Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

R.A. Trompert, J.G. Verwer

Runge-Kutta methods and local uniform grid refinement

Department of Numerical Mathematics Report NM-R9022 November

Runge-Kutta Methods and Local Uniform Grid Refinement

R.A. Trompert and J.G. Verwer
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

Local uniform grid refinement (LUGR) is an adaptive grid technique for computing solutions of partial differential equations possessing sharp spatial transitions. Using nested, finer-and-finer uniform subgrids, the LUGR technique refines the space grid locally around these transitions, so as to avoid discretization on a very fine grid covering the entire physical domain. This paper examines the LUGR technique for time-dependent problems when combined with static-regridding. Static-regridding means that in the course of the time evolution, the space grid is adapted at discrete times. The present paper considers the general class of Runge-Kutta methods for the numerical time integration. Following the method of lines approach, we develop a mathematical framework for the general Runge-Kutta LUGR method applied to multi-space dimensional problems. We hereby focus on parabolic problems, but a considerable part of the examination applies to hyperbolic problems as well. Much attention is paid to the local error analysis. The central issue here is a 'refinement condition' which is to underly the refinement strategy. By obeying this condition, spatial interpolation errors are controlled in a manner that the spatial accuracy obtained is comparable to the spatial accuracy on the finest grid if this grid would be used without any adaptation. A diagonally-implicit Runge-Kutta method is discussed for illustration purposes, both theoretically and numerically.

1980 Mathematics subject classification: Primary:65M50. Secondary:65M20.
1987 CR Categories: G.1.8.
Key Words & Phrases: partial differential equations, numerical mathematics, time-dependent problems, Runge-Kutta methods, adaptive grid methods, error analysis.

Note: This report will be submitted for publication elsewhere.

1. Introduction

Local uniform grid refinement (LUGR) is an adaptive grid technique for computing solutions of partial differential equations (PDEs) possessing sharp spatial transitions. Using nested, finer-and-finer, uniform subgrids, the LUGR technique refines the space grid locally around these transitions, so as to avoid discretization on a very fine grid covering the entire physical domain. LUGR should be contrasted with pointwise refinement which leads to truly nonuniform grids. In this paper we examine the LUGR technique for time-dependent problems. Thus, typical solutions aimed at are those possessing sharp moving transitions, such as steep fronts, emerging layers, moving pulses, etc. For time-dependent problems, LUGR is combined with static-regridding. Static-regridding means that in the

Report NM-R9022
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

course of the time evolution, the space grid is adapted at discrete times. Static-regridding should be contrasted with dynamic-regridding where the space grid moves continuously in the space-time domain as, e.g., in the case of the moving-finite-element method.

The idea of the method can be briefly described as follows. Given a coarse base space grid and a base temporal stepsize, nested, finer-and-finer uniform subgrids are generated, all within this base time step. These locally uniform subgrids possess nonphysical boundaries and on each of these subgrids an integration is carried out. They are generated up to a level of refinement good enough to resolve the anticipated fine scale structures. Having completed the refinement for the current base time step, the process is continued to the next base time step, while starting again from the coarse base grid. However, all refined subgrids computed at forward time levels are kept in storage as they are to be used for step continuation. Further, for step continuation always the most accurate solution is to be used that is available.

An attractive feature of the static-regridding LUGR approach is the possibility of dividing the whole solution process into the following computational procedures: spatial discretization, temporal integration, error estimation, regridding and interpolation. Depending on the application, these individual procedures may range from simple or straightforward to very sophisticated. This flexibility is attractive since it makes it possible to treat different types of PDE problems with almost one and the same code, assuming hereby that the grid structure and the associated data structure remain unchanged. It is of importance to realize that an important part of the development and coding of LUGR methods lies in the data structure. The choice of data structure is important for keeping the unavoidable overhead at an acceptable level, because at each time step, or certain number of steps, grids may be created or removed while also communication between grids of adjacent levels of refinement frequently takes place.

In this paper we consider the general class of Runge-Kutta methods for the numerical time integration. Following the method of lines approach, we develop a rigorous mathematical framework for the general Runge-Kutta LUGR method applied to multi-space dimensional problems. We hereby focus on parabolic problems, but a considerable part of the discussion applies to hyperbolic problems as well. The present paper is a continuation of [15] which deals exclusively with the implicit Euler method. Like in [15], much attention is paid to the local error analysis. The central issue here is the so-called 'refinement condition', which is to underly the refinement strategy. By obeying this condition, spatial interpolation errors are controlled in a manner that the spatial accuracy obtained is comparable to the spatial accuracy on the finest grid if this grid would be used without any adaptation. This means that, when taking the finest grid size as a reference point, the grid adaptation does not diminish accuracy. Non-numerical, technical subjects, such as the data structure and the memory use, are not discussed here. These are the same as in [15]. For related earlier work on LUGR methods, we refer to Berger and Oliger [4], Gropp [7, 8], Arney and Flaherty [3], and references therein.

An overview of the organization of the paper reads as follows. Section 2 is devoted to the general method formulation. Following the method of lines approach of [15], here we develop the mathematical framework that enables us to give a concise, but rigorous description of the Runge-Kutta LUGR method. This framework also enables us to set up the general error scheme, as discussed in Section 3. This general error scheme is further elaborated in Sections 4 and 5. Section 4 addresses the stability issue, while Section 5 is devoted to the local error analysis. Under a weak and natural assumption on the Runge-Kutta method, we prove that the 'uniform in h' temporal order of the method is at least equal to the stage order. We further derive the important, above mentioned 'refinement condition'. Noteworthy is that Sections 3 - 5 apply to the whole class of Runge-Kutta methods. As a result, the outcome of the analysis is of a rather general nature in the sense that for a specific, given Runge-Kutta method, a further elaboration is needed, not only when it comes to implementation, but also for providing more insight. Such an elaboration is presented in the remainder of the paper for a 3-stage diagonally-implicit Runge-Kutta (DIRK) method. Specifically, in Section 6 attention is given to the order reduction phenomenon and to how to implement the 'refinement condition' for this specific method. Section 7 deals with two numerical examples in two space dimensions. Finally, we conclude the paper with Section 8 discussing some important matters of practical interest which still need to be addressed.

2. The general method formulation

2.1. *The Runge-Kutta method*

Consider the initial value problem for a standard ODE system

$$\frac{d}{dt}U(t) = F(t, U(t)), \qquad 0 < t \leqslant T, \qquad U(0) = U^0. \tag{2.1}$$

The general one-step, s-stage RK scheme for the numerical solution of (2.1) is denoted by

$$U^{(i)} = U^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F(t_{n-1} + c_j \tau, U^{(j)}), \qquad 1 \leqslant i \leqslant s, \tag{2.2a}$$

$$U^{n} = U^{n-1} + \tau \sum_{i=1}^{s} b_{i} F(t_{n-1} + c_i \tau, U^{(i)}). \tag{2.2b}$$

This formula describes the numerical step from step point t_{n-1} to $t_n = t_{n-1} + \tau$, where $t_0 = 0$. The stepsize τ may vary with n, but this dependence is suppressed in our notation. Throughout this paper, upperscripts of approximation vectors will refer to time, while bracketed upperscripts are used for approximations at intermediate stages. However, the dependence of $U^{(i)}$ on the step index n is also suppressed. As usual, we suppose $c_i = a_{i1} + \cdots + a_{is}$. Hence $U^{(i)}$ is to be interpreted as an intermediate approximation to $U(t)$ at the intermediate step point $t = t_{n-1} + c_i \tau$. In the remainder it is convenient to combine (2.2a,b) into one formula. Denote, for a given s-stage scheme (2.2),

$$a_{s+1i} = b_i, \qquad 1 \leqslant i \leqslant s, \qquad U^{(s+1)} = U^n. \tag{2.3}$$

Then we rewrite (2.2) as

$$U^{(i)} = U^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F(t_{n-1} + c_j \tau, U^{(j)}), \qquad 1 \leqslant i \leqslant s+1. \tag{2.4}$$

The fact that $U^{(s+1)} = U^n$, and thus is an approximation at the main step point $t = t_n$, will always be clear from the context.

2.2. *The semi-discrete problem*

Consider a well-posed real abstract Cauchy problem

$$u_t = L(t,u), \qquad 0 < t \leqslant T, \qquad u(\underline{x},0) = u_0(\underline{x}), \tag{2.5}$$

where L represents a d-space dimensional partial differential operator which differentiates the unknown function $u(\underline{x},t)$ with respect to its space variable $\underline{x} = (x,y, \cdots)$ in a connected space domain Ω in IR^d. L is supposed to be of at most second order and provided with appropriate boundary conditions on the boundary $\partial\Omega$ of Ω. The boundary is taken to be locally parallel to the coordinate axes. The function $u(\underline{x},t)$ may be vector valued and, unless noted otherwise, the number of space dimensions d may be arbitrary. The function $u(\underline{x},t)$ is supposed to exist uniquely and to be as often differentiable on $(\Omega \bigcup \partial\Omega) \times [0,T]$ as the numerical analysis requires. Recall, however, that we specifically aim at non-smooth solutions like steep travelling wave fronts. Thus non-smoothness means here the occurrence of rapid transitions in the space-time domain $(\Omega \bigcup \partial\Omega) \times [0,T]$.

LUGR methods use local uniform grids which cover only a part of the physical domain. The size and the number of these local uniform grids mostly vary in time. Therefore, LUGR methods generate a sequence of operations on vectors in vector spaces with a variable dimension. This complicates the analysis of local and global errors. In [15] we got round this problem by expanding the fine grids in the mathematical formulation of the method, so that the entire domain is covered by them. Also here we use this 'grid expansion'. Temporal integration then takes place on one part of the expanded grid and interpolation on the other. Note that this grid expansion does not take place in the actual application but only in the mathematical formulation of the method. Nevertheless, the results of the error analysis presented in this paper remain valid for the applicated method.

Let $l \in \mathrm{IN}^+$. For $k = 1, \cdots, l$ we introduce uniform space grids ω_k with the uniformity meant direction wise. Each grid ω_k is supposed to cover the whole of the interior domain Ω (grid expansion). The grid ω_k has no points on $\partial\Omega$. The grid ω_1 is called the base grid and, given this grid, ω_2 is obtained from ω_1 by bisecting all sides of all cells of ω_1, etc. Hence the intersection of ω_k and ω_{k-1}

is just ω_{k-1} and the grids are nested in the natural way. Because $\partial\Omega$ is locally parallel to the coordinate axes, it is always possible to construct such a sequence of nested, uniform grids. Following the method of lines approach, to the problem (2.5) we now associate on each grid ω_k a real Cauchy problem for an explicit ODE system in $\mathbb{R}^{d_k}$,

$$\frac{d}{dt}U_k(t) = F_k(t, U_k(t)), \qquad 0 < t \leqslant T, \qquad U_k(0) = U_k^0, \tag{2.6}$$

which is defined by an appropriate finite-difference space-discretization of (2.5) and its boundary conditions. Thus, U_k and F_k are vectors representing the values of grid functions defined on the grid ω_k. Each component of U_k and F_k itself is vector valued if u is vector valued. F_k is determined by the type of grid, by the actual finite-difference formulas, and of course by the precise form of L and its boundary conditions. Hence, the boundary conditions have been worked into the semi-discrete system by eliminating semi-discrete values at $\partial\Omega$. The dimension d_k is determined by the spatial dimension, the grid spacing, and the number of PDEs. The initial vector U_k^0 for (2.6) is supposed to be exact.

In the remainder we let S_k with dim $(S_k) = d_k$ denote the linear, finite-dimensional vector space of grid functions defined on ω_k. Obviously, S_k coincides with $\mathbb{R}^{d_k}$ and U_k, F_k are specific elements of S_k. Let $u_k(t) \in S_k$ represent the natural (nodal wise) restriction of $u(\underline{x},t)$ to ω_k. In the space S_k the fully continuous problem (2.5) and the semi-discrete problem (2.6) are related by the local spatial discretization error $\alpha_k(t) \in S_k$ defined by

$$\alpha_k(t) = \frac{d}{dt}u_k(t) - F_k(t, u_k(t)), \qquad 0 \leqslant t \leqslant T. \tag{2.7}$$

In particular, the time continuous grid functions u_k and α_k are sufficiently often differentiable with respect to t and $\alpha_k(t)$ has the order of consistency of the finite-difference formulas employed. Finally, for a proper understanding of the method formulation, it is of importance to note once more that we consider elements $u_k(t)$, $U_k(t) \in S_k$ defined on space grids ω_k which cover the entire physical domain Ω.

2.3. The multi-level multi-stage RK method

Starting at the coarse base grid ω_1, this method carries out repeated integrations on subgrids of ω_k for $k = 2, \cdots, l$. Each of these integrations spans the same time interval $[t_{n-1}, t_n]$ and the subgrids, which we henceforth will call integration domains, possess internal nonphysical boundaries (grid interfaces). Characteristic for the method is that the integration domains are nested and that on each integration domain a new initial-boundary value problem is solved. Required initial values are defined by interpolation from the next coarser integration domain or taken from a possibly existing integration domain from the previous time interval. Boundary values required at internal boundaries are also interpolated from the next coarser integration domain. At each level of refinement, the integration domains are allowed to be disjunct and thus may consist of two or more subdomains. The nesting is continued up to a level fine enough to resolve the anticipated fine scale structure. This means that, given the base grid ω_1, l must be chosen sufficiently large. Having completed the integration on the finest, l^{th} level integration domain, the process is repeated for the next time interval $[t_n, t_{n+1}]$ by again starting from the coarse base grid ω_1. We note that all refined subgrids computed at forward time levels are kept in storage as they are used for step continuation. Further, for step continuation always the most accurate solution is used that is available.

The above described process is defined by the following formula, in which the lower index k denotes the grid level. For $k = 1$ we have

$$U_1^{(i)} = R_{l1} U_l^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_1(t_{n-1} + c_j \tau, U_1^{(j)}), \qquad 1 \leqslant i \leqslant s+1, \tag{2.8a}$$

and for $k = 2, \cdots, l$,

$$U_k^{(i)} = D_k^n \, [R_{lk} U_l^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_k(t_{n-1} + c_j \tau, U_k^{(j)})] + \tag{2.8b}$$

$$(I_k - D_k^n) \, [P_{k-1k} U_{k-1}^{(i)} + b_k^{(i)}], \qquad 1 \leqslant i \leqslant s+1.$$

Thus, within the time step from t_{n-1} to t_n, first (2.8a) is applied followed by (2.8b) for $k \geqslant 2$.

Observe that (2.8a) is a standard application on ω_1, except for the choice of initial function $R_{l1} U_l^{n-1}$. The following list of symbols is used in (2.8):

$U_k^{(s+1)} = U_k^n \in S_k$ is the approximation to $U_k(t_n)$ at the grid ω_k,

$U_k^{(i)} \in S_k$ is the i^{th} intermediate approximation at the grid ω_k,

$I_k \colon S_k \rightarrow S_k$ is the unit matrix,

$D_k^n \colon S_k \rightarrow S_k$ is a diagonal matrix with entries $(D_k^n)_{ii}$ either unity or zero,

$R_{lk} \colon S_l \rightarrow S_k$, $k = 1, \cdots, l$, is the natural restriction operator from ω_l to ω_k; $R_{ll} = I_l$,

$P_{k-1k} \colon S_{k-1} \rightarrow S_k$, $k = 2, \cdots, l$, is an interpolation operator from ω_{k-1} to ω_k,

$b_k^{(i)} \in S_k$ contains time-dependent terms emanating from the physical boundary $\partial\Omega$.

The above mentioned nesting property of the integration domains is induced by the actual grid adaptation strategy. This strategy determines at which nodes of ω_k integration or interpolation is carried out and defines the diagonal matrices D_k^n. Specifically, if at a certain node integration is to take place, then the associated diagonal entry of D_k^n, $(D_k^n)_{ii}$ say, is defined as $(D_k^n)_{ii} = 1$. For all remaining nodes, where interpolation is required, $(D_k^n)_{ii} = 0$. The nesting property itself cannot be recovered from the above formulation, as this is hidden in the actual definition of D_k^n.

The interpolation step on level $k \geqslant 2$ stands on its own and can be represented by

$$(I_k - D_k^n) U_k^{(i)} = (I_k - D_k^n) \, [P_{k-1k} U_{k-1}^{(i)} + b_k^{(i)}], \qquad 1 \leqslant i \leqslant s+1. \tag{2.9}$$

The gridfunction $b_k^{(i)}$ plays an auxiliary role here, but we need to include it due to the fact that physical boundary conditions have been worked into the semi-discrete system (method of lines). For the analysis presented in the remainder, $b_k^{(i)}$ plays no role whatsoever (it contains merely time-dependent terms and does not depend on the solution itself). Likewise, the integration step on the integration domain of level k can be represented by

$$D_k^n U_k^{(i)} = D_k^n \, [R_{lk} U_l^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_k(t_{n-1} + c_j \tau, \, U_k^{(j)})], \qquad 1 \leq i \leq s+1, \tag{2.10}$$

where, according to (2.8a), $D_l^n = I_1$. Noteworthy is that values at or beyond internal boundaries needed in the function evaluation in (2.10) are defined by (2.9). That is, formulation (2.8) automatically comprises the definition of Dirichlet type boundary values at internal boundaries at each RK stage. This follows from the observation that for nodes at and beyond internal boundaries the associated diagonal entry of D_k^n is zero (there is no integration at and beyond internal boundaries). Due to the internal boundaries, (2.10) cannot be considered uncoupled from the interpolation (2.9). Also observe that at each grid level the integration has the fine grid solution $D_k^n R_{lk} U_l^{n-1}$ as initial function.

In (2.8), the fully discrete approximations $U_k^{(i)}$ are defined on the whole of the grids ω_k and thus are also elements of S_k. The consequence is that for any $k \geqslant 2$ interpolation is considered to take place on the whole of ω_k. This, of course, is costly. In actual application, the interpolations are therefore restriced to the nested integration domains. This point will be discussed later in the paper. For the time being, it is assumed that numerical solutions are defined by (2.8) and thus are generated as grid functions in S_k for $k = 1, \cdots, l$ (grid expansion).

In (2.8), the number of grid levels l is fixed a priori, independent of time. In various interesting applications this so-called fixed-level mode of operation may be inefficient. For example, if a solution steepens up in time, less levels are needed in the initial part of the integration interval $(0,T]$ than at later times. This necessarily means that in the fixed-level mode of operation, at early times the number of levels must be taken larger than necessary, which is not efficient. On the other hand, the solution may also be less steep for later times, which again makes the fixed-level mode of operation inefficient when effort is wasted for later times. Obviously, the method should be capable of working with a variable number of levels. Henceforth, we call this the variable-level mode of operation.

For this mode of operation, (2.8) requires a little modification. Let l_{n-1}, l_n denote the required number of levels for the steps from t_{n-1} to t_n and t_n to t_{n+1}, respectively. Then, for the step from

t_{n-1} to t_n, (2.8) is modified as follows. For $k = 1$:

$$U_1^{(i)} = R_{l_{n-1}1} U_{l_{n-1}}^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_1(t_{n-1} + c_j \tau, U_1^{(j)}), \qquad 1 \leqslant i \leqslant s+1, \tag{2.11a}$$

for $k = 2, \cdots, l_{n-1}$:

$$U_k^{(i)} = D_k^n \, [R_{l_{n-1}k} U_{l_{n-1}}^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_k(t_{n-1} + c_j \tau, U_k^{(j)})] + \tag{2.11b}$$

$$(I_k - D_k^n) \, [P_{k-1k} U_{k-1}^{(i)} + b_k^{(i)}], \qquad 1 \leqslant i \leqslant s+1,$$

and, provided $l_n > l_{n-1}$, for $k = l_{n-1}+1, \cdots, l_n$:

$$U_k^{(i)} = P_{k-1k} U_{k-1}^{(i)} + b_k^{(i)}, \qquad 1 \leqslant i \leqslant s+1. \tag{2.11c}$$

Consequently, if the number of levels should increase for use in the next integration step, then so-called full interpolations (2.11c) are carried out at the end of the current step. This means that the required initial function, which is to be taken from the highest grid level that will be used, is always available. If $l_n \leqslant l_{n-1}$, then full interpolation is omitted and nothing really changes. For output at $t = t_n$, one may use either $U_{l_{n-1}}^n$ or $U_{l_n}^n$, assuming that the interpolation (2.11c) does not diminish accuracy. Needless to say that full interpolation is necessary only when the solution steepens up in time. Because we will let the number of levels depend exclusively on the spatial steepness, and because $\max_n \{l_n\}$ is finite, full interpolation will be carried out only for a finite number of steps, uniformly in τ. This means that full interpolation cannot have a strongly diminishing effect on global accuracy. Like the matrices D_k^n, the actual choice for the number of levels is part of the adaptation strategy.

We conclude this section with a minor modification for certain RK methods. As a rule, D_k^n depends only on the step number n and the level index k, and not on the stages. There exist RK methods for which all coefficients a_{1j} are zero, trivially so for all explicit methods, but it occurs for example also for the implicit Lobatto IIIA-methods ($s = 2$ yields the familiar trapezoidal rule). If this is the case, then it is more natural to define the 1st stage value in (2.8) as

$$U_k^{(1)} = R_{lk} U_l^{n-1} \tag{2.12}$$

and to avoid interpolation. This means that at this first stage D_k^n is to be replaced by the unit matrix I_k. No exception will be made for RK methods with this zero row property at a higher stage, since such methods are likely to be of little practical use.

3. The general error scheme

We will derive the general error scheme for the variable-level mode of operation. We also assume (2.12) in case all RK coefficients $a_{1j} = 0$. To save space, (2.11) is rewritten as

$$\begin{aligned} U_k^{(i)} = {} & D_k^n \, [R_{l_{n-1}k} U_{l_{n-1}}^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_k(t_{n-1} + c_j \tau, U_k^{(j)})] + \\ & (I_k - D_k^n) \, [P_{k-1k} U_{k-1}^{(i)} + b_k^{(i)}], \qquad 1 \leqslant i \leqslant s+1, \qquad 1 \leqslant k \leqslant l_n. \end{aligned} \tag{3.1}$$

Note that $D_1^n = I_1$ and $D_k^n = 0$ if $k > l_{n-1}$. Further, if $a_{1j} = 0$ $(1 \leqslant j \leqslant s)$, then D_k^n is to be replaced by I_k for $i = 1$, but only for $1 \leqslant k \leqslant l_{n-1}$. The rewriting of (2.11) into (3.1) introduces variables not existing in reality, viz. the grid functions $U_0^{(i)}$, $b_1^{(i)}$ and the operators P_{01} and $R_{l_{n-1}k}$ for $k > l_{n-1}$. We suppose in the remainder that any time such a dummy variable appears, it is replaced by zero. Formally we can use (3.1) due to the definiton of D_k^n. Hence, the introduction of the dummy variables has no influence on the result of the error analysis, even if they were not assigned the value zero. Further, we should also keep in mind that mostly $l_n \leqslant l_{n-1}$ which means that the full interpolation levels are absent.

The derivation of the error scheme parallels that in [15]. Consider the perturbed scheme

$$\begin{aligned} \tilde{U}_k^{(i)} = {} & D_k^n \, [R_{l_{n-1}k} \tilde{U}_{l_{n-1}}^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_k(t_{n-1} + c_j \tau, \tilde{U}_k^{(j)})] + \\ & (I_k - D_k^n) \, [P_{k-1k} \tilde{U}_{k-1}^{(i)} + b_k^{(i)}] + r_k^{(i)}, \qquad 1 \leqslant i \leqslant s+1, \qquad 1 \leqslant k \leqslant l_n, \end{aligned} \tag{3.2}$$

with the usual meaning for the local and global perturbations $r_k^{(i)}$ and $\tilde{U}_{l_{n-1}}^{n-1}$, $\tilde{U}_k^{(i)}$. Introduce the errors

$$e_k^n = \tilde{U}_k^n - U_k^n, \tag{3.3}$$

$$e_k^{(i)} = \tilde{U}_k^{(i)} - U_k^{(i)}, \qquad 1 \leqslant i \leqslant s+1, \qquad 1 \leqslant k \leqslant l_n,$$

and subtract (3.1) from (3.2) so as to obtain

$$\begin{aligned} e_k^{(i)} = D_k^n\,[R_{l_{n-1}k}e_{l_{n-1}}^{n-1} + \tau\sum_{j=1}^{s} a_{ij}M_k^{(j)}e_k^{(j)}] + \\ (I_k - D_k^n)P_{k-1k}e_{k-1}^{(i)} + r_k^{(i)}, \qquad 1 \leqslant i \leqslant s+1, \qquad 1 \leqslant k \leqslant l_n. \end{aligned} \tag{3.4}$$

$M_k^{(j)}$ is the integrated Jacobian matrix resulting from the use of the mean value theorem for vector functions:

$$F_k(t_{n-1} + c_j\tau, \tilde{U}_k^{(j)}) - F_k(t_{n-1} + c_j\tau, U_k^{(j)}) = M_k^{(j)}(\tilde{U}_k^{(j)} - U_k^{(j)}), \tag{3.5a}$$

$$M_k^{(j)} = \int_0^1 F'(t_{n-1} + c_j\tau, \theta\tilde{U}_k^{(j)} + (1-\theta)U_k^{(j)})d\theta. \tag{3.5b}$$

Clearly, the error scheme (3.4) is the most general one possible, since the matrices $M_k^{(j)}$ are defined by the nonlinear relation (3.5) and as yet no particular meaning has been given to the local perturbations $r_k^{(i)}$.

We next introduce the Kronecker product notation as commonly employed in the analysis of RK methods (see, e.g., [6], Section 5.1). Let I_{s+1} denote the unit matrix of order $s+1$ and denote $e = [1, \cdots 1]^T \in \mathrm{IR}^{s+1}$. Introduce the augmented vectors

$$\mathbf{e}_k^n = [e_k^{(1)\,T}, \cdots, e_k^{(s+1)\,T}]^T, \qquad \mathbf{r}_k^n = [r_k^{(1)\,T}, \cdots, r_k^{(s+1)\,T}]^T \tag{3.6}$$

in the augmented space $\mathbf{S}_k = \mathrm{IR}^{(s+1)d_k}$ and the matrix operators

$$\begin{aligned} &\mathbf{R}_{l_{n-1}k}\colon \mathbf{S}_{l_{n-1}} \to \mathbf{S}_k, \qquad &\mathbf{R}_{l_{n-1}k} = I_{s+1} \otimes R_{l_{n-1}k} = \mathrm{diag}(R_{l_{n-1}k}) \\ &\mathbf{P}_{k-1k}\colon \mathbf{S}_{k-1} \to \mathbf{S}_k, \qquad &\mathbf{P}_{k-1k} = I_{s+1} \otimes P_{k-1k} = \mathrm{diag}(P_{k-1k}), \\ &\mathbf{I}_k\colon \mathbf{S}_k \to \mathbf{S}_k, \qquad &\mathbf{I}_k = I_{s+1} \otimes I_k = \mathrm{diag}(I_k). \end{aligned} \tag{3.7}$$

Define

$$\mathbf{D}_k^n\colon \mathbf{S}_k \to \mathbf{S}_k, \qquad \mathbf{D}_k^n = \mathrm{diag}(I_k, D_k^n, \cdots, D_k^n) \tag{3.8a}$$

if $a_{1j} = 0$, $1 \leqslant j \leqslant s$ (cf. (2.12)), and otherwise

$$\mathbf{D}_k^n\colon \mathbf{S}_k \to \mathbf{S}_k, \qquad \mathbf{D}_k^n = I_{s+1} \otimes D_k^n = \operatorname{diag}(D_k^n). \tag{3.8b}$$

Finally, we introduce the augmented Jacobian operator

$$\mathbf{M}_k^n = \begin{bmatrix} a_{11}M_k^{(1)} & a_{12}M_k^{(2)} & \cdots & a_{1s}M_k^{(s)} & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{s1}M_k^{(1)} & a_{s2}M_k^{(2)} & \cdots & a_{ss}M_k^{(s)} & 0 \\ a_{s+11}M_k^{(1)} & a_{s+12}M_k^{(2)} & \cdots & a_{s+1s}M_k^{(s)} & 0 \end{bmatrix} \tag{3.9}$$

so that (3.4) can now be written in the compact form

$$\mathbf{Z}_k^n \mathbf{e}_k^n = \mathbf{D}_k^n \mathbf{R}_{l_{n-1}k}(e \otimes e_{l_{n-1}}^{n-1}) + (\mathbf{I}_k - \mathbf{D}_k^n)\mathbf{P}_{k-1k}\mathbf{e}_{k-1}^n + \mathbf{r}_k^n, \tag{3.10}$$

where $k = 1, \cdots, l_n$ and

$$\mathbf{Z}_k^n = \mathbf{I}_k - \tau \mathbf{D}_k^n \mathbf{M}_k^n. \tag{3.11}$$

In (3.10) we deal with an inner and an outer recursion. The inner recursion is connected with the grid refinement and has index k. The outer recursion is connected with the time stepping and has index n. For later use, we now rewrite (3.10) to an alternative form. Introduce

$$\mathbf{X}_k^n = (\mathbf{Z}_k^n)^{-1}(\mathbf{I}_k - \mathbf{D}_k^n)\mathbf{P}_{k-1k}, \tag{3.12a}$$
$$\Gamma_k^n = (\mathbf{Z}_k^n)^{-1}\mathbf{D}_k^n \mathbf{R}_{l_{n-1}k}, \tag{3.12b}$$
$$\phi_k^n = (\mathbf{Z}_k^n)^{-1}\mathbf{r}_k^n, \tag{3.12c}$$

where $k = 1, \cdots, l_n$. Note that

$$\mathbf{Z}_k^n = \mathbf{I}_k, \qquad \mathbf{X}_k^n = \mathbf{P}_{k-1k}, \qquad \Gamma_k^n = \mathbf{0}, \qquad \phi_k^n = \mathbf{r}_k^n \tag{3.13}$$

for the full interpolation levels $k = l_{n-1}+1, \cdots, l_n$. Using (3.12), (3.10) is rewritten as

$$\mathbf{e}_k^n = \mathbf{X}_k^n \mathbf{e}_{k-1}^n + \Gamma_k^n(e \otimes e_{l_{n-1}}^{n-1}) + \phi_k^n, \qquad k = 1, \cdots, l_n. \tag{3.14}$$

An elementary calculation then leads to the final form

$$\mathbf{e}_k^n = \mathbf{G}_k^n(e \otimes e_{l_{n-1}}^{n-1}) + \psi_k^n, \qquad k = 1, \cdots, l_n, \tag{3.15}$$

where $\mathbf{G}_k^n$ and ψ_k^n itself are also defined by recursions:

$$\mathbf{G}_1^n = \Gamma_1^n, \qquad \mathbf{G}_j^n = \mathbf{X}_j^n \mathbf{G}_{j-1}^n + \Gamma_j^n, \qquad j = 2, \cdots, k, \tag{3.16}$$

and

$$\psi_1^n = \phi_1^n, \qquad \psi_j^n = \mathbf{X}_j^n \psi_{j-1}^n + \phi_j^n, \qquad j = 2, \cdots, k. \tag{3.17}$$

The error equation (3.15) describes in a very general way the error propagation for increasing levels, up to level k, that occurs within one complete time step. When to be used as outer error recursion, i.e. error recursion for evolving time, we must put $k = l_n$ because we always use the highest level approximations $U_{l_{n-1}}^{n-1}$, $U_{l_n}^n$, $\cdots$ for step continuation. Hence,

$$\mathbf{e}_{l_n}^n = \mathbf{G}_{l_n}^n(e \otimes e_{l_{n-1}}^{n-1}) + \psi_{l_n}^n, \qquad n = 1,2, \cdots, \tag{3.18}$$

is the final error scheme for the highest level approximations. Both (3.15) and (3.18) do have the nature of a one-step RK error scheme in the augmented space. Similar as in the standard application of the RK method (single-level multi-stage), our main interest concerns the $(s+1)^{\text{th}}$ component vector. Note that the formulation (3.18) supposes that $U_{l_n}^n$ is taken for output rather than $U_{l_n-1}^n$.

4. Remarks on stability

The error equations (3.10), (3.15) and (3.18) look very similar to the associated ones of the multi-level implicit Euler method (see [15], formula (4.3) and (4.8) with $k = l$). In fact, if (2.2) is just the implicit Euler method, then the current multi-level multi-stage formulas completely coincide with their implicit Euler counterparts. In [15] we have presented a comprehensive analysis of the stability of the multi-level implicit Euler method. In spite of the intimate correspondence with the single-stage implicit Euler forms, and the availability of many results from the RK field [6], the multi-level multi-stage formulas are not so feasible for a comprehensive stability analysis. A technical difficulty originates from the property that at any RK stage, non-physical boundary values are defined by interpolating the solution of the corresponding stage from the next coarser grid. This implies that the internal RK stages play a more distinct role in the stability analysis than we are used to. For example, for constant coefficient linear problems, it is in the usual single-level application the familiar stability function of the final $(s+1)^{\text{th}}$ stage that fully determines the error propagation for evolving time. For multi-level multi-stage methods, even when applied to constant coefficient linear problems, the situation is essentially more complicated.

To see more clearly the difference with the standard single-level situation, consider, instead of (3.10), the single-level general error form

$$\mathbf{Z}\mathbf{e}^{n} = (e \otimes e^{n-1}) + \mathbf{r}^{n}, \tag{4.1}$$

where we assume that the problem is of constant coefficient linear type with matrix M ($\mathbf{Z}$ is then independent of n). The $(s+1)^{\text{th}}$ component $e^{(s+1)} = e^{n}$ of $\mathbf{e}^{n}$ is expressed in terms of the $(s+1)^{\text{th}}$ component of $\mathbf{Z}^{-1}(e \otimes e^{n-1})$ (stability) and $\mathbf{Z}^{-1}\mathbf{r}^{n}$ (local accuracy). In particular, the $(s+1)^{\text{th}}$ component of $\mathbf{Z}^{-1}(e \otimes e^{n-1})$ takes the simple form $R(\tau M)e^{n-1}$ where $R: \mathbb{C} \rightarrow \mathbb{C}$ represents the familiar stability function of the RK method, defined for the $(s+1)^{\text{th}}$ stage. Hence, here the internal stages play no longer a role.

Next consider the multi-level multi-stage case represented by (3.10). For simplicity we omit the local perturbation term $\mathbf{r}_{k}^{n}$, since it plays no role in stability considerations. Again assume that the problem is of constant coefficient linear type with matrix M_k ($\mathbf{Z}_{k}^{n}$ still depends on n now due to $\mathbf{D}_{k}^{n}$). Then, the multiplication of $(\mathbf{Z}_{k}^{n})^{-1}$ to $\mathbf{D}_{k}^{n}\mathbf{R}_{l_n - 1k}(e \otimes e_{l_n - 1}^{n-1})$ results, for the $(s+1)^{\text{th}}$ component vector, in a similar expression as in the standard case above, viz $R(\tau D_{k}^{n} M_{k})\ [D_{k}^{n} R_{l_n - 1k} e_{l_n - 1}^{n-1}]$. Hence, like above, properties of the stability function R and the matrix $\tau D_{k}^{n} M_{k}$ completely define the growth of this $(s+1)^{\text{th}}$ component and the internal stages play no role. However, the multiplication of $(\mathbf{Z}_{k}^{n})^{-1}$ to the interpolation term $(\mathbf{I}_{k} - \mathbf{D}_{k}^{n})\mathbf{P}_{k-1k}\mathbf{e}_{k-1}^{n}$ complicates matters. The reason is the occurrence of $\mathbf{e}_{k-1}^{n}$, the augmented vector containing all stage errors $e_{k-1}^{(i)}$. These usually differ from $e_{l_n - 1}^{n-1}$, so that the multiplication with $(\mathbf{Z}_{k}^{n})^{-1}$results in a $(s+1)^{\text{th}}$ component of $(\mathbf{Z}_{k}^{n})^{-1}(\mathbf{I}_{k} - \mathbf{D}_{k}^{n})\mathbf{P}_{k-1k}\mathbf{e}_{k-1}^{n}$ which is much more complicated than the other, standard one. Specifically, it involves all stage errors $(I_k - D_k^n)\ P_{k-1k} e_{k-1}^{(i)}$, multiplied by polynomial or rational matrix functions of various kinds connected to the RK stages. These functions are fully determined by $(\mathbf{Z}_k^n)^{-1}$through the array of coefficients of the RK formula. Thus, to proceed to the final $(s+1)^{\text{th}}$ stage error $e^{(s+1)} = e^n$ of level l_n formulated in (3.18), so as to obtain a stability expression of the type amplification operator times $e_{l_n-1}^{n-1}$, all stage errors $(I_k - D_k^n) P_{k-1k} e_{k-1}^{(i)}$ must be carried along and subsequently eliminated. This observation explains why the multi-level multi-stage formulas are not so feasible for a comprehensive stability analysis, although it can be done for constant coefficient linear problems, of course. However, a comprehensive and general nonlinear analysis, as presented in [15] for the multi-level single-stage implicit Euler scheme, seems most unlikely.

The discussion above outlines that in application the stability depends on at least three factors, viz (i) the common step-by-step stability of the underlying RK method which for linear problems is fully determined by the common stability function, (ii) certain stability properties connected with the internal stages, and (iii) the stability of the interpolation procedure. However, there exists a fourth factor which diminishes the combined role of the interpolation and the internal stages. This factor is connected with the adaptation strategy for selecting the actual integration domains. For accuracy reasons, this strategy should be such that the interpolation takes place exclusively in low error regions in the space domain, while in large error regions integrations are carried out. If this is the case, then the effect of any possible source of instability arising from the interpolation at internal stages, is likely to be of less importance than the common step-by-step stability of the RK method. In other words, if the RK integration on each of the integration domains is carried out in a stable way, then we believe that the stability behaviour of the multi-level multi-stage RK method will not differ significantly from the behaviour encountered when applying the RK method without adaptation. Finally, when considered on its own, linear interpolation is stable and hence has no effect on stability. In case higher order (polynomial Lagrangrian) interpolation is used, then the interpolation will have some effect. However, this again cannot be significant if the adaptations are such that the interpolation takes place exclusively in low error regions. This remark obviously does not apply to the full interpolation levels (2.11c). However, as said before, in an integration over a finite interval, there are only a few steps containing full interpolation levels, the number of which is independent of the total number of steps taken. Trivially, a few of such steps cannot cause instability.

In this paper no further attention will be paid to the stability analysis. Instead, we will tacitly assume that we have stability in actual application. This means that we assume that the transition from local to global, governed by the error equation (3.18), is such that the components $\psi_{l_j}^{(s+1)} = \psi_{l_j}^{j}$ of $\psi_{l_j}^{j}$, $1 \leq j \leq n$, are 'added up' in the usual way so that a standard stability inequality like

$$\|e_{l_n}^{n}\| \leq C\sum_{j=1}^{n} \|\psi_{l_j}^{j}\|, \qquad n = 1,2, \cdots \tag{4.2}$$

applies. Here, $C \geq 1$ is supposed to be a constant of moderate size, independent of temporal and spatial meshwidths, while also the 'addition' of local errors takes place under appropriate nonrestrictive conditions. As outlined above, it is reasonable to assume that this standard form of stability applies, as long the integrator integrates in a stable way. This stability assumption allows us to proceed with the local error analysis which is to reveal how to define the adaptation strategy for choosing the spatial integration domains at the various refinement levels. Obviously, this is one of the main issues in the analysis, implementation and application of adaptive grid methods.

5. The local errors

5.1. Preliminaries

In the following, $\|.\|$ denotes the conventional maximum norm. In spite of the fact that the maximum norm is seldom used in the analysis of RK methods (see Kraaijevanger [9], Spijker [13] and the references therein for exceptions), in the error analysis below we exclusively use the maximum norm as this norm is most natural for implementing adaptation strategies. Note that $\|.\|$ stands for the maximum norm in any space S_k or $\mathbf{S}_k$ under consideration, while the same symbol will be used for operators.

Consider the error equation (3.15). We will examine the total local error ψ_k^n obtained by associating the local perturbations $\mathbf{r}_k^n$ with the true PDE solution. Note that the global errors $\mathbf{e}_k^n$ then become global discretization errors, viz

$$\mathbf{e}_k^n = \mathbf{u}_k^n - \mathbf{U}_k^n, \qquad n = 0,1, \cdots, \qquad 1 \leqslant k \leqslant l_n. \tag{5.1}$$

After having obtained bounds for ψ_k^n and its main component $\psi_k^n = \psi_k^{(s+1)}$, the stability assumption underlying (4.2) can then be used to conclude convergence according to (4.3). For clarity, henceforth we will consistently call ψ_k^n the total local error, whereas $\mathbf{r}_k^n$ will be consistently called a residual, so as to distinguish from ψ_k^n. Note that ψ_k^n can be interpreted as the k^{th} level global error after one time step starting from the true PDE solution (put $e_{l_n-1}^{n-1} = 0$ in (3.15)).

In the previous Sections 3 and 4, we have tacitly used the natural assumption that any occurring augmented RK operator $\mathbf{Z}_k^n$ is invertible (under appropriate conditions on τ and ω_k). For the local error analysis it is convenient to introduce the following natural bound:

$$\|(\mathbf{Z}_k^n)^{-1}\mathbf{v}\| \leqslant C\|\mathbf{v}\| \tag{5.2}$$

for any occurring $\mathbf{v} \in \mathbf{S}_k$. Here, $C \geqslant 1$ denotes a constant independent of τ and ω_k, while τ itself satisfies $\tau \leqslant \tau_0$ with τ_0 possibly depending on ω_k. The constant C and stepsize bound τ_0 are assumed to take on appropriate values (C close to 1 and τ_0 not unduly restrictive).

Following [15] the main aim of the local error analysis is to derive a refinement or adaptivity condition that distributes space discretization and interpolation errors in such a way that the local spatial accuracy obtained on the grid ω_{l_n-1} is comparable to the local spatial accuracy on ω_{l_n-1} if this grid would be used without any adaptation. Assuming a stable time stepping process, this will then also be true for the final global spatial accuracy. Needless to say that this is one of the main goals of any adaptive grid method based on local uniform grid refinement.

5.2. The local error ψ_k^n

To begin with we replace, in the perturbed scheme (3.2), all $\tilde{U}_k^{(i)}$ values by the corresponding PDE solution values $u_k^{(i)}$. An elementary calculation then shows that, in the space $\mathbf{S}_k$, the resulting residual $\mathbf{r}_k^n$ can be expressed as

$$\mathbf{r}_k^n = \mathbf{D}_k^n(\beta_k^n + \tau\sigma_k^n) + (\mathbf{I}_k - \mathbf{D}_k^n)\gamma_k^n, \tag{5.3}$$

where

$$\beta_k^n = [\beta_k^{(1)\,T}, \cdots, \beta_k^{(s)\,T}, \beta_k^{(s+1)\,T}]^T, \tag{5.4a}$$
$$\sigma_k^n = (A \otimes I_k)\,[\alpha_k^{(1)\,T}, \cdots, \alpha_k^{(s)\,T}, \alpha_k^{(s+1)\,T}]^T, \tag{5.4b}$$
$$\gamma_k^n = [\gamma_k^{(1)\,T}, \cdots, \gamma_k^{(s)\,T}, \gamma_k^{(s+1)\,T}]^T, \tag{5.4c}$$

The component $\beta_k^{(i)}$ is the PDE residual defined for the i^{th} RK stage:

$$\beta_k^{(i)} = u_k(t_{n-1} + c_i\tau) - u_k(t_{n-1}) - \tau\sum_{j=1}^{s} a_{ij}\frac{d}{dt}u_k(t_{n-1} + c_j\tau). \tag{5.5}$$

The component $\alpha_k^{(i)}$ is the PDE residual defined by the semi-discretization:

$$\alpha_k^{(i)} = \frac{d}{dt}u_k(t_{n-1} + c_i\tau) - F_k(t_{n-1} + c_i\tau, u_k(t_{n-1} + c_i\tau)). \tag{5.6}$$

Following common use, α_k^n, and likewise σ_k^n and their components, will also be called local space discretization error. The matrix A represents the $(s+1) \times (s+1)$ matrix of RK coefficients a_{ij} and reads

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1s} & 0 \\ . & . & . & . \\ . & . & . & . \\ a_{s1} & \cdots & a_{ss} & 0 \\ a_{s+11} & \cdots & a_{s+1s} & 0 \end{bmatrix} \tag{5.7}$$

Hence, the i^{th} component $\sigma_k^{(i)}$ of σ_k^n is given by

$$\sigma_k^{(i)} = \sum_{j=1}^{s} a_{ij}\alpha_k^{(j)}. \tag{5.8}$$

Finally, the component $\gamma_k^{(i)}$ is the PDE residual defined by the interpolation:

$$\gamma_k^{(i)} = u_k(t_{n-1} + c_i\tau) - P_{k-1k}u_{k-1}(t_{n-1} + c_i\tau) - b_k(t_{n-1} + c_i\tau) \tag{5.9}$$

and $\gamma_k^{(i)}$ and γ_k^n will also be called interpolation error. Observe that any component vector

$$r_k^{(i)} = D_k^n(\beta_k^{(i)} + \tau\sigma_k^{(i)}) + (I_k - D_k^n)\gamma_k^{(i)} \tag{5.10}$$

of $\mathbf{r}_k^n$, associated to stage i and level k , is now determined completely by the true PDE solution $u = u(\underline{x}, t)$. As usual, any residual component $r_k^{(i)}$ can be Taylor expanded assuming sufficient differentiability of u.

We are now ready to determine the local error ψ_k^n defined by recursion (3.17). Different from the residual $\mathbf{r}_k^n$, which is exclusively determined by level k quantities, ψ_k^n contains contributions from all levels $j \leqslant k$. Using the convention

$$\prod_{i=k}^{k+1}\mathbf{X}_k^n = \mathbf{I}_k, \qquad k = 1, \cdots, l_n, \tag{5.11}$$

and (3.12c) and (5.3), we get

$$\begin{aligned}\psi_k^n &= \sum_{j=1}^{k}(\prod_{i=k}^{j+1}\mathbf{X}_i^n)\phi_j^n \\ &= \sum_{j=1}^{k}(\prod_{i=k}^{j+1}\mathbf{X}_i^n)(\mathbf{Z}_j^n)^{-1}[\mathbf{D}_j^n(\beta_j^n + \tau\sigma_j^n) + (\mathbf{I}_j - \mathbf{D}_j^n)\gamma_j^n].\end{aligned} \tag{5.12}$$

A natural splitting into a temporal and a spatial local error is

$$\psi_k^n = \psi_{k,s}^n + \psi_{k,t}^n, \tag{5.13}$$

where

$$\psi_{k,s}^n = \sum_{j=1}^{k}(\prod_{i=k}^{j+1}\mathbf{X}_i^n)(\mathbf{Z}_j^n)^{-1}[\tau\mathbf{D}_j^n\sigma_j^n + (\mathbf{I}_j - \mathbf{D}_j^n)\gamma_j^n], \tag{5.14}$$

$$\psi_{k,t}^n = \sum_{j=1}^{k}(\prod_{i=k}^{j+1}\mathbf{X}_i^n)(\mathbf{Z}_j^n)^{-1}\mathbf{D}_j^n\beta_j^n. \tag{5.15}$$

The local space error $\psi_{k,s}^n$ contains only contributions from the spatial approximation, viz local space discretization errors σ_j^n and spatial interpolation errors γ_j^n. The local time error $\psi_{k,t}^n$ contains only contributions β_j^n from the time integration. Hence, due to the splitting (5.13), for the spatial local error analysis we may restrict ourselves to $\psi_{k,s}^n$ and for the temporal local error analysis to $\psi_{k,t}^n$.

5.3. The local space error $\psi_{k,s}^n$

We rewrite $\psi_{k,s}^n$ as

$$\begin{aligned} \psi_{k,s}^n &= (\mathbf{Z}_k^n)^{-1}(\mathbf{I}_k - \mathbf{D}_k^n)\mathbf{P}_{k-1k} \star \\ &\quad \sum_{j=1}^{k-1}\Big(\prod_{i=k-1}^{j+1}\mathbf{X}_i^n\Big)(\mathbf{Z}_j^n)^{-1}[\tau\mathbf{D}_j^n\sigma_j^n + (\mathbf{I}_j - \mathbf{D}_j^n)\gamma_j^n] + \\ &\quad (\mathbf{Z}_k^n)^{-1}[\tau\mathbf{D}_k^n\sigma_k^n + (\mathbf{I}_k - \mathbf{D}_k^n)\gamma_k^n] \\ &= (\mathbf{Z}_k^n)^{-1}[\tau\mathbf{D}_k^n\sigma_k^n + (\mathbf{I}_k - \mathbf{D}_k^n)\rho_k^n], \end{aligned} \tag{5.16}$$

where

$$\rho_1^n = \mathbf{0}, \tag{5.17a}$$

$$\rho_k^n = \gamma_k^n + \mathbf{P}_{k-1k}\psi_{k-1,s}^n, \qquad k = 2, \cdots, l_n. \tag{5.17b}$$

In (5.16), the local space discretization error $\mathbf{D}_k^n\sigma_k^n$, defined at the level k integration domain, is separated from the local spatial error part $(\mathbf{I}_k - \mathbf{D}_k^n)\rho_k^n$ living outside this domain. From (5.17) we see that ρ_k^n contains the level k interpolation error γ_k^n and the prolongated local space error $\mathbf{P}_{k-1k}\psi_{k-1,s}^n$. At the full interpolation levels, (5.16) simplifies to

$$\psi_{k,s}^n = \gamma_k^n + \mathbf{P}_{k-1k}\psi_{k-1,s}^n, \qquad k = l_{n-1}+1, \cdots, l_n. \tag{5.18}$$

The separation of errors in (5.16) enables us to formulate the important refinement condition:

$$\|(\mathbf{Z}_{l_n-1}^n)^{-1}(\mathbf{I}_{l_n-1} - \mathbf{D}_{l_n-1}^n)\rho_{l_n-1}^n\| \leqslant c\|(\mathbf{Z}_{l_n-1}^n)^{-1}\tau\mathbf{D}_{l_n-1}^n\sigma_{l_n-1}^n\|, \tag{5.19}$$

where $c > 0$ denotes a threshold factor to be specified later. Substitution in (5.16) yields

$$\|\psi_{l_n-1,s}^n\| \leqslant (1+c)\|(\mathbf{Z}_{l_n-1}^n)^{-1}\tau\mathbf{D}_{l_n-1}^n\sigma_{l_n-1}^n\|. \tag{5.20}$$

Hence, apart from the factor $(1+c)$, the local space error at the finest integration level is smaller than or equal to the local space discretization error on its integration domain. In other words, by imposing (5.19), we have virtually removed the error contribution from interpolation committed on all levels

$k \leqslant l_{n-1}$. Inequality (5.20) is in agreement with our goal of developing an adaptation strategy that generates integration domains in such a way that the spatial accuracy obtained on the finest level is comparable to the spatial accuracy obtained on this level if the corresponding grid would be used without any adaptation at all.

The refinement condition (5.19) is to be interpreted as a constraint on the matrices $\mathbf{D}_k^n$ for $2 \leqslant k \leqslant l_{n-1}$. A trivial solution is provided by the choice $\mathbf{D}_{l_{n-1}}^n = \mathbf{I}_{l_{n-1}}$. This choice makes no sense of course due to the nesting property. Taking into account this nesting property, to satisfy (5.19) constraints must be imposed on all matrices $\mathbf{D}_k^n$, for the simple reason that at any level interpolation errors show up and enter $\rho_{l_{n-1}}^n$. These constraints follow from the following derivation. Let, for brevity, $l = l_{n-1}$. With a simple calculation [15], we can rewrite ρ_l^n as

$$\rho_l^n = \lambda_l^n + \mathbf{P}_{l-1l} \sum_{k=2}^{l-1} (\prod_{i=l-1}^{k+1} \mathbf{X}_i^n)(\mathbf{Z}_k^n)^{-1}(\mathbf{I}_k - \mathbf{D}_k^n)\lambda_k^n, \tag{5.21}$$

where

$$\lambda_k^n = \gamma_k^n + \mathbf{P}_{k-1k}(\mathbf{Z}_{k-1}^n)^{-1} \tau \mathbf{D}_{k-1}^n \sigma_{k-1}^n, \qquad k = 2, \cdots, l, \tag{5.22}$$

contains the interpolation error at level k and the prolongation of the spatial discretization error of level $k-1$ to k. Note that for $k = l - 1$ convention (5.11) applies. This λ-function will be used for determining the matrices $\mathbf{D}_k^n$. Let C denote the norm bound of $(\mathbf{Z}_k^n)^{-1}$(see (5.2)). Let $C_I \geqslant 1$ be a constant, independent of k, such that

$$\|\mathbf{P}_{k-1k}\| \leqslant C_I. \tag{5.23}$$

Note that for linear interpolation $C_I = 1$, while for higher order Lagrangian interpolation $C_I > 1$. Now,

$$\| \prod_{i=l-1}^{k+1} \mathbf{X}_i^n \| \leqslant C_X \leqslant (CC_I)^{l-k-1} \tag{5.24}$$

and using (5.21) we get

$$\|(\mathbf{Z}_l^n)^{-1}(\mathbf{I}_l - \mathbf{D}_l^n)\rho_l^n\| \leqslant \tilde{C} \max\nolimits_{2 \leqslant k \leqslant l}\|(\mathbf{I}_k - \mathbf{D}_k^n)\lambda_k^n\| \tag{5.25}$$

with

$$\tilde{C} = C(1 + C_I(l - 2)CC_X) = C + (l - 2)(CC_I)^{l-k} \tag{5.26}$$

Hence, if for each $k = 2, \cdots, l$, the matrices $\mathbf{D}_k^n$ are selected such that

$$\|(\mathbf{I}_k - \mathbf{D}_k^n)\lambda_k^n\| \leqslant \frac{c}{\tilde{C}}\|(\mathbf{Z}_l^n)^{-1}\tau\mathbf{D}_l^n\sigma_l^n\|, \qquad l = l_{n-1}, \tag{5.27}$$

then the refinement condition (5.19) is satisfied.

In the remainder, (5.27) thus replaces (5.19). This condition says that, outside the integration domain of any level, the sum of the interpolation error and the prolongation of the spatial discretization error from the previous coarser level, shall be smaller than or equal to the final spatial discretization error of the highest level, multiplied by the constant $\frac{c}{\tilde{C}}$. Clearly, this imposes a severe restriction on the size of the interpolation and spatial discretization errors of the lower levels. On the other hand, this restriction is also natural, because, when going to a higher level within the current base time step, we never return to a grid point where the solution has been interpolated (nesting property). Noteworthy is that in the spatial refinement condition also the temporal stepsize τ features. In particular, if $\tau \rightarrow 0$, then the interpolation errors will prevail and $\mathbf{D}_k^n \rightarrow \mathbf{I}_k$. Recall that we interpolate at each base time step so that interpolation errors can accumulate linearly with the number of base time steps. This threat is present in static-regridding methods using heuristic criteria, like, e.g., equidistribution of arclength or curvature (see e.g. Revilla [10]). Our refinement condition prevents this unwanted accumulation of interpolation errors.

The refinement condition (5.27) is not applicable to the full interpolation levels as at these levels $\mathbf{D}_k^n = \mathbf{0}$. For simplicity, we now consider only one full interpolation level and note that this is sufficient for practical purposes. In case $l_n = l_{n-1} + 1$, we thus have, instead of (5.20), the local space error bound

$$\begin{aligned} \|\psi^n_{l_n,s}\| &\leqslant \|\gamma^n_{l_n}\| + C_I\|\psi^n_{l_n-1,s}\| \\ &\leqslant \|\gamma^n_{l_n}\| + C_I(1+c)\tau\|(\mathbf{Z}^n_{l_n-1})^{-1}\mathbf{D}^n_{l_n-1}\sigma^n_{l_n-1}\|. \end{aligned} \tag{5.28}$$

Recall that full interpolation occurs only in a finite number of steps, uniformly in τ. When adding all local errors for a convergence proof, like in (4.2), this fact should be taken into account so as to avoid an overly pessimistic summation like

$$\sum_{j=1}^{n} \|\gamma^j_{l_j}\| \geqslant \frac{T}{\tau}\min_n\|\gamma^n_{l_n}\|. \tag{5.29}$$

This term blows up for $\tau\to 0$ and is not in correspondence with reality. With a more subtle summation, taking into account the finite number of full interpolations, the $\frac{1}{\tau}$-term is avoided.

In conclusion, by imposing the refinement condition (5.27), the local space error bounds (5.20), (5.28) are valid. In a concrete implementation, the two bounds can be used to monitor the size of the spatial accuracy, while accordingly (5.27) is then used for selecting the actual integration domains. Such an implementation is method dependent and therefore best to describe for a selected method. An illustration for a DIRK method is presented further on this paper. Finally we briefly comment on the threshold factor c present in (5.27). The error bound (5.20) suggests to choose c not too large. However, if we take c very small, then the effect will be that the greater part of the diagonal entries of $\mathbf{D}^n_k$ are put to unity to satisfy the refinement condition. This of course has the effect that the integration domains will become quite large.

5.4. The local time error $\psi^n_{k,t}$

Consider (5.15). Since the same temporal stepsize is used at all integration levels, and β^n_k does not depend on the mesh width, we have

$$\beta^n_k = \mathbf{R}_{l_n-1k}\beta^n_{l_n-1}, \qquad 1 \leqslant k \leqslant l_{n-1}, \tag{5.30}$$

so that (5.15) yields

$$\psi^n_{k,t} = \sum_{j=1}^{k}\Big(\prod_{i=k}^{j+1}\mathbf{X}^n_i\Big)(\mathbf{Z}^n_j)^{-1}\mathbf{D}^n_j\mathbf{R}_{l_n-1j}\beta^n_{l_n-1}. \tag{5.31}$$

By comparison with the recursion (3.16) for the amplification operators G_k^n, one can see that

$$\psi_{k,t}^n = \mathbf{G}_k^n \beta_{l_{n-1}}^n, \qquad 1 \leqslant k \leqslant l_{n-1}. \tag{5.32}$$

This formula shows the dependence of the local time error on the temporal residual of the finest integration level. Alternatively, we may write, similar as for the local space error,

$$\psi_{1,t}^n = (\mathbf{Z}_1^n)^{-1}\mathbf{D}_1^n \mathbf{R}_{l_{n-1}1} \beta_{l_{n-1}}^n. \tag{5.33}$$

$$\psi_{k,t}^n = \mathbf{X}_k^n \sum_{j=1}^{k-1} (\prod_{i=k-1}^{j+1} \mathbf{X}_i^n)(\mathbf{Z}_j^n)^{-1}\mathbf{D}_j^n \mathbf{R}_{l_{n-1}j} \beta_{l_{n-1}}^n + (\mathbf{Z}_k^n)^{-1}\mathbf{D}_k^n \mathbf{R}_{l_{n-1}k} \beta_{l_{n-1}}^n$$

$$= (\mathbf{Z}_k^n)^{-1}[\mathbf{D}_k^n \mathbf{R}_{l_{n-1}k} \beta_{l_{n-1}}^n + (\mathbf{I}_k - \mathbf{D}_k^n)\mathbf{P}_{k-1k}\psi_{k-1,t}^n], \qquad k = 2, \cdots, l_{n-1}.$$

This representation shows more insight than (5.32). At each integration level we recover the local time error contribution committed on the integration domain, viz $(\mathbf{Z}_k^n)^{-1}[\mathbf{D}_k^n \mathbf{R}_{l_{n-1}k} \beta_{l_{n-1}}^n]$, and the prolongation of the previous local time error of the next coarser level, viz $(\mathbf{Z}_k^n)^{-1}[(\mathbf{I}_k - \mathbf{D}_k^n)\mathbf{P}_{k-1k}\psi_{k-1,t}^n]$.

Let $\tilde{p}$ denote the stage order of the RK method, that is,

$$\tilde{p} = \min_{1 \leqslant i \leqslant s+1}\{\tilde{p}^{(i)}\}, \tag{5.34}$$

where $\tilde{p}^{(i)}$ is the (quadrature) order of the i^{th} temporal residual $\beta_k^{(i)}$ given in (5.4). Hence,

$$\beta_k^{(i)} = \mathcal{O}(\tau^{\tilde{p}^{(i)}+1}), \qquad \tau \rightarrow 0. \tag{5.35}$$

From the norm bound (5.2) it directly follows that

$$\psi_{1,t}^n = \mathcal{O}(\tau^{\tilde{p}+1}), \qquad \tau \rightarrow 0. \tag{5.36}$$

Next, for $k = 2, \cdots, l_{n-1}$, we have

$$\|\psi_{k,t}^n\| \leqslant \|(\mathbf{Z}_k^n)^{-1}\| \, \|\mathbf{D}_k^n \mathbf{R}_{l_{n-1}k} \beta_{l_{n-1}}^n + (\mathbf{I}_k - \mathbf{D}_k^n)\mathbf{P}_{k-1k}\psi_{k-1,t}^n\| \tag{5.37}$$

$$\leqslant C \max\{\|\mathbf{D}_k^n \mathbf{R}_{l_{n-1}k} \beta_{l_{n-1}}^n\|, \|(\mathbf{I}_k - \mathbf{D}_k^n)\mathbf{P}_{k-1k}\psi_{k-1,t}^n\|\}.$$

Using (5.36), and the fact that the first term is $\mathcal{O}(\tau^{\tilde{p}+1})$, we then also have

$$\psi^n_{k,t} = \mathcal{O}(\tau^{\tilde{p}+1}), \qquad 1 \leqslant k \leqslant l_{n-1}. \tag{5.38}$$

In particular, for any i^{th} stage, including the $(s+1)^{\text{th}}$ output stage,

$$\psi^{(i)}_{k,t} = \mathcal{O}(\tau^{\tilde{p}+1}), \qquad 1 \leqslant k \leqslant l_{n-1}. \tag{5.39}$$

Of interest is that, apart from the norm bounds C for $(\mathbf{Z}^n_k)^{-1}$ and C_I for $\mathbf{P}_{j-1j}$, the order constant involved depends exclusively on bounds for one or more temporal derivatives of the PDE solution u. The bounds for these temporal derivatives of u emanate from the residual result (5.35).

Finally, no integration takes place at a full interpolation level, so that

$$\psi^n_{k,t} = \mathbf{P}_{k-1k}\psi^n_{k-1,t}, \qquad l_{n-1}+1 \leqslant k \leqslant l_n. \tag{5.40}$$

Like in the previous section, we consider only one full interpolation level and note again that this is sufficient for practical purposes. In case $l_n = l_{n-1}+1$, we thus have the same temporal order as for $\psi^n_{k,t}$, $1 \leqslant k \leqslant l_{n-1}$.

The stage order plays a crucial role in the B-convergence theory for stiff ODEs, and, accordingly, also in convergence theories for 'Method of Lines' schemes for time-dependent PDEs (see, e.g., [11, 12] the references therein). As a rule, $\tilde{p} < p$, p denoting the conventional order of the RK method. One reason is that p is associated to the $(s+1)^{\text{th}}$ output stage of the method only, whereas the stage order takes all stages into account. However, when comparing p and $\tilde{p}$, the crucial consideration is that $\tilde{p}$ is associated to residuals, whereas p is associated to the local ODE error (of the output stage) of the RK method. This local ODE error underlies the conventional Taylor expansion of the RK operator and hence involves derivatives of the ODE operator F. Specifically, in this expansion, terms may be present which are "large" and not related to smoothness properties of the solution itself, in contrast with the expansion of residuals. In method of lines applications, these "large" terms depend on the spatial mesh width and typically blow up as this mesh width approaches zero. This means, in theory, a reduction of order from p to $\tilde{p}$, which is known to originate mainly from the boundary conditions [11, 12]. Therefore, the order observed in actual application can be lower than p. The order observed is necessarily problem dependent and lies always in the interval $[\tilde{p}, p]$. For explicit methods and many of the DIRK methods, the stage order $\tilde{p} = 1$. Consequently, for such methods the temporal accuracy on fine grids may be disappointing. On the other hand, it also frequently happens that accuracy predictions based on the stage order turn out to be rather conservative. In the next section we will further illustrate the order reduction phenomenon for a multi-level 3-stage DIRK method having $p = 3$, $\tilde{p} = 2$.

6. Error analysis for a 3-stage DIRK method

In this section we will elaborate the local error analysis for a typical 3-stage DIRK method which later on will be used for numerical illustrations. The elaboration focusses on the order reduction phenomenon in the local time error and on the grid refinement condition in connection with the local space error.

6.1. The DIRK method

The DIRK method is due to [2] (see [5]) and defined by the Butcher array

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 2\theta & \theta & \theta & 0 \\ 1 & b_1 & b_2 & \theta \\ \hline & b_1 & b_2 & \theta \end{array} \qquad \begin{aligned} \theta &= (3+\sqrt{3})/6 \\ b_1 &= 3/2-\theta-1/(4\theta) \\ b_2 &= -1/2+1/(4\theta) \end{aligned} \tag{6.1}$$

It is strongly A-stable, has classical order $p = 3$, stage order $\tilde{p} = 2$, and uses only two effective stages (first row of coefficients is zero). Note that stage one and two define the trapezoidal rule and that stage three and four, the output stage, are identical. Also note that $c_2 = 2\theta > 1$. Hence, the second stage evaluation of $F(t, U)$ takes place outside the step interval $[t_{n-1}, t_n]$. For certain applications this may be inconvenient. For example, if $F(t, U)$ is not defined beyond the physical end time.

6.2. Elaboration of the local time error

For simplicity of presentation, we will assume that the semi-discrete problem (2.6) is of constant coefficient linear type,

$$\frac{d}{dt}U_k(t) = M_k U_k(t) + f_k(t). \tag{6.2}$$

We stipulate that the linear case reveals the essentials of the order reduction phenomenon. Also for simplicity, we put $l_{n-1} = 2$. Conclusions for the higher level case immediately follow. Thus, our task is to examine

$$\begin{aligned} \psi^n_{1,t} &= (\mathbf{Z}^n_1)^{-1}\mathbf{R}_{21}\beta^n_2, \\ \psi^n_{2,t} &= (\mathbf{Z}^n_2)^{-1}[\mathbf{D}^n_2\beta^n_2 + (\mathbf{I}_2 - \mathbf{D}^n_2)\mathbf{P}_{12}\psi^n_{1,t}]. \end{aligned} \tag{6.3}$$

The components of β^n_2 satisfy (Taylor expand (5.5))

$$\begin{aligned} \beta^{(1)}_2 &= 0, \\ \beta^{(2)}_2 &= -\frac{2\theta^3}{3}\tau^3\frac{d^3}{dt^3}u_2(t_{n-1}) + \mathcal{O}(\tau^4), \\ \beta^{(3)}_2 &= (\frac{1}{24} - \frac{\theta}{6} - \frac{\theta^2}{3} + \frac{2\theta^3}{3})\tau^4\frac{d^4}{dt^4}u_2(t_{n-1}) + \mathcal{O}(\tau^5), \\ \beta^{(4)}_2 &= \beta^{(3)}_2. \end{aligned} \tag{6.4}$$

For any $\mathbf{v}_k \in \mathbf{S}_k$ having $\mathrm{v}^{(1)}_k = 0$, the components $\mathrm{w}^{(i)}_k$ of $\mathbf{w}_k = (\mathbf{Z}^n_k)^{-1}\mathbf{v}_k$ read

$$\begin{aligned} \mathrm{w}^{(1)}_k &= 0, \\ \mathrm{w}^{(2)}_k &= (I_k - \theta\tau D^n_k M_k)^{-1}\mathrm{v}^{(2)}_k, \\ \mathrm{w}^{(3)}_k &= (I_k - \theta\tau D^n_k M_k)^{-2}b_2\tau D^n_k M_k \mathrm{v}^{(2)}_k + (I_k - \theta\tau D^n_k M_k)^{-1}\mathrm{v}^{(3)}_k, \\ \mathrm{w}^{(4)}_k &= w^{(3)}_k. \end{aligned} \tag{6.5}$$

We note in passing that the bound (5.2) may be derived from the more specific bound

$$\|(I_k - \theta\tau D^n_k M_k)^{-1}\mathrm{v}_k\| \leqslant (1-\theta\tau\mu)^{-1}\|\mathrm{v}_k\|, \qquad 1-\theta\tau\mu > 0, \tag{6.6}$$

with μ independent of (the mesh width of) M_k. This bound applies in all cases where the well-known implicit Euler method integrates in a stable way [6]. The constant μ represents the logarithmic norm

$\mu_\infty[D_k^n M_k]$.

Put the level index $k = 1$. We have

$$\psi_{1,t}^{(1)} = 0, \tag{6.7}$$

$$\psi_{1,t}^{(2)} = (I_1 - \theta\tau M_1)^{-1} R_{21} \beta_2^{(2)}$$

$$= -\frac{2\theta^3}{3}(I_1 - \theta\tau M_1)^{-1}\tau^3 R_{21}\frac{d^3}{dt^3}u_2(t_{n-1}) + \mathcal{O}(\tau^4),$$

$$\psi_{1,t}^{(3)} = (I_1 - \theta\tau M_1)^{-2} b_2 \tau M_1 R_{21} \beta_2^{(2)} + (I_1 - \theta\tau M_1)^{-1} R_{21} \beta_2^{(3)}$$

$$= -b_2\frac{2\theta^3}{3}(I_1 - \theta\tau M_1)^{-2} M_1 R_{21}\tau^4\frac{d^3}{dt^3}u_2(t_{n-1}) + \mathcal{O}(\tau^4),$$

$$\psi_{1,t}^{(4)} = \psi_{1,t}^{(3)}.$$

Using the boundedness of the operators $(I_1 - \theta\tau M_1)^{-1}$, $(I_1 - \theta\tau M_1)^{-2}\tau M_1$, the order result (5.39) with stage order $\tilde{p} = 2$ is now directly recovered. Also the classical order $p = 3$ follows from $\psi_{1,t}^{(4)}$, when interpreted as the local ODE error (of the output stage) of the DIRK scheme. However, for classical order $p = 3$, the order constant depends on the size of the grid function $M_1 R_{21}\frac{d^3}{dt^3}u_2(t_{n-1}) = M_1\frac{d^3}{dt^3}u_1(t_{n-1})$. The crucial point is now that in general not all components of this grid function are bounded, thus causing order reduction.

By way of illustration, let M_1 be the standard, tridiagonal finite-difference matrix

$$M_1 = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot & \cdot \end{bmatrix} \tag{6.8}$$

for approximating in 1D the second derivative u_{xx} at equidistant nodal values $x_j = jh,\ j = 1,2, \cdots$ Hence, the first row of M_1 corresponds with $j = 1$ and $j = 0$ corresponds with the boundary point $x = 0$, where a Dirichlet condition has been imposed. Then the first component of $M_1\frac{d^3}{dt^3}u_1(t_{n-1})$

is given by

$$\frac{1}{h^2}(-2u_{ttt}(h,t_{n-1}) + u_{ttt}(2h,t_{n-1})) = u_{tttxx}(h,t_{n-1}) + \frac{1}{h^2}u_{ttt}(0,t_{n-1}) + \mathcal{O}(h^2) \tag{6.9}$$

and it follows that this component is $\mathcal{O}(\frac{1}{h^2})$. Consequently, this component blows up as the mesh width tends to zero. Only if the true solution u satisfies $u_{ttt}(0,t_{n-1}) = 0$, then this component is a consistent approximation like the remaining ones and there will be no blow up. The condition $u_{ttt}(0,t_{n-1}) = 0$ is an unnatural boundary condition, merely introduced by the time-stepping scheme. At the semi-discrete level, and for the general linear system (6.2), this additional boundary condition reads

$$M_k \frac{d^3}{dt^3} u_k(t_{n-1}) = \mathcal{O}(1), \tag{6.10}$$

uniformly in the mesh width. With (6.10) we have $\psi_{1,t}^{(4)} = \mathcal{O}(\tau^4)$, that is, order of consistency 3 uniformly in the mesh width.

This brief illustration explains the essence of the origin of order reduction. Needless to say that the restriction to one space dimension is irrelevant. The simplicity of the example in fact indicates that in applications the threat of "blow up" is always present. On the other hand, the difference between order 2 and 3 is not really dramatic. Noteworthy is that the discussion concerns the local error. It is known, for normal single-grid applications, that the reduction for the global error is usually less due to cancellation effects in the transition from local to global errors. One then often finds that the order to be expected lies between $\tilde{p}+1$ and p, rather than between $\tilde{p}$ and p. Thus, with this cancellation effect, the DIRK scheme does not suffer from global reduction.

Next we put the level index $k = 2$ and consider again the general linear system (6.2). Since also $l_{n-1} = 2$, it suffices to examine the local error of the output stage which is calculated as

$$\psi_{2,t}^{(4)} = \psi_{2,t}^{(3)} = (I_2 - \theta\tau D_2^n M_2)^{-2} b_2 \tau D_2^n M_2 [D_2^n \beta_2^{(2)} + (I_2 - D_2^n) P_{12} \psi_{1,t}^{(2)}] + \tag{6.11}$$

$$(I_2 - \theta\tau D_2^n M_2)^{-1} [D_2^n \beta_2^{(3)} + (I_2 - D_2^n) P_{12} \psi_{1,t}^{(3)}]$$

$$= (I_2 - \theta\tau D_2^n M_2)^{-2} b_2 \tau D_2^n M_2 [D_2^n \beta_2^{(2)} + (I_2 - D_2^n) P_{12} \psi_{1,t}^{(2)}] +$$

$$(I_2 - \theta\tau D_2^n M_2)^{-1} (I_2 - D_2^n) P_{12} \psi_{1,t}^{(3)} + \mathcal{O}(\tau^4).$$

Using boundedness of the operators, the order result (5.39) with stage order $\tilde{p} = 2$ is directly recovered. Inspection of the various terms also reveals the classical order $p = 3$. In connection with $p = 3$, we again examine the possibility of order reduction, particularly so originating from internal boundaries.

Distinguishing local error components outside and inside the integration domain, we can write, after some elementary calculations,

$$(I_2 - D_2^n) \psi_{2,t}^{(4)} = (I_2 - D_2^n) P_{12} \psi_{1,t}^{(4)}, \tag{6.12a}$$

$$D_2^n \psi_{2,t}^{(4)} = (I_2 - \theta\tau D_2^n M_2)^{-2} b_2 \tau D_2^n M_2 [D_2^n \beta_2^{(2)} + (I_2 - D_2^n) P_{12} \psi_{1,t}^{(2)}] + \tag{6.12b}$$

$$[(I_2 - \theta\tau D_2^n M_2)^{-1} - I_2](I_2 - D_2^n) P_{12} \psi_{1,t}^{(4)} + \mathcal{O}(\tau^4).$$

Apart from the interpolation, the outside local error (6.12a) is completely determined by level 1 properties, so that a reduction at level 1 will also be felt at level 2 components outside the integration domain. However, the reduction will also be felt inside the level 2 integration domain, since the inside local error (6.12b) depends on boundary values computed at level 1. An interesting question is now, will the internal boundaries cause order reduction in case the physical ones do not. To examine this question, we henceforth suppose that no reduction will take place at the physical boundary and assume the additional boundary condition (6.10). Then we have $\psi_{1,t}^{(4)} = \mathcal{O}(\tau^4)$, so that

$$\begin{aligned} \psi_{2,t}^{(4)} &= (I_2 - \theta\tau D_2^n M_2)^{-2} b_2 \tau D_2^n M_2 [D_2^n \beta_2^{(2)} + (I_2 - D_2^n) P_{12} \psi_{1,t}^{(2)}] + \mathcal{O}(\tau^4) \\ &= - b_2 \frac{2\theta^3}{3} (I_2 - \theta\tau D_2^n M_2)^{-2} \tau^4 D_2^n M_2 \star \\ &\quad [D_2^n \frac{d^3}{dt^3} u_2(t_{n-1}) + (I_2 - D_2^n) P_{12} \frac{d^3}{dt^3} u_1(t_{n-1})] + \mathcal{O}(\tau^4). \end{aligned} \tag{6.13}$$

Substitution of the interpolation error (see (5.9))

$$\gamma_2(t_{n-1}) = u_2(t_{n-1}) - P_{12}u_1(t_{n-1}) - b_2(t_{n-1}), \tag{6.14}$$

yields

$$\psi_{2,t}^{(4)} = b_2\frac{2\theta^3}{3}(I_2 - \theta\tau D_2^n M_2)^{-2}\tau^4 D_2^n M_2(I_2 - D_2^n)\frac{d^3}{dt^3}\gamma_2(t_{n-1}) + \mathcal{O}(\tau^4). \tag{6.15}$$

We note in passing that (6.10) implies "homogeneity in boundary conditions" in the sense that the 3rd derivative of the boundary term $b_2(t)$ vanishes. It follows from (6.15) that

$$\tilde{\gamma} \equiv D_2^n M_2(I_2 - D_2^n)\frac{d^3}{dt^3}\gamma_2(t_{n-1}) = \mathcal{O}(1), \tag{6.16}$$

uniformly in the mesh width, implies $\psi_{2,t}^{(4)} = \mathcal{O}(\tau^4)$.

Hence, assuming that at the physical boundary no order reduction takes place, an important conclusion is that the internal boundaries do not cause order reduction if the interpolation condition (6.16) holds. Fortunately, in applications this condition is easily satisfied. Sufficient is that

$$\|M_2\| \, \|\frac{d^3}{dt^3}\gamma_2(t_{n-1})\| = \mathcal{O}(1), \tag{6.17}$$

saying that the accuracy order of the interpolation procedure should be greater than or equal to the spatial order of the diffential operator (not to be confused with the order of consistency of the spatial difference operator). For example, for second order in space problems it suffices to use simple linear interpolation. Noteworthy is that while $\tilde{\gamma}$ has zero components outside the integration domain, the argument function $(I_2 - D_2^n)\frac{d^3}{dt^3}\gamma_2(t_{n-1})$ has zero components inside (the boundary of the domain is outside). Further, by definition, at coinciding nodal values the interpolation error component is zero. Consequently, virtually all components of $\tilde{\gamma}$ are zero, except for a few associated to nodal values near the boundary. This specifically depends on the coupling properties of the finite difference operator, but also on the dimension. An extreme situation arises for the one-space dimensional 3-point operator (6.8). Here all components of $\tilde{\gamma}$ are zero, both for simple linear and higher order Lagrangian interpolation. It is easy to figure out that this extreme situation cannot occur in more space dimensions.

To sum up, the above local time error analysis has revealed: if the interpolation condition (6.17) is satisfied, then the multi-level 3-stage DIRK scheme can only suffer from order reduction if there is reduction in the normal single-level application. In other words, the multi-level use itself does not cause reduction. The restriction to the linear problem class (6.2) is not essential. It is likely that the same conclusion can be drawn for nonlinear problems, but only at the expense of much more tedious derivations. Also the restriction to the two-level case is not essential. The same conclusion applies to the multi-level case. Of importance is the interpolation condition (6.17). No doubt the same condition will be encountered when analysing different time integrators. Finally, it is worth to mention that for certain combinations of physical boundary conditions, it will be possible to enforce the additional condition (6.10). This amounts to a tranformation of $u(x,t)$ resulting in homogeneity at the boundary. See [12] for a simple illustration.

6.3. Elaboration of the refinement condition

Of crucial importance for the success of any multi-level method is the performance of the implemented grid adaptation algorithm. We propose the refinement condition (5.27) to underly this algorithm. Satisfying this condition implies that space discretization and interpolation errors are distributed in such a way that when arriving at the finest integration level in use, the influence of the unavoidable interpolations has been eliminated by a judicious choice based on error analysis of the integration domains. However, given an integration method, the general refinement condition (5.27) needs to be simplified for practical use. Two main simplifications can be distinguished:

(i) The first simplification has to do with the augmented form. Working with (5.27) requires computing in $\mathbf{S}_k$ which we consider far too expensive. Consequently, (5.27) must be replaced by an appropriate approximating condition in S_k, preferably connected with the output stage. It is always possible to carry this out, since the refinement condition is concerned with spatial interpolation and discretization errors. Apart from various multiplying bounded operators, these errors are identical over the stages, indicating that it suffices to work with the output stage. However, when required feasible for implementation, the actual replacement is to be method dependent. We will describe this now for the present DIRK method.

Consider (5.22), (5.27). First we replace the variable Jacobian $M_k^{(i)}$ occurring in $\mathbf{Z}_k^n$ by an approximation M_k constant over the stages. M_k is taken to be the Jacobian matrix, or an approximation thereof, computed at the begin of the time step. M_k is available as it is also used in the iterative Newton process for solving the implicit relations. Trivially, for linear problems (6.2) we have the exact relation $M_k^{(i)} = M_k$. Second, the augmented spatial error σ_k^n is approximated as

$$\sigma_k^n = \begin{bmatrix} & 0 & \\ \theta\alpha_k^{(1)} + & \theta\alpha_k^{(2)} & \\ b_1\alpha_k^{(1)} + & b_2\alpha_k^{(2)} & + \ \theta\alpha_k^{(3)} \\ b_1\alpha_k^{(1)} + & b_2\alpha_k^{(2)} & + \ \theta\alpha_k^{(3)} \end{bmatrix} \approx \begin{bmatrix} 0 \\ 2\theta\alpha_k^{(3)} \\ \alpha_k^{(3)} \\ \alpha_k^{(3)} \end{bmatrix} \tag{6.18}$$

Note that we here truncate $\mathcal{O}(\tau)$ terms and that $\alpha_k^{(3)} = \alpha_k^n = \alpha_k(t_n)$. Next, by using (6.5), the components of the spatial error function $\mathbf{w}_k = (\mathbf{Z}_k^n)^{-1}\mathbf{D}_k^n\sigma_k^n$ are approximated by

$$w_k^{(1)} = 0, \tag{6.19}$$

$$w_k^{(2)} \approx 2\theta(I_k - \theta\tau D_k^n M_k)^{-1} D_k^n \alpha_k^n,$$

$$w_k^{(3)} \approx (I_k - \theta\tau D_k^n M_k)^{-1}(2b_2\theta(I_k - \theta\tau D_k^n M_k)^{-1}\tau D_k^n M_k + I_k) D_k^n \alpha_k^n,$$

$$w_k^{(4)} = w_k^{(3)}.$$

As to be expected, at each of the genuine stages we recover a proportionality with the local space discretization error $D_k^n\alpha_k^n$. This justifies to select one particular stage, so as to get rid of working in $\mathbf{S}_k$. The natural choice is the output stage. We choose the approximation

$$w_k^{(4)} \approx (1 - 2b_2)(I_k - \theta\tau D_k^n M_k)^{-1} D_k^n \alpha_k^n, \tag{6.20}$$

which has been invoked to avoid two forward-backward substitutions. Note that we have used the somewhat crude approximation

$$[2b_2\theta(I_k - \theta\tau D_k^n M_k)^{-1}\tau D_k^n M_k + I_k]D_k^n\alpha_k^n \approx (1 - 2b_2)D_k^n\alpha_k^n. \tag{6.21}$$

In first approximation, (6.21) is exact if $D_k^n\alpha_k^n$ is taken to be an eigenvector belonging to the maximal eigenvalue. On the other hand, the operator in (6.21) is bounded, which justifies this approximation step.

We are now ready to replace the regridding condition (5.27) by its counterpart in S_k:

$$\|(I_k - D_k^n)\lambda_k^n\| \leqslant \frac{c}{l - 1}\|(1 - 2b_2)\tau(Z_l^n)^{-1}D_l^n\alpha_l^n\|, \qquad k = 2, \cdots, l, \tag{6.22}$$

where $l = l_{n-1}$,

$$\lambda_k^n = \gamma_k^n + (1 - 2b_2)\tau P_{k-1k}(Z_{k-1}^n)^{-1}D_{k-1}^n\alpha_{k-1}^n \tag{6.23}$$

and

$$Z_k^n = I_k - \theta\tau D_k^n M_k. \tag{6.24}$$

Observe that $\|(I_k - D_k^n)\lambda_k^n\| = \|(\mathbf{I}_k - \mathbf{D}_k^n)\boldsymbol{\lambda}_k^n\| + \mathcal{O}(\tau)$. The choice $l - 1$ for the constant $\tilde{C}$ is exact in case of linear interpolation, provided $C \leqslant 1$ (see (5.26), (5.2)). We will use $\tilde{C} = l - 1$ also in other situations. Apart from the constant $1 - 2b_2$, condition (6.22) is completely identical to the regridding condition found for the implicit Euler method in [15] (formula (6.9)).

(ii) The second simplification has to do with the nesting property and restricted interpolation. Once at level $k-1$ the integration is completed, (6.22) is used to select the integration domain for level k (the actual definition of D_k^n). This selection process is carried out by the so-called flagging procedure. This procedure scans level k points and flags those points for which the refinement condition is violated to be placed within the new integration domain. Our mathematical framework prescribes that the scan be carried out on the whole of ω_k, as the interpolation error γ_k^n is defined on the whole of ω_k. This, of course, is very time consuming because it requires interpolation over the whole of ω_k. We therefore apply restricted interpolation. Restricted interpolation means that the interpolation is restricted to level k points lying within the $(k-1)^{\text{th}}$ integration domain. Subsequently, the scan is also restricted to the $(k-1)^{\text{th}}$ integration domain. Actually, this way the nesting of the integration domains is enforced. In [15] it is shown that restricted interpolation leads to nearly the same integration domains as found with full interpolation. In other words, full interpolation is redundant. Finally, the flagging procedure contains some safety measures (buffering) which enhances the reliability of the restricted interpolation (see [15]). Of course, this procedure also implements numerical estimators for γ_k^n, $(Z_{k-1}^n)^{-1} D_{k-1}^n \alpha_{k-1}^n$ and $\|(Z_l^n)^{-1} D_l^n \alpha_l^n\|$. To save space, we again refer to [15]. The choice for the various estimators is discussed there at length.

To sum up, we have described how the original refinement condition (5.27) can be implemented for practical use. We have concluded that it is justified to base the actual implementation on the feasible condition (6.22), completely similar as for the implicit Euler method. Needless to say that in actual application the various simplifications and approximations can result in deviations. However, these deviations will remain small, so that the spatial accuracy obtained on the finest grid will remain close to the spatial accuracy on this grid when applied uniformly over the entire physical domain. The numerical illustrations of the next section serve to support this claim.

7. Numerical examples

The purpose of this section is to illustrate numerically the outcome of imposing the simplified refinement condition (6.22), derived from the general condition (5.27) for the particular DIRK method (6.1). Recall that, in theory, this general condition guarantees local space errors at most equal to the maximum of the local space error of the finest grid in use when applied without adaptation, except for the constant c . This constant, however, is independent of the grid spacing. Hence, assuming stability, our theory dictates that the usual convergence behaviour of the discretization method applied without adaptation is to be maintained when this discretization method is combined with our LUGR technique. We claim that this theoretical result can be put into practice in a straightforward way. We also claim that the simplifications mentioned in Section 6 do not adversely affect the convergence behaviour.

Two numerical examples are presented, both being two-space dimensional. The first example serves to illustrate the above mentioned claims on convergence. This example problem is solved using the 'fixed-level mode of operation'. The second example serves to illustrate the performance of the method when applied in the 'variable-level mode of operation'. This mode of operation is very useful if the solution shape strongly changes in time, e.g., when steep layers or wave fronts emerge at later times and at earlier times large gradients are absent. In such situations an important consideration is that new levels are created in time in order to keep the accuracy at approximately the same level. On the other hand, these new levels should not be created when they are not yet needed, for reasons of efficiency.

7.1. Example problem I.

We have borrowed this example problem from Adjerid & Flaherty [1]. The equation is linear and parabolic and given by

$$u_t = u_{xx} + u_{yy} - u_x - u_y + f(x,y,t), \qquad 0 < x,y < 1, \qquad t > 0. \tag{7.1}$$

The initial condition at $t = 0$, the Dirichlet boundary conditions for $t > 0$, and the forcing function f are such that the exact solution is known and given by

$$u(x,y,t) = 1 - \tanh(25(x - t) + 5(y - 1)). \tag{7.2}$$

This solution is a skew wave propagating through the domain from left to right. The wave starts near the left boundary and approaches the right boundary at approximately $t = 0.8$. We integrate over the time interval [0,0.6]. We emphasize that the choice of the PDE operator is of minor importance here. For convenience of testing, we therefore selected a scalar, linear problem. What counts is the specific solution form. The present solution is suitable to subdue the LUGR method to a convergence test.

The spatial discretization is based on second order, symmetric differences. Simple linear interpolation is used and the constant c, introduced in the refinement condition, is put equal to one. Four computations were performed using, respectively, 1,2,3 and 4 levels. The mesh width in both x- and y

direction of the base grid is 0.05. Note that the number of levels is fixed during a computation over the time interval [0,0.6]. We also fix the stepsize τ during such a computation. However, since per computation the spatial mesh width is halved within the integration domains, we simultaneously halve τ. Because the stage order of the DIRK method is 2, like the order of the spatial discretization, per computation a gain factor of approximately 4 then should be found for the total global errors. To compare the accuracy of the LUGR computations with the accuracy obtained on a single uniform grid, we have also solved the problem in the standard way using the same values for τ and the mesh width of the finest level. It is emphasized that the values for τ and the mesh width in space are always such that the space error dominates. For illustration purposes this is necessary since otherwise no valid conclusion can be drawn on the performance of the spatial refinement condition.

τ	no. of	single	t	
	levels	grid	0.3	0.6
0.1	1	20x20	0.17319	0.17401
0.05	2		0.02728	0.02815
		40x40	0.02789	0.02810
0.025	3		0.00624	0.00716
		80x80	0.00680	0.00684
0.0125	4		0.00177	0.00174
		160x160	0.00168	0.00169

TABLE 7.1. Example problem I. Maxima of global errors computed at the finest available level. Comparison with the accuracy obtained on a single uniform grid.

The results of the computations are contained in Table 7.1. We see that the LUGR solutions converge according to the theory and, also, that these solutions are as accurate as the standard, uniform grid solutions. This correspondence in accuracy is striking. Together with the choice of the PDE operator (recall the constant $\frac{c}{\tilde{C}}$), the simplifications mentioned in Section 6 play a role here, but also the fact that in the actual flagging procedure some safety measures have been built, like buffering. Buffering of course helps in keeping the LUGR accuracy close to the standard accuracy. We can conclude that the present experiment is very successful in that the implemented simplified refinement condition (6.22) works as anticipated. Finally, Figure 7.1 shows the grids of the 2,3 and 4-level computations at two different times. Note that the grids nicely align with the wave front.

7.2. Example problem II.

The equation is again linear and parabolic,

$$u_t = u_{xx} + u_{yy} + f(x,y,t), \qquad 0 < x,y < 1, \qquad t > 0. \tag{7.3}$$

The initial condition at $t = 0$, the Dirichlet boundary conditions for $t > 0$, and the forcing function f are adapted to the exact solution

$$u(x,y,t) = 1 - \tanh(100[(x - 0.5)^2 + (y - 0.5)^2 - t + 0.025]). \tag{7.4}$$

This solution rapidly varies its shape as time elapses and has been constructed with the aim of illustrating the 'variable-level mode of operation'. At $t = 0$ the solution is almost zero over the entire domain. As time elapses it steepens up at the center point [0.5,0.5], developing a circular wave front. This circular wave front starts to propagate towards the boundaries when $u(0.5,0.5,t) \approx 2$ and during the propagation the front becomes steeper and steeper. When the front has passed a point (x,y), the solution $u(x,y,t)$ approximates the value 2. We solve the problem over the time interval [0,0.1], which is sufficiently large to see all phenomena happen.

The refinement condition (5.27) and its simplified version (6.22) tell us where to integrate on a finer level. When using the 'fixed-level mode of operation', and supposing that the number of levels is fixed a priori, this suffices. When using the 'variable-level mode of operation', we also need a criterion to decide when to change the current number of levels. A natural thing to do is to associate this criterion to the spatial local error value. In the present experiment we employ the spatial local error expression as used in (6.22). Note this involves no extra costs. Specifically, within each base time step we monitor the number of grid levels with the criterion

$$(1+c)(1-2b_2)\tau\|(Z_k^n)^{-1}D_k^n\alpha_k^n\| < \tau\mathrm{TOL}, \tag{7.5}$$

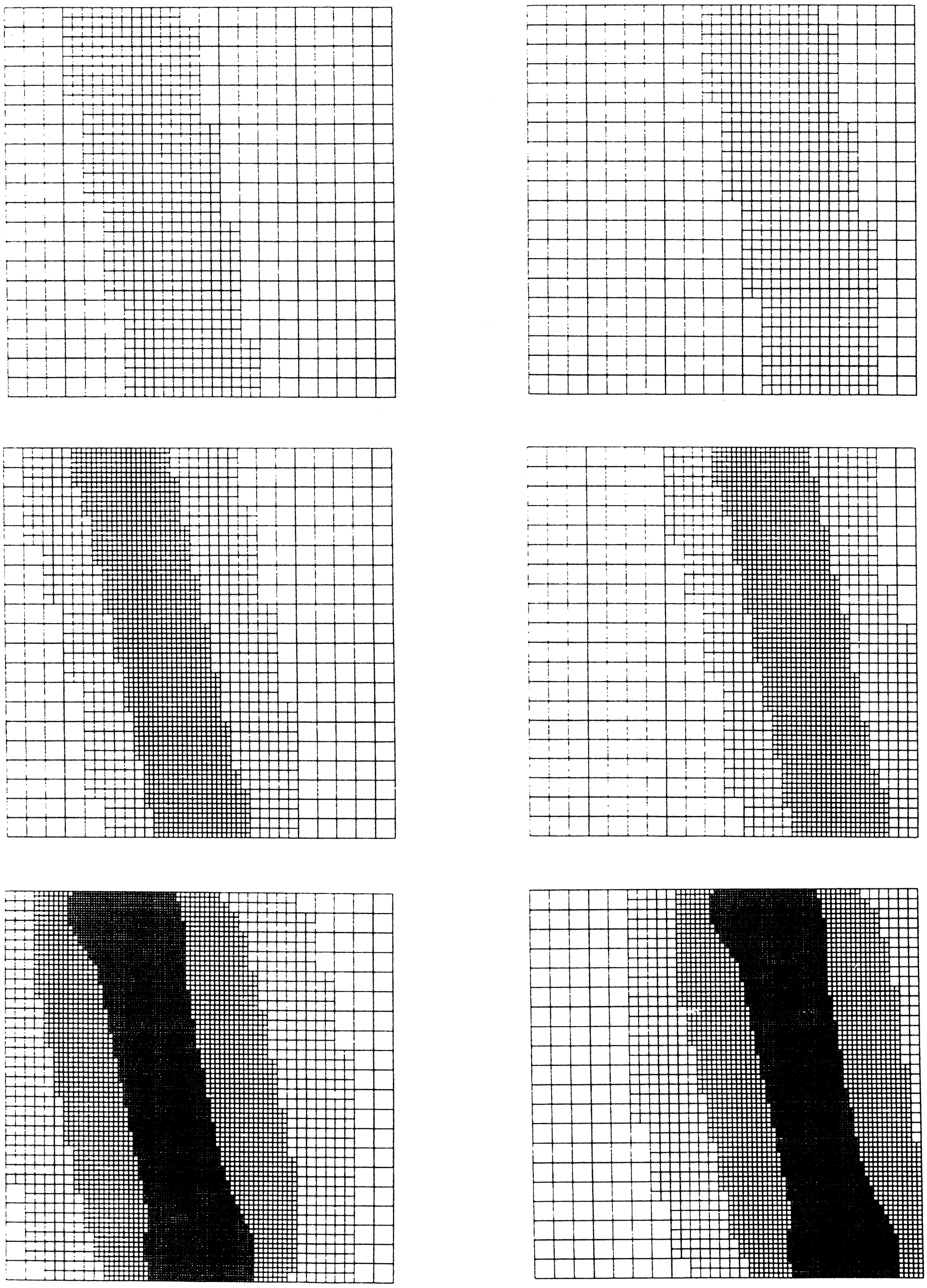


FIGURE 7.1. Example Problem I. Grids of the 2,3 and 4-level computations at $t = 0.3$ and $t = 0.6$.

where TOL represents a monitor tolerance value to be specified and, in the application, $(Z_k^n)^{-1}D_k^n\alpha_k^n$ is replaced by an asymptotically correct estimator. Starting with $k = 1$, this inequality is checked after each level integration. If it is violated, then k is increased by 1. Otherwise it is decided that enough levels have been introduced and l_{n-1} is assigned the current value for k. Recall that l_{n-1} denotes the number of levels in use during the base step from t_{n-1} to t_n. Hence, the idea is to select l_{n-1} in such a way that the local error expression in (7.5) is kept close to τTOL.

In most applications we will come across a few so-called full interpolations, viz when $l_n = l_{n-1} + 1$ (see Section 5). In that case the error bound (5.28) applies (also in a simplified version). This error bound takes into account the full interpolation error which is committed at the old time value. This eventual interpolation error is neglected in (7.5). We justify this heuristic decision with the observation that full interpolation can take place only in a few number of steps. However, to remain on the safe side, we now use 4th order Lagrangian interpolation instead of 2nd order linear. It is obvious that full interpolation should not diminish the quality of the approximations, since otherwise the estimation of the discretization and interpolation errors used by the refinement condition is hindered. The full interpolation should also not interfere with the estimation of the number of levels needed in the step to follow. Therefore, the additional errors stemming from full interpolation have to be restricted in some manner. In the present experiment 4th order interpolation has turned out to work satisfactorily.

We are now ready to discuss the actual experiment carried out with problem (7.3) - (7.4). This experiment concerns one run over the time interval [0,0.1]. The constant c of the refinement condition is again put equal to 1. The stepsize $\tau = 0.001$ and is kept constant. The value of 0.001 is sufficiently small to guarantee that spatial effects dominate. The mesh width in both x- and y direction of the base grid is 0.05 and the tolerance parameter TOL $=$ 50. Results are collected in Tables 7.2-7.3 and Figure 7.2.

For a subset of time points, including those where a new grid level is added, Table 7.2 shows the course of the number of grid levels and the maximum of the global error measured at the finest

t	no. of levels	global error
0.01 0.017	1	0.01074 0.03171
0.018 0.02 0.03 0.039	2	0.01222 0.01117 0.01612 0.02523
0.04 0.05 0.06 0.07 0.072	3	0.01392 0.01493 0.01668 0.02168 0.02136
0.073 0.08 0.09 0.1	4	0.01289 0.01191 0.00722 0.00713

TABLE 7.2. Example problem II. Maxima of global errors computed at the finest grid at various time points, including those where a new grid is introduced.

available grid. An important conclusion is that while the circular wave front develops, the algorithm keeps the error at a fairly constant level. This anticipated behaviour is in line with the idea behind the error monitor (7.5).

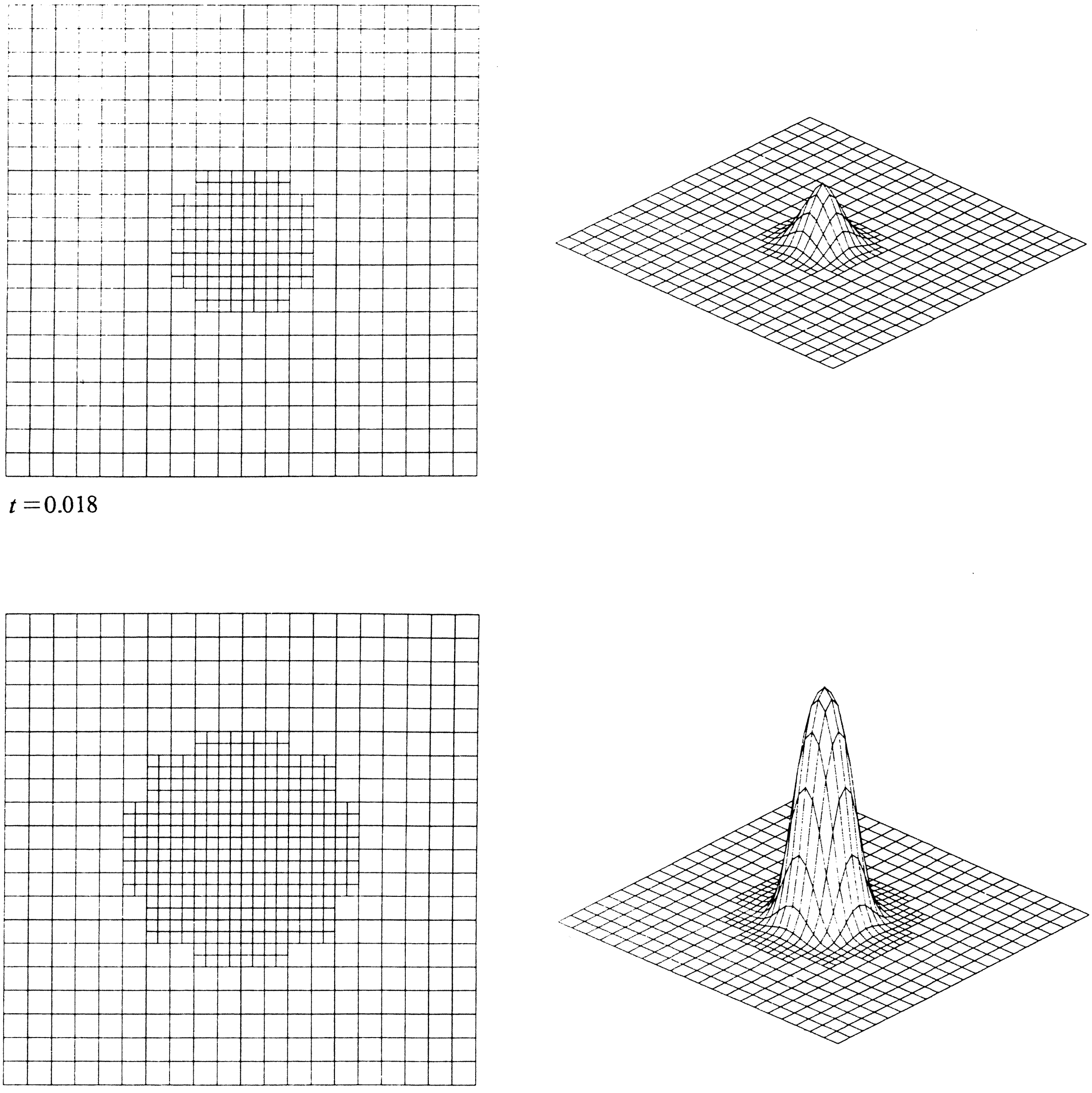
A collection of pictures, showing the evolution of the local uniform grids and the computed solution of (7.3) is contained in Figure 7.2. These pictures illustrate that the shape of the grids is accurately adapted to the circular wave front form (symmetry). This again shows that the simplified refinement condition, which tells us where to refine, works as anticipated. On the other hand, the number of levels needed is not always computed optimally. This happens, e.g., at $t = 0.04$ and $t = 0.073$, time points where a new grid level is used for the first time. The grid pictures show that at these time points the new fine grid almost completely overlaps the existing one, indicating that the new fine grid is introduced too late (the solution steepens up). Fortunately, Table 7.2 shows that this small deficiency does not diminish the accuracy for evolving time. Also note that at later points of time this phenomenon disappears, see $t = 0.05$ and $t = 0.1$. This is of course what should happen

t	level	approx.	exact
0.017	1	0.04616	0.05307
0.018	1	0.05165	0.05793
	2	0.01246	0.01468
0.039	1	0.12511	0.28075
	2	0.04678	0.05579
0.04	1	0.13972	0.33177
	2	0.05084	0.06329
	3	0.01495	0.01467
0.072	1	0.31630	1.48171
	2	0.18625	0.30880
	3	0.04685	0.05130
0.073	1	0.34144	1.39739
	2	0.18078	0.29537
	3	0.05088	0.05367
	4	0.01683	0.01382

TABLE 7.3. Example problem II. Exact values and numerical estimates of the spatial local error expression (7.5). Note that the stepsize $\tau = 0.001$ is contained in these values.

due to the ever increasing solution gradients.

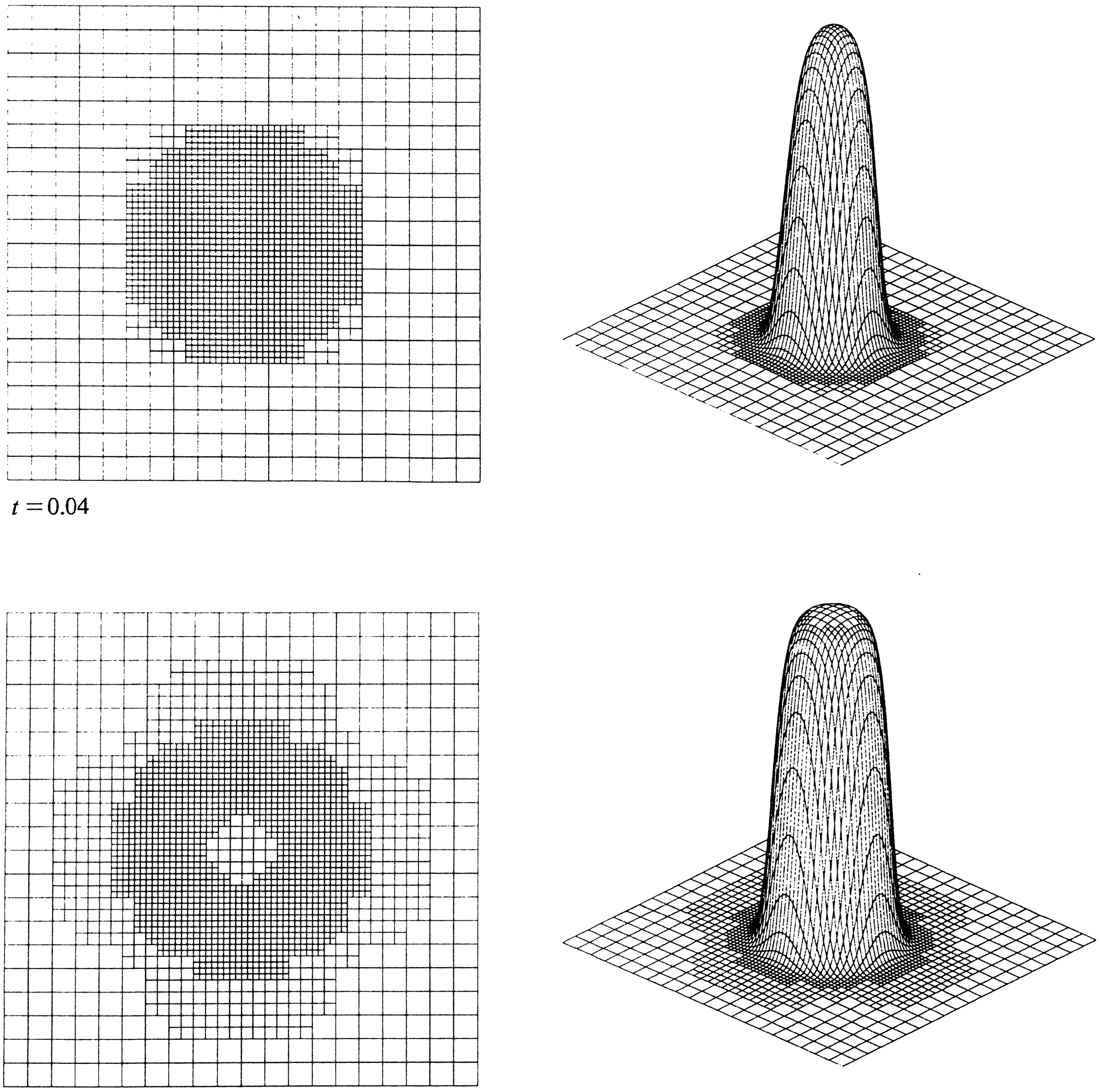
The precise origin of this small deficiency is not clear. The error introduced by the full interpolation can play a role here (this error is not monitored by (7.5)). More likely is, however, that it emanates from the lack of asymptotics at the coarser grids. This lack of asymptotics is inherent to any monitoring process that starts on coarse grids and therefore very difficult to overcome, if possible at all. To provide some insight in the asymptotics for the estimator of (7.5), we have added Table 7.3. This table shows the exact, analytical values for (7.5) with their estimated numerical values at time points just before and after the introduction of a new grid level. First, we see that at corresponding levels before and after the listed time points the numerical estimations are in fairly good agreement with one another, even on the coarse base grid. This supports the conclusion that the full interpolation is sufficiently accurate for not interfering the selection of number of levels. Second, there is excellent agreement between the exact and numerical values on the fine grids. However, particularly at later times, the coarse grid values are not in good agreement with one another. This means that we are outside the asymptotic regime and this is likely to cause some disturbances in the selection of the right



t =0.018

t =0.03

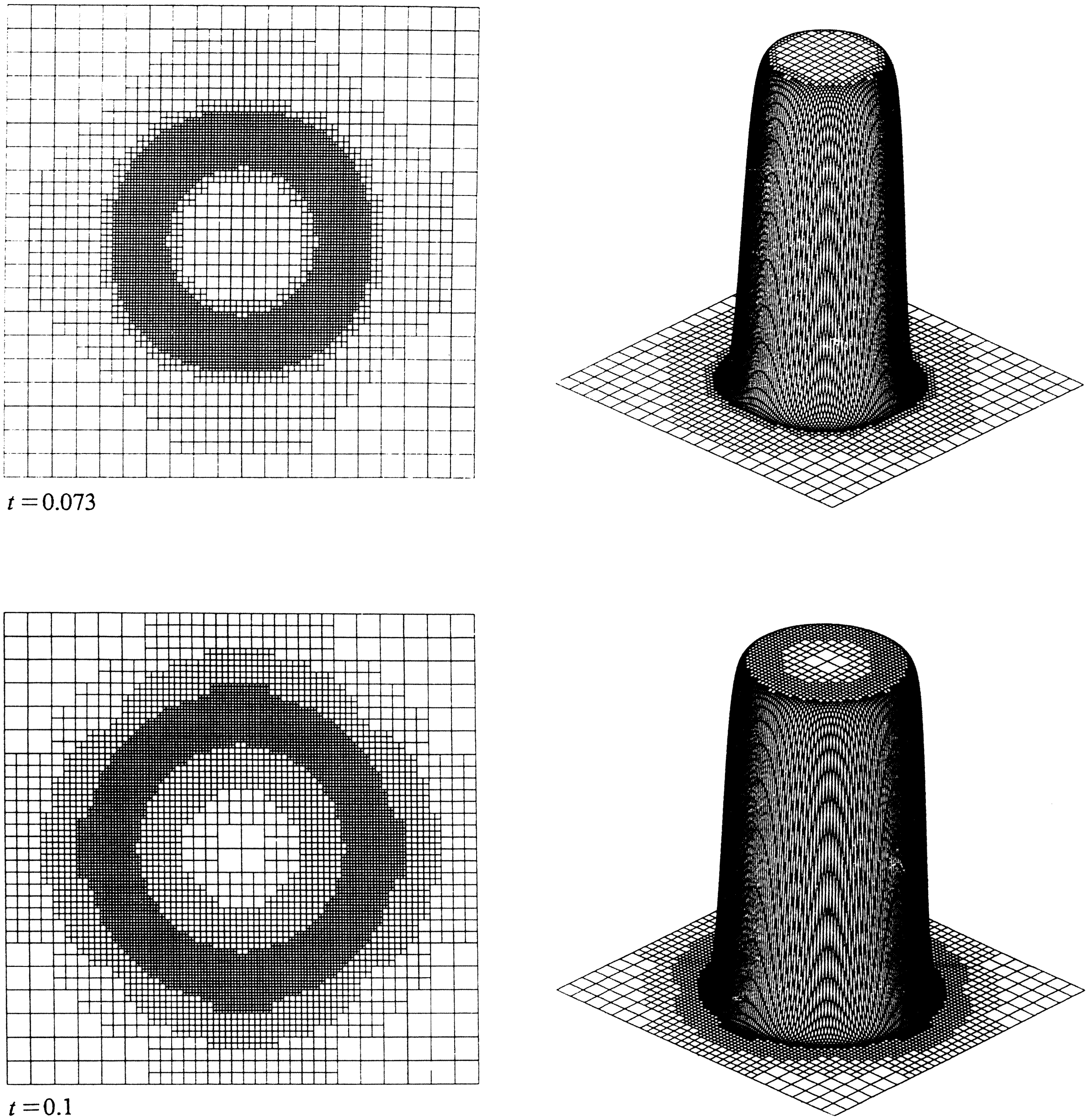
FIGURE 7.2. Example Problem II. The course of the local uniform grids and the computed solution of (7.3).



t = 0.04

t = 0.05

FIGURE 7.2. continued.



$t = 0.073$

$t = 0.1$

FIGURE 7.2. continued.

number of levels. We wish to emphasize once more that in spite of this lack of asymptotics, the overall behaviour of the algorithm is very satisfactorily.

Let us conclude this section with a remark on the choice of the monitor parameter TOL, in connection with the discrepancy between the current value of TOL = 50 and the global accuracy shown in Table 7.2. A discrepancy like this is unavoidable, due to damping of global errors. Note that we have a parabolic problem and that the DIRK method mimics the damping property of the parabolic operator. Among others, the method is strongly A-stable. Part of the discrepancy may also originate from cancellation between various temporal and spatial terms. This damping of global errors, and this eventual cancellation, has not been taken into account in our error analysis which focusses on local errors, in particular on local error bounds. For precise estimation purposes our analysis is simply too general. On the other hand, the present example once more shows that local error bounds like (7.5) can be much too conservative (the simplified form is not essential for the present discussion). Altogether, this means that in application local error expressions like (7.5) are better be interpreted as error monitors than as precise local error estimators. In connection with grid selection purposes, our practical experience is that with this interpretation the (simplified) spatial local error expression is reliable and works very satisfactorily.

8. Summary and future research

The paper can be divided in two main parts. The first part is constituted by Sections 2 - 5. Following our earlier work [15] on the implicit Euler method, here we set up the mathematical framework for the general Runge-Kutta LUGR method. In this part the examination is mainly directed to the question of how to maintain the spatial accuracy when a general Runge-Kutta method is combined with an LUGR technique. The result of this examination is the refinement condition (5.27). Some attention is paid to the question of temporal accuracy, primarily in connection with the order reduction phenomenon. The remaining part of Sections 6 - 7 serves to illustrate, both analytically and numerically, how to elaborate the general results of the first part for a particular integration scheme.

For this purpose we have selected the DIRK method (6.1), but other choices of methods are conceivable of course (see also [14]).

An important subject for future research is that of efficiency of the time-stepping scheme itself when combined with the LUGR technique. Two important issues not addressed in this work concern the use of variable time steps and the solution of the arising systems of linear and nonlinear algebraic equations, in case of an implicit scheme. Straightforward use of variable time step algorithms, as successfully applied in single-grid method of lines computations, renders problems due to the fact that approximations obtained with an LUGR method are always difficult to numerically differentiate in time. The reason for this is that part of the components is obtained from a numerical integration, part from interpolation or injection. The resulting 'non-smoothness' is then felt when computing higher temporal derivatives. More precisely, the higher temporal derivatives are estimated in a rough way, resulting in disturbances in the stepsize selection (see also [14]). To our experience, smoothing or filtering procedures provide only a partial remedy here.

Concerning the second issue, by nature of the LUGR approach approximations are computed in varying dimensions, even within one base time step. For DIRK or alternative implicit methods this obviously implies that the numerical algebra effort, to be made in solving systems of algebraic equations, becomes highly important. In the numerical experiments reported here, we have paid no attention to the efficiency of the numerical algebra computations and simply used an available sparse matrix technique (same as in [15]). This technique, however, is known to yield a considerable overhead when used in the solution of time-dependent problems. It is most likely that sophisticated iterative solution procedures will be much more effective. An important consideration hereby is that always accurate initial guesses are available.

References

1. S. Adjerid and J.E. Flaherty (1988). A Local Refinement Finite Element Method for Two Dimensional Parabolic Systems, *SIAM J. Sci. Stat Comput.*, 9, 792-881.

2. R. ALT (1973). *Thèse de Troisième Cycle*, Université de Paris 6.

3. D.C. ARNEY and J.E. FLAHERTY (1989). An Adaptive Local Mesh Refinement Method for Time-Dependent Partial Differential Equations, *Appl. Numer. Math.*, 5, 257-274.

4. M.J. BERGER and J. OLIGER (1984). Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations, *J. Comput. Phys.*, 53, 484-512.

5. M. CROUZEIX and P.A. RAVIART (1980). *Approximation des Problèmes d'Evolution. Première Partie: Etude des Méthodes Linéaires a Pas Multiples et des Méthodes de Runge-Kutta,* Unpublished Lecture Notes, Université de Rennes, France.

6. K. DEKKER and J.G. VERWER (1984). *Stability of Runge-Kutta Methods for Stiff Nonlinear Differential Equations*, North-Holland, Amsterdam-New York-Oxford.

7. W.D. GROPP (1987). Local Uniform Mesh Refinement on Vector and Parallel Processors, in *Large Scale Scientific Computing*, 349-367, ed. P. DEUFLHARD, B. ENGQUIST, Birkhäuser Series Progress in Scientific Computing.

8. W.D. GROPP (1987). Local Uniform Mesh Refinement with Moving Grids, *SIAM J. Sci. Stat. Comput.*, 8, 292-304.

9. J.F.B.M. KRAAIJEVANGER (1989). *Contractivity of Runge-Kutta Methods,* Report TW-89-12, Dept. of Math. and Comp. Sc., Leiden University.

10. M.A. REVILLA (1986). Simple Time and Space Adaptation in One-Dimensional Evolutionary Partial Differential Equations, *Int. J. Numer. Meth. Eng.*, 23, 2263-2275.

11. J.M SANZ-SERNA and J.G. VERWER (1989). Stability and Convergence at the PDE/Stiff ODE Interface, *Appl. Numer. Math.*, 5, 117-132.

12. J.M. SANZ-SERNA, J.G. VERWER, and W.H. HUNDSDORFER (1987). Convergence and Order Reduction of Runge-Kutta Schemes Applied to Evolutionary Problems in Partial Differential Equations, *Numer. Math.*, 50, 405-418.

13. M.N. SPIJKER (1985). Stepsize Restrictions for Stability of One-Step Methods in the Numerical Solution of Initial Value Problems, *Math. Comp.*, 45, 377-392.

14. R.A. Trompert and J.G. Verwer (1989). *A Static-Regridding Method for Two Dimensional Parabolic Partial Differential Equations,* Report NM-R8923, Centre for Mathematics and Computer Science, Amsterdam (to appear in Appl. Numer. Math.).

15. R.A. Trompert and J.G. Verwer (1990). *Analysis of the Implicit Euler Local Uniform Grid Refinement Method,* Report NM-R9011, Centre for Mathematics and Computer Science, Amsterdam.