Methods for Parallel Integration of Stiff Systems of ODEs *

Stig Skelboe †

Abstract

This paper presents a class of parallel numerical integration methods for stiff systems of ordinary differential equations which can be partitioned into loosely coupled sub-systems. The formulas are called decoupled backward differentiation formulas, and they are derived from the classical formulas by restricting the implicit part to the diagonal sub-system. With one or several sub-systems allocated to each processor, information only has to be exchanged after completion of a step but not during the solution of the nonlinear algebraic equations.

The main emphasis is on the formula of order 1, the decoupled implicit Euler formula. It is proved that this formula even for a wide range of multirate formulations has an asymptotic global error expansion permitting extrapolation. Besides, sufficient conditions for absolute stability are presented.

1 Introduction

Define a system of ordinary differential equations,

$$y' = f(t, y),\ y(t_0) = y_0 \text{ and } t \geq t_0 \tag{1.1}$$

where $y : R \rightarrow R^S$, f is Lipschitz continuous in y and $f : R \times R^S \rightarrow R^S$. Stable systems of differential equations are considered stiff when the step size of the discretization by an *explicit* integration method is limited by stability of the discretization and not by accuracy. Efficient numerical integration of stiff systems therefore require *implicit* integration methods, e.g. implicit linear multistep formulas or implicit Runge-Kutta methods.

To fix ideas, consider the discretization of (1.1) by the backward Euler formula,

$$y_n = y_{n-1} + hf(t_n, y_n) \tag{1.2}$$

where $y_n \approx y(t_n)$ and $h = t_n - t_{n-1}$ for $n = 1, 2, \ldots$ In order to compute a new discrete approximation y_n from the previous one, y_{n-1}, the implicit algebraic problem (1.2) has to be solved using e.g. a Newton iteration,

$$y_n^{(m+1)} = y_n^{(m)} + J_n^{-1}(y_n^{(m)}) f_n(y_n^{(m)}) \tag{1.3}$$

*Published in BIT 32 (1992), 689-701.

†Department of Computer Science, University of Copenhagen, Universitetsparken 1, DK–2100 Copenhagen, Denmark

where $f_n(y) = y - y_{n-1} - hf(t_n, y)$ and $J_n(y) = \partial f_n/\partial y|_y$.

A parallelization of the backward Euler formula therefore amounts to a parallelization of (1.3) which involves the parallel computation of f_n, J_n and the parallel solution of systems of linear equations. While an efficient parallelization of f_n and J_n is often a trivial matter, efficient parallel solution of systems of linear equations has proved difficult, especially when J_n is sparse with random pattern of non-zero elements.

The difficulties associated with parallelizing (1.3) efficiently have spurred the development of alternative approaches, notably the waveform relaxation method [1] where (1.1) is partitioned into a number of coupled sub-systems. Each sub-system is integrated independently (trivial parallelization) over a time window. After the completion of an integration sweep, the solutions (waveforms) are exchanged and the integration over the time window is repeated with the updated solutions from the "other" sub-systems. The relaxation process is repeated until convergence.

The waveform relaxation method parallelizes well and gains considerable efficiency from the partitioning of the original problem into smaller sub-systems and from the integration of each sub-system with an optimal selection of step sizes (multirate integration). The main weakness of waveform relaxation is the iteration process itself which may suffer slow convergence or even divergence. Another disadvantage is the need for storing the waveforms which adds substantially to the memory requirement.

This paper introduces two different methods for the parallel numerical integration of stiff systems of ordinary differential equations. They are based on ideas from multirate integration methods [2] and exploit a partitioning of the original problem. This partitioning permits a decoupling into independent problems of the algebraic system resulting from the discretization by a backward differentiation formula.

The monotonic max-norm stability is the theoretical condition for applying the decoupling successfully. This concept is therefore defined in Section 2. The decoupled implicit Euler method with its superior stability properties are described in Section 3 while Section 4 proves the existence of a global error expansion and outlines the use of extrapolation to obtain any order of integration accuracy while maintaining stability. Section 5 describes the decoupled backward differentiation formulas which are derived from the classical formulas, and Section 6 gives an example of the methods applied to a realistic problem.

2 Monotonic Max-Norm Stability

Let the original problem (1.1) be partitioned as follows,

$$\begin{pmatrix} y_1' \\ y_2' \\ \vdots \\ y_q' \end{pmatrix} = \begin{pmatrix} f_1\,(t,y) \\ f_2\,(t,y) \\ \vdots \\ f_q\,(t,y) \end{pmatrix}, \quad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_q \end{pmatrix}, \quad y(t_0) = \begin{pmatrix} y_{1,0} \\ y_{2,0} \\ \vdots \\ y_{q,0} \end{pmatrix} \tag{2.1}$$

where $y_r : R \to R^{s_r}$, $f_r : R \times R^S \to R^{s_r}$ and $\sum_{i=1}^{q} s_i = S$. When necessary, the partitioning of y will be stated explicitly like in $f_r(t, y_1, y_2, \ldots, y_q)$. For sub-system r, y_r is called the *local* integration variable and y_i, $i \neq r$ are called *external* variables.

The following stability condition introduced in [2] plays a crucial role for the stability of decoupled implicit integration methods.

Definition: Monotonic max-norm stability

The partitioned system (2.1) is said to be monotonically max-norm stable, if there exist norms $\|\cdot\|_r$, such that

$$\|u_r - v_r + \lambda\,[f_r(t,u) - f_r(t,v)]\,\|_r \;\geq\; \|u_r - v_r\|_r + \lambda \sum_{j=1}^{q} a_{rj}(t,u,v)\|u_j - v_j\|_j \qquad (2.2)$$

for all $t \geq t_0,\ u, v \in \Omega_t,\ \lambda \leq 0$, where $\Omega_t \subseteq R^S$ and the following condition holds for the logarithmic max-norm of the q × q matrix (a_{rj})

$$\mu_\infty\,[(a_{rj}(t,u,v))] \leq 0 \qquad (2.3)$$

□

The norms used in (2.2) can be related as follows using a common norm $\|\cdot\|$

$$d_j\|y_j\|_j \leq \|y_j\| \leq \tilde{d}_j\|y_j\|_j,\quad j = 1,2,\ldots,q$$

The condition for monotonic max-norm stability (2.2) is very general and neither particularly intuitive, nor very easy to apply in practice. Theorem 3 in [2] gives sufficient conditions for monotonic max-norm stability of a partitioned system. This Theorem and the associated remarks complement the general definition of monotonic max-norm stability.

3 Decoupled Implicit Euler Method

The decoupled implicit Euler method is defined as follows, where the r'th sub-system is discretized by the backward Euler formula:

$$y_{r,n} = y_{r,n-1} + h_{r,n} f_r(t_{r,n}, \tilde{y}_{1,n}, \ldots, \tilde{y}_{r-1,n}, y_{r,n}, \tilde{y}_{r+1,n}, \ldots, \tilde{y}_{q,n}), \qquad (3.1)$$

where $n = 1,2,\ldots,\ t_{r,n} = t_0 + \sum_{j=1}^{n} h_{r,j}$ and the external variables $\tilde{y}_{i,n}$ are convex combinations of values in $\{y_{i,k} \mid k \geq 0\}$ for $i \neq r$. The method is called "decoupled" because the algebraic system resulting from the discretization is decoupled into a number of independent algebraic problems.

The decoupled implicit Euler method is particularly well suited for parallel implementation. Each processor can solve one or several systems like (3.1) and information exchange with other processors is only required between solution steps where sub-system solutions are exchanged.

The convexity of $\tilde{y}_{i,n}$ is necessary for the stability of the decoupled implicit Euler method. All convex combination coefficients might in general be different, but the convex combinations would typically be either a zero-order extra- or interpolation:

$$\tilde{y}_{i,n} = y_{i,k} \text{ or } y_{i,k+1}, \quad \text{where } \ t_{r,n} \in (t_{i,k}, t_{i,k+1}]$$

or a first-order interpolation:

$$\tilde{y}_{i,n} = \frac{t_{i,k+1} - t_{r,n}}{t_{i,k+1} - t_{i,k}} y_{i,k} + \frac{t_{r,n} - t_{i,k}}{t_{i,k+1} - t_{i,k}} y_{i,k+1}$$

where $t_{r,n} \in [t_{i,k}, t_{i,k+1}]$.

Let $\{\tilde{u}_{i,n}\}$ be a sequence of convex combinations of $\{u_{i,n}\}$ and let $\{\tilde{v}_{i,n}\}$ be a sequence of convex combinations of $\{v_{i,n}\}$ where $\{u_{i,n}\}$ and $\{v_{i,n}\}$ are arbitrary sequences. The convex combinations $\{\tilde{u}_{i,n}\}$ and $\{\tilde{v}_{i,n}\}$ are defined to be equivalent if

$$(u_{i,n} = v_{i,n} \text{ for all } i, n) \Rightarrow (\tilde{u}_{i,n} = \tilde{v}_{i,n} \text{ for all } i, n)$$

The stability properties of the decoupled implicit Euler method are described in the following Theorem,

Theorem 1

Assume that the partitioned system (2.1) is monotonically max-norm stable and consider any two solutions $\{u_{r,n}\}, \{v_{r,n}\}$ computed from the decoupled implicit Euler method (3.1) using the same grid points $\{t_{r,n}\}$ and equivalent convex combinations $\{\tilde{u}_{r,n}\}$ and $\{\tilde{v}_{r,n}\}$. Then

$$\sup_{r,n} \|u_{r,n} - v_{r,n}\|_r \ \leq \max_r \|u_{r,0} - v_{r,0}\|_r,$$

provided that $(\tilde{u}_{1,n}, \ldots, u_{r,n} \ldots, \tilde{u}_{q,n})$ and $(\tilde{v}_{1,n}, \ldots, v_{r,n}, \ldots, \tilde{v}_{q,n})$ belong to $\Omega_{t_{r,n}}$ for all r, n

Proof: The Theorem is a corollary of Theorem 4 in [2].

There are two crucial conditions of the stability result of Theorem 1, namely the monotonic max-norm stability and the choice of the external variables which must be convex combinations of previously computed solution components.

The following Theorem is needed for the global error estimation of the decoupled implicit Euler methods.

Theorem 2

Let (3.1) define the function ϕ_r for all $r = 1, 2, ..., q$ and $n = 1, 2, ...$ such that

$$y_{r,n} = y_{r,n-1} + h_{r,n}\phi_r(h_{r,n}, t_{r,n}, \tilde{y}_{1,n}, \ldots, \tilde{y}_{r-1,n}, y_{r,n-1}, \tilde{y}_{r+1,n}, \ldots, \tilde{y}_{q,n}) \tag{3.2}$$

Assume that the partitioned system (2.1) is monotonically max-norm stable, and that $(\tilde{y}_{1,n}, \ldots, \tilde{y}_{r-1,n}, y_{r,n}, \tilde{y}_{r+1,n}, \ldots, \tilde{y}_{q,n}) \in \Omega_{t_{r,n}}$ and that f_r is continuous in y_r. Then ϕ_r exists and $I + h_{r,n}\phi_r$ is Lipschitz continuous for y-variables where $(\tilde{y}_{1,n}, \ldots, \tilde{y}_{r-1,n}, y_{r,n}, \tilde{y}_{r+1,n}, \ldots, \tilde{y}_{q,n}) \in \Omega_{t_{r,n}}$ with Lipschitz constant 1 for $y_{r,n}$ and $\tilde{d}_r/d_j$ for $\tilde{y}_{j,n}$.

Proof:

The monotonic max-norm stability implies

$$\|u_{r,n} - v_{r,n} - h_{r,n}[f_r(t_{r,n}, \tilde{y}_{1,n}, \ldots, u_{r,n}, \ldots, \tilde{y}_{q,n}) - f_r(t_{r,n}, \tilde{y}_{1,n}, \ldots, v_{r,n}, \ldots, \tilde{y}_{q,n})]\|_r$$

$$= \|u_{r,n-1} - v_{r,n-1}\|_r \geq (1 - h_{r,n}a_{rr}(t_{r,n}))\|u_{r,n} - v_{r,n}\|_r$$

where $a_{rr}(t_{r,n}) = \max_{u,v \in \Omega_{t_{r,n}}} a_{rr}(t_{r,n}, u, v)$.

According to [3] Lemma 2, (3.1) can therefore be solved uniquely for $y_{r,n}$ and $y_{r,n}$ is a Lipshitz continuous function of $y_{r,n-1}$ with Lipschitz constant $(1 - h_{r,n}a_{rr}(t_{r,n}))^{-1} \leq 1$.

For $u_{r,n-1} = v_{r,n-1}$ and u_j and v_j, respectively, substituted for $\tilde{y}_{j,n}$, the monotonic max-norm stability leads to the relation

$$0 \geq (1 - h_{r,n}a_{rr})\|u_{r,n} - v_{r,n}\|_r - h_{r,n}a_{rj}\|u_j - v_j\|_j$$

$$\Leftrightarrow \|u_{r,n} - v_{r,n}\|_r \leq \frac{h_{r,n}a_{rj}}{1 - h_{r,n}a_{rr}}\|u_j - v_j\|_j$$

$$\Rightarrow \|u_{r,n} - v_{r,n}\| \leq \tilde{d}_r/d_j\|u_j - v_j\|$$

□

The decoupled implicit Euler method as defined in (3.1) can be used with a completely arbitrary step size selection and still retain stability when the conditions of Theorem 1 are fulfilled. However, to obtain adequate *accuracy*, restrictions on step size are required. Such restrictions can be introduced through multirate compound steps where two different examples are given below.

Definition: Basic Compound Step

For $r = 1, 2, ..., q$ define a compound step over the interval $[t_0, t_0 + Nh]$:

$$y_{r,np_r} = y_{r,(n-1)p_r} + p_r h f_r(t_{np_r}, y_{1,0}, \ldots, y_{r-1,0}, y_{r,np_r}, y_{r+1,0}, \ldots, y_{q,0})$$

for $n = 1, 2, \ldots, N/p_r$ where p_r divides N. □

The basic compound step as well as the slowest-first compound step stated below are defined for the interval $[t_0, t_0 + Nh]$. This is done to simplify notation, and the definition of the compound steps are readily extended to cover any interval of width Nh. Any numerical integration algorithm based on these methods would in general employ a sequence of compound steps. A compound step bears some similarities to a step of a Runge-Kutta method.

The basic compound step uses the simplest convex combination corresponding to a zero order extrapolation. Although the order of accuracy is not improved, the absolute accuracy will in general be better when linear interpolation is used whenever possible. This is the objective of the slowest-first compound step:

Definition: Slowest-first Compound Step

The slowest-first compound step is defined recursively over the interval $[t_0, t_0 + Nh]$. The sub-systems have been ordered such that they are integrated using increasingly

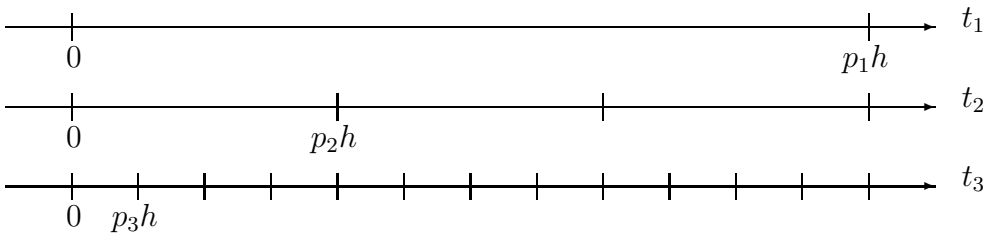

Figure 1: Example of step size distribution for a slowest-first compound step

smaller step sizes. Sub-system r is integrated with step size $p_r h$ where $p_r \leq p_{r-1}$, $p_1 = N$, $p_0 = p_1$ and p_r divides p_{r-1}.

The general *recursive* definition for $r = 1, 2, \ldots, q-1$ is as follows:

> p_{r-1}/p_r steps of sub-system r are computed as one step of sub-system r followed by p_r/p_{r+1} steps of sub-system $r+1$, then another step of sub-system r, etc.

Sub-system 1 is integrated first as follows:

$$y_{1,N} = y_{1,0} + Nhf_1(t_N, y_{1,N}, y_{2,0}, \ldots, y_{q,0}) \tag{3.3}$$

Then follows sub-systems $2, 3, \ldots, q$ as specified by the recursion. The following general formulas are used:

$$y_{r,np_r} = y_{r,(n-1)p_r} + p_r h f_r(t_{np_r}, \tilde{y}_{1,np_r}, \ldots, \tilde{y}_{r-1,np_r}, y_{r,np_r}, y_{r+1,(n-1)p_r}, \ldots, y_{q,(n-1)p_r}) \tag{3.4}$$

for $n = 1, 2, \ldots, N/p_r$. The convex combinations are defined by

$$\tilde{y}_{i,np_r} = (1 - np_r/p_i + \lfloor np_r/p_i \rfloor) y_{i,\lfloor np_r/p_i \rfloor p_i} + (np_r/p_i - \lfloor np_r/p_i \rfloor) y_{i,(\lfloor np_r/p_i \rfloor + 1)p_i} \tag{3.5}$$

for $i = 1, 2, \ldots, r-1$. ($\lfloor \cdot \rfloor$ denotes rounding towards zero) □

The slowest-first compound step and the involved interpolation is illustrated using Fig. 1. First y_{1,p_1} is computed using $y_{2,0}$ and $y_{3,0}$. Then one step of sub-system 2 is computed interpolating linearly between $(y_{1,0}, y_{1,p_1})$ and using $y_{3,0}$. Now follows $p_2/p_3 (= 4)$ steps of sub-system 3 interpolating linearly between the pairs $(y_{1,0}, y_{1,p1})$ and $(y_{2,0}, y_{2,p_2})$. The maximum depth of the recursion has now been reached and another step of sub-system 2 is computed, p_2/p_3 more steps of sub-system 3, etc.

The basic compound step preserves the parallelity of the decoupled implicit Euler method while the slowest-first compound step imposes a *sequence* of computation which reduces parallelity.

A slightly modified version of the slowest-first compound step may be introduced fully in parallel on P processors in the following case: The first P sub-systems are integrated using the same step size $p_1 h = p_2 h = \ldots = p_P h$, the next P sub-systems are integrated using step size $p_{P+1} h = p_{P+2} h = \ldots = p_{2P} h$, etc.

4 Extrapolated Decoupled Implicit Euler Method

Many applications require better accuracy than offered by the decoupled implicit Euler method. The central property of decoupling and the good stability properties can be retained while the accuracy is improved by using extrapolation. The main result of this section is Theorem 5 which states the existence of global error expansions for both the basic and the slowest-first compound steps. Several other compound steps could be defined, but the proof of existence of global error expansions would basically be the same.

Definition: Increment function Φ.

Define the increment function Φ based on either the basic or the slowest-first compound steps such that,

$$y_N = y_0 + H\Phi(H, t_0, y_0)$$

where $H = Nh$. Φ is defined implicitly as the solution of a system of equations. □

Lemma 3

Let f be Lipschitz continuous in y and let the assumptions of Theorem 2 be fulfilled. Then the increment function Φ based on either the basic or the slowest-first compound steps is Lipschitz continuous in y.

Proof:

The Lemma is only proved for the slowest-first compound step. The proof is similar and simpler for the basic compound step. The proof is by induction in the sequence of computations leading to Φ as specified in the definition of the slowest-first compound step.

1) According to Theorem 2, $y_{1,N}$ defined by (3.3) is a Lipschits continuous function of $y_{1,0}, y_{2,0}, \ldots, y_{q,0}$.

2) Assume that the sequence of solution values recursively specified for the slowest-first compound step, $y_{1,p_1}, y_{2,p_2}, \ldots, y_{q,p_q}, y_{q,2p_q}, \ldots, y_{q,p_{q-1}/p_q}, y_{q-1,2p_{q-1}}, \ldots, y_{r,(n-1)p_r}$, are all Lipschitz continuous functions of $y_{1,0}, y_{2,0}, \ldots, y_{q,0}$.

The next solution value y_{r,np_r} is defined by (3.4). From the definition of y_{r,np_r}, the induction assumptions and Theorem 2, it follows that y_{r,np_r} is also a Lipschitz continuous function of $y_{1,0}, y_{2,0}, \ldots, y_{q,0}$.

By the induction axiom it is thus proved that all the interior solution points, including the final ones of the compound step,

$$y_{r,np_r} \text{ for } r = 1, 2, \ldots, q \text{ and } n = 1, 2, \ldots, N/p_r$$

are all Lipschitz continuous functions of $y_{1,0}, y_{2,0}, \ldots, y_{q,0}$.

The increment function Φ can be expressed as

$$\Phi_r(H, t_0, y_{1,0}, y_{2,0}, \ldots, y_{q,0})$$

$$= \frac{p_r}{N} \sum_{n=1}^{N/p_r} f_r(t_{np_r}, \tilde{y}_{1,np_r}, \ldots, \tilde{y}_{r-1,np_r}, y_{r,np_r}, y_{r+1,(n-1)p_r}, \ldots, y_{q,(n-1)p_r}) \qquad (4.1)$$

From the definition of $\tilde{y}_{i,np_r}$ in (3.5), the Lipschitz continuity of f_r and the Lipschitz continuity of y_{r,np_r} for $r = 1, 2, \ldots, q$ and $n = 1, 2, \ldots, N/p_r$, it follows that Φ_r is Lipschitz continuous in $y_{1,0}, y_{2,0}, \ldots, y_{q,0}$ for $r = 1, 2, \ldots, q$. □

Lemma 4

For a function $y(t) \in \Omega_t$ and the increment function Φ based on a monotonic max-norm stable system (2.1), the local error has the form

$$y(t+H) - y(t) - H\Phi(H, t, y(t)) = e_2(t)H^2 + e_3(t)H^3 + \ldots + e_M(t)H^M + O(H^{M+1}) \quad (4.2)$$

when f is $(M-1)$ times differentiable.

Proof:

The local error can also be expressed as $y(t+H) - y_N$ where $y_0 = y(t)$. First, y_N will be expressed as an expansion in H and then this expansion will be compared with the expansion of $y(t+H)$ to give the desired form. The existence and form of y_N is proved using an induction argument similar to the argument used in the previous proof. It will only be proved for the slowest-first compound step since it is similar, but simpler, for the basic compound step.

1) Assume that $y_{1,N}$ can be expanded as follows,

$$y_{1,N} = y_{1,0} + e_{1,0}^{(1)}H + e_{1,0}^{(2)}H^2 + \ldots + e_{1,0}^{(M)}H^M + O(H^{M+1})$$

Substitute this expansion into (3.3) and collect terms of equal power in H:

$$e_{1,0}^{(1)} = f_1(t_0, y_{1,0}, y_{2,0}, \ldots, y_{q,0})$$

$$e_{1,0}^{(2)} = f_{1,t}(t_0, y_{1,0}, y_{2,0}, \ldots, y_{q,0}) + f_{1,y_1}(t_0, y_{1,0}, y_{2,0}, \ldots, y_{q,0}) f_1(t_0, y_{1,0}, y_{2,0}, \ldots, y_{q,0})$$

where $f_{r,t}$ denotes $\partial f_r/\partial t$ and f_{r,y_i} denotes $\partial f_r/\partial y_i$

It is straightforward to continue the substitution process to prove that the terms up to and including $e_{1,0}^{(M)}$ exist if f is $(M-1)$ times differentiable.

2) Assume that expansions

$$y_{r,i} = y_{r,0} + e_{r,i}^{(1)}H + e_{r,i}^{(2)}H^2 + \ldots + e_{r,i}^{(M)}H^M + O(H^{M+1}) \quad (4.3)$$

exist for the sequence of solution values recursively specified for the slowest-first compound step, up to and including $y_{r,(n-1)p_r}$. The next solution value, y_{r,np_r} is computed using (3.4).

Assuming the expansion

$$y_{r,np_r} = y_{r,0} + e_{r,np_r}^{(1)}H + e_{r,np_r}^{(2)}H^2 + \ldots + e_{r,np_r}^{(M)}H^M + O(H^{M+1})$$

Substitution into (3.4) gives

$$e_{r,np_r}^{(1)} = e_{r,(n-1)p_r}^{(1)} + \frac{p_r}{N} f_r(t_0, y_{1,0}, y_{2,0}, \ldots, y_{q,0})$$

It is straightforward to continue the substitution process, although the terms become complicated, and to prove that $e_{r,np_r}^{(2)}, \ldots, e_{r,np_r}^{(M)}$ exist if f is $(M-1)$ times differentiable.

It has thus been proved that y_N has a Taylor series expansion in H. By subtracting this expansion from

$$y(t_0 + H) = y_0 + fH + \frac{1}{2}(f_t + f_y f)H^2 + O(H^3)$$

the desired local error expansion (4.2) is obtained ($e_0(t) = e_1(t) = 0$). □

Theorem 5

Let the initial value problem (2.1) be solved using the method

$$y_{(n+1)N} = y_{nN} + H\Phi(H, t_{nN}, y_{nN}) \tag{4.4}$$

where Φ is the previously defined increment function, and let $y_H(t_0 + nH) = y_{nN}$.

Assume that

1. f is Lipschitz continuous in y.
2. The system (2.1) is monotonically max-norm stable for $y(t) \in \Omega_t$. (The conditions of Theorem 2 are fulfilled.)
3. f is $(M-1)$ times differentiable.

Then the global error $y(t) - y_H(t)$ has an expansion of the form

$$y(t) - y_H(t) = E_1(t)H + E_2(t)H^2 + \ldots + E_{M-1}(t)H^{M-1} + O(H^M)$$

Proof:

The proof follows from Theorem 2, Lemmas 3 and 4 and from Theorem 8.1 in [4].

□

When the conditions of Theorem 5 are fulfilled, the basic or the slowest-first compound steps can be used as the basis of an extrapolation method [4].

Starting with $y_H(t_0) = y_0$, use (4.4) to compute

$$y_H(t_0 + nH),\ y_{H/2}(t_0 + nH/2),\ y_{H/4}(t_0 + nH/4), \ldots \quad \text{for } n = 1, 2, \ldots$$

Extrapolated values can be computed as

$$\begin{aligned}
\hat{y}_H(t_0 + nH) &= 2y_{H/2}(t_0 + nH) - y_H(t_0 + nH) \\
\hat{y}_{H/2}(t_0 + nH) &= 2y_{H/4}(t_0 + nH) - y_{H/2}(t_0 + nH) \\
\hat{\hat{y}}_H(t_0 + nH) &= (4\hat{y}_{H/2}(t_0 + nH) - \hat{y}_H(t_0 + nH))/3
\end{aligned}$$

$$\ldots$$

$\hat{y}_H$ has global order 2 and $\hat{\hat{y}}_H$ has global order 3. Assuming that f is sufficiently smooth, the extrapolation technique can be used to obtain an arbitrarily high order of accuracy.

This approach to extrapolation is called passive: the extrapolated values are not used for further computation. The global error expansion is valid also for varying step sizes if

the variation is sufficiently smooth, but it may be convenient to use an extrapolated value after a change of step size. This may affect the stability properties adversely, but as long as only a finite number of changes of step size are performed, the effect will remain finite.

The computation of each of the sequences $y_H, y_{H/2}, \ldots$ can be done in parallel since they are independent. This adds to the parallelity offered by the decoupled implicit Euler formula, be it in the basic form or a multirate version.

5 Decoupled Backward Differentiation Formulas

The basic step of an order k decoupled backward differentiation formula is

$$y_{r,n}^{(1)} = \sum_{j=1}^{k} \alpha_j y_{r,n-j} + h_n \beta_0 f_r(t_n, y_{1,n-1}, \ldots, y_{r-1,n-1}, y_{r,n}^{(1)}, y_{r+1,n-1}, \ldots, y_{q,n-1})$$

for $r = 1, 2, \ldots, q$. The coefficients $\alpha_1, \alpha_2, \ldots, \alpha_k, \beta_0$ characterize a standard k-step backward differentiation formula. It is crucial for the stability properties that the external variables are $y_{i,n-1}$, $i \neq r$. Any attempt to *extrapolate* the external variables to a value at t_n will degrade the stability properties seriously.

The corrector steps are as follows,

$$y_{r,n}^{(m+1)} = \sum_{j=1}^{k} \alpha_j y_{r,n-j} + h_n \beta_0 f_r(t_n, y_{1,n}^{(m)}, \ldots, y_{r-1,n}^{(m)}, y_{r,n}^{(m+1)}, y_{r+1,n}^{(m)}, \ldots, y_{q,n}^{(m)})$$

for $r = 1, 2, \ldots, q$ and $m = 1, 2, \ldots, k-1$.

For each corrector step, the order of accuracy is increased by one until order k is reached. The same result could be obtained by using an order 2 method for the first corrector step, an order 3 method for the second corrector step etc. However, there is an obvious advantage in using the final order k method for the basic step and all the corrector steps, namely that the Jacobian for the Newton iteration has the same $h_n\beta_0$ coefficient (c.f. (1.3)), and in many cases the final Jacobian of the basic step and its factorization can be used throughout the corrector steps.

The main drawback of the extrapolated decoupled implicit Euler method is the computational expense involved in solving the problem for a sequence of different step sizes. The decoupled backward differentiation formulas are better in this respect and they are similar to the decoupled implicit Euler method in terms of potential for parallelity.

However, the stability properties of the decoupled backward differentiation formulas are in general slightly worse than those of the classical backward differentiation formulas because of the decoupling. So far, there is only experimental evidence that the decoupled backward differentiation formulas will have good stability properties when used for the integration of monotonically max-norm stable systems of differential equations.

The decoupled backward differentiation formulas can also be formulated as a multirate compound step.

Definition: Decoupled BDF compound step

For $r = 1, 2, \ldots, q$ define the BDF basic compound step over the interval $[t_0, t_0 + Nh]$:

$$y^{(1)}_{r,np_r} = \sum_{j=1}^{k} \alpha_j y^{(1)}_{r,(n-j)p_r} + p_r h \beta_0 f_r(t_{np_r}, y_{1,0}, \ldots, y_{r-1,0}, y^{(1)}_{r,np_r}, y_{r+1,0}, \ldots, y_{q,0})$$

for $n = 1, 2, \ldots, N/p_r$ where p_r divides N.

The corrector steps are computed as follows for $m = 1, 2, \ldots, k-1$,

$$y^{(m+1)}_{r,np_r} = \sum_{j=1}^{k} \alpha_j y^{(m+1)}_{r,(n-j)p_r} + p_r h \beta_0 f_r(t_{np_r}, \tilde{y}^{(m)}_{1,np_r}, \ldots, \tilde{y}^{(m)}_{r-1,np_r}, y^{(m+1)}_{r,np_r}, \tilde{y}^{(m)}_{r+1,np_r}, \ldots, \tilde{y}^{(m)}_{q,np_r})$$

for $n = 1, 2, \ldots, N/p_r$ and $r = 1, 2, \ldots, q$.

$\tilde{y}^{(m)}_{i,np_r}$ is computed from $y^{(m)}_{i,np_i}$, $n = 0, 1, \ldots, N/p_i$ using interpolation of order m.

□

The weak link of the decoupled compound step is clearly the interpolation where interpolation of order higher than 1 required for decoupled BDF compound steps of order higher than 2 may degrade the stability properties seriously.

6 Example

The following example was presented in [2].

$$V' = f(t, V) \text{ where } f(t, V) = C^{-1}(g(V) + i(t)),\ y:\ R \to R^4 \text{ and } f:\ R \times R^4 \to R^4 \quad (6.1)$$

$$C = \begin{pmatrix} C_D + C_S & -C_D & 0 & 0 \\ -C_D & 2C_D + C_S & -C_D & 0 \\ 0 & -C_D & 2C_D + C_S & -C_D \\ 0 & 0 & -C_D & C_D + C_S \end{pmatrix}, \quad V = \begin{pmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{pmatrix}$$

$$g(V) = \begin{pmatrix} -V_1 G \\ (V_{DD} - V_2)G - i_1(V_1, V_2) \\ (V_{DD} - V_3)G - i_2(V_2, V_3) \\ (V_{DD} - V_4)G - i_3(V_3, V_4) \end{pmatrix}, \quad i(t) = \begin{pmatrix} i_0(t) \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\begin{aligned} i_r(V_r, V_{r+1}) &= 2\beta(V_r - V_{th} - V_{r+1}/2)V_{r+1} \text{ for } V_r \geq V_{th}, V_{r+1} < V_r - V_{th} && (6.2) \\ i_r(V_r, V_{r+1}) &= \beta(V_r - V_{th})^2 \text{ for } V_r \geq V_{th}, V_{r+1} \geq V_r - V_{th} && (6.3) \\ i_r(V_r, V_{r+1}) &= 0 \text{ for } V_r < V_{th} && (6.4) \end{aligned}$$

$$C_S = 10 C_D \text{ and } \beta = G/[2(V_{DD} - V_{th})]$$

The variables i_r are continuously differentiable in V_r and V_{r+1}, but $\partial^2 i_r / \partial V_r^2$ and $\partial^2 i_r / \partial V_{r+1}^2$ are discontinuous for $V_{r+1} = V_r - V_{th}$ where the definition of i_r changes between

t H	$5_{10}-7$ 10^{-7}	$5_{10}-8$ 10^{-8}	$5_{10}-9$ 10^{-9}
$2^p \approx$	3.2658 3.3987 3.1083 2.7443	3.9459 3.9610 3.9142 3.8595	3.9269 3.9300 3.8631 3.7042

Table 1: Results for the left-hand side of (6.5).

(6.2) and (6.3). This means that f defined in (6.1) is once differentiable ($M = 2$). Because of the limited smoothness of the solution, extrapolation can only be applied once to remove the $E_1(t)H$ error term. This amounts to a second order integration formula. Similarly, a second order decoupled backward differentiation formula can be used.

Problem (6.1) was separated into 4 scalar sub-systems and integrated numerically with $C_D = 10^{-14}$, $G = 10^{-3}$, $V_{th} = 0.9$ and $V_{DD} = 5$. The function $i_0(t)$ was defined as

$$\begin{aligned} i_0(t) &= [V_{th} + (1 - \cos(10^6 t))(V_{DD} - V_{th})/2]G \text{ for } t \in [0, 10^{-6}\pi] \\ i_0(t) &= V_{DD}G \text{ for } t > 10^{-6}\pi \end{aligned}$$

V_0 was chosen to make $V(t)$ constant for a constant $i_0 = V_{th}G$. The eigenvalues of $\partial f/\partial V$ are of the order of magnitude 10^{10} which means that the problem is stiff for step sizes greater than 10^{-9}.

First the extrapolated decoupled implicit Euler method was tested using the same step size for all sub-systems. Assuming that there exists a global error expansion of sufficiently high order, the order of accuracy can be estimated using

$$\frac{V_H(t) - V_{H/2}(t)}{V_{H/2}(t) - V_{H/4}(t)} \approx 2^p \tag{6.5}$$

where the differences and division is applied componentwise.

The entries of Table 1 clearly approximate 2^2 for small h corresponding to a second order method as expected from the extrapolation. However, the model on which (6.5) is based is not valid because of the discontinuity in the second derivative of f so therefore the entries of Table 1 do not exhibit a convincing convergence towards 4.

The second order decoupled backward differentiation formula was also tested using the same step size for all sub-systems. A computation similar to Table 1 gave erratic results since the method does not have a global error expansion for this problem. There are two reasons for this. First, as mentioned before, the lack of smoothness of f and second, the procedure used for generating the first solution point after the initial value (the extrapolated decoupled implicit Euler method was used).

Then a reference solution was generated over the interval $[0, 3.15_{10} - 6]$ using the extrapolated decoupled implicit Euler method with step size $H = 10^{-8}$. This solution is compared with the solution of the second order decoupled backward differentiation

t H	$3_{10}-6$ $2_{10}-7$	$3.1_{10}-6$ 10^{-7}	$3.1_{10}-6$ $5_{10}-8$
Decoupled	0.0056	0.0015	0.0004
2nd order	-0.0066	-0.0012	-0.0003
BDF	0.0386	0.0100	0.0024
	-0.0343	-0.0062	-0.0014
Extrapolated	0.0077	0.0018	0.0004
decoupled	-0.0337	-0.0072	-0.0017
implicit	0.0710	0.0144	0.0033
Euler	-0.1156	-0.0242	-0.0054

Table 2: $V_H(t) - V_{10^{-8}}(t)$

formula in Table 2 where the difference vectors are shown. The corresponding results for the extrapolated decoupled implicit Euler method are given for comparison.

The errors of the two different methods are comparable, and each halving of the step size leads to a reduction of the distance to the reference solution by a factor of approximately 4. In conclusion, the methods perform as expected.

The monotonic max-norm stability of (6.1) partitioned into 4 scalar components was proved in [2] for $V \in [0, V_{DD}] \times \ldots \times [0, V_{DD}]$. The solutions computed for Table 2 actually stray slightly outside that domain but without any consequences for the stability of the integration methods.

Acknowledgement

This work was done while the Author visited Coordinated Science Laboratory, University of Illinois at Urbana-Champaign. The support provided by CSL is greatly appreciated.

References

[1] Lelarasmee E., Ruehli A.E., and Sangiovanni-Vincentelli A.L., "The Waveform Relaxation Method for Time-Domain Analysis of Large Scale Integrated Circuits", IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, 1(1982), pp. 131-145.

[2] Sand, J. and Skelboe, S., "Stability of backward Euler multirate methods and convergence of waveform relaxation", To appear in BIT.

[3] Söderlind, G., "On nonlinear difference and differential equations", BIT, Vol. 24(1984), pp. 667-680.

[4] Hairer, E., Nørsett, S. P. and Wanner, G., "Solving ordinary differential equations I", Springer, 1987.