Konrad-Zuse-Zentrum für Informationstechnik Berlin

Jens Lang

Adaptive FEM for Reaction–Diffusion Equations

Invited talk, Grid Adaptation in Computational PDEs Conference, Edinburgh 1996

Preprint SC 96–28 (September 1996)

Jens Lang

Adaptive FEM for Reaction–Diffusion Equations

Abstract. An integrated time–space adaptive finite element method for solving mixed systems of nonlinear parabolic, elliptic, and differential algebraic equations is presented. The approach is independent of the spatial dimension. For the discretization in time we use singly diagonally linearly implicit Runge–Kutta methods of Rosenbrock type. Local time errors for the step size control are defined by an embedded strategy. A multilevel finite element Galerkin method is subsequently applied for the discretization in space. A posteriori estimates of local spatial discretization errors are obtained solving local problems with higher order approximation. Superconvergence arguments allow to simplify the required computations. Two different strategies to obtain the start grid of the multilevel process are compared. The devised method is applied to a solid–solid combustion problem.

KEY WORDS: Adaptive Finite Elements, Rosenbrock Methods, A Posteriori Error Estimates, Rothe Method, Reaction–Diffusion Equations

1 Introduction

A large class of industrially relevant problems governed by reaction and diffusion processes is set up by time–dependent systems of PDEs of the form

$$\begin{aligned} H(x,t,u)\,u_t &= \nabla\cdot(D(x,t,u)\nabla u) + F(x,t,u,\nabla u)\,, \\ &\quad x\in\Omega\subset \mathbb{R}^n\,, n=1,2,3,\ t>0\,, \end{aligned} \tag{1.1}$$

subject to boundary and initial conditions for the solution vector $u = (u_1, ..., u_m)^T$. These equations are quite general and include degenerate cases where the heat capacity matrix H and the diffusion matrix D are singular. The nonlinear vector F describes the reaction mechanism and allows to take into account mild convection. We assume that (1.1) has an isolated solution.

Our first investigation [21] involved one–dimensional reaction–diffusion systems of semilinear type. In contrast to the widely used standard method of lines (MOL) approach [1, 6, 14, 25] we first discretized in time and place the adaptation in space in the interior of the time step. This so–called adaptive Rothe method was proposed and studied in [7] for linear parabolic problems. We employed adaptive two–stage Runge–Kutta methods and a linear finite element Galerkin procedure. Due to superconvergence properties the spatial error estimates were computed as quadratic corrections to the solution on each element. These estimates were used to control a multilevel process which has been proven to be a successfull approach to many problems requiring adaptive discretization in space [11, 8]. The resulting algorithm has demonstrated its reliability for a variety of real–life chemical problems [20].

These techniques were extended to two–dimensional problems using piecewise linear polynomials on unstructed triangular grids as solution spaces and a Rosenbrock method of order three for temporal discretization. The h–refinement process was controlled by a posteriori error estimates based on interpolation techniques [19]. To include more challenging practically relevant problems we improved this estimation technique by the computation of approximate local residuals [15]. The computation of these estimates is much more expensive than the first method, but on the other hand they produce a quite accurate global measure of the actual spatial error. Anisotropic refinement is another difficult topic which does not seem to be satisfactorily solved up to now. In [16] we did a first step in this direction proposing a directional refinement along a given line.

In the spirit of adaptive elliptic solvers classical multilevel methods start from a solution–independent mesh. It has to capture the computational domain and has to yield a solution which permits to determine the required refinement to reach a prescribed tolerance. However, in a time integration process the permanent start with a fixed coarse mesh can quickly become wasteful. In this paper an efficient alternative is presented. A solution–dependent starting mesh for the new time is directly derived from the actual mesh. Dynamic tree structures are used to manage grid enhancement and robust coarsening as well.

The paper is outlined as follows. In Section 2 we present a short comparison between the adaptive MOL and Rothe approach to give better insight in the role of the discretization sequence. Following the time–space discretization sequence we describe the time integrator and the finite element method in Sections 3 and 4, respectively. A code based on our adaptive techniques is applied to solve a two–dimensional combustion problem. This example is also used in part to appraise the performance of the current method starting with a solution–dependent mesh, relative to the standard multilevel method starting with a fixed coarse mesh. It turns out that an appropriate reduction of the required refinement levels saves computational cost.

2 Adaptive Strategies

The standard method of lines employs the space–time discretization sequence. Discretizing first in space, the initial problem (1.1) is transformed into a DAE–system which can be solved directly by an appropriate time integrator. In real–life applications highly nonlinear, stiff, sparse, large, and block–structured systems have to be solved numerically.

Adaptive Space–Time Sequence (MOL)

- $\mathrm{TOL}_x \geq \mathrm{TOL}_t$
- Mesh adaptation after time step

$$\boxed{G_n, \varepsilon_x} \longrightarrow \boxed{G_{n+1}}$$

- Step size reduction: $\|\varepsilon_x\| > \mathrm{TOL}_x$, $\|\varepsilon_t\| > \mathrm{TOL}_t$

$\downarrow$ $\uparrow$

Mesh adaptation for $\|\varepsilon_x\| > \mathrm{TOL}_x$

One mesh adaptation $d = 0$

$\downarrow$ $\uparrow$

Adaptive Time–Space Sequence (Rothe)

- $\mathrm{TOL}_x \leq \mathrm{TOL}_t$
- Mesh adaptation during time step

$$\boxed{G_n, \varepsilon_x} \longrightarrow \begin{array}{c} \boxed{G^0_{n+1}, \varepsilon_x} \\ \vdots \\ \boxed{G^d_{n+1}, \varepsilon_x} \end{array}$$

- Step size reduction: $\|\varepsilon_t\| > \mathrm{TOL}_t$

Fig. 1: Space–time discretization versus time–space discretization

Implicit integrators such as DASSL [26] or RADAU5 [18] avoid time–step restrictions forced by stability requirements, and have proved to perform very well for the above problem class. Unfortunately, they require the inner solution of large nonlinear systems usually done by Newton–like iterations. One procedure to reduce drastically the complexity of the time integration

process and the computational amount of work per time step is to put an approximation of the Jacobian matrix directly into the discretization scheme. This leads to the well–known class of linearly implicit methods. One of the most efficient integrator of this type is the extrapolation code LIMEX [12].
In practical computations, the spatial discretization error should dominate the temporal error for global error control [23]. This is forced by the requirement $TOL_x \geq TOL_t$. In the standard MOL approach the mesh is adapted after an accepted time step. An a posteriori error estimate ε_x of the local spatial errors related to the current mesh G_n, is employed to construct a new mesh G_{n+1}. Next, the new solution and the corresponding approximate discretization errors ε_x and ε_t are computed. Time–step reductions occur if at least one of the prescribed tolerances TOL_x or TOL_t is not reached. In both cases the spatial discretization is not changed because it was derived by analyzing the old solution on G_n (see Fig.1).
In recent years efficient methods have been developed which also allow to adapt the spatial discretization during the time step. Starting with an initial mesh G^0_{n+1} derived from G_n the approximation property in space is successively improved until the prescribed tolerance TOL_x is achieved for a final mesh G^d_{n+1}, $d \geq 0$. Here, excellent stationary multilevel solvers such as KASKADE [13] or PLTMG [3] are available to handle this adaptation.
In consequence, we now discretize first in time and then in space. In the mathematical literature this complementary type of method is often referred to as Rothe method. The time–dependent system (1.1) is understood as a system of differential–algebraic equations in a Hilbert space of infinite dimension. This permits us to consider the spatial discretization as a perturbation of the discrete orbit in that space [7, 21]. The requirement $TOL_x \leq TOL_t$ keeps this spatial perturbation below a certain level such that the generated time–step sequence is nearly the same as in the case of no perturbation. Time–step reductions take place if the estimated temporal discretization error is greater than TOL_t.
Another possibility to get solution–dependent spatial discretizations are moving grid methods where the coordinates of the spatial points are no longer chosen at the beginning of each time step. These methods are able to succeed in more sophisticated one–dimensional problems. In spite of the recent progress been made these methods are still far away from a robust solver for complex problems in higher dimensions.
For an extended discussion of the MOL and Rothe approach the interested reader is referred to [10].

3 Time Discretization and Step–Size Control

Following the adaptive Rothe method our aim is to compute a well–fitted spatial mesh at any time. Due to rich spatial dynamics frequent mesh adaptation is necessary. Consequently, time integrators using a wide stencil of different time levels to construct higher order solutions such as BDF or extrapolation methods seem not be favorable here. Storage requirements, the amount of work for the control of interpolation errors and searching algorithms to find spatial points in each of these grids become quickly larger and larger, especially for complex 2D and 3D problems. In such a situation one–step methods of Rosenbrock type have some natural advantages: These need only two different grids, one to hold the old solution and one to construct the new one. Compared to the other methods the solution of only linear and decoupled systems of equations is required. They are easy to implement and accepted to integrate stiff ODEs efficiently for moderate accuracy requirements.
Rosenbrock methods have been studied for DAEs with a solution–dependent matrix multiplying the time derivative and being either invertible or singular [22, 27]. Rewriting the system (1.1) as

$$\begin{aligned} u_t &= z \\ 0 &= F(x,t,u,\nabla u) + \nabla \cdot (D(x,t,u)\nabla u) - H(x,t,u)z \end{aligned} \tag{3.1}$$

and adding the trivial equation $\partial_t t = 1$ we formally get a differential–algebraic system of the form

$$B\,v = f(v)\,,\; v = (u,z,t) \tag{3.2}$$

with a constant matrix $B = \mathrm{diag}(I,0,1)$. A Rosenbrock scheme for system (3.2) reads now as follows

$$\left(\frac{1}{\gamma\tau_n}B - \partial_v f(v_{n-1}\right) l_j \;=\; f\left(v_{n-1} + \sum_{i=1}^{j-1} a_{ji} l_i\right) + \frac{1}{\tau_n} B \sum_{i=1}^{j-1} c_{ji} l_i \tag{3.3}$$

$$v_n \;=\; v_{n-1} + \sum_{j=1}^{s} b_j l_j \tag{3.4}$$

and suitable chosen values for the coefficients γ, a_{ji}, c_{ji} and b_j. The new solution v_n is computed by summing up all stage values $l_j,\ j = 1, ..., s$. These are determined by linear equations using only one small matrix of dimension of the solution vector and several function evaluation on the right–hand side.

Replacing the coefficients b_j in (3.4) by different coefficients $\hat{b}_j$ a second solution $\hat{v}_n$ of inferior order can be obtained. The difference $\|v_n - \hat{v}_n\|$ satisfactorily estimates the error introduced by the temporal discretization. This approach is well–known as embedding strategy and often used in the context of Runge–Kutta methods.
Let ϵ_{n-1} be the estimated error in time of the previous time step. A value for the new time step is classically chosen according to

$$\tau_n^{(1)} := \rho \left(\frac{TOL_t}{\epsilon_{n-1}}\right)^{1/(p+1)} \tau_{n-1} \tag{3.5}$$

where ρ denotes a safety factor and p is the local order of the solution $\hat{v}_n$. It is well known that this control mechanism often leads to a nonsmooth behaviour of the time integration process. After a drastic step–size reduction the temporal error becomes very small. The proposed new time step will be too optimistic leading to repeated rejections. Taking into account more informations from the previous steps we get a better step–size selection of the form

$$\tau_n^{(2)} := \rho \, \frac{\tau_{n-1}}{\tau_{n-2}} \left(\frac{TOL_t \; \epsilon_{n-2}}{\epsilon_{n-1} \; \epsilon_{n-1}}\right)^{1/(p+1)} \tau_{n-1} \tag{3.6}$$

as introduced in [17] for implicit methods. Additionally, we have experienced that the following two modifications are usefull: (1) After successive rejections the standard method (3.5) with smaller p should be applied. (2) Since the proposed step $\tau_n^{(2)}$ is often too optimistic when steps have to be increased, we choose in this case

$$\tau_n := \min\,(\tau_n^{(1)}, \tau_n^{(2)}). \tag{3.7}$$

In [16] a study of the time–step selection for a one–dimensional oscillating flame is reported. Applying the described procedure the resulting reduction in CPU time was above 20%. Of course, it is not very large, but it has been obtained with merely no additional cost.

4 Spatial Discretization and Grid Control

Let us now turn over to the spatial discretization. The linear partial differential equations (3.3) have to be solved for each stage value. In the spirit of full adaptivity this is being done by a self–adaptive multilevel finite element method (MFEM) as implemented in the KASKADE package [13]. Replacing the solution space by an appropriate sequence of discrete spaces with successively increasing dimensions the size of the arising linear systems necessary to achieve the prescribed accuracy can be drastically reduced with respect to uniform methods [11, 8].
Starting with the weak formulation of (3.3) the finite element solutions $l_j^k \in S_k^q$ have to satisfy the equations

$$\left(A_k\, l_j^k\,,\, w\right) = (r_j\,,\, w)\;,\;\forall w \in S_k^q\;,\; j = 1, ..., s, \tag{4.1}$$

$$A_k := \left(\frac{1}{\gamma\tau_n} B - \partial_v f(v_{n-1}^k)\right)\,. \tag{4.2}$$

Here, the finite element subspace S_k^q corresponds to the mesh G_{n+1}^k and consists of all continuous functions which are polynomials of order q on each finite element. The function $r_j = r_j(l_1^k, ..., l_{j-1}^k)$ stands for the right–hand side of the j–th equation in (3.3).
In a second step an a posteriori error estimation has to be computed to give specific assessment of the error distribution of the solution

$$v_n^k = v_{n-1}^k + \sum_{j=1}^{s} b_j\, l_j^k\,. \tag{4.3}$$

An obvious possibility is to utilize interpolation techniques which play an important role in the traditional finite element approximation theory [19]. The corresponding error indicators can be relatively cheap computed; they are able to improve the grids in many practical cases. Note that for linear elements, i.e. $q = 1$ in (4.1), this approach is equivalent to the control of the curvature of the solution. Unfortunately, a priori error constants specified for a general class of solutions have to be involved. This may lead to pessimistic estimations for the actual error forcing more degrees of freedom than really needed. A second type of error estimation is based on the computation of approximate local residuals employing higher order finite elements. These estimators are much more expensive than the first one, but they produce quite accurate and robust estimators. This approach is now described in detail.

Defining the spatial discretization error $e_j := l_j - l_j^k$ for each stage value of (4.1) an approximation $e_j^k \in S_k^{q+1}$ is given by the equations

$$\left(A_k\, e_j^k \;,\; w\right) = (\hat{r}_j \;,\; w)\;,\; \forall w \in S_k^{q+1}\,, \tag{4.4}$$

$$\hat{r}_j := r_j(e_1^k + l_1^k, ..., e_{j-1}^k + l_{j-1}^k) - A_k\, l_j^k\,. \tag{4.5}$$

Observe that due to the nonlinearity of r_j we get a nonlinear error propagation from the first to the last stage value. The approximate discretization error $e^k \in S_k^{q+1}$ for the finite element solution $v_n^k \in S_k^q$ reads now as follows

$$e^k = v_{n-1} - v_{n-1}^k + \sum_{j=1}^{s} b_j\, e_j^k\,. \tag{4.6}$$

The first contribution to the spatial error results from the fact that the fine grid components of v_{n-1} can not be represented on those parts of the grid G_{n+1}^k where the latter is coarser than the grid of the previous time step. In the case of only one mesh adaptation typically for the standard MOL approch (d=0, see Fig. 1), one has to take special care of that error because there is no chance to reduce it by time–step reduction.
Of course, an estimation process in space which includes the equations (4.4) is too expensive. The amount of work would be the same as for the resolution of (4.1) with S_k^{q+1} instead of S_k^q. One common method is to compute error estimates only in terms of local quantities [2]. In fact, we simply solve the equations (4.4) on small subdomains $\theta_i \subset \Omega$, $i = 1, ...M$, $\cup\overline{\theta_i} = \overline{\Omega}$ imposing homogenous Dirichlet boundary conditions. Denoting by θ such a domain with a fixed index i which is obviously the union of several finite elements, a local approximation $\epsilon_j^k(\theta)$ of the stage error e_j^k fulfills

$$\begin{aligned} \left(A_k\, \epsilon_j^k \;,\; w\right) &= (\hat{r}_j \;,\; w)\;,\; \forall w \in S_k^{q+1}(\theta)\,, && (4.7)\\ \epsilon_j^k &= 0\,,\; x \in \partial\theta,\; \text{if } \partial\theta \cap \partial\Omega = \emptyset\,. && (4.8) \end{aligned}$$

If $\partial\theta\cap\partial\Omega$ is not empty the boundary conditions have to be modified according to the boundary conditions on $\partial\Omega$.
Let us further consider a hierarchical decomposition

$$S_k^{q+1}(\theta) = S_k^q(\theta) \oplus W_k^{q+1}(\theta) \tag{4.9}$$

where $W_k^{q+1}(\theta)$ is the subspace that corresponds to the span of all additional basis functions needed to enrich the space $S_k^q(\theta)$ to order $q + 1$. Assuming superconvergence of the solution $v_n^k \in S_k^q$ in the nodal points we can approximate the local error $\epsilon_j^k(\theta)$ only in the space $W_k^{q+1}(\theta)$, which further saves computational cost. Indeed, using the hierarchical basis to resolve (4.7), one observes that the components lying in $S_k^q(\theta)$ will change very little from the previous computation. This technique is widely used for the adaptive solution of partial differential equations [4, 9, 11]. It remains to compute $\epsilon_j^k \in W_k^{q+1}(\theta)$ satisfying the equations

$$\begin{aligned} \Big(A_k\, \epsilon_j^k \;,\; w \Big) &= (\hat{r}_j \;,\; w) \;,\; \forall w \in W_k^{q+1}(\theta) \,, && (4.10) \\ \epsilon_j^k &= 0 \,,\; x \in \partial\theta ,\; \text{if } \partial\theta \cap \partial\Omega = \emptyset \,. && (4.11) \end{aligned}$$

The dimension of the local stiffness matrix in (4.10) is very small, such that an LU–decomposition is available by Gaussian elimination. This decomposition can then be employed for all stage errors. In practical computations one often observes that the higher–order errors are dominated by the first stage error ϵ_1^k. Setting then $s = 1$ and $b_1 = 1/\gamma$ in (4.6) the discretization error e^k is approximated by a lower order method in time known as semi–implicit Euler method [15]. For an analogous discussion within the MOL approach we refer to [24].

We now define local error estimators for each subdomain θ_i, $i = 1, ..., M$,

$$\eta_i := \| v_{n-1} - v_{n-1}^k + \sum_{j=1}^{s} b_j \epsilon_j^k \|_{\theta_i} \qquad (4.12)$$

and set the global spatial discretization error

$$\varepsilon_x = \left(\sum_{i=1}^{M} \eta_i^2 \right)^{1/2} \,. \qquad (4.13)$$

The values η_i are used to judge the quality of the underlying discretization in the domain θ_i. If η_i exceeds a certain error barrier, all finite elements which are contained in θ_i are marked for refinement. This estimate is quite optimal in the sense that it is a lower and upper bound of the true spatial error up to multiplicative constants which are independent of the mesh and solution. Moreover, in practical computations one expects that they are not large.

One fundamental question of the adaptive Rothe method is how to choose G_{n+1}^0, the starting grid for the new solution. In the spirit of classical elliptic solvers often a time–fixed coarse grid G^0 is taken [7]. The resulting iterative solution process is then speeded up by the natural use of the nested grid structure. Moreover, no special coarsening routines and algorithms are

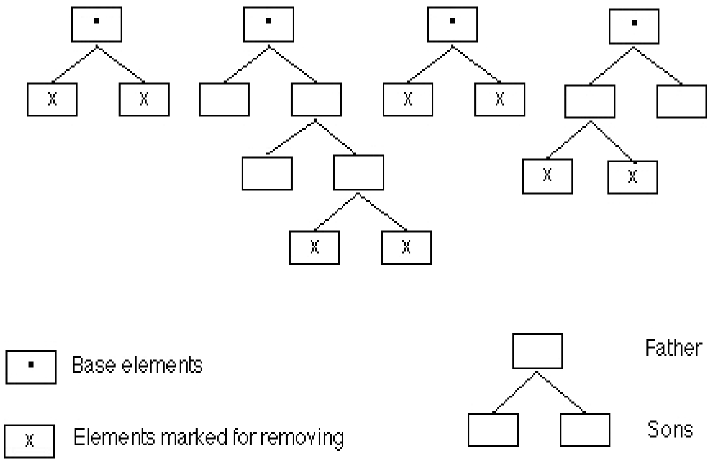

Fig. 2: One–dimensional tree structure used when removing elements

needed. On the other hand we have to pay for more assembly and estimation cost during the refinement process. Obviously, in situations where the solution changes very slowly in time, the permanent start with a fixed coarse mesh would be wasteful. But not only in this case one has the intuitive feeling that the new first mesh should be an approximation of the final mesh from the last time, i.e. $G^0_{n+1} \sim G^d_n$. Therefore, a considerably more efficient alternative is to remove degrees of freedom only by analyzing the current solution to find regions where the errors are small enough. Of course, this requires data structures that allow grid enhancement and robust coarsening as well. Great care must be taken to avoid spoiled grids when elements are removed.

We first select candidates for coarsening employing the tree structure of the old mesh G^d_n. An element is marked if it has a father which does not have a refined son. Fig.2 shows the result for a given one–dimensional tree structure. This strategy can be directly applied to more sophisticated tree structures in higher dimensions.

In a second step we take into account the local error behavior of all marked elements. So, regions of small error can be identified by the η–values. Supposing an asymptotic behavior of the form $\eta \sim ch^p$ as the characteristic mesh size h of the domain θ tends to zero, a simple prediction of the η–value after coarsening will be approximately

$$\eta_{pred} \sim c(\alpha h)^p \sim \alpha^p \eta \,. \tag{4.14}$$

Here, the value αh describes the characteristic mesh size after coarsening. Many refinement strategies such as bisection in 1D or the standard "red–

green" procedure in 2D and 3D use $\alpha = 2$. Clearly, we should coarsen the domain θ if η_{pred} is still above the refinement barrier. Doing this, we have the justified hope that the new region will be not immediately refined in the next step. Practical experience has shown that the above described "trimming–tree" strategy is rather satisfactory.

5 An Example

To illustrate the performance for the above described method we solve a solid–solid combustion problem [28] employing a Rosenbrock method of order three with embedded order two ($s = 3$) and linear finite elements on unstructed triangular meshes ($q = 1$). The local error estimate of the finite element steering the adaptive mesh refinement is obtained solving local problems on all subdomains θ which are the union of two neighboring triangles having one common edge. In order to produce a nearly optimal mesh, all domains θ having an error η larger than a certain barrier η_{thresh} are refined. To compute η_{thresh} we use a simple prediction scheme introduced in [2] to forecast what may happen to the η-value if the subdomain θ is subdivided. Given a prescribed global tolerance TOL we use $TOL_x = TOL/3.0$ and $TOL_t = TOL/2.0$ to control the discretization errors in a weighted L_2–norm [15]. We first apply the trimming–tree strategy for getting a new starting mesh. The computations were done on an IBM RS6000 computer.
The combustion problem under consideration ranges among the most demanding for spatial adaptivity when the thin reaction sheet is to be resolved numerically. We consider the ignition and propagation of the alloying reaction front to study the automatic adjustion of the time step. Assuming constant thermodynamic properties the equations to be solved for the temperature $T(x, y, t)$ and the product concentration $Y(x, y, t)$ read

$$\partial_t T = \kappa \nabla^2 T + Qw \tag{5.1}$$

$$\partial_t Y = w \qquad (x, y) \in \Omega = [0, 1.5] \times [0, 1] \tag{5.2}$$

$$w := \frac{D}{R^2} \frac{(1-Y)^{1/3}}{1-(1-Y)^{1/3}} e^{-E/T}$$

with the initial conditions

$$T(x, y, 0) = 300.0,\ Y(x, y, 0) = 0.001\,.$$

Homogenous Neumann boundary conditions are applied on all boundaries except in the starting phase where the system is ignited by a linearly increasing Dirichlet condition on the left boundary, i.e.

$$\begin{aligned} T(0, y, t) &= 300.0 + 3.4E4\,t \quad &, t \leq 0.02, \\ \partial_x T(0, y, t) &= 0 &, t \geq 0.02\,. \end{aligned}$$

We further set $E = 1.1E4$, $Q = 2.7E3$, $\kappa = 1E-4$, $D/R_{ref}^2 = 5.8E3$. A variation of the material properties in space is included by setting $R = R_{ref}$

for $y \in [0.2, 0.8]$ and linearly decreasing particle size to $R = R_{ref}/4$ at $y = 0$ and $y = 1$. Note that there is no diffusion term in the second equation leading to very steep fronts in the profiles of the product concentration. We choose $TOL = 2E-3$ to reflect correctly the temporal dynamic of the flame propagation.

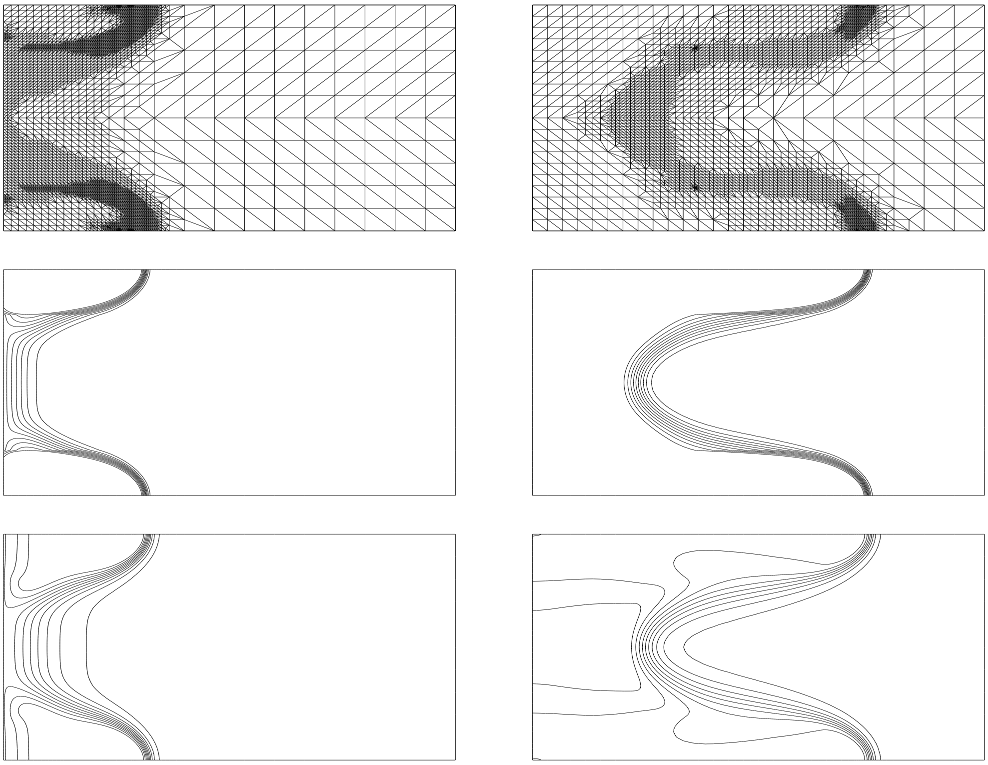

Fig. 3: Adaptive grid, level lines of product concentration and temperature at $t = 0.02$ (left column, top to bottom) and $t = 0.03$ (right column).

Fig. 3 shows the propagation of the reaction front through the nonuniformly packed reactor. The flame travels much more faster near the upper and lower boundary of the domain due to the higher reaction rate. The highly nonuniform grids give an impression of the saving in terms of degrees of freedom with respect to uniform discretizations.
The temporal development of the degrees of freedom is plotted in Fig. 4 which is also concerned with the adaptivity in time. It is mainly the steepness of Y requiring the observed refinement. In a first phase of the ignition the solid is only pre–heated and the time step is slowly decreased. In a second phase the reaction front starts to propagate and the time step becomes about

$\tau = 1E-4$. Of course, an efficient transition between these two phases would become almost impossible without automatic time step control. The peak at $t = 0.02$ is related to the fact that a change of the boundary conditions is requested. This imposes one very small time step to exactly reach this point. Let us now compare the two different strategies for choosing G^0_{n+1}, the coarse–to–fine strategy starting with a time–fixed coarse grid, and the trimming–tree strategy using an approximation of the final old mesh.

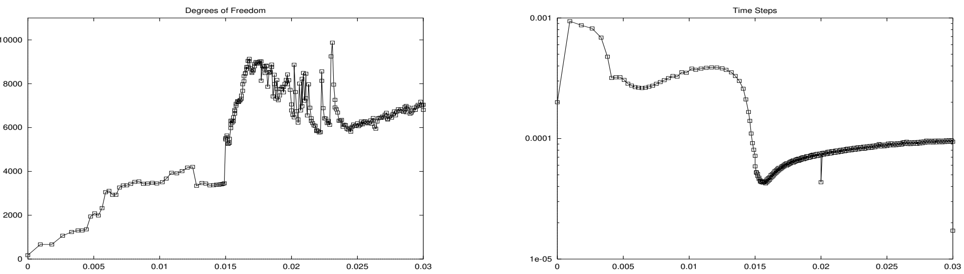

Fig. 4: Temporal development of degrees of freedom (left), and selected time–steps during the integration (right) for TOL=2E-3.

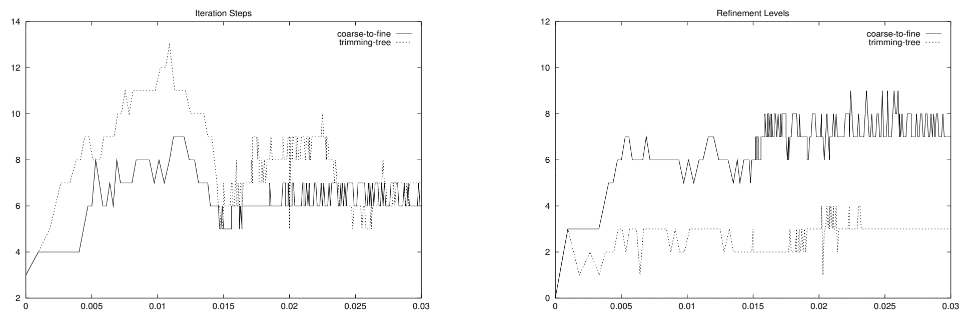

Fig. 5: Temporal development of iteration steps per refinement level (left), and number of required refinement levels (right) for the trimming–tree ($\cdots$) and the coarse–to–fine strategy (—).

First we look at the number of iteration steps per time step forced by the BICGStab algorithm [29] with ILU–preconditioning. This number is devided by the number of required refinement levels. In Fig. 5 one observes that only few iteration steps per refinement level are necessary to get the solutions. These numbers are nearly independent of the degrees of freedom ranging

from approximately 500 up to $10,000$ during the course of the simulation. Thus, optimal complexity is reached in both cases.
Tab. 1 shows that the trimming–tree strategy performs faster with respect to the computational cost. It saves considerable time for assembly, estimation and the iterative process due to the drastic reduction of required refinement levels (see Fig. 5).

	Coarse–to–Fine	Trimming–Tree
Assemble & Estimate	8,258 sec	7,391 sec
Iterate	5,967 sec	3,463 sec
Refine	1,030 sec	0,858 sec
All	15,727 sec	12,340 sec

Tab. 1: Computational cost for the coarse–to–fine and trimming–tree strategy

Acknowledgement

The author wishes to thank P. Deuflhard for inspiring discussions on stiff ODE integrators and adaptive multilevel methods. R. Roitzsch provided invaluable advise based on his wide experience in the field of programming, and I am indebted to him for his assistance with several of my problems.

Bibliography

[1] S. Adjerid, J. E. Flaherty, Second–order finite element approximations and a posteriori error estimation for two–dimensional parabolic systems, *Numer. Math.* 53 (1988) 183–198

[2] I. Babuška, W. Rheinboldt, Error estimates for adaptive finite element computations, *SIAM J. Numer. Anal.* 15 (1978) 736–754

[3] R. E. Bank, *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations, Users Guide 6.0* (SIAM, Philadelphia, 1990)

[4] R. E. Bank, R. K. Smith, A posteriori error estimates based on hierarchical bases, *SIAM J. Numer. Anal.* 30 (1993) 921–935

[5] M. Berzins, P. M. Dew, R. M. Furzeland, Developing software for time-dependent problems using the method of lines and differential-algebraic integrators, *Appl. Numer. Math.* 5 (1989) 375–397

[6] M. Bieterman, I. Babuška, An adaptive method of lines with error control for parabolic equations of the reaction–diffusion type, *J. Comp. Phys.* 63 (1986) 33–66

[7] F. A. Bornemann, An adaptive multilevel approach to parabolic equations I. General theory and 1D–implementation, *IMPACT Comput. Sci. Engrg.* 2 (1990) 279–317

[8] F. A. Bornemann, B. Erdmann, R. Kornhuber, Adaptive multilevel methods in three space dimensions, *Int. J. Numer. Meth. Engrg.* 36 (1993) 3187–3203

[9] F. A. Bornemann, B. Erdmann, R. Kornhuber, A posteriori error estimates for elliptic problems in two and three space dimensions, *SIAM J. Numer. Anal.* 33 (1996) 1188–1204

[10] P. Deuflhard, J. Lang, U. Nowak, Adaptive algorithms in dynamical process simulation, in: H. Neunzert, ed., *Progress in Industrial Mathematics at ECMI 94.* (Wiley–Teubner, 1996) 122–137

[11] P. Deuflhard, P. Leinen, H. Yserentant, Concepts of an adaptive hierarchical finite element code, *IMPACT Comput. Sci. Engrg.* 1 (1989) 3–35

[12] P. Deuflhard, U. Nowak, Extrapolation integrators for quasilinear implicit ODEs, in: P. Deuflhard, B. Engquist, eds., *Large Scale Scientific Computing*, PSC 7 (Birkhäuser, Basel, 1987) 37–50

[13] B. Erdmann, J. Lang, R. Roitzsch, KASKADE – Manual, Technical Report TR 93–5, Konrad–Zuse–Zentrum für Informationstechnik Berlin, Germany, 1993

[14] J. E. Flaherty, P. Moore, Integrated space–time adaptive hp–refinement methods for parabolic systems, *Appl. Numer. Math.* 16 (1995) 317–341

[15] J. Fröhlich, J. Lang, Twodimensional cascadic finite element computations of combustion problems, Preprint SC 96–5, Konrad–Zuse–Zentrum für Informationstechnik Berlin, Germany, 1996, submitted to Comp. Meth. Appl. Mech. & Engrg.

[16] J. Fröhlich, J. Lang, R. Roitzsch, Selfadaptive finite element computations with smooth time controller and anisotropic refinement, Preprint SC 96–16, Konrad–Zuse–Zentrum für Informationstechnik Berlin, Germany, 1996

[17] K. Gustafsson, Control of Error and Convergence in ODE Solvers, Ph.D. Thesis, Department of Automatic Control, Lund Institute of Technology, 1992

[18] E. Hairer, G. Wanner, *Solving Ordinary Differential Equations II, Stiff and Differential–Algebraic Problems.* (Springer, Berlin–Heidelberg, 1991)

[19] J. Lang, Two–dimensional fully adaptive solutions of reaction–diffusion equations, *Appl. Numer. Math.* 18 (1995) 223–240

[20] J. Lang, High–resolution self–adaptive computations on chemical reaction–diffusion problems with internal boundaries, *Chem. Engrg. Sci.* 51 (1996) 1055–1070

[21] J. Lang, A. Walter, An adaptive Rothe method for nonlinear reaction–diffusion systems *Appl. Numer. Math.* 13 (1993) 135–146

[22] Ch. Lubich, M. Roche, Rosenbrock methods for differential–algebraic systems with solution–dependent singular matrix multiplying the derivative, *Computing* 43 (1990) 325–342

[23] J. Lawson, M. Berzins, P. M. Dew, Balancing space and time errors in the method of lines for parabolic equations, *SIAM J. Sci. Stat. Comput.* 12 (1991) 573–594

[24] P. Moore, A posteriori error estimates with finite element semi– and fully discrete methods for nonlinear parabolic equations in one space dimension, *SIAM J. Numer. Anal.* 31 (1994) 149–169

[25] U. Nowak, J. Frauhammer, U. Nieken, A fully adaptive algorithm for parabolic partial differential equations in one space dimension, *Comput. & Chem. Engrg.* 20 (1996) 547–561

[26] L. R. Petzold, A description of DASSL: a differential–algebraic system solver, in: Proc. IMACS World Congress 1982

[27] M. Roche, Rosenbrock methods for differential algebraic equations, *Numer. Math.* 52 (1988) 45–63

[28] M. D. Smooke, M. L. Koszykowski, Two–dimensional fully adaptive solutions of solid–solid alloying reactions, *J. Comp. Phys.* 62 (1986) 1–25

[29] H. A. van der Vorst, BI–CGSTAB: a fast and smoothly converging variant of BI–CG for the solution of nonsymmetric linear systems, *SIAM J. Sci. Statist. Comp.* 13 (1992) 631–644