1991

J.G. Verwer, R.A. Trompert

Local uniform grid refinement for time-dependent partial differential equations

Department of Numerical Mathematics Report NM-R9105 February

CWI, nationaal instituut voor onderzoek op het gebied van wiskunde en informatica

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Research (N.W.O.).

Local Uniform Grid Refinement for Time-Dependent Partial Differential Equations

J.G. Verwer and R.A. Trompert
CWI, Dept. of Numerical Mathematics
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 only at discrete values of time. We present a local refinement theory underlying the criterion that the multi-level LUGR scheme maintains the spatial accuracy of the finest grid in use when applied in the single-fixed-grid mode. This way we provide automatic control on the inevitable accumulation of interpolation errors for evolving time. Following the method of lines approach, we first develop this theory for the implicit Euler LUGR method applied to a general nonlinear PDE problem. We next show how results can be extended to general Runge-Kutta and linear multistep LUGR methods. The local refinement theory is numerically illustrated for two parabolic model problems.

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, adaptive-grid methods, error analysis, method of lines.

Note: This report was prepared for presentation at the 14th Biennial Conference on Numerical Analysis, 25th - 28th June 1991, Dundee, Scotland and will be submitted to the proceedings.

Report NM-R9105
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

1. INTRODUCTION

Standard numerical methods integrate on a fixed spatial grid, a priori chosen for the whole temporal integration interval. For solutions exhibiting strong local phenomena, like steep moving fronts, emerging layers, etc., this may necessitate a very fine grid covering the entire spatial domain for all values of time. Needless to say that this then may involve unacceptably high computational costs, notably for multi-space dimensional problems. In such cases adaptive-grid methods may provide a remedy. An adaptive-grid method refines the space grid only locally, thus striving for a substantial reduction in number of grid points and computer costs.

Two main categories of adaptive methods are distinguished, viz dynamic and static methods. Dynamic (in time) methods adapt the grid in a continuous-time manner, like classical Lagrangian methods. As examples we mention the moving-finite-element method of Miller [15] and the moving-finite-difference method discussed in Verwer et al. [19]. These methods usually not only provide a local spatial refinement, but also allow larger time steps due to their Lagrangian nature. As a rule, the dynamic approach is feasible for 1D problems, but much less so in 2D and 3D. The second main category of methods, the static (in time) methods, adapt the grid only at discrete times. Static methods are in principle well feasible in any space dimension, but they do not soften temporal solution behaviour like in the dynamic approach, since the actual time stepping is carried out on a non-moving grid.

We are interested in the development of adaptive-grid techniques for time-dependent PDE problems which are suitable for general implementation and are, as much as possible, as feasible as fixed-single-grid methods. The present paper is devoted to a static technique called local uniform grid refinement (LUGR). Our main purpose here is to discuss this technique and to show how it can be combined with standard integration methods of the linear multistep and Runge-Kutta type. Following the method-of-lines approach, we start our discussion with the implicit Euler method. Later in the paper we show how the results obtained on the implicit Euler LUGR method can be extended to general Runge-Kutta and linear multistep methods. Throughout the paper the discussion focusses on the local refinement analysis problem.

2. THE LUGR TECHNIQUE

Although its mathematical elaboration is complicated, in essence the LUGR technique is simple and straightforward. Therefore we begin with an outline of the technique and the grid structure involved (cf. [17, 18] and [16]). Let $\Omega \times [0,T]$ be the space-time domain (in any dimension) with boundary $\partial\Omega$ parallel to the co-ordinate axes. Let ω_1 be a coarse, uniform grid covering Ω. This grid is called the base grid. Starting at ω_1, one base time step with the implicit Euler LUGR method consists of repeated integrations on nested, finer-and-finer local subgrids, possessing internal nonphysical boundaries (grid interfaces). Each of the single integrations spans the same step interval. Hence on each subgrid a new initial-boundary value problem is solved, over the base time step interval in use. Required initial values are defined by interpolation from the next coarser subgrid or taken from a possibly existing subgrid from the previous time step interval. Boundary values required at internal boundaries are also interpolated from the next coarser subgrid. Internal boundaries are treated as Dirichlet boundaries. The generation of the nested subgrids is continued up to a level considered fine enough for resolving the fine scale structure at hand. Having

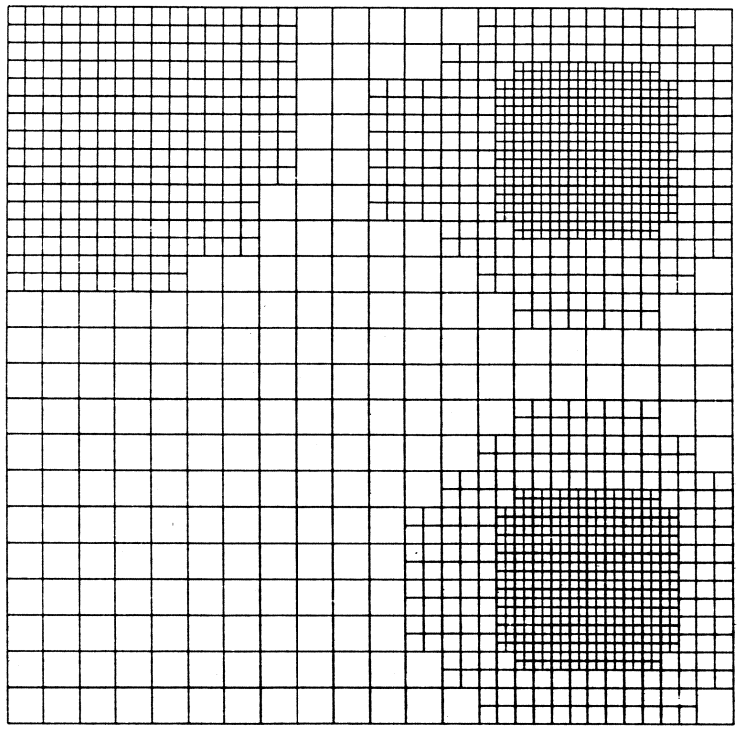

Figure 1. Example of a 2D composite grid for a particular point of time.

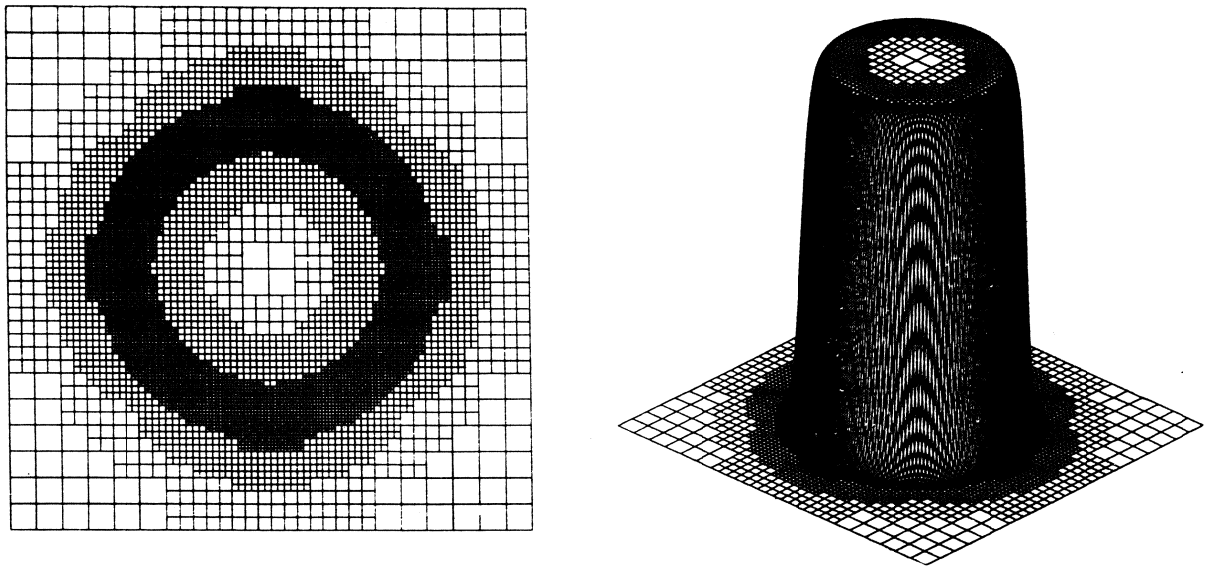

Figure 2. Circular wave front computed on a circular 4-level grid.

completed the integration on the finest level, the process is repeated for the next base time step interval until the physical end time t = T is reached.

Thus, for each base time step the computation starts at the base grid, while using the most accurate solution available, since fine grid solution values are always injected in coinciding coarse grid points.

Moreover, grid points already living at a certain level of refinement are used for step continuation (all fine subgrids are kept in storage). In conclusion, each base step consists of the following operations:

1. *Integrate on coarse base grid.*
2. *Determine new fine subgrid at forward time.*
3. *Interpolate internal boundary values at forward time.*
4. *Interpolate new internal boundary values at backward time.*
5. *Interpolate new initial values at backward time.*
6. *Integrate on subgrid, using the same steplength.*
7. *Inject fine grid values at coinciding coarse grid points.*
8. *Goto 2 until the desired number of refinement levels is reached.*

Figure 1 shows an example of a composite LUGR grid in 2D for a certain point of time. Note that we allow disjunct subgrids and that these need not be a rectangle. Further one sees that the actual refinement is cellular and carried out by bisection of sides of coarse grid cells. An important point to notice is that the repeated use of all varying subgrids, from coarse to fine, is essential in our method. This way we generate the required boundary conditions at the internal boundaries and keep the subgrids uniform. This approach is necessarily a bit wasteful in situations where sharp transitions move very slowly, e.g. when approaching steady state. On the other hand, the workload on the coarser grids will normally be small and we consider the use of uniform grids attractive. Uniform grids allow an efficient use of vector based algorithms and finite-difference approximations on uniform grids are more accurate and faster to compute than on nonuniform grids. In this respect the current approach is to be contrasted with pointwise refinement leading to truly nonuniform grids. Pointwise refinement techniques also require a more complex and expensive data structure.

Previous work on LUGR methods has been published by Berger and Oliger [4], Gropp [10,11], and others. In [4], and also in Arney and Flaherty [3], LUGR methods are examined which are based on noncellular refinement and truly rectangular local subgrids which may rotate and overlap to align with an evolving fine scale structure. We avoid these difficulties. Our local subgrids may not overlap and internal boundaries are always parallel to co-ordinate axes. On the other hand, our subgrids need not be a rectangle. Figure 2 shows how a circular wave front is handled with our grid structure.

3. MATHEMATICAL FORMULATION

We will now give a formulation of the implicit Euler LUGR method which enables us to set up a general analysis of the interplay between spatial discretization and interpolation errors.

3.1. The semi-discrete problem.

Consider a well-posed real abstract Cauchy problem

$$(3.1) \qquad u_t = L(\underline{x},t,u), \quad (\underline{x},t) \in \Omega \times (0,T], \qquad u(\underline{x},t,0) = u_0(\underline{x}),$$

where L is a d-space dimensional PDE operator, of at most second order. L is supposed to be provided with appropriate boundary conditions for $t>0$ such that the true solution $u(\underline{x},t)$ uniquely exists and is as often differentiable on $(\Omega \cup \partial\Omega) \times [0,T]$ as the numerical analysis requires. Recall, however, that we aim at non-smooth solutions, like steep travelling fronts. Thus, non-smoothness means here the occurrence of rapid transitions in the space-time domain, with a sufficient degree of differentiability. The function $u(\underline{x},t)$ may be vector-valued and the number of space dimensions may be arbitrary.

LUGR methods use local subgrids of varying size in time and thus generate approximation vectors of a varying dimension. This varying dimension complicates the error analysis. For the sake of analysis, we

circumvent this problem by expanding the fine local subgrids over the entire domain Ω. Integration then takes place on part of the expanded grid and interpolation on its complement. We stipulate that this grid expansion does not take place in the application of the method, merely in the formulation.

Let ω_k, $1 \le k \le r$, be uniform space grids, with the uniformity meant directionwise. The integer r is the number of refinement levels. Each grid covers the whole of Ω (grid expansion) and has no points on $\partial\Omega$. Given the base grid ω_1, ω_2 is constructed by bisecting all sides of all cells of ω_1, etc., so that the grids are naturally nested. Because $\partial\Omega$ is locally parallel to the co-ordinate axes, it is always possible to construct such a set of grids. To the PDE problem (3.1) we now associate on each ω_k a real Cauchy problem for an explicit ODE system

$$(3.2) \qquad \frac{d}{dt} U_k(t) = F_k(t,U_k(t)), \qquad 0 < t \le T, \quad U_k(0) = U_k^0,$$

obtained by an appropriate spatial discretization of (3.1) on ω_k. Hence U_k and F_k are vectors representing grid functions on ω_k. It is assumed that the boundary conditions have been worked into (3.2) by elimination of variables on $\partial\Omega$.

Let d_k be the length of U_k. In the remainder we let S_k with $\dim(S_k) = d_k$ denote the vector space of all grid functions on ω_k. Specific elements of S_k are U_k, F_k and u_k, where $u_k = u_k(t)$ represents the natural restriction of the true PDE solution $u(\underline{x},t)$ to ω_k. In the space S_k the initial value problems (3.1), (3.2) are related by the local space (discretization) error

$$(3.3) \qquad \alpha_k(t) = \frac{d}{dt} u_k(t) - F_k(t,u_k(t)), \qquad 0 \le t \le T.$$

In particular, the continuous time grid functions u_k and α_k are supposed to be sufficiently often differentiable in t, while α_k has the order of consistency of the spatial finite-difference formulas employed.

3.2. The implicit Euler LUGR formula.

The base time step from t_{n-1} to t_n is formulated by

$$(3.4a) \qquad U_1^n = R_{r1} U_r^{n-1} + \tau F_1(t_n,U_1^n),$$

$$(3.4b) \qquad U_k^n = D_k^n [R_{rk} U_r^{n-1} + \tau F_k(t_n,U_k^n)] + (I_k - D_k^n) [P_{k-1k} U_{k-1}^n + b_k^n], \qquad k = 2, \dots, r,$$

where τ is the stepsize in time and

$U_k^n \in S_k$ is the approximation to $u_k(t_n)$ at ω_k,

I_k: $S_k \to S_k$ is the unit matrix,

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

R_{rk}: $S_r \to S_k$ is the natural restriction operator from ω_r to ω_k, $R_{rr} = I_r$,

P_{k-1k}: $S_{k-1} \to S_k$ is an interpolation operator from ω_{k-1} to ω_k,

$b_k^n \in S_k$ contains time-dependent terms emanating from the boundary $\partial\Omega$.

Inspection of (3.4b) reveals that the diagonal matrices D_k^n are used to define the integration domains, which are just the local subgrids without grid expansion. Specifically, if at a certain node integration takes

place, then $(D_k^n)_{ii} = 1$, while at all remaining nodes where interpolation is carried out, $(D_k^n)_{ii} = 0$. The actual selection of integration and interpolation nodes is made by the regridding strategy which is discussed later. The nesting property of the integration domains is also induced by this strategy and cannot be recovered from the above formulation, as it is hidden in the actual definition of D_k^n.

The interpolation step on level $k \geq 2$ can be written as

$$(3.5) \qquad (I_k - D_k^n)\, U_k^n = (I_k - D_k^n)\, [P_{k-1k} U_{k-1}^n + b_k^n].$$

The gridfunction b_k^n plays an auxiliary role here, but we need to include it due to the fact that boundary conditions on $\partial\Omega$ have been worked into the semi-discrete system. For the analysis it plays no role, whatsoever. Noteworthy is that the choice of interpolation is still free. The integration step on level $k \geq 1$ can be represented by $(D_1^n = I_1)$

$$(3.6) \qquad D_k^n U_k^n = D_k^n [R_{rk} U_r^{n-1} + \tau F_k(t_n, U_k^n)].$$

Values at or beyond internal boundaries needed in $F_k(t_n, U_k^n)$ are defined by (3.5). Thus (3.4) automatically prescribes values at internal boundaries. Recall that for nodes at and beyond internal boundaries, the associated entry of D_k^n is zero, by definition. Also note that (3.6) is coupled with (3.5) through these internal boundaries. Finally, due to the injection, at each grid level the fine grid solution $D_k^n R_{rk} U_r^{n-1}$ is used as initial function.

An important point to notice is that (3.4) - (3.6) live in the space S_k, due to the grid expansion. This means that the interpolation is considered to take place on the whole of ω_k. However, in actual application we interpolate only over the current integration domain. This deviation is justified as our regridding strategy is such that as far as the choice of the next finer integration domain is concerned, it makes no difference whether we interpolate over the whole of ω_k or restrict it to the current domain. Our strategy, together with this issue of restricted interpolation, is discussed at length in [17]. In the next section we will briefly review the main underlying ideas.

Finally, in (3.4) the number of levels, r, is a priori fixed independent of time. This fixed-level mode of operation may be inefficient and it is obvious that the method should be capable of working in a variable-level mode of operation. For general Runge-Kutta methods we discuss this in [18]. In this paper we restrict ourselves to fixed r, for reasons of space.

4. THE REGRIDDING STRATEGY

A strategy should fulfil two basic accuracy requirements. It should induce a sufficient local refinement in regions where the spatial errors are larger than elsewhere, and it should involve automatic control of the inevitable interpolation errors. This second requirement is often neglected, but is of significant importance. The reason is that if we regrid at each base time step, we interpolate at each base time step. Thus the interpolation errors can accumulate linearly with the base time steps, so that, reducing τ may eventually result in error growth, rather than in error decay. Although less, this threat remains if we would not regrid at every base time step, but per certain number > 1 of such steps.

In [17] we have developed a strategy which meets both requirements. We demand that the refinement is such that the spatial accuracy on the composite final grid is comparable to the spatial accuracy on ω_r if integration takes place on the whole of ω_r. As far as accuracy is concerned, this is the maximum we can ask for. In addition, this way we force the interpolation error to remain negligible when compared with the common spatial error on ω_r. This means that the accumulation of the interpolation error is automatically controlled. In this section we will outline the analysis upon which our regridding strategy is based.

In (3.4) the matrices D_k^n determine the integration domains. Two extreme choices are I_k and 0, which imply, respectively, integration or interpolation over the entire domain Ω. In our analysis the diagonal matrices D_k^n act as control matrices used to satisfy the above mentioned accuracy demand. Introduce the global error

$$(4.1) \qquad e_k^n = u_k^n - U_k^n, \qquad u_k^n = u_k(t_n),$$

and the perturbed scheme

$$(4.2a) \qquad u_1^n = R_{r1} u_r^{n-1} + \tau F_1(t_n, u_1^n) + \delta_1^n,$$

$$(4.2b) \qquad u_k^n = D_k^n [R_{rk} u_r^{n-1} + \tau F_k(t_n, u_k^n)] + (I_k - D_k^n) [P_{k-1k} u_{k-1}^n + b_k^n] + \delta_k^n, \qquad k = 2, \ldots, r.$$

Here, δ_k^n is the defect left by substituting u_k^n into (3.4). Let $D_1^n = I_1$. Because $u_k^{n-1} = R_{rk} u_r^{n-1}$, we can write

$$(4.3) \qquad \delta_k^n = D_k^n (\tau \alpha_k^n + \beta_k^n) + (I_k - D_k^n) \gamma_k^n, \qquad k = 1, \ldots, r,$$

where

$$(4.4a) \qquad \alpha_k(t) = \frac{d}{dt} u_k(t) - F_k(t, u_k(t)) \qquad \text{(local space error)}$$

$$(4.4b) \qquad \beta_k(t) = u_k(t) - u_k(t - \tau) - \tau \frac{d}{dt} u_k(t) \qquad \text{(local time error)}$$

$$(4.4c) \qquad \gamma_k(t) = u_k(t) - P_{k-1k} u_{k-1}(t) - b_k(t) \qquad \text{(interpolation error)}$$

Naturally, δ_k^n is composed of the three common local errors encountered in the space S_k. We omit here to discuss the asymptotics on these errors, as this is not relevant for the remainder.

Next subtract (3.4) from (4.2). This yields

$$(4.5a) \qquad Z_1^n e_1^n = R_{r1} e_r^{n-1} + \delta_1^n,$$

$$(4.5b) \qquad Z_k^n e_k^n = D_k^n R_{rk} e_r^{n-1} + (I_k - D_k^n) P_{k-1k} e_{k-1}^n + \delta_k^n, \qquad k = 2, \ldots, r,$$

where

$$(4.6) \qquad Z_k^n = I_k - \tau D_k^n \int_0^1 \frac{\partial}{\partial U} F(t_n, \theta u_k^n + (1-\theta) U_k^n) \, d\theta$$

is the integrated Jacobian matrix resulting from applying the mean value theorem for vector functions. Note that so far we consider the general nonlinear, semi-discrete problem (3.2). In the remainder it is now tacitly assumed that the matrix (4.6) is regular, which is a most natural condition satisfied in any realistic application. We also introduce the interpolation operator $X_k^n : S_{k-1} \rightarrow S_k$ given by

$$(4.7) \qquad X_k^n = (Z_k^n)^{-1} (I_k - D_k^n) P_{k-1k}.$$

The global errors e_k^n then can be shown to satisfy an inner-outer recurrence of the one-step type

(4.8) $$e_k^n = G_k^n e_r^{n-1} + \psi_k^n , \qquad n = 1, 2, \ldots ; k = 1, \ldots , r,$$

where the step operator G_k^n and the local error ψ_k^n are also recursively defined:

(4.9) $$G_1^n = (Z_1^n)^{-1} R_{r1} \qquad G_k^n = X_k^n G_{k-1}^n + (Z_k^n)^{-1} D_k^n R_{rk}, \qquad k = 2, \ldots, r,$$

(4.10) $$\psi_1^n = (Z_1^n)^{-1} \delta_1^n, \qquad \psi_k^n = X_k^n \psi_{k-1}^n + (Z_k^n)^{-1} \delta_k^n, \qquad k = 2, \ldots, r.$$

The occurrence of the backward finest level error e_r^{n-1} in (4.8) emanates from the injection. We are primarily interested in these finest level errors. G_k^n represents the step operator that advances e_r^{n-1} to the forward level k and ψ_k^n denotes the full local error at this level. Note that both G_k^n and ψ_k^n depend on quantities living on all grids ω_j, $1 \le j \le k$. Specifically, G_k^n is composed of the involved integration, interpolation and restriction operators, while the local error ψ_k^n is composed of the three types of local errors encountered.

As outlined above, a regridding strategy should find a proper balance between the two types of spatial errors and at the same time control the accumulation of interpolation errors for evolving time. The following result can be shown. First we split ψ_k^n into its temporal and spatial part. That is, we write

(4.11) $$\psi_k^n = \psi_{kt}^n + \psi_{ks}^n.$$

From (4.3), (4.9) and (4.10) we then can deduce that $\psi_{kt}^n = G_k^n \beta_r^n$ and

(4.12) $$\psi_{1s}^n = (Z_1^n)^{-1} \tau \alpha_1^n, \qquad \psi_{ks}^n = (Z_k^n)^{-1} [\tau D_k^n \alpha_k^n + (I_k - D_k^n)\rho_k^n], \qquad k = 2, \ldots , r,$$

where the new grid function ρ_k^n satisfies

(4.13) $$\rho_1^n = 0, \qquad \rho_k^n = \gamma_k^n + P_{k-1k} \psi_{k-1s}^n, \qquad k = 2, \ldots , r.$$

Because our main interest lies in the local refinement analysis, we now focuss on the spatial local error expressions (4.12), (4.13) which contain all possible local contributions from the spatial discretization and interpolation. Specifically, in (4.12) we have separated the common local space error

(4.14) $$\tau D_k^n \alpha_k^n,$$

restricted to the level-k integration domain, from all other space error contributions living outside this integration domain. These are collected in

(4.15) $$(I_k - D_k^n) \rho_k^n.$$

From (4.13) we see that the error ρ_k^n contains the level-k interpolation error γ_k^n plus the prolongated spatial local error $P_{k-1k} \psi_{k-1s}^n$ from the next coarser grid. This separation of spatial discretization and interpolation errors is very useful, since it enables us to constraint the diagonal matrices D_k^n in such a way that the space discretization error contribution (4.14) dominates its parasitic counterpart (4.15), which is entirely due to interpolation.

Let $\| \cdot \|$ be the maximum norm in the space S_k. The definition of the control matrices D_k^n is then accomplished through the so-called refinement condition. This condition reads

$$(4.16) \qquad \| (Z_k^n)^{-1} \tau D_k^n \alpha_k^n \| \geq \frac{1}{c} \| (Z_k^n)^{-1} (I_k - D_k^n) \rho_k^n \|, \qquad k = 2, \ldots, r,$$

where $c > 0$ is a control parameter to be specified. Substitution into (4.8) and taking norm bounds, gives for $k = r$ the global error inequality

$$(4.17) \qquad \| e_r^n \| \leq \| G_r^n e_r^{n-1} \| + \| G_r^n \beta_r^n \| + (1+c) \| (Z_r^n)^{-1} \tau D_r^n \alpha_r^n \|.$$

The crucial observation is now that by imposing (4.16), the parasitic interpolation error (4.15) has been virtually removed. Apart from the factor 1+c, the overall spatial error at the finest grid level is bounded by the common spatial error bound $\| (Z_r^n)^{-1} \| \| \tau \alpha_r^n \|$, which is also found on ω_k without any adaptation at all. This result is in agreement with the two requirements on the regridding strategy mentioned above.

For practical use the refinement condition (4.16) needs to be brought into a workable form. Needless to say that this involves various aspects of implementation, like estimation of local space and interpolation errors. We conclude this section by briefly outlining how we have dealt with (4.16). The main point is that we have replaced it by the related condition

$$(4.18) \qquad \| (I_k - D_k^n) (\gamma_k^n + P_{k-1k} (Z_{k-1}^n)^{-1} \tau D_{k-1}^n \alpha_{k-1}^n) \| \leq \frac{c}{r-1} \| (Z_r^n)^{-1} \tau D_r^n \alpha_r^n \|, \qquad k = 2, \ldots, r,$$

which is more feasible for implementation. If implicit Euler is contractive and the interpolation stable, i.e., $\| (Z_k^n)^{-1} \| \leq 1$ and $\| P_{k-1k} \| = 1$, then (4.18) can be proved to be stronger than (4.16). If implicit Euler is not truly contractive, but merely stable, then the bound 1 needs to be replaced by (unknown) stability bounds for $(Z_k^n)^{-1}$ which also enter (4.18). However, these are close to 1 so that also in this case (4.18) remains valid, approximately. We also have the experience that the stability condition on the interpolation is not crucial. Note that this condition is somewhat restrictive. For example, it holds for standard linear interpolants, but not for higher order Lagrangian ones.

In actual application (4.18) thus determines the integration domains within each base time step. This goes as follows. Suppose that at level-(k-1) a solution has been computed. Condition (4.18) is then checked in a flagging procedure which scans all level-k points lying within the level-(k-1) integration domain (recall the nesting property). Specifically, at these level-k points we check the inequality

$$(4.19) \qquad |(\lambda_k^n)_i| > \frac{c}{r-1} \| (Z_r^n)^{-1} \tau D_r^n \alpha_r^n \|, \qquad \lambda_k^n = \gamma_k^n + P_{k-1k} (Z_{k-1}^n)^{-1} \tau D_{k-1}^n \alpha_{k-1}^n.$$

If (4.19) is true, then it is decided that the point will remain within the integration domain. This means that the refinement condition (4.18) is then satisfied by putting $(D_k^n)_{ii} = 1$. Otherwise we put $(D_k^n)_{ii} = 0$, saying that the grid point will lie outside this domain. This way the refinement conditions (4.16) and (4.18) are satisfied.

Inspection of λ_k^n reveals that the integration is redone at a grid point if the sum of the interpolation and prolongated spatial discretization error is larger than the maximum spatial discretization error on ω_k. This requirement is quite restrictive for the interpolation, because the discretization errors are multiplied with τ and the interpolation error is not. This actually means that for decreasing τ the interpolation becomes more and more dominant. In other words, the smaller τ, the more points will be flagged into the new integration domain, so as to keep the interpolation error sufficiently small. This is what really should happen, because when going to a higher level within the current base time step, we never return to a point where the

solution has been interpolated. This means that the error will be carried along to the next base time step. Our strategy only allows this if the interpolation error is so small that the final spatial accuracy is not affected, according to the global error bound (4.17). Naturally, these observations suggest to use higher order interpolation. In application the use of higher order interpolation indeed leads to smaller integration domains than found with linear interpolation. We stress, however, that simple linear interpolation can be used.

The control parameter c plays a minor role. We usually take $c \approx 1$. Note that the larger c, the easier it will be to satisfy the refinement condition in the flagging procedure. Thus a 'large' value for c will lead to 'small' integration domains. At first sight this seems attractive. However, according to (4.17), a 'large' value for c will most likely result in 'large' spatial errors. Apparently, a 'large' value for c is not in accordance with our two regridding requirements.

Finally, to save space, we omit a discussion on the stability properties of the implicit Euler LUGR method. However, as a rule stability is not affected by interpolation. If the semi-discrete PDE system (3.2) is dissipative in the maximum norm, and the interpolation is stable, one can even prove that the maximum norm contractivity of implicit Euler is maintained ($\| G_r^n \| \leq 1$). We also recall that our analysis takes place in the space S_k (grid expansion), but that the interpolation and the subsequent scanning of grid points in the flagging procedure is restricted to the current integration domain. In [17] it is shown that the deviation caused by the restricted interpolation is allowed.

5. NUMERICAL ILLUSTRATION

We will numerically illustrate the outcome of imposing the refinement condition for a parabolic model problem. See [17] for details on estimation, implementation and strategy aspects. The model is hypothetical and due to [1]:

$$(5.1) \qquad u_t = u_{xx} + u_{yy} + f(x,y,t), \quad 0 < x,y < 1, \quad t > 0.$$

The initial function, the Dirichlet boundary conditions, and the source term f are such that

$$(5.2) \qquad u(x,y,t) = \exp\left(-80[(x - .25(2 + \sin(\pi t)))^2 + (y - .25(2 + \cos(\pi t)))^2]\right).$$

This solution is a cone that is initially centered at (1/2,3/4) and symmetrically rotates around (1/2,1/2) in the clockwise direction. The speed of rotation is constant with period 2. This problem is not a very difficult one in the sense that the spatial gradients are not very large, that is, the cone is not very steep. However, the problem is suitable to subdue the LUGR method to a convergence test and to check the refinement condition. The spatial discretization is based on standard 2-nd order differencing (implicit Euler is then contractive).

We have carried out two identical experiments. In the first linear interpolation has been used and in the second 4-th order Lagrangian. In both the solution is computed two times over the interval $0 \leq t \leq 2$ using a constant τ. In the first computation r = 3 and in the second r = 4, using a uniform 10 x 10 base grid. When going from r = 3 to r = 4, τ is decreased with the factor 2^2, since the smallest meshwidth is halved and the Euler method is of order one only. Then, in line with our analysis, the maximal global error should also decrease with this factor 2^2, approximately.

Table 1 shows the maxima of global errors restricted to the finest available subgrid. Inspection of this table clearly reveals the 2-nd order (this conclusion also follows if we would examine the global errors at the entire composite grid). Note the striking correspondence with the single fine-grid error. We conclude that the interpolation error is well controlled and that the spatial accuracy of the single finest grid is maintained.

At this point we should emphasize that in spite of the relatively large values for τ, the spatial error dominates the global errors shown in the table. In other words, conclusions on the spatial error behaviour induced by the local refinement algorithm can be drawn from these results. The threshold factor c = 1 has apparently no influence on the error. We owe this to the fact that the refinement condition has been derived from errors bounds. Furthermore, buffering in the flagging procedure may also have some interfering effect.

Table 1. Results for model problem (5.1)-(5.2).

τ	r	interpolation	single grid	error at t = 2
1/8	3	linear		0.01369
		4-th order		0.01376
			40 x 40	0.01389
1/32	4	linear		0.00340
		4-th order		0.00359
			80 x 80	0.00347

As anticipated by our strategy, the choice of interpolant has no notable influence on the error. The use of the two different interpolants is expressed in the slightly different integration domains shown in Figure 3. As expected, at the higher levels linear interpolation gives rise to somewhat larger domains. This means that the use of linear interpolation is more expensive. For both interpolants the moving domains accurately reflect the symmetric rotation of the cone, which once again nicely illustrates the reliability of the implemented refinement condition with the various estimators.

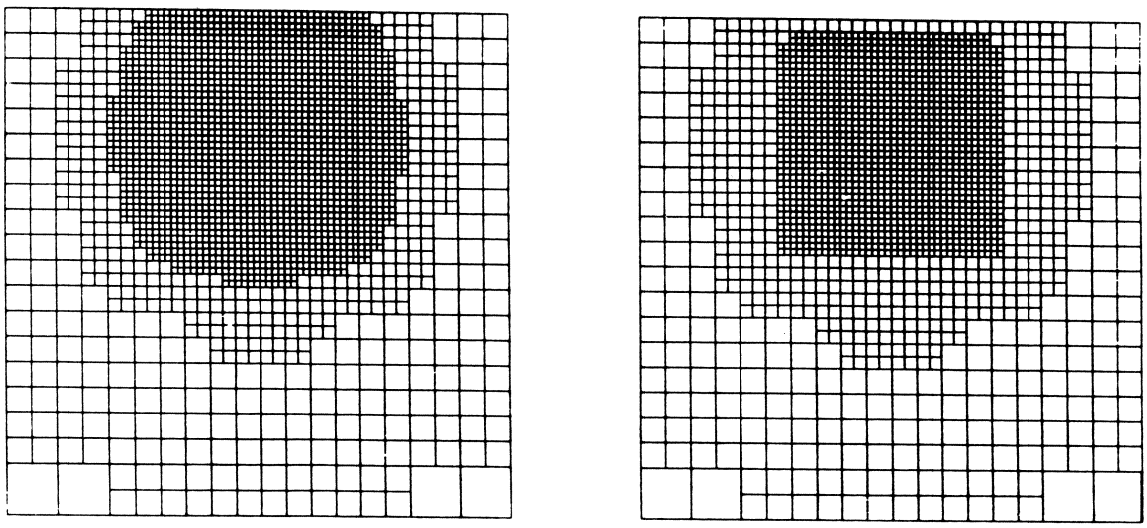

Figure 3. Problem (5.1)-(5.2). Composite grids of the r = 4 runs at t = 2. The left picture corresponds with linear interpolation and the right one with 4-th order Lagrangian.

6. RUNGE-KUTTA AND LINEAR MULTISTEP METHODS

Implicit Euler has excellent stability properties, but in connection with accuracy its order one is often considered to be too restrictive. Higher order methods are found in the familiar Runge-Kutta and linear multistep families. Given such a method, the question arises how to extend the implicit Euler LUGR technique, together with the analysis, so as to obtain a similar control of interpolation errors and maintenance of fine-grid spatial accuracy as found for (3.4).

6.1. Runge-Kutta methods.

Suppressing index k in the ODE formula (3.2), we denote the general one-step, s-stage RK scheme by

$$(6.1) \qquad U^{(i)} = U^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F(t_{n-1} + c_j\tau, U^{(j)}) \quad (1 \le i \le s+1), \qquad U^n = U^{(s+1)}.$$

Note that bracketed upperscripts are used for the intermediate stage values. In contravention of the usual notation, we use one and the same expression for the intermediate stages and the (s+1)-st output stage. The fact that $U^{(s+1)}$ is the approximation at the main step point t_n is clear from the context. We adhere to the convention $c_i = a_{i1} + \ldots + a_{is}$. Hence $U^{(i)}$ is an approximation at the intermediate point $t_{n-1} + c_i\tau$. Note that class (6.1) is supposed to represent the general RK class, containing all implicit and explicit methods.

The general Runge-Kutta LUGR formula now reads

$$(6.2a) \qquad U_1^{(i)} = R_{r1}U_r^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_1(t_{n-1} + c_j\tau, U_1^{(j)}) \qquad (1 \le i \le s+1)$$

$$(6.2b) \qquad U_k^{(i)} = D_k^n [R_{rk}U_r^{n-1} + \tau \sum_{j=1}^{s} a_{ij} F_k(t_{n-1} + c_j\tau, U_k^{(j)})] +$$

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

where $k = 2, \ldots, r$. Hence, $U_k^{(i)} \in S_k$ is the approximation to $u_k(t_{n-1} + c_i\tau)$ at grid ω_k. This formula is similar to the implicit Euler LUGR formula (3.4). In essence there is no difference with the implicit Euler formulation due to the fact that the interpolation step is carried out stagewise. In application the same operations are carried out, the only difference being that the interpolation is done at all stages and a more complicated integration formula is used. However, there exists one small exception for methods having $a_{1j} = 0$, $1 \le j \le s$. For such methods, including all explicit ones and the implicit Lobatto IIIA-methods (trapezoidal rule), no integration is carried out at the first stage. Now, due to the fact that the regridding matrix D_k^n depends on the stepnumber n and the level index k, and not on the stage index i, a consequence is that for these methods interpolation is carried out at the first stage, which is redundant since there is no integration there. It is therefore more natural to put $U_k^{(1)} = R_{rk}U_r^{n-1}$, which is what we do. We make no exception for RK methods having this zero coefficient row property at a higher stage, as these are likely to be of little practical interest.

The close correspondence with the implicit Euler formulation is seen very clearly if we rewrite (6.1), (6.2) in the compact Kronecker-product form notation (see, e.g. [8], Section 5.1). For example, if we denote

$$(6.3) \qquad \mathbf{U^n} = [U^{(1)T}, \ldots, U^{(s+1)T}]^T, \qquad \mathbf{F}(t_n, \mathbf{U^n}) = [F^{(1)T}, \ldots, F^{(s)T}, 0^T]^T,$$

where $F^{(i)} = F(t_{n-1} + c_i\tau, U^{(i)})$, and put $a_{is+1} = 0$, then (6.1) then can be rewritten as

(6.4) $$U^n = e \otimes U^{n-1} + \tau(A \otimes I)\, F(t_n, U^n),$$

where $e = [1, \ldots, 1]^T$ and A is the coefficient matrix (a_{ij}). This general RK formula can be interpreted as an 'implicit Euler formula' in the augmented vector space. It shall be clear that (6.2) can be rewritten in a similar way, leading to an 'implicit Euler LUGR formula' in the augmented spaces $\mathbf{S_k} = S_k^{s+1}$. This interpretation is very helpful for error analysis directed to controlling interpolation errors and maintenance of fine-grid spatial accuracy. Actually, this is one of the principal reasons to write the RK formula in the somewhat unusual form (6.1).

The error analysis we are referring to is given in [18]. There we derive a refinement condition similar to (4.16). Because this condition lives in $\mathbf{S_k}$, it must be elaborated further into a condition in S_k, similar to (4.18), so as to reduce overhead in the actual application. Our paper [18] contains an example of such an elaboration for the strongly A-stable DIRK method

(6.5)

0	0	0	0	$\theta = (3 + \sqrt{3})/6$
2θ	θ	θ	0	$b_1 = 1.5 - \theta - (4\theta)^{-1}$
1	b_1	b_2	θ	$b_2 = -0.5 + (4\theta)^{-1}$
	b_1	b_2	θ	

This method has classical order of consistency 3, stage order 2 [8], and is due to [2,7]. For a linear PDE problem we have proved that this lower stage order causes no order reduction at the internal (Dirichlet) boundaries, provided the accuracy order of the interpolation ≥ the order of the PDE operator. Hence, for a parabolic PDE simple linear interpolation is allowed.

We now present results of a numerical example for the DIRK method (6.5). The example is similar as in Section 5 and serves to illustrate the outcome of imposing the regridding condition, i.e., control of interpolation errors and maintenance of fine-grid spatial accuracy (for details see [18]). The equation is again linear and parabolic and given by

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

Note, however, that the choice of PDE operator is not our main concern here. This is the type of solution to be computed. Following [3], the initial function, the Dirichlet boundary conditions, and the source term f are such that

(6.7) $$u(x,y,t) = 1 - \tanh(25(x - t) + 5(y - t)).$$

This solution is a skew wave, propagating from left to right. The wave starts near the left boundary and arrives at the right boundary at approximately t = 0.8. We use the interval [0,0.6]. Symmetric 2-nd order finite-differences are used for the spatial discretization and the interpolation is linear. We give results from 4 integrations using, respectively, 1, 2, 3 and 4 levels. In all 4 cases the base grid is 20 x 20. During an integration τ is fixed, but when a grid level is added, we also halve τ. In view of the 2-nd order in space and the stage order 2 of the DIRK scheme, a gain factor of 4 in the global errors is to be expected. For comparison we also solved the problem on single-fine grids. The stepsizes in time and space are such that

the space error dominates the time error, so that we are able to draw valid conclusions on the performance of the LUGR method.

Table 2 shows maxima of global errors restricted to the finest subgrid (examining the composite grid covering the whole of Ω leads to the same conclusions). We see that the LUGR solutions converge according to the theory: the errors decrease with approximately the gain factor 4 for increasing r. We see that the errors are also very close to the single-finest grid errors. Hence, the results are in full accordance with the theory underlying the local refinement. Figure 4 shows grids obtained with r = 2, 3 and 4 at t = 0.3 and t = 0.6. Note that the grids nicely align with the wave front. In particular, this figure illustrates that if τ is decreasing, the local subgrids grow in order to satisfy the regridding condition (see inequality (4.8) and recall that we have used simple linear interpolation).

Table 2. Results for model problem (6.6)-(6.7).

τ	r	single grid	error at t = 0.3	error at t = 0.6
1/10	1	20x20	0.17319	0.17401
1/20	2		0.02728	0.02815
		40x40	0.02789	0.02810
1/40	3		0.00624	0.00716
		80x80	0.00680	0.00684
1/80	4		0.00177	0.00174
		160x160	0.00168	0.00169

6.2. Linear multistep methods.

Extending the implicit Euler local refinement analysis to one-step RK methods is complicated by the multi-stage character of these methods. For linear multistep (LM) methods this is much simpler, since there are no intermediate stages. The most popular LM method for time-dependent PDEs is the backward-differentiation (BDF) method. Therefore, to save space, we confine ourselves here to BDF.

Suppressing the level index k in the ODE formula (3.2), we denote the familiar s-step BDF formula by

$$U^n = a_1 U^{n-1} + \ldots + a_s U^{n-s} + \tau\theta_s F(t_n, U^n). \tag{6.8}$$

Due to its close analogy with implicit Euler (s = 1), in the spaces S_k the LUGR formulation for the BDF method immediately follows (see (3.4)):

$$U_1^n = R_{r1} \mathbf{U}_r^{n-1} + \tau\theta_s F_1(t_n, U_1^n), \tag{6.9a}$$

$$U_k^n = D_k^n [R_{rk} \mathbf{U}_r^{n-1} + \tau\theta_s F_k(t_n, U_k^n)] + (I_k - D_k^n) [P_{k-1k} U_{k-1}^n + b_k^n], \qquad k = 2, \ldots, r, \tag{6.9b}$$

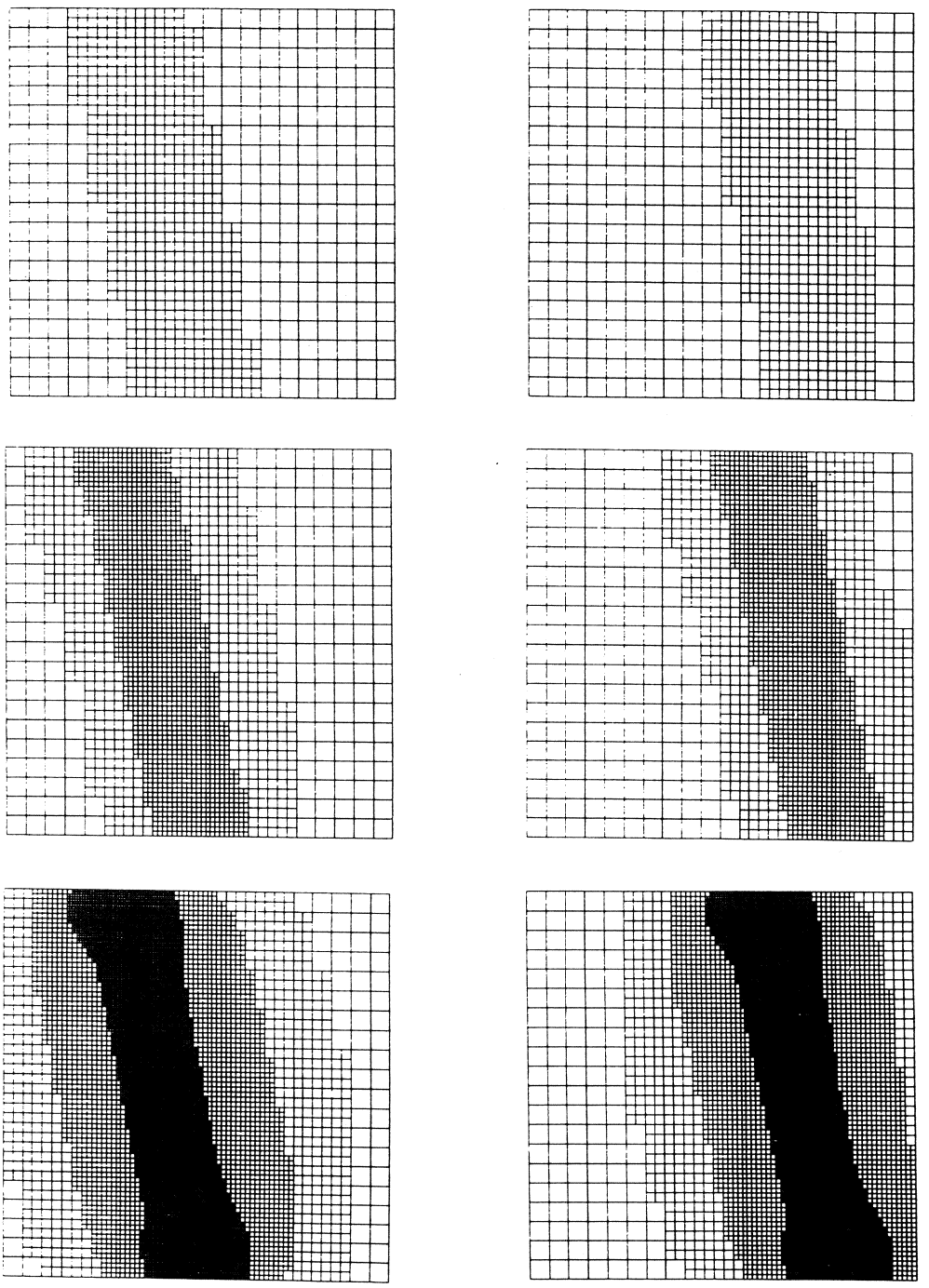

Figure 4. Problem (6.6)-(6.7). Composite grids for r = 2, 3 and 4 at t = 0.3 and t = 0.6.

where $U_r^{n-1} = a_1 U_r^{n-1} + ... + a_s U_r^{n-s}$ collects the past solution values at the finest-grid level r. Except for this past-values vector, (6.9) and (3.4) are completely identical, and so is their local refinement analysis. The reason is that we here merely deal with the question of how to balance the local space and interpolation errors committed at the current point of time $t = t_n$. The past values do of course show up in any global error analysis, but not in the local error analysis of which this particular local refinement analysis is part of. In conclusion, for (6.9) the control matrices D_k^n are defined by the same refinement condition as for implicit Euler.

The correspondence with the implicit Euler case relies on the premise of local subgrid expansion leading to formulation in the spaces S_k. In theory we then interpolate over the whole of the grids ω_k which is of course too expensive. As mentioned before, in application we use restricted interpolation (to the current integration domain) and can also justify this (see Section 3). However, for linear multistep methods like BDF we now encounter the practical difficulty that due to the restricted interpolation, not all components of the past-solution vector U_r^{n-1} required at $t = t_n$ will be available, simply because the integration domains move for evolving time. This difficulty can only be solved for by additional interpolation and injection of missing past solution values, assuming we keep regridding at every base time step. This necessarily involves a so-called flying restart per base time-step. The flying restart possibility has been proposed before by Furzeland [9] (see also Berzins et al. [5]). If we do not regrid at every base time step, then a cold restart is of course also possible. However, delaying regridding has the disadvantage that more points will be needed to resolve moving fine-scale structures. In these respects, LM methods are less amenable for static regridding than the one-step RK methods for which the backward interpolation difficulty does not exist. We note that the approach of delaying the regridding is followed by Bieterman [6].

So far we haven't implemented our LUGR technique in the BDF method, but we plan to do so in the future. The strong potential of the BDF method for the method of lines warrants further investigations.

7. SOME REMARKS ON IMPLICITNESS

In this paper we have focussed on the analysis of the local spatial refinement and refrained from examining efficiency questions. For the actual application one should have sufficient insight into the issues of efficiency and overhead, so as to be able to judge whether the tradeoff between the expected gain in efficiency due to reduction in number of grid points, and loss of efficiency due to extra overhead, is sufficiently large to justify adaptive grids.

In this respect, a subject of major importance is implicitness. When using an explicit time-stepping scheme, the only additional overhead emanates from data structure operations. Because for LUGR methods the data structure is not so complicated, this overhead is readily earned back. As a rule, explicit time-stepping is attractive to combine with LUGR, provided stability renders no problems. In contrast, implicitness leads to overhead costs for solving systems of linear or nonlinear algebraic equations. For LUGR methods implicitness has a still larger impact than for common single-grid methods, because multilevel local refinement results in a new system of linear or nonlinear algebraic equations to solve per level per base time step (we now tacitly assume adaptation at each base time step). Needless to say that the use of direct solvers requiring the Jacobian matrix be given in explicit form then will be expensive. These overhead costs can be reduced by not adapting the grids at each base time step, of course at the expense of accuracy. An attractive alternative may be provided by matrix-free iteration methods. A matrix-free iteration method does not require the Jacobian matrix be given explicitly, as it is based on matrix-vector product operations. Matrix-free iteration methods are, in theory, also directly applicable to nonlinear problems (inexact Newton methods). These methods are accelerated using some form of (problem dependent) preconditioning. Recently quite some research is in progress on using these methods in implicit method-of-lines schemes (see [12-14]). In view of our grid structure, multigrid is of course also a natural approach to

combine with our locally uniform, nested grid approach, provided the iteration scheme can be used matrix-free. Our current research code works with the Harwell sparse matrix solver MA28 combined with standard Newton iteration. Although MA28 is well suited to cope with the nonregular band structures we meet, for our application it is rather time consuming for the reasons just discussed.

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, 792-811.

[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. Berzins, P.M. Dew and R.M. Furzeland (1989), *Developing software for time-dependent problems using the method of lines and differential-algebraic integrators.* Appl. Numer. Math. 5, 375 - 398.

[6] M.B. Bieterman (1983), *A posteriori error estimation and adaptive finite elements grids for parabolic equations.* In: Adaptive computational methods for partial differential equations, eds. I. Babuska, J. Chandra and J.E. Flaherty, SIAM Publications, pp. 123 - 143.

[7] M. Crouzeix and P.A. Raviart (1980), *Approximation des problèmes d'evolution. Première partie: étude des méthodes linéaires à pas multiples et des méthodes de Runge-Kutta.* Unpublished Lecture Notes, Université de Rennes, France.

[8] K. Dekker and J.G. Verwer (1984), *Stability of Runge-Kutta methods for stiff nonlinear differential equations.* North-Holland, Amsterdam - New York - Oxford.

[9] R.M. Furzeland (1985), *The construction of adaptive space meshes.* Report TNER.85.022, Shell Research Ltd, Thornton Research Centre, Chester, England.

[10] W.D. Gropp (1987), *Local uniform mesh refinement on vector and parallel processors.* In: Large Scale Scientific Computing, eds. P. Deuflhard and B. Engquist, Birkhäuser Series Progress in Scientific Computing, Vol. 7, pp. 349-367.

[11] W.D. Gropp (1987), *Local uniform mesh refinement with moving grids.* SIAM J. Sci. Stat. Comput. 8, 292-304.

[12] A.C. Hindmarsh and P.N. Brown (1987), *Reduced storage techniques in the numerical method of lines.* Preprint UCRL - 96261, Lawrence Livermore National Laboratory.

[13] A.C. Hindmarsh and S.P. Nørsett (1988), *KRYSI, an ODE solver combining a semi-implicit Runge-Kutta method and a preconditioned Krylov method.* Report UCID - 21422, Lawrence Livermore National Laboratory.

[14] A.C. Hindmarsh (1989), *Combining the method of lines, stiff integrators, and Krylov methods.* Lecture presented at the SIAM Annual Meeting, San Diego, July 1989.

[15] K. Miller (1986), *Recent results on finite-element methods with moving nodes.* In: Accuracy estimates and adaptive refinements in finite element computations, eds. I. Babuska, O.C. Zienkiewicz, J. Gago and E.R. de A. Oliveira, Wiley, pp. 325 - 338.

[16] R.A. Trompert and J.G. Verwer (1989), *A static-regridding method for two-dimensional parabolic partial differential equations.* Report NM-R8923, CWI, Amsterdam (to appear in Appl. Numer. Math.).

[17] R.A. Trompert and J.G. Verwer (1990), *Analysis of the implicit Euler local uniform grid refinement method.* Report NM-R9011, CWI, Amsterdam.

[18] R.A. Trompert and J.G. Verwer (1990), *Runge-Kutta methods and local uniform grid refinement.* Report NM-R9022, CWI, Amsterdam.

[19] J.G. Verwer, J.G. Blom, R.M. Furzeland and P.A. Zegeling (1989), *A moving-grid method for one-dimensional PDEs based on the method of lines.* In: Adaptive methods for partial differential equations, eds. J.E. Flaherty, P.J. Paslow, M.S. Shephard and J.D. Vasilakis, SIAM Publications, pp. 160 - 175.

J.G. Verwer and R.A. Trompert
CWI, Dept. of Numerical Mathematics
Kruislaan 413
1098 SJ Amsterdam
The Netherlands
email: janv@cwi.nl