n \in \{y_n\}$ will not affect the stability of the combined method because the value of y_n does not participate in the further computations (see again Fig. 1). This proves the theorem. ■

Theorem 3: The computations will in general be unstable when the Trapezoidal Rule is used together with the Active Richardson Extrapolation.

Proof: Since the Trapezoidal Rule is a second-order method, equation (5) with $p = 2$ can be written as

$$\textbf{(23)} \qquad y_n = \frac{4w_n - z_n}{3},$$

where

$$\textbf{(24)} \qquad z_n = \left[\frac{1 + 0.5\mu}{1 - 0.5\mu}\right] y_{n-1}$$

and (since two consecutive small time-steps with stepsize $0.5h$ are to be carried out)

$$\textbf{(25)} \qquad w_n = \left[\frac{1 + 0.25\mu}{1 - 0.25\mu}\right]^2 y_{n-1}.$$

Substitute the expressions (24) and (25) in (23). The result is:

$$\textbf{(26)} \qquad y_n = \left\{\frac{4}{3}\left[\frac{1 + 0.25\mu}{1 - 0.25\mu}\right]^2 - \frac{1}{3}\left[\frac{1 + 0.5\mu}{1 - 0.5\mu}\right]\right\} y_{n-1}.$$

The last equality can be represented in the form $\mathbf{y_n = R_{TR+RICH}(\mu)\, y_{n-1}}$ with

$$\textbf{(27)} \qquad \mathbf{R_{TR+RICH}(\mu) = \frac{4}{3}\left[\frac{1+0.25\mu}{1-0.25\mu}\right]^2 - \frac{1}{3}\left[\frac{1+0.5\mu}{1-0.5\mu}\right].}$$

Equality (27) can be rewritten as

$$\textbf{(28)} \qquad \mathbf{R_{TR+RICH}(\mu) = \frac{4}{3}\,\frac{\frac{1}{\mu^2}+\frac{0.5}{\mu}+0.0625}{\frac{1}{\mu^2}-\frac{0.5}{\mu}+0.0625} - \frac{1}{3}\,\frac{\frac{1}{\mu}+0.5}{\frac{1}{\mu}-0.5}\ .}$$

It can easily be seen now that the following relation holds:

$$\textbf{(29)} \qquad \mathbf{\lim_{\mu \to \infty}\left(R_{TR+RICH}(\mu)\right) = \frac{5}{3}}$$

and, thus, $\left|\mathbf{R_{TR+RICH}}\right|$ will be greater than one when $\left|\mu\right|$ is sufficiently large, which means that the computational process will be unstable when the problem solved is stiff. This completes the proof of the theorem. ■

Remark 4: The Implicit Mid-point Rule (see, for example, Lambert, 1991) will in general also produce unstable results when it is combined with the Richardson Extrapolation. Indeed, for linear problems the Implicit Mid-point Rule and the Trapezoidal Rule coincide. Thus, the stability function of the Implicit Mid-point Rule is given by (14); i.e. it is the same as the stability function of the Trapezoidal Rule. The conclusion is that the stability properties of the Implicit Mid-point Rule and the Trapezoidal Rule are the same. ■

Remark 5: The assertions of Theorem 2 and Theorem 3 are also stated in Dahlquist (1963). These two theorems are given here in order both to facilitate the reading of this paper and, what is even more important, to emphasize the fact that one must be careful when the Active Richardson Extrapolation is used. ■

Theorem 3 and Remark 4 show that in general the application of the Active Richardson Extrapolation may cause instability of the computational process. The question is whether it is possible to ensure stability when the Active Richardson Extrapolation is combined with some particular numerical methods. An answer to this question will be given in the next section.

6. Active Richardson Extrapolation applied with the Backward Euler Formula

Theorem 4: The combined method consisting of the Active Richardson Extrapolation and the Backward Euler Formula is L-stable.

Proof: It follows from Theorem 1 and Definition 2 that in order to prove the assertion of this theorem it is necessary (a) to derive the stability function $\mathbf{R(\mu) = R_{BE+RICH}(\mu)}$ of the combined numerical method (the Active Richardson Extrapolation + the Backward Euler Formula) and (b) to show that the relationships (20), (21) and (22) hold. This means that the theorem can be proved in four steps. The first step will be the derivation of the stability function of the combined numerical method. The validity of each of the three relationships mentioned above has to be established in the next three steps.

(A) Stability function of the combined method: It is clear that performing one large step and two small steps with the Backward Euler Formula starting with the approximation $\mathbf{y_{n-1}}$ will result in the following formula:

$$\mathbf{(30)} \qquad \mathbf{y_n = \left\{ 2\left[\frac{1}{1-0.5\mu}\right]^2 - \left[\frac{1}{1-\mu}\right] \right\} y_{n-1}},$$

which means that the stability function of the combined method is given by

$$\mathbf{(31)} \qquad \mathbf{R(\mu) = \frac{2}{(1-0.5\mu)^2} - \frac{1}{1-\mu}}.$$

The last equality is equivalent to

$$\mathbf{(32)} \qquad \mathbf{R(\mu) = \frac{2(1-\mu)-(1-0.5\mu)^2}{(1-0.5\mu)^2(1-\mu)}}$$

and the polynomials $\mathbf{P(\mu)}$ and $\mathbf{Q(\mu)}$ from (19) are given by

$$\mathbf{(33)} \qquad \mathbf{P(\mu) = 2(1-\mu)-(1-0.5\mu)^2}$$

and

$$\mathbf{(34)} \qquad \mathbf{Q(\mu) = (1-0.5\mu)^2(1-\mu).}$$

The relationships (32), (33) and (34), which were derived above, will play a key role in the proof of the following three steps.

(B) Verification of (20): Equality (20) alone is equivalent to the requirement that the method is stable on the imaginary axis (Hairer and Wanner, 1991). It is shown in Hairer and Wanner (1991) that the stability of the numerical method on the imaginary axis is ensured if

$$\mathbf{(35)} \qquad \mathbf{E(\beta) \geq 0}$$

for all real values of β, where $\mathbf{E(\beta)}$ is defined by

(36) $$E(\beta) = Q(i\beta)Q(-i\beta) - P(i\beta)P(-i\beta).$$

Consider the first term in the right-hand-side of (36). Successful transformations of this term are given below.

(37) $$Q(i\beta)Q(-i\beta) = (1 - i0.5\beta)^2(1 - i\beta)(1 + i0.5\beta)^2(1 + i\beta),$$

(38) $$Q(i\beta)Q(-i\beta) = [(1 - i0.5\beta)(1 + i0.5\beta)]^2(1 - i\beta)(1 + i\beta),$$

(39) $$Q(i\beta)Q(-i\beta) = (1 + 0.25\beta^2)^2(1 + \beta^2),$$

(40) $$Q(i\beta)Q(-i\beta) = (0.0625\beta^4 + 0.5\beta^2 + 1)(\beta^2 + 1),$$

(41) $$Q(i\beta)Q(-i\beta) = 0.0625\beta^6 + 0.5625\beta^4 + 1.5\beta^2 + 1.$$

Similar transformations of the second term in (36) are represented below.

(42) $$P(i\beta)P(-i\beta) = [2(1 - i\beta) - (1 - i0.5\beta)^2][2(1 + i\beta) - (1 + i0.5\beta)^2],$$

(43) $$P(i\beta)P(-i\beta) = 4(1 - i\beta)(1 + i\beta) - 2(1 - i0.5\beta)^2(1 + i\beta) - 2(1 - i\beta)(1 + i0.5\beta)^2 + (1 - i0.5\beta)^2(1 + i0.5\beta)^2,$$

(44) $$P(i\beta)P(-i\beta) = 4(1 + \beta^2) - 2[(1 - i0.5\beta)^2 + (1 + i0.5\beta)^2] - 2i\beta[(1 - i0.5\beta)^2 - (1 + i0.5\beta)^2] + [(1 - i0.5\beta)(1 + i0.5\beta)]^2,$$

(45) $$P(i\beta)P(-i\beta) = 4 + 4\beta^2 - 4 + \beta^2 - 4\beta^2 + (1 + 0.25\beta^2)^2,$$

(46) $$P(i\beta)P(-i\beta) = \beta^2 + 1 + 0.5\beta^2 + 0.0625\beta^4,$$

(47) $$P(i\beta)P(-i\beta) = 0.0625\beta^4 + 1.5\beta^2 + 1.$$

Substitute the expression in the right-hand-sides of (41) and (47) in (36). The result is:

(48) $$E(\beta) = (0.0625\beta^6 + 0.5625\beta^4 + 1.5\beta^2 + 1) - (0.0625\beta^4 + 1.5\beta^2 + 1),$$

which is equivalent to

(49) $$E(\beta) = 0.0625\beta^6 + 0.5\beta^4.$$

The right-hand-side of (49) is clearly non-negative for any value of β. This means that (35) is satisfied and, therefore, the combined method (the Richardson Extrapolation + the Backward Euler Formula) is stable on the imaginary axis.

(C) Verification of (21): $\mathbf{R(\mu)}$ is a ratio of two polynomials, $\mathbf{P(\mu)}$ and $\mathbf{Q(\mu)}$; see (19). It is well-known that polynomials are analytic functions and a ratio of two polynomials is analytic function in $\mathbf{C^-}$ if the denominator $\mathbf{Q(\mu)}$ has no roots in $\mathbf{C^-}$. The roots of the denominator $\mathbf{Q(\mu)}$ of the rational function $\mathbf{R(\mu)}$ are $\mu_1 = 1$ (single root) and $\mu_{2,3} = 2$ (double root). This means that $\mathbf{R(\mu)}$ is analytic in $\mathbf{C^-}$ and, therefore, (21) holds.

(D) Verification of (22): The results proved in (B) and (C) show that the combined method based on the use of the Richardson Extrapolation with the Backward Euler Formula is A-stable. Therefore, according to Definition 2, the method will be also L-stable when (22) holds. It is immediately seen that both terms in the right-hand-side of (30) tend to zero as $\mu \to \infty$. Thus, the combined method is L-stable, which completes the proof of the theorem. ■

Remark 6: The fact that the combination of the Richardson Extrapolation and the Backward Euler Formula is A-stable is demonstrated geometrically in Fig 9.5 on p. 151 in Hairer and Wanner (1991) by plotting the stability region of the combined method. An analytic proof of the A-stability of the combined method is given in the first part of Theorem 4. The ideas on which the proof is based are rather general and can successfully be used in connection with more complicated numerical methods (research results related to a class of numerical methods for solving systems of ODEs will be published in the near future). It should also be stressed here that the final result proved in the second part of Theorem 4 shows that the combined method has much better stability properties: not only is it A-stable, but L-stability is also proved. ■

7. Numerical experiments

Many numerical experiments were performed in order to show that (a) it is possible to improve the accuracy of the numerical results when the Richardson Extrapolation is used (also if very difficult non-linear atmospheric chemistry schemes are to be treated), (b) the computations can become unstable when the combination of the Trapezoidal Rule and the Active Richardson Extrapolation is used, (c) the application of the Active Richardson Extrapolation with the Backward Euler Formula is leading to a stable computational process, (d) the computing time can sometimes be reduced considerably by utilizing a specially designed sparse matrix solver and (e) the computational cost of the Richardson Extrapolation is much less than that of the corresponding underlying method when a prescribed accuracy has to be achieved.

7.1. The atmospheric chemistry scheme used in the experiments

An atmospheric chemistry scheme containing $\mathbf{s = 56}$ species has been selected and used in the experiments results of which will be presented below. Such schemes are used in several well-known environmental models (for example, in the EMEP models, Simpson, 2003, and UNI-DEM, Zlatev and Dimov, 2006). The atmospheric chemistry scheme is described mathematically as a non-linear system of type (1) containing $\mathbf{56}$ ODEs. This numerical example is extremely difficult because (a) it is badly scaled and (b) some chemical species vary very quickly during the periods of changes from day-time to night-time and from night-time to day-time.

The badly scaling is demonstrated by the results given in Table 1, where the maximal, minimal and mean values of the concentrations of several chemical species during a time-interval of 24 hours are given. It is clearly seen that while some chemical species (as nitrogen di-oxide and ozone) does not vary too much, other chemical species vary in a very wide range (sometimes by many orders of magnitude; see also Fig. 3 and Fig. 4).

The steep gradients of some of the concentrations in the critical parts of the time-interval (changes from day-time to night-time and from night-time to day-time) are demonstrated in the plots drawn in Fig. 3 and Fig. 4. The concentrations of some species are growing during the day (an example is given in Fig. 3), while other concentrations are growing during the night (see Fig. 4).

Table 1: Maximal, minimal and mean values of concentrations of some chemical species during a time-period of 24 hours. The units are numbers of molecules per cubic centimetre.

Species	Maximal value	Minimal value	Mean value
NO	2.5E+09	8.4E+04	5.5E+08
NO_2	2.4E+10	3.7E+08	4.3E+09
Ozone	1.8E+12	1.4E+12	1.5E+12
OH	2.3E+07	3.3E+04	6.3E+06
Isoprene	3.7E+09	1.1E+06	1.5E+09

7.3. Organization of the computations

The atmospheric chemistry scheme discussed in the previous sub-section was handled on the time-interval $[a,b]=[43200,129600]$. The value $a=43200$ corresponds to twelve o'clock at the noon, while $b=129600$ corresponds to twelve o'clock on the next day. Thus, the length of the time-interval is 24 hours (86400 seconds) and it contains important changes from day-time to night-time and from night-time to day-time.

Several sequences of 19 runs have been treated in different experiments. In each experiment the first run is performed by using $N=168$ time-steps (this means that the time-stepsize is $h \approx 514.285$ seconds). After that the stepsize h was halved eighteen times (this implies that the number N of time-steps is doubled in the beginning of every successive run). The behavior of the errors in each sequence of 19 runs was studied. The error made in an arbitrary run is measured in the following way. Denote:

$$\textbf{(50)} \qquad ERR_m = \max_{k=1,2,\ldots,56} \left(\frac{\left| y_{m,k} - y^{ref}_{m,k} \right|}{\max\left(\left| y^{ref}_{m,k} \right|, 1.0 \right)} \right),$$

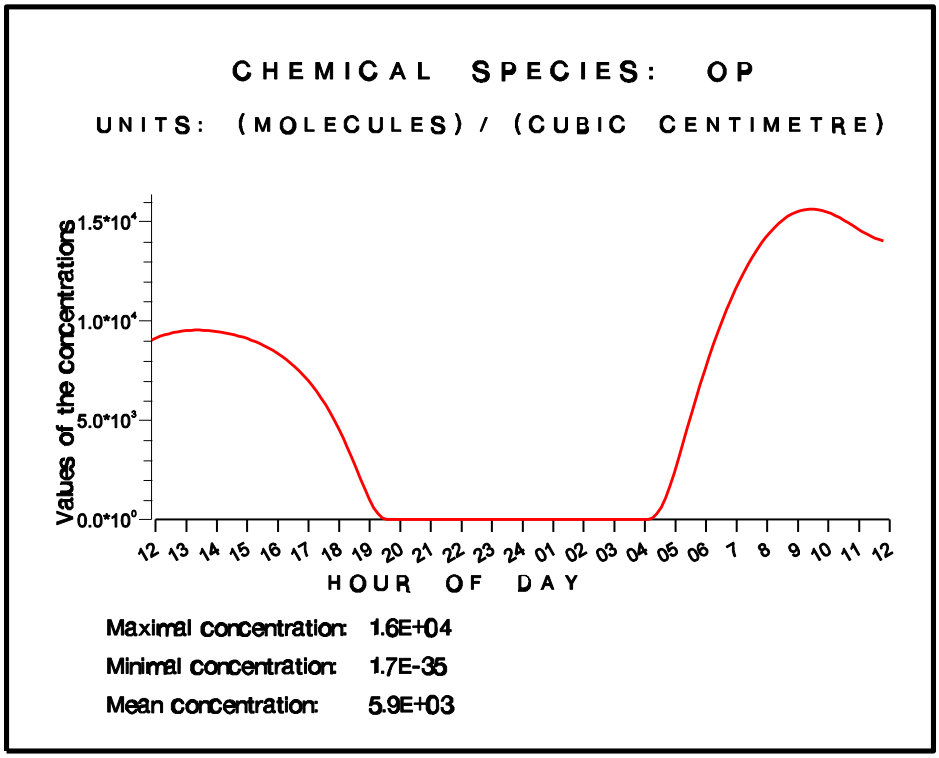

Figure 3: Diurnal variation of the concentrations of the chemical species **OP** .

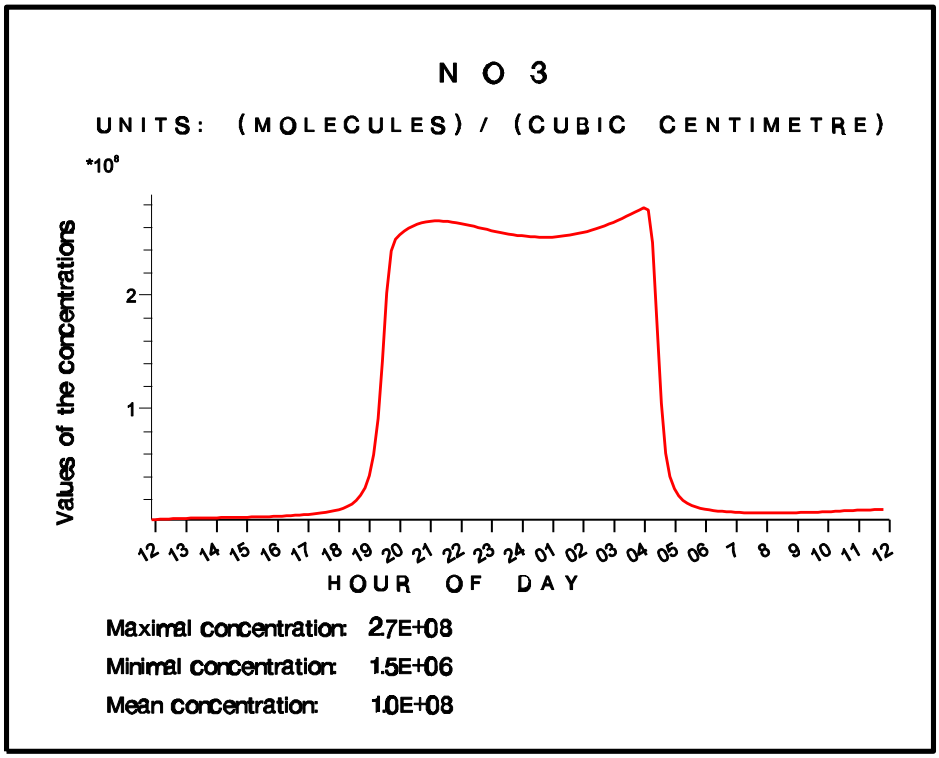

Figure 4: Diurnal variation of the concentrations of the chemical species NO_3 .

where $y_{m,k}$ and $y^{ref}_{m,k}$ are the calculated value and the reference solution of the k^{th} chemical species at time $t_m = t_0 + mh_0$ (where $m = 1,2,\ldots,168$ and $h_0 \approx 514.285$ is the time-stepsize that has been used in the first run). The reference solution was calculated by using a three-stage fifth-order L-stable fully implicit Runge-Kutta algorithm (see Butcher, 2003 or Hairer and Wanner, 1991) with $N = 52848230$ and $h_{ref} \approx 0.0000\mathbf{6131}$. It is clear from the above discussion that only the values of the reference solution at the grid-points of the coarsest grid used in the first run have been stored and applied in the evaluation of the error (it is, of course, also possible to store all values of the reference solution, but such an action will increase tremendously the storage requirements).

The global error made during the computations is estimated by

$$\mathbf{(51)} \qquad ERR = \max_{m=1,2,\ldots,168} \left(ERR_m \right).$$

The crucial task is to eliminate the influence of the rounding errors when the quantities involved in (50) and (51) are calculated. Normally this task can easily be accomplished when double precision arithmetic is used. Unfortunately, this is not true when the atmospheric chemistry scheme is handled. The difficulty can be explained as follows. If the problem is stiff, and the atmospheric chemistry scheme is, as mentioned above, a stiff non-linear system of ODEs, then implicit numerical methods are to be used. The application of such numerical methods leads to the solution of systems of non-linear algebraic equations, which are normally treated at each time-step by the Newton Iterative Method (to be discussed in the next sub-section). This means that long sequences of systems of linear algebraic equations are to be handled. Normally this does not cause great problems. However, the atmospheric chemistry schemes are very badly scaled and the condition numbers of the involved matrices are very large. It was found (by applying an LAPACK subroutine for finding eigenvalues and condition numbers from Anderson et al., 1992) that the condition numbers of the matrices involved in the Newton Iterative Process vary in the interval $[4.56E+08,\ 9.27E+12]$. Simple application of error analysis arguments from Wilkinson (1963) indicate that there is a danger that the rounding errors will affect the fourth significant digit of the approximate solution on most of the existing computers when double precision arithmetic is used. Therefore, all computations reported in the next sub-sections were performed by using quadruple-precision (i.e. by using REAL*16 declarations for the real numbers and, thus, about 32-digit arithmetic) in order to eliminate the influence of the rounding errors affecting the first 16 significant digits of the computed approximate solutions.

7.4. Stopping criteria

Denote by J the Jacobian matrix of the vector function f from (1). The application of implicit methods in the solving (1), which is necessary when the problem is stiff, leads to the solution of a long sequence of non-linear systems of algebraic equations. The Newton Iterative Procedure is often used in the solution of these systems. Assume that the computations at step n are to be carried out. Then a linear system of algebraic equations:

$$\mathbf{(52)} \qquad (I - \gamma h\, J_n^k)\, \Delta y_n^k = c_n^k$$

has to be solved at the $\mathbf{k}^{\mathbf{th}}$ step of the Newton Iterative Procedure. The constant γ depends on the particular method for solving systems of ODEs ($\gamma = 1$ for the Backward Euler Formula, while $\gamma = 0.5$ when the Trapezoidal Rule is selected). Vector $\mathbf{c}_n^k$ depends also on the numerical method for solving ODEs: $\mathbf{c}_n^k = -\mathbf{y}_n^{k-1} + \mathbf{y}_{n-1} + \mathbf{h}\mathbf{f}(\mathbf{t}_n, \mathbf{y}_n^k)$ for the Backward Euler Formula and $\mathbf{c}_n^k = -\mathbf{y}_n^{k-1} + \mathbf{y}_{n-1} + 0.5\mathbf{h}\left[\mathbf{f}(\mathbf{t}_{n-1}, \mathbf{y}_{n-1}) + \mathbf{f}(\mathbf{t}_n, \mathbf{y}_n^k)\right]$ for the Trapezoidal Rule. The current approximation of the solution of (1) is updated by

$$\mathbf{(53)} \qquad \mathbf{y}_n^k = \mathbf{y}_n^{k-1} + \Delta\mathbf{y}_n^k .$$

Normally, a Modified Newton Iterative Procedure in which $\mathbf{I} - \gamma\mathbf{h}\mathbf{J}_n^k$ is replaced by some approximation $\mathbf{A}_m^j$ is used. Matrix $\mathbf{A}_m^j$ is as a rule obtained at iteration $\mathbf{j}$ during some time-step $\mathbf{m} \le \mathbf{n}$ .The order of the Modified Newton Iterative Procedure is one, while the classical Newton Iterative Procedure carried out by (52) – (53) is of order two (at least when the starting approximation is sufficiently close to the exact solution; see Kantorovich and Akilov, 1964, or Zlatev, 1981). This will as a rule lead to an increase of the number of iterations when the Modified Newton Iterative Procedure is used. On the other hand, the number of matrix factorizations is normally reduced substantially when the Modified Newton Iterative Procedure is used and this results as a rule in reduction of the computing time.

It is important to stop correctly the iterative process defined by (52) and (53). Assume that the Backward Euler Formula is used directly and that the $\mathbf{i}^{\mathbf{th}}$ job mentioned in the previous sub-section, i.e. $\mathbf{i} \in \{1, 2, \ldots, 19\}$, is to be treated. Consider the quantities:

$$\mathbf{(54)} \qquad \mathbf{EST}_n = \frac{\left\| \Delta\mathbf{y}_n^k \right\|}{\max\left(\left\| \mathbf{y}_n^k \right\|, 1.0 \right)}, \qquad \mathbf{ACCUR} = \max\left(10^{-i-2}, 10^{-28}\right).$$

The iterative process carried out at time step $\mathbf{n}$ is stopped when $\mathbf{EST}_n$ becomes less than $\mathbf{ACCUR}$ and $\mathbf{y}_n$ is set to be equal to the last iterate $\mathbf{y}_n^k$ when this happens. It should be mentioned here that sometimes it is desirable to ensure that the stopping criterion used in the solution of the non-linear systems of algebraic equations will have no influence on the convergence properties of the selected numerical methods for solving systems of ODEs. It is necessary to use much smaller values of $\mathbf{ACCUR}$ if this is the case. $\mathbf{ACCUR} = 10^{-28}$ was used in the calculation of the results presented in Table 2 – Table 5 and in Table 7. This choice (together with the fact that quadruple precision is used in the computations) guarantees that neither the stopping criterion used in the Newton Iterative Procedure nor the rounding errors would affect the convergence of the numerical method for solving systems of ODEs when the experiments related to Table 2 – Table 5 and Table 7 were run.

Assume that $\mathbf{h}_i$ is the time-stepsize used in the $\mathbf{i}^{\mathbf{th}}$ run. Consider some time-step $\mathbf{n}$. If the Newton Iterative Procedure is not convergent or if it is slowly convergent then the time-stepsize is reduced by a factor of two. This could happen several times at a given time-step. The remaining part of the interval $[\mathbf{t}_{n-1}, \mathbf{t}_n]$ is calculated by using the reduced time-stepsize, however, the computations at the next time-step $\mathbf{n}+1$ are started with a time-stepsize $\mathbf{h}_i$ (i.e. calculations with a reduced time-

stepsize are carried out only when there are difficulties with the convergence of the Newton Iterative Procedure).

7.5. Instability of the Active Richardson Extrapolation when the Trapezoidal Rule is used

Assume that the Trapezoidal Rule is used. One should expect the Passive Richardson Extrapolation to be stable (Theorem 2), while the Active Richardson Extrapolation will in general lead to unstable computations (Theorem 3). Many experiments performed with the atmospheric chemistry scheme demonstrate the validity of these two statements. Some of the obtained results are given in Table 2.

Several important conclusions can be drawn from the results shown in Table 2 (it should be mentioned here that many other runs were also performed and the conclusions were similar):

- The order of the Trapezoidal Rule is two. Therefore, it should be expected that doubling the number $\mathbf{N}$ of time-steps (which leads to a decrease of the time-stepsize $\mathbf{h = (129600 - 43200) / N = 86400 / N}$ by a factor of two) will in general result in an improvement of the accuracy by a factor of four. It is seen that in the beginning this is the case. However, after the seventh run the rate of convergence is shifting from four to two. It is not clear why the rate of convergence is deteriorated.
- The application of the Active Richardson Extrapolation with the Trapezoidal Rule leads to unstable computations. This is clearly a consequence of Theorem 2. It is only necessary to explain here how the instability is detected. Two stability checks are carried out. The first check is based on monitoring the norm of the calculated approximate solutions: if this norm becomes $\mathbf{10^{10}}$ times the norm of the initial vector, then the computations are stopped and the computational process is declared to be unstable. The second check is based on the convergence of the Newton Iterative Process. It was mentioned in the previous sub-section that if this process is not convergent or very slowly convergent at some time-step $\mathbf{n}$, then the stepsize $\mathbf{h}$ is halved. This can happen several times at time-step $\mathbf{n}$. If the reduced time-stepsize become less than $\mathbf{10^{-5}\, h}$, then the computational process is stopped and declared to be unstable. If the time-stepsize has been reduced at time-step $\mathbf{n}$, then the remaining calculations in the interval from $\mathbf{t_{n-1}}$ to $\mathbf{t_n}$ are performed with the reduced time-stepsize (with the reduced time-stepsizes if the time-stepsize has been reduced several times), however an attempt is carried out to perform the next time-step $\mathbf{n+1}$ (i.e. to proceed from $\mathbf{t_n}$ to $\mathbf{t_{n+1}}$) with the time-stepsize $\mathbf{h = (129600 - 43200) / N = 86400 / N}$ used in the current run.
- The order of the Passive Richardson Extrapolation with the Trapezoidal Rule is three. Therefore, it should be expected that doubling the number $\mathbf{N}$ of time-steps, which leads to a decrease of the time-stepsize $\mathbf{h = (129600 - 43200) / N = 86400 / N}$ by a factor of two, will in general result in an improvement of the accuracy by a factor of eight. It is seen from Table 2 that this is not the case (excepting perhaps the first three runs). However, it is also seen that the Passive Richardson Extrapolation with the Trapezoidal Rule gives consistently more accurate results than those obtained when the Trapezoidal Rule is applied directly.

Table 2: Numerical results obtained in 19 runs of (a) the direct implementation of the Trapezoidal Rule, (b) the Active Richardson Extrapolation with the Trapezoidal Rule and (c) the Passive Richardson Extrapolation with the Trapezoidal Rule are given. The error obtained by (51) is given in the columns under "Accuracy". The ratios of two successive errors are given in the columns under "Rate". "Unstable" means that the code detected that the computations are not stable, while "n.a." stands for not applicable.

Job Number	Number of steps	Direct Implementation		Richardson Extrapolation			
				Active		Passive	
		Accuracy	Rate	Accuracy	Rate	Accuracy	Rate
1	168	3.605E-01	-	Unstable	n.a.	4.028E-02	-
2	336	7.785E-02	4.631	Unstable	n.a.	3.246E-03	12.407
3	672	1.965E-02	3.961	Unstable	n.a.	1.329E-03	2.443
4	1344	4.915E-03	3.998	Unstable	n.a.	1.462E-04	9.091
5	2688	1.228E-03	4.001	Unstable	n.a.	5.823E-05	2.510
6	5376	3.071E-04	4.000	Unstable	n.a.	3.765E-05	1.547
7	10752	7.677E-05	4.000	Unstable	n.a.	2.229E-05	1.689
8	21504	2.811E-05	2.731	Unstable	n.a.	1.216E-05	1.833
9	43008	1.615E-05	1.741	Unstable	n.a.	6.300E-06	1.930
10	86016	8.761E-06	1.843	Unstable	n.a.	3.188E-06	1.976
11	172032	4.581E-06	1.912	Unstable	n.a.	1.600E-06	1.993
12	344064	2.345E-06	1.954	Unstable	n.a.	8.007E-07	1.998
13	688128	1.187E-06	1.976	Unstable	n.a.	4.005E-07	1.999
14	1376256	5.970E-07	1.988	Unstable	n.a.	2.002E-07	2.000
15	2752512	2.994E-07	1.994	Unstable	n.a.	1.001E-07	2.000
16	5505024	1.499E-07	1.997	Unstable	n.a.	5.005E-08	2.000
17	11010048	7.503E-08	1.998	Unstable	n.a.	2.503E-08	2.000
18	22020096	3.753E-08	1.999	Unstable	n.a.	1.252E-08	2.000
19	44040192	1.877E-08	2.000	Unstable	n.a.	6.257E-09	2.000

7.6. Using the Richardson Extrapolation with the Backward Euler Formula

Assume that the Richardson Extrapolation is used together with Backward Euler Formula. Both the active and the passive implementation of the Richardson Extrapolation should be stable in this case (Theorem 4). The results in Table 3 show clearly that the stability is preserved. The following conclusions can additionally be drawn from the results in Table 3 as well as from the results obtained in several other runs.

- The order of the Backward Euler Formula is one. Therefore, it should be expected that doubling the number $\mathbf{N}$ of time-steps (which leads to a decrease of the time-stepsize $\mathbf{h = (129600 - 43200) / N = 86400 / N}$ by a factor of two) will in general result in an improvement of the accuracy by a factor approximately equal to two. It is seen that this is the case for all eighteen runs after the first one.
- The application of the Active Richardson Extrapolation with the Backward Euler Formula leads, as predicted by Theorem 3, to stable computations. The order of the combined method is two and it should be expected that doubling the number $\mathbf{N}$ of time-steps will in general lead to an improvement of the accuracy by a factor approximately equal to four. It is seen that the Active Richardson Extrapolation behaves as a numerical method of order two when it is combined with the Backward Euler Formula.

- The order of the combination of the Passive Richardson Extrapolation with the Backward Euler Formula should also be two and it is seen that the combined method behaves as a second-order numerical method.

Table 3: Numerical results obtained in 19 runs of (a) the direct implementation of the Backward Euler Formula, (b) the Active Richardson Extrapolation with the Backward Euler Formula and (c) the Passive Richardson Extrapolation with the Backward Euler Formula are given. The error obtained by (51) is given in the columns under "Accuracy". The ratios of two successive errors are given in the columns under "Rate".

Job Number	Number of steps	Direct Implementation		Richardson Extrapolation			
				Active		Passive	
		Accuracy	Rate	Accuracy	Rate	Accuracy	Rate
1	168	2.564E+00	-	3.337E-01	-	3.337E-01	-
2	336	1.271E+00	2.017	1.719E-01	1.942	2.981E-01	1.120
3	672	6.227E-01	2.041	5.473E-02	3.140	2.526E-02	11.801
4	1344	3.063E-01	2.033	7.708E-03	7.100	6.727E-03	3.749
5	2688	1.516E-01	2.020	1.960E-03	3.933	1.739E-03	3.874
6	5376	7.536E-02	2.011	5.453E-04	3.594	4.417E-04	3.937
7	10752	3.757E-02	2.006	1.455E-04	3.749	1.113E-04	3.969
8	21504	1.876E-02	2.003	3.765E-05	3.864	2.793E-05	3.984
9	43008	9.371E-03	2.002	9.583E-06	3.929	6.997E-06	3.992
10	86016	4.684E-03	2.001	2.418E-06	3.963	1.751E-06	3.996
11	172032	2.341E-03	2.000	6.072E-07	3.981	4.379E-07	3.998
12	344064	1.171E-03	2.000	1.522E-07	3.991	1.095E-07	3.999
13	688128	5.853E-04	2.000	3.809E-08	3.995	2.844E-08	3.850
14	1376256	2.926E-04	2.000	9.526E-09	3.998	7.266E-09	3.914
15	2752512	1.463E-04	2.000	2.382E-09	4.000	1.836E-09	3.957
16	5505024	7.315E-05	2.000	5.951E-10	4.002	4.613E-10	3.981
17	11010048	3.658E-05	2.000	1.484E-10	4.011	1.153E-10	4.001
18	22020096	1.829E-05	2.000	3.665E-11	4.047	2.853E-11	4.042
19	44040192	9.144E-05	2.000	8.727E-12	4.200	6.796E-12	4.197

7.7. Combining the Richardson Extrapolation with the Marchuk-Strang Splitting Procedure

Consider the following initial value problem:

$$\mathbf{(55)} \qquad \frac{\mathbf{dy}}{\mathbf{dt}} = \mathbf{f}_1(\mathbf{t},\mathbf{y}) + \mathbf{f}_2(\mathbf{t},\mathbf{y}), \quad \mathbf{t} \in [\mathbf{a},\mathbf{b}], \quad \mathbf{b} > \mathbf{a}, \quad \mathbf{y} \in \Re^{\mathbf{s}}, \quad \mathbf{f}_1 \in \Re^{\mathbf{s}}, \quad \mathbf{f}_2 \in \Re^{\mathbf{s}}, \quad \mathbf{s} \geq \mathbf{1},$$

instead of (1). The Active Richardson Extrapolation can be combined with the Backward Euler Formula and the Marchuk-Strang Splitting Procedure (about this splitting procedure see Marchuk, 1968, Strang, 1968, Dimov et al., 2004) by performing successively three computational steps:

Step 1: Use a large time-stepsize $\mathbf{h}$ to calculate an approximation $\mathbf{z}_n$ of $\mathbf{y}(\mathbf{t}_n)$ starting with the approximation $\mathbf{y}_{n-1}$ obtained at the previous time-step:

$$\mathbf{(56)} \qquad \mathbf{z}_n^{(1)} = \mathbf{y}_{n-1} + \mathbf{0.5h}\mathbf{f}_1(\mathbf{t}_n, \mathbf{z}_n^{(1)}),$$

$$\mathbf{(57)} \qquad \mathbf{z}_n^{(2)} = \mathbf{z}_n^{(1)} + \mathbf{h}\mathbf{f}_2(\mathbf{t}_n, \mathbf{z}_n^{(2)}),$$

(58) $\mathbf{z_n^{(3)} = z_n^{(2)} + 0.5hf_1(t_n, z_n^{(3)})},$

(59) $\mathbf{z_n = z_n^{(3)}}.$ ■

Step 2: Perform two small time-steps with a time-stepsize $\mathbf{0.5h}$ to calculate a second approximation $\mathbf{w_n}$ to $\mathbf{y(t_n)}$ starting again with the approximation $\mathbf{y_{n-1}}$ obtained at the previous time-step:

(60) $\mathbf{w_{n-0.5}^{(1)} = y_{n-1} + 0.25hf_1(t_n, w_{n-0.5}^{(1)})},$

(61) $\mathbf{w_{n-0.5}^{(2)} = w_{n-0.5}^{(1)} + 0.5hf_2(t_n, w_{n-0.5}^{(2)})},$

(62) $\mathbf{w_{n-0.5}^{(3)} = w_{n-0.5}^{(2)} + 0.25hf_1(t_n, w_{n-0.5}^{(3)})},$

(63) $\mathbf{w_n^{(1)} = w_{n-0.5}^{(3)} + 0.25hf_1(t_n, w_n^{(1)})},$

(64) $\mathbf{w_n^{(2)} = w_n^{(1)} + 0.5hf_2(t_n, w_n^{(2)})},$

(65) $\mathbf{w_n^{(3)} = w_n^{(2)} + 0.25hf_1(t_n, w_n^{(3)})},$

(66) $\mathbf{w_n = w_n^{(3)}}.$ ■

Step 3: Apply the formula for computing the Richardson Extrapolation with $\mathbf{p = 1}$ to compute an improved approximation $\mathbf{y_n}$ of $\mathbf{y(t_n)}$:

(67) $\mathbf{y_n = 2w_n - z_n}.$ ■

Note that everything is prepared for the computation of the next approximation $\mathbf{y_{n+1}}$ after the performance of the calculations with (56) – (67). It should also be mentioned here that the computational process based on (56) – (67) can obviously be started, because it is assumed that an initial value vector $\mathbf{y_0 = y(a)}$ is given.

Remark 7: Combining the Passive Richardson Extrapolation with the Backward Euler Formula and the Marchuk-Strang Splitting Procedure can be achieved by replacing $\mathbf{y_{n-1}}$ with $\mathbf{z_{n-1}}$ in Step 1 and with $\mathbf{w_{n-1}}$ in Step 2. ■

Remark 8: The Richardson Extrapolation and the Marchuk-Strang Splitting Procedure can also be combined with the Trapezoidal Rule: one has to apply the Trapezoidal Rule instead of the Backward Euler Formula in the right-hand sides of (56) – (58) and (60) – (65). Moreover, it is more appropriate to use the Richardson Extrapolation with $\mathbf{p = 2}$ which will give

(68) $$\mathbf{y_n = \frac{4w_n - z_n}{3}}$$

instead of (67). ■

Remark 9: The Marchuk-Strang Splitting Procedure can be implemented directly (i.e. no Richardson Extrapolation). The needed formulae can be obtained from (56) - (59) by replacing $\mathbf{z}$ with $\mathbf{y}$. ■

Some results obtained by using the combination: the Richardson Extrapolation + the Marchuk-Strang Splitting Procedure + the Trapezoidal Rule are given in Table 4. Results obtained when the Trapezoidal Rule is replaced by the Backward Euler Formula are given in Table 5. The same numerical example as in the previous sub-section is used. The right-hand-side of (1) is split into two functions, see (55), by using the following rules: (a) all species that react with ozone are combined in the first component $\mathbf{f}_1$, while the remaining species form $\mathbf{f}_2$.

Table 4: Numerical results obtained in 19 runs of (a) the direct implementation of the Marchik-Strang Splitting Procedure and the Trapezoidal Rule, (b) the Active Richardson Extrapolation with the Marchik-Strang Splitting Procedure and the Trapezoidal Rule and (c) the Passive Richardson Extrapolation with the Marchik-Strang Splitting Procedure and the Trapezoidal Rule are given. The error obtained by (51) is given in the columns under "Accuracy". The ratios of two successive errors are given in the columns under "Rate". "Unstable" means that the code detected that the computations are not stable, while "n.a." stands for not applicable.

Job Number	Number of steps	Direct Implementation		Richardson Extrapolation			
				Active		Passive	
		Accuracy	Rate	Accuracy	Rate	Accuracy	Rate
1	168	2.230E+00	-	Unstable	n.a.	6.826E-01	-
2	336	1.919E-01	11.626	Unstable	n.a.	2.594E-01	2.631
3	672	5.531E-02	3.469	Unstable	n.a.	8.970E-02	2.892
4	1344	1.360E-02	4.068	Unstable	n.a.	3.266E-02	2.746
5	2688	3.711E-03	3.664	Unstable	n.a.	1.312E-02	2.489
6	5376	9.472E-04	3.918	2.440E-04	-	5.757E-03	2.280
7	10752	2.384E-04	3.973	4.119E-05	5.922	2.677E-03	2.150
8	21504	5.980E-05	3.987	8.548E-06	4.819	1.289E-03	2.078
9	43008	1.501E-05	3.983	3.187E-06	2.682	6.317E-04	2.040
10	86016	4.578E-06	3.279	1.600E-06	1.992	3.128E-04	2.020
11	172032	2.344E-06	1.953	8.007E-07	1.998	1.556E-04	2.010
12	344064	1.187E-06	1.976	4.005E-07	1.999	7.760E-05	2.005
13	688128	5.970E-07	1.988	2.002E-07	2.000	3.875E-05	2.003
14	1376256	2.994E-07	1.994	1.001E-07	2.000	1.936E-05	2.001
15	2752512	1.499E-07	1.997	5.006E-08	2.000	9.679E-06	2.000
16	5505024	7.503E-08	1.998	2.503E-08	2.000	4.839E-06	2.000
17	11010048	3.753E-08	1.999	1.252E-08	2.000	2.419E-06	2.000
18	22020096	1.877E-08	2.000	6.257E-09	2.000	1.210E-06	2.000
19	44040192	9.385E-09	2.000	3.129E-09	2.000	6.051E-7	2.000

Three conclusions can be drawn by studying carefully the results shown in Table 4:

- If the Richardson Extrapolation is not used, then the order of the combined method (the Marchuk-Strang Splitting Procedure + the Trapezoidal Rule) is two. Therefore, it should be expected that doubling the number $\mathbf{N}$ of time-steps will in general result in an improvement of

the accuracy by a factor of four. It is seen that this expectation is fulfilled (excepting the results after the tenth run).

- The application of the Active Richardson Extrapolation with the Marchuk-Strang Splitting Procedure and the Trapezoidal Rule causes instability in the first five runs, (this is a consequence of Theorem 2), but after that the computations are stable and the convergence rate is pretty good (in runs 6 and 7 at least). The combined method is of order three. This means that doubling the number of time-steps should result in an improvement of the accuracy by a factor of eight. This factor is not achieved, but in runs 7 and 8 the factors are considerably larger than four.
- The order of the Passive Richardson Extrapolation with the Marchuk-Strang Splitting Procedure and the Trapezoidal Rule is also three. Therefore, it should be expected that doubling the number **N** of time-steps will in general lead to an improvement of the accuracy by a factor of eight. It is seen from Table 4 that this is not the case. In fact the convergence rate is rather poor: the combined method behaves as a first-order method. It is not very clear why this is so.

Table 5: Numerical results obtained in 19 runs of (a) the direct implementation of the Marchik-Strang Splitting Procedure and the Backward Euler Formula, (b) the Active Richardson Extrapolation with the Marchik-Strang Splitting Procedure and the Backward Euler Formula and (c) the Passive Richardson Extrapolation with the Marchik-Strang Splitting Procedure and the Backward Euler Formula are given. The error obtained by (51) is given in the columns under "Accuracy". The ratios of two successive errors are given in the columns under "Rate".

Job Number	Number of steps	Direct Implementation		Richardson Extrapolation			
				Active		Passive	
		Accuracy	Rate	Accuracy	Rate	Accuracy	Rate
1	168	2.456E+00	-	2.133E-01	-	8.484E-01	-
2	336	1.011E+00	2.430	1.090E-01	1.957	2.333E-01	3.636
3	672	4.901E-01	2.062	4.465E-02	2.441	6.850E-02	3.407
4	1344	2.389E-01	2.051	1.677E-02	2.662	3.315E-02	2.066
5	2688	1.175E-01	2.033	6.512E-03	2.576	1.621E-02	2.046
6	5376	5.819E-02	2.019	2.559E-03	2.545	7.940E-03	2.041
7	10752	2.819E-02	2.010	1.059E-03	2.417	3.893E-03	2.040
8	21504	1.444E-02	2.005	4.168E-03	2.540	1.913E-03	2.034
9	43008	7.208E-03	2.001	1.543E-04	2.702	9.449E-04	2.025
10	86016	3.602E-03	2.001	5.253E-05	2.937	4.687E-04	2.016
11	172032	1.800E-03	2.001	1.651E-05	3.183	2.333E-04	2.009
12	344064	9.000E-04	2.000	4.865E-06	3.393	1.164E-04	2.005
13	688128	4.499E-04	2.000	1.439E-06	3.381	5.809E-04	2.003
14	1376256	2.250E-04	2.000	3.341E-07	3.822	2.905E-04	2.001
15	2752512	1.125E-04	2.000	8.554E-08	3.906	1.552E-04	2.000
16	5505024	5.623E-05	2.000	2.165E-08	3.523	7.258E-05	2.000
17	11010048	2.812E-05	2.000	5.445E-09	3.975	3.629E-05	2.000
18	22020096	1.406E-05	2.000	1.365E-09	3.989	1.814E-05	2.000
19	44040192	7.030E-06	2.000	3.391E-10	3.999	9.071E-06	2.000

Three conclusions can be drawn by studying carefully the results shown in Table 5:

- The order of the combined method (the Marchuk-Strang Splitting Procedure + the Backward Euler Formula) is one, because the error from the first-order Backward Euler Formula will be dominating over the error from the second-order Marchuk-Strang Splitting Procedure. Therefore, it should be expected that doubling the number **N** of time-steps will in general

result in an improvement of the accuracy by a factor of two. It is seen that this expectation is fulfilled.

- The application of the Active Richardson Extrapolation with the Marchuk-Strang Splitting Procedure and the Backward Euler Formula leads to a second-order numerical method. One should expect factor "Rate" to be approximately four. It is clearly seen that "Rate" is increased when the number of time-steps grows and the method behaves as a second-order numerical method at the end of computations.
- The order of the Passive Richardson Extrapolation with the Marchuk-Strang Splitting Procedure and the Backward Euler Formula should also be two. Therefore, it should be expected that doubling the number **N** of time-steps will in general lead to an improvement of the accuracy by a factor of four. It is seen from Table 5 that this is not the case. In fact the convergence rate is rather poor: the combined method behaves as a first-order method (excepting the first two runs). It is not very clear why this is so.

7.8. Treatment of linear systems of algebraic equations

The stiffness of the atmospheric chemistry scheme requires the use of implicit methods for solving systems of ODEs and, thus, solving linear systems of algebraic equations, which is a rather costly procedure. The performance of five different algorithms for solving linear systems of algebraic equations was tested. A short description of these algorithms is given below:

- **DENSE:** Direct method (based on the Gaussian Elimination) is used in the solution of (52) and the sparsity of the Jacobian matrix is not exploited. LAPACK routines (Anderson et al., 1992) for solving linear systems with dense coefficient matrices are called in this algorithm.
- **SPARSE-0:** Regular sparse matrix technique based on the pivotal strategy introduced in Zlatev (1980) is used. The algorithm is fully described in Zlatev et al. (1981) and Zlatev (1991). Other sparse matrix codes, as those applied in Duff et al. (1986), Demmel (1997) or Demmel et al. (1999a, 1999b), can also be applied. Comprehensive comparisons reported in Zlatev (1980), Zlatev et al. (1982), Zlatev and Dimov (2006) show that the algorithms from Zlatev et al. (1981) are at least competitive with the other algorithms.
- **SPARSE-1:** As **SPARSE-0**, but it is allowed to drop (to replace by zero) small elements (both before the start of the Gaussian Elimination and during every stage of the Gaussian Elimination). The implementation of the dropping device is based on the following rule. Consider stage $\mathbf{k}$ ($\mathbf{k=0,1,\ldots,s-1}$) of the Gaussian Elimination and the active part of the matrix at this stage containing elements $\mathbf{a_{ij}^k}$ with $\mathbf{i,j=s-k,s-k+1,\ldots,s}$. Let $\mathbf{a_{ij}^k}$ be an arbitrary element in the active part of the matrix at stage $\mathbf{k}$ and denote by $\mathbf{a_i^k}$ the largest in absolute value non-zero element in the active part of row $\mathbf{i}$ at stage $\mathbf{k}$ of the Gaussian Elimination. If $\left|\mathbf{a_{ij}^k}\right|/\mathbf{a_i^k} < \mathbf{RELTOL}$ is satisfied, then $\mathbf{a_{ij}^k}$ is dropped (not used in the further computations). $\mathbf{RELTOL=0.1}$ was used in this paper. An attempt to regain the accuracy lost because of the dropping procedure is carried out by using iterative refinement in an inner loop within any iteration of the Newton method. The iterative refinement algorithm used in **SPARSE-1** was introduced in Zlatev (1982); see also Zlatev (1991). Other iterative methods were also tried (the methods are based on preconditioned conjugate gradient

techniques and discussed in detail in Gallivan et al., 2003, Zlatev and Dimov, 2006). It was found out that iterative refinement works very well for the atmospheric chemistry scheme.

- **SPARSE-2:** As **SPARSE-1**, but no attempt is carried out to regain the accuracy lost during the Gaussian Elimination by performing iterative refinement. The application of this algorithm in solution of non-linear algebraic equations arising when implicit methods for solving stiff systems of ODEs are used can be considered as a Modified Newton Iterative Method in which the exact Jacobian matrix $\mathbf{I}-\Delta\mathbf{t}\,\mathbf{J}_{\mathbf{n}}^{\mathbf{k}}$ ($\mathbf{n}$ being the current time-step, while $\mathbf{k}$ is the current iteration number) is replaced by some $\mathbf{A}\approx\mathbf{I}-\Delta\mathbf{t}\,\mathbf{J}_{\mathbf{n}}^{\mathbf{k}}$. It should be noted here that approximations of the Jacobian matrix are also applied in the previous three algorithms. This is a commonly used approach: the same Jacobian matrix and its factorization are kept as long as possible when stiff systems of ODEs are solved (see, for example, Shampine, 1993). In the previous three algorithms the exact Jacobian matrix is calculated and used when the Newton Iterative Procedure does not converge or is slowly convergent. The stepsize is reduced when this happens during the calculations with **SPARSE-2**. This means that some extra time-steps are occasionally carried out when **SPARSE-2** is used, but the experiments indicate that this extra computing time is fully compensated by the fact that the inner loop performed at every time-step when **SPARSE-1** is used is skipped in **SPARSE-2**.
- **SPARSE-3:** As **SPARSE-2**, but a special sparse code was developed and implemented in **SPARSE-3**. This code has three major properties: (a) there are no loops, (b) no indirect addressing is used and (c) no integer arrays are applied. The algorithm is especially designed for the atmospheric chemistry scheme used in this paper and cannot directly be used in the treatment of other problems (while the previous four algorithms can be applied in the solution of any linear and sparse system of algebraic equations). **SPARSE-3** is described in Zlatev and Dimov (2006).

Numerical results, which were obtained when the Backward Euler Formula is used directly in the treatment of the atmospheric chemistry scheme, are given in Table 6. The following conclusions can be drawn by studying the results:

- It was expected that the dense code would be competitive with the sparse codes, because the matrix is rather small. This is clearly not the case. The application of any of the four sparse codes leads to a very considerable reduction of the computing time. It should be mentioned here that the dense code was competitive with the sparse codes when a smaller atmospheric chemistry scheme (with $\mathbf{s=35}$ instead of $\mathbf{s=56}$) was used. The smaller chemistry scheme was developed by Gery et al. (1989). Results obtained by this scheme are given in Alxandrov et al. (1997). The computing time for the dense Gaussian elimination is proportional to the number $\mathbf{O(s^3)}$ of arithmetic operations. It is difficult to evaluate the complexity of the sparse algorithms. A crude estimation can be obtained as follows. Denote by $\mathbf{r}$ and $\mathbf{c}$ the largest numbers of non-zero elements per row and per column respectively which appear in the active part of the matrix during the Gaussian Elimination. Then the number of arithmetic operations needed to solve the system of linear equations by sparse matrix technique can be evaluated by $\mathbf{O(rcs)}$. The term $\mathbf{O(rcs)}$ is, of course, an overestimation, but it allows us to explain why the sparse codes perform much better than the dense code when $\mathbf{s=56}$. The numbers of arithmetic operations grow roughly speaking linearly with $\mathbf{s}$ when sparse codes are used (because as a rule both $\mathbf{r}$ and $\mathbf{c}$ do not vary too

much when **s** is varied), while $\mathbf{O(s^3)}$ is telling us that small changes of **s** will results in substantial changes of the computing time when the dense algorithm is used.

- The specially designed for the atmospheric chemistry scheme code, **SPARSE-3**, performs rather well, but the two sparse codes **SPARSE-0** and **SPARSE-2**, which are more general and can be used in many other situations related to the solution of non-linear systems of ODEs, are not performing too badly. In some cases, **SPARSE-0** is performing even better than **SPARSE-3.**
- The code **SPARSE-1,** where both an outer iteration loop (the Modified Newton Iterative Procedure in connection with the non-linear systems of algebraic equations that are to be handled at every time-step) and an inner iteration loop (iterative refinement has to be used in the solution of the linear system of algebraic equations at every iteration of the Modified Newton Iterative Procedure) are to be carried out, is not very efficient in comparison with the other two sparse codes, but it is still considerably better than the dense code.

Table 6: Numerical results obtained in 19 runs with the Backward Euler Formula and five algorithms for the treatment of linear systems of algebraic equations are given in this table. The computing times, measured in seconds, are given in columns 3-7. The accuracy achieved when the SPARSE-3 algorithm is used is shown in the eighth column.

Job	Steps	DENSE	SPARSE-0	SPARSE-1	SPARSE-2	SPARSE-3	Accuracy
1	168	4.13	0.72	0.98	0.60	0.37	2.695E+00
2	336	9.17	1.54	3.30	2.34	1.26	1.288E+00
3	672	19.20	3.16	4.57	6.56	3.25	6.221E-01
4	1344	38.79	6.39	9.03	12.00	7.54	3.063E-01
5	2688	77.82	12.45	21.08	20.88	16.04	1.516E-01
6	5376	156.59	34.93	41.41	36.40	31.82	7.536E-02
7	10752	315.58	51.45	83.35	68.69	58.20	3.757E-02
8	21504	632.55	101.86	158.27	122.21	106.46	1.876E-02
9	43008	1267.63	203.30	304.56	230.09	196.42	9.371E-03
10	86016	2537.99	405.28	594.22	435.84	362.47	4.684E-03
11	172032	5077.83	806.23	1198.27	840.57	666.49	2.341E-03
12	344064	10160.90	1609.74	2390.99	1619.54	1266.13	1.171E-03
13	688128	20329.90	3214.60	4618.13	3102.74	2379.69	5.853E-04
14	1376256	40668.82	6416.62	9344.34	6144.19	4621.48	2.926E-04
15	2752512	81312.97	12778.42	18285.49	11949.24	9103.15	1.463E-04
16	5505024	162612.03	25529.66	35826.72	23587.94	17884.93	7.315E-05
17	11010048		50957.76	71087.69	45279.74	35657.37	3.658E-05
18	22020048		101917.18	147284.36	90449.57	70930.82	1.829E-05
19	44040192		203405.37		183371.06	139802.35	9.144E-06

- The accuracy obtained when **SPARSE-3** is run is given in the eighth column. The accuracy obtained with the other four codes is practically the same. Moreover, the accuracy shown in Table 6 is practically the same as that given in the third column of Table 3. The results presented in Table 3 are obtained by using a very stringent error $\mathbf{ACCUR = 10^{-28}}$ (as mentioned above, in order to ensure that the convergence properties of the numerical method for solving systems of ODEs are not affected by the selected stopping criterion), while $\mathbf{ACCUR=max\left(10^{-i-2},10^{-28}\right)}$ is used to calculate the results presented in Table 6. The results indicate clearly that the use of a flexible value of **ACCUR** does not affect the accuracy of the results, but leads to considerable reductions of the computing times (which can be concluded by taking the results in the columns three and four of Table 7 and

comparing them with the corresponding results in Table 6). Therefore, the choice of a proper value for parameter **ACCUR** seems to be very important.

7.9. Checking the computational cost of the Richardson Extrapolation

Three time-steps (one large and two small) with the underlying numerical method are necessary when one time-step of the Richardson Extrapolation is performed. This means that if the Richardson Extrapolation and the underlying numerical method are used with the same time-stepsize, then the computational cost of the Richardson Extrapolation will be about three times greater than that of the underlying numerical method. In many practical situations this factor will be less than three, but considerably larger than two (because the number of Newton iterations needed for each of the two small time-steps will normally be smaller than the corresponding number for the large time-step). However, the use of the Richardson Extrapolation leads also to an improved accuracy of the calculated approximations. Therefore, it is not relevant (and not fair either) to compare the Richardson Extrapolation with the direct application of the underlying numerical method for solving systems of ODEs under the assumption that both devices are run with equal number of time-steps. It is much more relevant to investigate how much work is needed in order to achieve the same accuracy with each of the two devices. The computing times needed in the efforts to achieve prescribed accuracy are given in Table 7. If the desired accuracy is $\mathbf{10^{-k}}$ ($\mathbf{k = -1, -2, \ldots, -11}$), then the computing time achieved in the first run in which the quantity **ERR** from (51) becomes less than $\mathbf{10^{-k}}$ is given in Table 7. The stringent stopping criterion $\mathbf{ACCUR = 10^{-28}}$ was used to obtain the results given in Table 7.

Table 7: Comparison of the computational costs (measured by the CPU times given in seconds) needed to achieve prescribed accuracy in the cases where (a) the Backward Euler Formula is implemented directly, (b) the Active Richardson Extrapolation with the Backward Euler Formula is used and (c) the Passive Richardson Extrapolation with the Backward Euler Formula is applied. The computing times measured in seconds are given in the columns under "Accuracy". The numbers of time-steps needed to obtain the desired accuracy are given in the columns under "Steps".

Desired accuracy of the solution	Direct Implementation of the Backward Euler Formula		Richardson Extrapolation			
			Active		Passive	
	CPU time	Steps	CPU time	Steps	CPU time	Steps
[1.0E-01, 1.0E-02]	274	5376	304	672	307	672
[1.0E-02, 1.0E-03]	862	43008	374	1344	378	1344
[1.0E-03, 1.0E-04]	7144	688128	661	5376	661	5376
[1.0E-04, 1.0E-05]	42384	5505024	1428	21504	1429	21504
[1.0E-05, 1.0E-06]	265421	44040192	2240	43008	2240	43008
[1.0E-06, 1.0E-07]	Not achieved in the 19 runs		6386	172032	6398	172032
[1.0E-07, 1.0E-08]	Not achieved in the 19 runs		19834	688128	19885	688128
[1-0E-08, 1.0E-09]	Not achieved in the 19 runs		35237	1376356	35245	1376256
[1.0E-09, 1.0E-10]	Not achieved in the 19 runs		119791	5505024	119796	5505024
[1.0E-10, 1.0E-11]	Not achieved in the 19 runs		410087	22020096	410846	22020048
[1.0E-11, 1.0E-12]	Not achieved in the 19 runs		777872	44040192	778974	44040192

Four conclusions can be drawn by studying the results shown in Table 7:

- The direct use of the Backward Euler Formula is slightly more efficient with regard to the computing time than the two implementations of the Richardson Extrapolation when the desired

accuracy is very low (**ERR** being less than $\mathbf{10^{-1}}$ and greater than $\mathbf{10^{-2}}$); compare the CPU times in the first row of Table 7.

- Accuracy better than $\mathbf{10^{-6}}$ has not been achieved in the 19 runs with the Backward Euler Formula reported in Table 3, while even accuracy better than $\mathbf{10^{-11}}$ is achievable when the Richardson extrapolation is used (see lines 5-11 in Table 7).
- The two implementations of the Richardson Extrapolation become much more efficient than the Backward Euler Formula when the accuracy requirement is increased (see the second, the third and the fourth lines of Table 7). If it desirable to achieve accuracy better than $\mathbf{10^{-3}}$, then the computing time spent with the Richardson Extrapolation is more than ten times smaller than the corresponding computing time for the Backward Euler Formula (compare the CPU times on the third line of Table 7). The reduction is by a factor approximately equal to thirty if the desired accuracy is $\mathbf{10^{-4}}$ (see the fourth line in Table 7).
- The major conclusion is that not only is the Richardson Extrapolation a powerful tool for improving the accuracy of the underlying numerical method, but it is also extremely efficient with regard to the computational cost (this being especially true when the accuracy requirement is not extremely low).

8. Concluding remarks

Several properties of the Richardson Extrapolations were studied in the previous sections of this paper. Some theorems related to the stability of the computational process were formulated and proved. Numerical results were given to demonstrate (a) the improvement of the accuracy by applying the Richardson Extrapolation and (b) the great savings in computing time achieved when a prescribed accuracy is required.

There are still many open problems which will be studied in the near future. Three of the open problems are listed below:

- It seems plausible to conjecture that Theorem 4 could be extended to any L-stable method for solving systems of ODEs (or, at least, for some other L-stable methods).
- It is desirable to obtain some results for strongly stable numerical methods for solving systems of ODEs, i.e. for numerical methods with $|\mathbf{R(\mu)}| \leq \mathbf{1}$ and $\mathbf{lim}\left(|\mathbf{R(\mu)}|\right) < \mathbf{1}$ as $\mu \to \infty$ (see more details about the definition of strongly stable methods for systems of ODEs in Hundsdorfer and Verwer, 2003).
- The comparison of the numerical results for the Active Richardson Extrapolation in Table 2 with the corresponding results in Table 4 indicates that the splitting procedures have some stabilizing effect on the numerical results. It is interesting to try to prove in a rigorous way when such a property of the splitting procedures takes place.

Acknowledgements:

Ágnes Havasi is a grantee of the Bolyai János Scholarship and her work was supported by Hungarian National Research Foundation (OTKA), grant: F61016.

A part of this work was done in New Zealand, during István Faragó's visit at the Otago University and supported by Hungarian National Research Foundation (OTKA), grant: K67819.

The numerical experiments were performed at the Danish Centre for Scientific Computing at the Technical University of Denmark. We should like to thank the specialists from the Centre for helping us to achieve high performance when the experiments were run.

References:

V. Alexandrov, A. Sameh, Y. Siddique and Z. Zlatev: *Numerical integration of chemical ODE problems arising in air pollution models*, Environmental Modelling and Assessment, **Vol. 2 (1997),** pp. 365-377.

E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen: *LAPACK: Users' Guide***.** SIAM, Philadelphia, **1992.**

K. Burrage: *Parallel and Sequential Methods for Ordinary Differential Equations*, Oxford University Press, Oxford-New York, **1992.**

J. C. Butcher: *Numerical Methods for Ordinary Differential Equations*, Second edition, Wiley, New York, **2003.**

G. Dahlquist: *A special stability problem for linear multistep methods*, BIT, **Vol. 3 (1963),** pp. 27-43.

I. Dimov, I. Farago, A. Havasi and Z. Zlatev: *Operator splitting and commutativity analysis of the Danish Eulerian Model*, Mathematics and Computers in Simulation, **Vol. 67 (2004),** pp. 217-233.

J. W. Demmel: *Applied Numerical Linear Algebra*, SIAM, Philadelphia, **1997.**

J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li and J. W. H. Liu: *An supernodal approach to sparse partial pivoting*, SIAM Journal of Matrix Analysis and Applications, **Vol. 20 (1999a),** pp. 720-755.

J. W. Demmel, J. R. Gilbert and X. S. Li: *An asynchronous parallel supernodal algorithm for sparse Gaussian elimination*, SIAM Journal of Matrix Analysis and Applications, **Vol. 20 (1999b),** pp. 915-952.

I. S. Duff, A. M. Erisman and J. K. Duff: *Direct Methods for Sparse Matrices*, Oxford University Press, Oxford-London, **1986.**

I. Faragó: *Numerical treatment of linear parabolic problems*, Doctoral dissertation, Eötvös Loránd University, Budapest, **2008.**

K. A. Gallivan, A. H. Sameh and Z. Zlatev: *"Comparison of ten methods for the solution of large and sparse linear algebraic systems "*. In: **"Numerical Methods and Applications"** (I. Dimov, I. Lirkov, S. Margenov and Z. Zlatev, eds.), pp. 24-35. Lecture Notes in Computer Science, No. 2542, Spinger-Verlag, Berlin, **2003.**

M. W. Gery, G. Z. Whitten, J. P. Killus and M. C. Dodge: *A photochemical kinetics mechanism for urban and regional modeling*, Journal of Geophysical Research, **Vol. 94 (1989),** pp. 12925-12956.

E. Hairer and G. Wanner: *Solving Ordinary Differential Equations: II Stiff and Differential-Algebraic Problems*, Springer-Verlag, Berlin, **1991.**

P. Henrici: *Discrete Variable Methods in Ordinary Differential Equations*, Wiley, New York, **1968.**

W. Hundsdorfer and J. G. Verwer: *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, Springer-Verlag, Berlin, **2003.**

L. V. Kantorovich and G. P. Akilov: *Functional Analysis in Normed Spaces*, Pergamon Press, Oxford, **1964.**

J. D. Lambert: *Numerical Methods for Ordinary Differential Equations*, Wiley, New York, **1991.**

G. I. Marchuk: *Some application of splitting-up methods to the solution of mathematical physics problems*. Applikace Matematiky, **Vol. 13, No. 2 (1968),** pp. 103-132.

L. F. Richardson: *The deferred approach to limit, I - Single lattice*, Philosophical Transactions of the Royal Society, London, **Ser. A, Vol. 226 (1927),** pp. 299-349.

L. F. Shampine: *Numerical Solution of Ordinary Differential Equation*, Chapman and Hall, New York-London, **1993.**

L. F. Shampine and M. K. Gordon: *Computer Solution of Ordinary Differential Equations: The Initial Value Problem*, W. H. Freeman and Company, San Francisco, **1975.**

D. Simpson, H. Fagerli, J. E. Jonson, S. G. Tsyro, P. Wind, J.-P.Tuovinen: *Transboundary Acidification, Eutrophication and Ground Level Ozone in Europe, Part I. Unified EMEP Model Description.* EMEP/MSC-W Status Report 1/2003. Norwegian Meteorological Institute, Oslo, Norway, **2003.**

G. Strang: *On the construction and comparison of difference schemes*, SIAM Journal on Numerical Analysis, **Vol. 5 (1968),** pp. 505-517.

J. H. Wilkinson: *The algebraic eigenvalue problem*, Oxford University Press, Oxford-London, **1965.**

C. R. Wylie: *Advanced Engineering Mathematics*, McGraw-Hill Kogakusha, Tokyo, **1975.**

Z. Zlatev: *On some pivotal strategies in Gaussian elimination by sparse technique*, SIAM Journal on Numerical Analysis, **Vol. 17 (1980),** pp. 18-30.

Z. Zlatev: *Modified diagonally implicit Runge-Kutta methods*, SIAM Journal on Scientific and Statistical Computing, **Vol. 2 (1981),** pp. 321-334.

Z. Zlatev: *Use of iterative refinement in the solution of sparse linear systems*, SIAM Journal on Numerical Analysis, **Vol. 19 (1982),** pp. 381-399.

Z. Zlatev: *Computational Methods for General Sparse Matrices*, Kluwer, Dordrecht-Boston-London, **1991.**

Z. Zlatev: *Computer Treatment of Large Air Pollution Models*, Kluwer, Dordrecht-Boston-London, **1995.**

Z. Zlatev and I. Dimov: *Computational and Environmental Challenges in Environmental Modelling*, Elsevier, Amsterdam-Boston-Heidelberg-London-New York-Oxford-Paris-San Diego-San Francisco-Singapore-Sidney-Tokyo, **2006.**

Z. Zlatev, J. Wasniewsky and K. Shaumburg: *Y12M – Solution of Large and Sparse Systems of Linear Algebraic Equations*, Spinger-Verlag, Berlin-Heidelberg-New York, **1981**.

Z. Zlatev, J. Wasniewsky and K. Shaumburg: *Comparison of two algorithms for solving large linear systems*, SIAM Journal on Scientific and Statistical Computing, **Vol. 3 (1982),** pp. 486-501.