Konrad-Zuse-Zentrum für Informationstechnik Berlin

Jens Lang

High–Resolution Selfadaptive Computations on Chemical Reaction–Diffusion Problems with Internal Boundaries

Preprint SC 94–9 (May 1994)

Jens Lang

High–Resolution Selfadaptive Computations on Chemical Reaction–Diffusion Problems with Internal Boundaries

Abstract. Large chemical computations show the need for full adaptivity supporting the development of robust and highly efficient programs. For solutions possessing sharp moving spatial transitions, as travelling wavefronts or emerging boundary and internal layers, an automatic adjustment of both the space and the time stepsize is generally accepted to be more successful in efficient resolving critical regions of high spatial and temporal activity. In contrast to the widespread discretization sequence first space then time the reversed sequence first time then space is employed. Full adaptivity of the proposed algorithm is realized by combining embedded time discretization and multilevel finite element space discretization. In this paper the algorithm is described for one–dimensional problems. The numerical results show the significantly new perspectives opened by this approach.

Keywords: initial boundary value problem, adaptive Rothe method, embedded Runge–Kutta method, multilevel finite element method

AMS Classification: 65J15, 65M30, 65M50

Chapter 1

Introduction

Many chemical phenomena are governed by reaction–diffusion processes. The resulting initial boundary value problems consist of parabolic differential equations that are coupled through a highly nonlinear source term, often representing the reaction process of the chemical components.

One widespread approach to solve such a kind of partial differential equations is the method of lines (MOL). It involves the discretization sequence first space then time. In order to get a simple and therefore "fast implementable" discretization, two rather popular strategies are in use: uniform finite differences in space and implicit Euler discretization in time. Sometimes even an explicit Euler discretization is applied. However, with increasing complexity of the above problems these schemes can easily give disastrous results that are obviously – or sometimes not so obviously – incorrect.

One of the remarkable features of large chemical models being composed of various subsystems, is the different character of the arising differential equations. For example, the transport of chemical substances can be caused by convection, diffusion or reaction mechanism. Here, special conditions on internal boundaries have to be used to couple all these subsystems. Furthermore, in frequent cases algebraic conditions, possibly in differential form, enter into the problem formulation. Of course, these features require some care when aiming to devise reliable numerical methods to solve such a problem. Adaptive techniques are characterized by the fact that the numerical approximation changes and evolves with the solution. They are often the only way to provide the computational codes necessary to solve some of the difficult problems arising in chemical computations.

Employing the MOL approach the whole power of time integrators such as LSODE, DASSL and LIMEX ([16], [9], [6]) developed in a long research tradition is available to solve the semi–discrete problem accurately and efficiently, see e.g. [8], [19] for a thorough overview. Adaptivity in the context of ordinary differential equations (ODE's) means a stepsize control with respect to the time discretization. It turns out that this control is not enough for a class of challenging chemical problems such as problems with travelling reaction zones and internal boundary layers, which additionally require a time–dependent adaption of the spatial discretization. Two different strategies are well–known: the static and the dynamic regridding. Static regridding aims at finding a better space grid after an appropriate number of time steps using some information about the estimated spatial discretization error. It is clear, that this approach introduces an element of discontinuity into the time integration process and leads to a space grid which generally comes "too late"

[3]. In opposite, dynamic regridding is a possibility to include the current spatial nodes into the solution process making them time–dependent. That means, they now become additional unknowns of the whole system. Such moving–grid techniques have been used successfully for one space dimension [13], [1]. In two space dimensions, however, their application is less trivial, and it is not always clear how to extend the underlying grid selection procedure [21]. Therefore, 2D moving–grid methods have hardly been applied to real–life problems. For further information, the interested reader is referred to [7] and [15], where several real–life examples from chemical computing are reported.

Another possibility is the discretization sequence first time then space – classically known as Rothe method [18]. This sequence allows to compute the space discretization during the time integration process. Therefore, the Rothe method should be preferred to overcome the above difficulties. A further advantage is the fact, that the theory of the whole adaptive process can be formulated independently of the space dimension. First basic investigations have been done for linear parabolic PDEs in [4] and later extended to diffusion–reaction systems in [12].

In this paper an implementation of the fully–adaptive Rothe method for one–dimensional reaction–diffusion problems with internal boundaries is described. The time discretization is based on a special embedded Runge–Kutta method of order 3(2), which is designed to keep its accuracy even in the case of differential–algebraic equations. The occurring elliptic subproblems are solved by a multilevel finite element method (MFEM) with linear elements using fundamental ideas of [5]. In consequence, a space grid well suited to the problem under consideration is obtained in order to assure a desired accuracy. The resulting algorithm represents a significant improvement over the previous one developed in [11] since it is now possible to handle differential–algebraic equations with internal boundaries. The program used for the above computations is based on the code KARDOS implemented by the author [10]. The excellent behavior of the proposed approach is demonstrated on some real life chemical problems such as a chemical catalytic process and a vertical bubble reactor.

Finally, it should be mentioned that in principle an extension to the higher dimensional case can be done without any difficulties. Clearly, the corresponding realizations are much more expensive.

Chapter 2

Problem Formulation

Let $u = (u_1, u_2, ..., u_r)$ be a vector function and $\Omega = \Omega_1 \cup \Omega_2$ the union of two open bounded domains in $\mathbb{R}$, that have a common boundary point. The boundary $\partial\Omega$ of Ω is splitted into two subsets Γ_{int} and Γ_{ext}, see Figure 2.1.

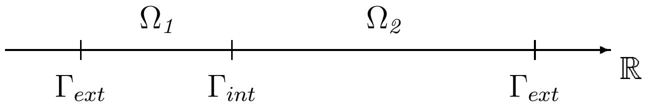

Fig. 2.1: *Decomposition of the computational domain*

We consider the system of parabolic initial boundary value problems

$$
\begin{aligned}
P(x)u_t - (D(x)u_x)_x &= F(u), \quad x \in \Omega \subset \mathbb{R}, \quad t \in (0, t_e], \\
u(0, x) &= u_0(x), \\
\alpha(x)u + \beta(x)u_x &= \gamma(t, x), \quad x \in \Gamma_{ext} \subset \partial\Omega, \\
\alpha_L^1(x^-)u(x^-) + \beta_L^1(x^-)u_x(x^-) &= \alpha_R^1(x^+)u(x^+) + \beta_R^1(x^+)u_x(x^+), \\
\alpha_R^2(x^+)u(x^+) + \beta_R^2(x^+)u_x(x^+) &= \alpha_L^2(x^-)u(x^-) + \beta_L^2(x^-)u_x(x^-), \\
& \qquad\qquad x^-, x^+ \in \Gamma_{int} \subset \partial\Omega
\end{aligned}
\tag{2.1}
$$

as a model of reaction–diffusion processes with one internal boundary. On the external boundary Γ_{ext} Dirichlet or Cauchy boundary conditions are given, where the function γ may be time–dependent to handle external heat–up processes. The internal boundary conditions on Γ_{int} realize the coupling between the processes in Ω_1 and Ω_2. The matrices $P(x)$ and $D(x)$ may vanish on subsets of Ω. In the case $P(x) = 0$ for some $x \in \Omega$, we get algebraic equations, possibly in differential form if simultaneously $D(x) \neq 0$. It is clear, that the special situation of one internal point can be easily extended to more internal points.

Note, that the solution u need not be continuous on Γ_{int}, for the identity $x^- = x^+$ does not imply $u(x^-) = u(x^+)$. That means, a standard continuous discretization method has to be significantly modified to handle such kind of internal boundary conditions.

Chapter 3

Adaptive Time Discretization

In order to discretize the time first and then the space, we consider the reaction–diffusion equations (2.1) as a pure time problem. That means, for the moment we only deal with $u(t)$ satisfying the equation

$$\begin{aligned} Pu_t &= f(u), \quad t \in (0, t_e], \\ u(0) &= u_0, \end{aligned} \tag{3.1}$$

where

$$f(u) = F(u) - Au \; . \tag{3.2}$$

The operator A denotes the weak representation of the diffusion operator in (2.1) with weakly imposed boundary conditions.

We can use a highly efficient time integrator to solve this problem. In general, chemical reaction systems are characterized by the coexistence of fast and slow reactions, making the whole system very stiff. An explicit integration is therefore often inappropriate since it may lead to wrong solutions. The use of an implicit method promises better results. However, a fully implicit discretization requires the solution of nonlinear systems. This is avoided by a method, which replaces nonlinear systems by a fixed sequence of linear systems. These algorithms are often called linearly implicit Runge–Kutta methods. Among them, Rosenbrock methods have shown to give very satisfactory results for stiff equations. We use the following 3–stage embedded Rosenbrock method:

$$(P - \gamma_{ii}\tau f_u(u_0))k_i \;=\; \tau f(u_0 + \sum_{j=1}^{i-1} \alpha_{ij}k_j) + \tau f_u(u_0) \sum_{j=1}^{i-1} \gamma_{ij}k_j \; , \quad i = 1, 2, 3,$$

$$\begin{aligned} u_1 &= u_0 + \sum_{j=1}^{3} b_j k_j \; , \\ \hat{u}_1 &= u_0 + \sum_{j=1}^{3} \hat{b}_j k_j \; , \end{aligned} \tag{3.3}$$

where τ is the (variable) time step. The solutions u_1 and $\hat{u}_1$ should have different order using the same intermediate values k_j, $j = 1, 2, 3$, but with different coefficients. Yet this form is not suited to be implemented directly, because along with function evaluations there are still matrix$*$vector products on the right–hand side, that are too costly. They can be avoided by the transformation

$$l_i := \sum_{j=1}^{i} \gamma_{ij} k_j \ , \quad i = 1, 2, 3, \tag{3.4}$$

leading to the new system

$$\begin{aligned} \left(\frac{1}{\tau \gamma_{ii}} P - f_u(u_0) \right) l_i &= f(u_0 + \sum_{j=1}^{i-1} a_{ij} l_j) + P \sum_{j=1}^{i-1} \frac{c_{ij}}{\tau} l_j \ , \quad i = 1, 2, 3, \\ u_1 &= u_0 + \sum_{j=1}^{3} m_j l_j \ , \\ \hat{u}_1 &= u_0 + \sum_{j=1}^{3} \hat{m}_j l_j \ . \end{aligned} \tag{3.5}$$

Setting $\gamma_{11} = \gamma_{22} = \gamma_{33} := \gamma$ only one matrix computation on the left–hand side is needed. Furthermore, the equation (3.5) can then be solved in three successive stages of smaller dimension than the whole system. The traditional problem of choosing the remaining coefficients leads to a nonlinear algebraic system. We use a special set of coefficients (see Table 3.1), which guarantees the order 3 and 2 of the solutions u_1 and $\hat{u}_1$ respectively, and L–stability in (3.5). The corresponding method without the above transformation was given in [17], where further results on convergence and consistency can be found.

Tab. 3.1: Special coefficients

γ	=	$\gamma_{11} = \gamma_{22}$	$= \gamma_{33}$	=	0.435866521508459
γ_{21}	=	0.1685887625570998			
γ_{31}	=	4.943922277836421			
γ_{32}	=	1.000000000000000			
m_1	=	2.236727045296589	$\hat{m}_1$	=	2.059356167645941
m_2	=	2.250067730969645	$\hat{m}_2$	=	0.169401431934653
m_3	=	-0.209251404439032	$\hat{m}_3$	=	0.0
a_{21}	=	1.605996252195329	c_{21}	=	0.8874044410657823
a_{31}	=	1.605996252195329	c_{31}	=	23.98747971635035
a_{32}	=	0.0	c_{32}	=	5.263722371562130

Due to the difference of the approximated values of order 3 and 2 we have a good estimator of the main error term describing the local error of the second order method

$$\epsilon_t := \|u_1 - \hat{u}_1\| \ . \tag{3.6}$$

Given a tolerance tol_t for the time discretization, the usual proposal for the new step size is

$$\tau_{new} := \rho \left(\frac{tol_t}{\epsilon_t} \right)^{\frac{1}{3}} \tau \ , \tag{3.7}$$

where ρ denotes a safety factor. The norm in (3.6) has to be chosen very carefully to reflect accurately the scale of the problem. We use a weighted root–mean–square norm, given by

$$\|u_1 - \hat{u}_1\|_{0,w} := \left(\frac{1}{r} \sum_{i=1}^{r} \frac{\|u_{1,i} - \hat{u}_{1,i}\|_0^2}{w_i^2} \right)^{\frac{1}{2}} \ , \tag{3.8}$$

where $\| \cdot \|_0$ stands for the usual L_2–norm over Ω, that means

$$\|u_{1,i}\|_0 := \left(\int_{\Omega} u_{1,i}^2 \, dx \right)^{\frac{1}{2}} .$$

Recall, that here the index $i = 1, \dots, r$ denotes the i–th component of the function u_1, consequently, (3.8) permits to scale each component of the error ϵ_t individually by a time–dependent weight which is determined as follows. Setting two threshold factors

$$\begin{aligned} \eta_i^r &:= RTOL_i * max_t \|u_{1,i}\|_0 \ , \\ \eta_i^a &:= ATOL_i * |\Omega|^{1/2} \ , \end{aligned} \tag{3.9}$$

the weights w_i are

$$w_i := \begin{cases} \eta_i^a & \text{if} \quad \|u_{1,i}\|_0 < \eta_i^a \\ \eta_i^r & \text{if} \quad \eta_i^a \le \|u_{1,i}\|_0 < \eta_i^r \ , \qquad i = 1, \dots, r \\ \|u_{1,i}\|_0 & \text{if} \quad \|u_{1,i}\|_0 \ge \eta_i^r \ . \end{cases} \tag{3.10}$$

The tolerance $ATOL_i$ should indicate the absolute value at which the i–th component is essentially insignificant. On the other hand, the value $RTOL_i$ affects the relative accuracy of the i–th component with respect to its maximal value taken over all the time. This control turns out to be quite efficient and robust for a wide class of problems.

Now a discussion of the boundary conditions related to the different stages of the above Rosenbrock method is appropriate. The first stage reads

$$\left(\frac{1}{\tau\gamma}P - f_u(u_0)\right) l_1 = f(u_0). \tag{3.11}$$

Replacing the function f according to (3.2), we get

$$\left(\frac{1}{\tau\gamma}P - F_u(u_0)\right) l_1 - \frac{\partial}{\partial x}\left(D\frac{\partial l_1}{\partial x}\right) = F(u_0) + \frac{\partial}{\partial x}\left(D\frac{\partial u_0}{\partial x}\right). \tag{3.12}$$

The other two equations can be derived analogously. The boundary conditions of all the intermediate values $l_i,\ i = 1,2,3$, have to be chosen in such a way, that the solution u_1 fulfills the boundary conditions in (2.1). Inserting the final terms of equation (3.5) for the solution u_1 into these conditions and using the consistency relation for the method (3.3)

$$\sum_{j=1}^{3} b_j = 1 \ , \tag{3.13}$$

a simple computation leads to the corresponding boundary conditions

$$\begin{aligned}
\alpha(x)l_i + \beta(x)\frac{\partial l_i}{\partial x} &= (\gamma(t_0+\tau,x) - \gamma(t_0,x))\sum_{j=1}^{i}\gamma_{ij} \ , \\
&\qquad x \in \Gamma_{ext}, \\
\alpha_L^1(x^-)l_i(x^-) + \beta_L^1(x^-)\frac{\partial l_i}{\partial x}(x^-) &= \alpha_R^1(x^+)l_i(x^+) + \beta_R^1(x^+)\frac{\partial l_i}{\partial x}(x^+) \ , \\
\alpha_R^2(x^+)l_i(x^+) + \beta_R^2(x^+)\frac{\partial l_i}{\partial x}(x^+) &= \alpha_L^2(x^-)l_i(x^-) + \beta_L^2(x^-)\frac{\partial l_i}{\partial x}(x^-) \ , \\
&\qquad x^-, x^+ \in \Gamma_{int},\ i = 1,2,3.
\end{aligned} \tag{3.14}$$

Chapter 4

Adaptive Space Discretization

To implement one time step, one elliptic problem has to be solved approximately at each stage of the Rosenbrock method. In the spirit of full adaptivity, a self–adaptive spatial discretization method is the appropriate choice to solve these problems with high accuracy. We use a MFEM, which is an excellent tool to ensure that a prescribed tolerance tol_x for the spatial discretization error is achieved. The general principle of the multilevel technique mainly consists of replacing the solution space by a sequence of discrete spaces with successively increasing dimension to improve the approximation accuracy [5]. Self–adaptivity here calls for an efficient criterion optimizing this sequence in order to minimize the space discretization error. Summing up these parts, an adaptive MFEM is characterized by four modules: the finite element assembling code, the error estimation technique, the refinement strategy, and last but not least the solver of the linear equations. These steps are now described in detail.

The starting point for the use of the FEM is the weak formulation of the corresponding boundary value problem. The stage problems (3.5) are equivalent to the variational formulations

$$B_\tau(l_i, v) = r_i(v) \quad \text{for all } v \in V, \quad i = 1, 2, 3, \tag{4.1}$$

where

$$B_\tau(l_i, v) = \int\limits_{\Omega} \left(\left(\frac{1}{\tau\gamma} P - F_u(u_0) \right) l_i v + D \frac{\partial l_i}{\partial x} \frac{\partial v}{\partial x} \right) dx - \int\limits_{\partial\Omega} D \frac{\partial l_i}{\partial n} v \, ds$$

denotes the time–dependent bilinear form associated with the operator on the left–hand side of (3.12), r_i stands for the corresponding right–hand side operator of the i–th stage, and

$$V := \left\{ v = (v_1, \ldots, v_r) \in [H^1(\Omega)]^r \text{ vanishing on the Dirichlet boundary} \right\}.$$

The solution $u \in [H^1(\Omega)]^r$ has to satisfy the boundary conditions.
For $\Omega = \Omega_1 \cup \Omega_2 = (\xi_1, \xi_2) \cup (\xi_2, \xi_3)$ let

$$\triangle_n := \{ \xi_1 = x_0 < x_1 < \cdots < x_l = \xi_2 = x_{l+1} < \cdots < x_n = \xi_3 \} \tag{4.2}$$

be a partition of the domain Ω into finite elements $I_j := (x_{j-1}, x_j),\ j = 1, \ldots, n,\ j \neq l+1$. Denoting by $S(\triangle_n)$ the set of all continuous functions $v_n = (v_{n,1}, \ldots, v_{n,r})$ on Ω with components $v_{n,i}$ that are linear on each $I_j \in \triangle_n$

and possibly discontinuous at $x = \xi_2$, we define the finite element solution and trial space by

$$\begin{aligned} S_n &:= \{v_n \in S(\triangle_n) \text{ satisfying the Dirichlet conditions}\}\,, \\ T_n &:= \{v_n \in S(\triangle_n) \text{ vanishing on the Dirichlet boundary}\}\,. \end{aligned} \tag{4.3}$$

As parameters to describe a function $v_n \in S(\triangle_n)$ we choose the values $v_n(x_j)$ of v_n at the nodes $x_j,\ j = 0, \ldots, n$, of $\triangle_n$. The corresponding basis functions $\phi_j \in S(\triangle_n)$ are then componentwise defined by

$$\phi_{j,i}(x_k) = \delta_{jk} \equiv \begin{cases} 1 & \text{if} \quad j = k, \\ 0 & \text{if} \quad j \neq k, \qquad j, k = 0, \ldots, n. \end{cases}$$

Each component of the function $v_n \in S(\triangle_n)$ has the representation

$$v_{n,i}(x) = \sum_{j=0}^{n} \omega_j \phi_{j,i}(x), \quad \omega_j = v_{n,i}(x_j), \quad x \in \Omega \cup \partial\Omega.$$

The nodes on the Dirichlet boundary are excluded for all $v_n \in T_n$ since $v_n = 0$ there.
We can now formulate the following FEM to solve the stage problems (3.5) starting from the variational formulation (4.1):
Find $l_i^n \in S_n$ such that

$$B_\tau(l_i^n, v_n) = r_i(v_n) \quad \text{for all } v_n \in T_n, \quad i = 1, 2, 3. \tag{4.4}$$

We see that (4.4) is equivalent to the linear systems of equations

$$B\zeta_i = r_i\ , \quad i = 1, 2, 3,$$

where B, the stiffness matrix, consists of elements $b_{jk} = B_\tau(\phi_j, \phi_k)$, $\zeta_i = (\zeta_i^j)$ and $r_i = (r_i^j)$ are vectors with elements $\zeta_i^j = l_i^n(x_j)$, $r_i^j = r_i(\phi_j)$. During the assembling process the computation of the stiffness matrix B and the right–hand side vector r_i is done by summing the contributions from the different finite elements. The resulting stiffness matrix is the same for all the stages so that the three boundary value problems differ only in their right–hand side. Up to now we have taken into account the different external and internal boundary conditions only for the definition of the solution and trial spaces in (4.3). In what follows, the implementation of the boundary values will be worked out in detail.

External Dirichlet boundary conditions.

$$l_i^n(x) = (\gamma(t_0+\tau,x) - \gamma(t_0,x)) \sum_{j=1}^{i} \gamma_{ij}, \quad x \in \Gamma_{ext}.$$

To realize general Dirichlet boundary conditions within the FEM, the whole system (4.4) is enlarged by the algebraic equations for the nodes on the Dirichlet boundaries, although these values are not really unknown. Therefore, the test space T_n is simply replaced by $S(\triangle_n)$. The external Dirichlet values are imposed by the explicit use of the boundary condition as seen in the following for the stiffness matrix B and the right–hand side vector r_i considering a Dirichlet node with index j:

$$\begin{pmatrix} & & & 0 & & & \\ & & & \vdots & & & \\ & & & 0 & & & \\ 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ & & & 0 & & & \\ & & & \vdots & & & \\ & & & 0 & & & \end{pmatrix} \qquad \begin{pmatrix} r_i^0 - b_{0j} l_i^n(x) \\ \vdots \\ r_i^{j-1} - b_{j-1j} l_i^n(x) \\ l_i^n(x) \\ r_i^{j+1} - b_{j+1j} l_i^n(x) \\ \vdots \\ r_i^m - b_{mj} l_i^n(x) \end{pmatrix}$$

Internal Dirichlet boundary conditions.
They appear in the general form

$$\begin{aligned} l_i^n(x^-) &= \alpha_R^1(x^+) l_i^n(x^+), \\ l_i^n(x^+) &= \alpha_L^2(x^-) l_i^n(x^-), \quad x^-, x^+ \in \Gamma_{int}\ , \end{aligned}$$

not necessarily for both points x^+ and x^-. It is clear, that these algebraic conditions have to be taken into account in the final linear system. Assuming that the system is already enlarged by the equations for the Dirichlet nodes, we manipulate the corresponding rows and columns in the following way:

$$\begin{pmatrix} & \vdots & & \vdots & \\ 0 \cdots 0 & 1 & 0 \cdots 0 & -\alpha_R^1(x^+) & 0 \cdots 0 \\ & \vdots & & \vdots & \\ 0 \cdots 0 & -\alpha_L^2(x^-) & 0 \cdots 0 & 1 & 0 \cdots 0 \\ & \vdots & & \vdots & \end{pmatrix} \quad \begin{pmatrix} \vdots \\ 0 \\ \vdots \\ 0 \\ \vdots \end{pmatrix}$$

External and internal Cauchy boundary conditions.

$$\begin{aligned}
\alpha(x) l_i^n + \beta(x) \frac{\partial l_i^n}{\partial x} &= (\gamma(t_0+\tau, x) - \gamma(t_0, x)) \sum_{j=1}^{i} \gamma_{ij} \ , \\
&\qquad x \in \Gamma_{ext}, \\
\alpha_L^1(x^-) l_i^n(x^-) + \beta_L^1(x^-) \frac{\partial l_i^n}{\partial x}(x^-) &= \alpha_R^1(x^+) l_i^n(x^+) \ , \\
\alpha_R^2(x^+) l_i^n(x^+) + \beta_R^2(x^+) \frac{\partial l_i^n}{\partial x}(x^+) &= \alpha_L^2(x^-) l_i^n(x^-) \ , \\
&\qquad x^-, x^+ \in \Gamma_{int}.
\end{aligned}$$

These conditions, well–known as natural boundary conditions within the FEM theory, are very easy to include. The derivation in the normal direction $\partial l_i / \partial n$ in the bilinear form in (4.1) has to be replaced with the help of the above Cauchy conditions, leading to additional entries in the stiffness matrix and the right–hand sides.

A special set of boundary conditions.
Often the so–called transmission condition is encountered at the inner boundary, i.e. continuity of the solution and of the diffusive flux is imposed. This reads

$$\begin{aligned}
l_i^n(x^-) &= l_i^n(x^+), \\
D(x^+) \frac{\partial l_i^n}{\partial x}(x^+) &= D(x^-) \frac{\partial l_i^n}{\partial x}(x^-), \quad x^-, x^+ \in \Gamma_{int}.
\end{aligned}$$

Hence, a continuous FEM can be applied, that means the two different nodes in x^- and x^+ coincide. Furthermore, no internal boundary integral has to be computed, for

$$\int_{\Gamma_{int}} D \frac{\partial l_i^n}{\partial n} v_n \, ds = \int_{x^+} D \frac{\partial l_i^n}{\partial x} v_n \, ds - \int_{x^-} D \frac{\partial l_i^n}{\partial x} v_n \, ds = 0.$$

Next, the error estimator of the space discretization will be described. Assuming that the approximated solution is given as a piecewise linear function over a fixed grid, the aim is obviously to find critical regions with high spatial discretization error where additional nodes should be inserted to improve the solution. This process leads to an equidistribution of the spatial discretization error. Unfortunately, the error is typically not precisely known and can in general only be estimated. One rather popular error estimation technique is the use of higher order local approximations. This can be done by solving the same elliptic problems locally on each finite element imposing the current FEM approximation of the solution as Dirichlet boundary condition [2]. In the present case quadratic solution and trial functions instead of linear ones are applied.

With the time integrator of the previous section, the new solution u_1^n on $\triangle_n$ is described by three different stage problems. An estimation process including all these problems would become too expensive. A natural way then is the use of the first stage only. This restriction is justified by the fact, that

$$u_E^n := u_0 + \frac{l_1^n}{\gamma} \tag{4.5}$$

is exactly the semi–implicit Euler solution of (3.1) on the partition $\triangle_n$. Hence it can be assumed, that a space grid well fitted to the solution of the first stage problem works well for the solution u_1, too. Consequently, let us consider the local elliptic problem

$$\left(\frac{1}{\gamma\tau}P - F_u(u_0)\right)\bar{l}_1 - \frac{\partial}{\partial x}\left(D\frac{\partial \bar{l}_1}{\partial x}\right) = F(u_0) + \frac{\partial}{\partial x}\left(D\frac{\partial u_0}{\partial x}\right),\ x \in I_j,$$

$$\bar{l}_1(x) = l_1^n(x),\ x \in \partial I_j. \tag{4.6}$$

A weak formulation in terms of the local error $\varepsilon_j := \bar{l}_1 - l_1^n \in [H_0^1(I_j)]^r$ is the equation

$$B_\tau(\varepsilon_j, v) = r_1(v) - B_\tau(l_1^n, v) \quad \text{for all } v \in [H_0^1(I_j)]^r. \tag{4.7}$$

Denoting by $Q_j \subset [H_0^1(I_j)]^r$ the subspace consisting only of the function ϕ_j with components that are the quadratic bubble functions

$$\phi_{j,i} := -\frac{4}{(x_j - x_{j-1})^2}(x - x_{j-1})(x - x_j),\ x \in I_j,\ i = 1, \ldots, r,$$

the quadratic FEM approximation $\tilde{\varepsilon}_j$ of ε_j then reads

$$B_\tau(\tilde{\varepsilon}_j, \phi_j) = r_1(\phi_j) - B_\tau(l_1^n, \phi_j), \phi_j \in Q_j\ . \tag{4.8}$$

Note, that the realization of this estimation technique requires the solution of an $r \times r$ linear system for each finite element, which can be done numerically stable by a simple Gauss elimination. Finally, we end up with a reasonable local error estimator δ_j for the semi–implicit Euler solution (4.5) that reads

$$\delta_j := \left\| \frac{\tilde{\varepsilon}_j}{\gamma} \right\|_{0,w} .$$

The corresponding global error estimation is

$$\epsilon_x := \left(\sum_{j=0, j\neq l+1}^{n} \delta_j^2 \right)^{1/2} .$$

According to the above remarks, this error estimation is now used to improve the finite element solution u_1^n, until a given spatial tolerance tol_x is reached in the entire domain. This is achieved by the simple refinement strategy

$$\text{Refine } I_j \text{ if } \delta_j > cut := \frac{1}{n} \sum_{j=1, j\neq l+1}^{n} \delta_j .$$

More sophisticated strategies are available. The refinement process restarts from the coarsest mesh in each time step. Of course, this is not always necessary and modifications are possible.

We conclude this section by some remarks concerning the solution of the linear equations. In recent years clever direct sparse and band solvers have been developed as attractive methods for the solution of certain classes of sparse nonsymmetric linear systems arising from partial differential equations. They are very useful for solving problems with a sequence of right–hand sides supplied in succession such as in diagonally implicit Runge–Kutta methods. However, their application is limited to those general sparse systems in which fill–in is not a problem. If the factorization of the matrix tends to have many more none–zero entries than the original matrix, more storage and computational time is required. Especially, in the case of higher space dimensions these drawbacks are often encountered. Figure 4.1 shows the flow diagram for a direct solution of all the stage problems required for one time step of our algorithm. One observes that during the improvement of the spatial discretization only the first stage of the Rosenbrock method has to be solved. On the final grid all the stage values are computed using the same matrix factorization for three different right–hand sides.

An other possibility is the use of an iterative solver. A lot of iterative methods solving nonsymmetric systems of linear equations have been proposed. For a good comparative study see, e.g. [14]. Iterative methods are a practical alternative to direct methods whenever a good starting guess is available. This is the case for time–dependent problems, where the solution at each step can be used as starting value for the next step.

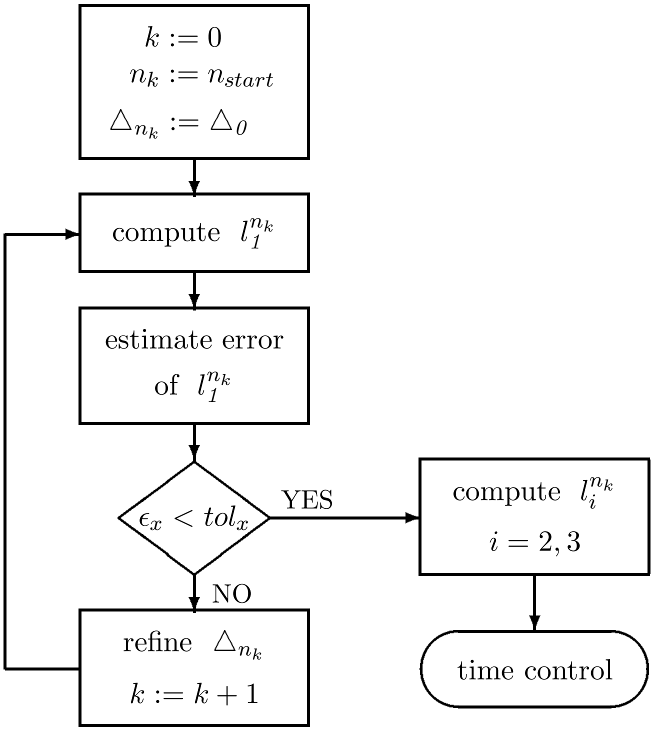

Fig. 4.1: Flow diagram for a direct solver

The rate of convergence of these methods can usually be speeded up by preconditioning or by the exploitation of a multilevel structure. In Figure 4.2 the flow diagram for a corresponding iterative solution process is pictured. On each refinement level all the stage problems are solved in order to get good starting values for the iteration on the final grid.

Now what is the best choice? Several comparisons between direct and iterative methods suggest that both can be applied with reliable success for the one–dimensional problems considered below. However, the question has to be discussed once again for higher space dimensions. In that case preconditioned iterative methods seem to have the potential for reducing dramatically the storage and computational time required by direct methods.

In the numerical examples presented below the BI–CGSTAB algorithm introduced in [20] was used. Its convergence behaviour is very smooth and in most cases it converges considerably faster than other methods. As stopping criterion a general threshold for the residuum is applied.

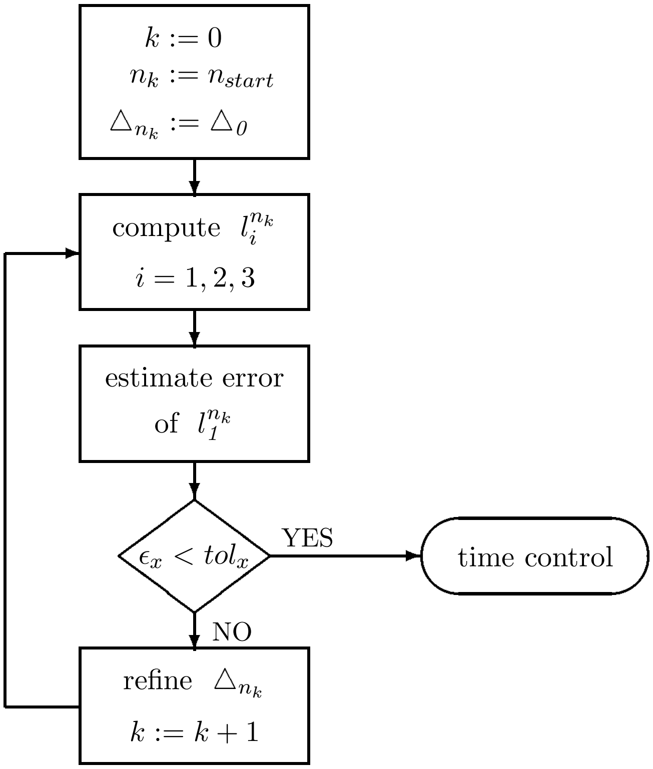

Fig. 4.2: Flow diagram for an iterative solver

Chapter 5

Numerical Results

In this section the capability of the described fully adaptive algorithm to solve real–life chemical problems is demonstrated. We shall test the performance on a class of solutions arising from reactions at phase boundaries. Obviously, these solutions have a high degree of spatial activity which ranges from steep moving wave fronts to emerging and dying layers.
All calculations were performed on a SPARC Workstation IPX.

Problem I: Reaction and transport by heterogeneous catalysis.
Catalysts are useful to generate porous media with transport and reaction pores. In such systems the predominantly diffusive transport of chemical substances is connected with adsorption/reaction processes. Figure 5.1 shows a part of a technically relevant honeycombed medium with four quadrilateral channels and the catalyst as wall.

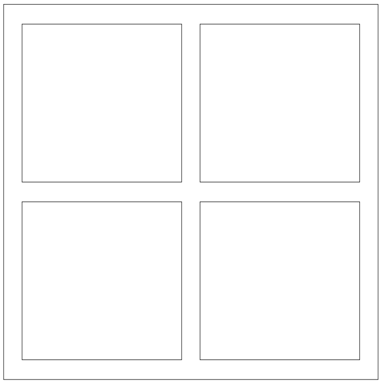

Figure 5.1: *Honeycomb system*

To get a first impression of the different phenomena an extremely simplified one–dimensional model with only one pore coordinate is often employed. Firstly, the adsorption/reaction mechanism in the porous catalyst is studied intensively, assuming a laminar streaming along the wall. This leads to a so–called 2–phase model pictured in Figure 5.2.
The according partial differential equations can be derived from the material balances. A typical 2–component system with $u = (u_1, u_2)$, including a special reaction term reads as follows:

Material balances in the channel:

$$c_1 \frac{\partial u_i}{\partial t} = -c_2 \beta_i \left(u_i - u_i(b^-) \right), \quad x \in (b, b+a), \quad t > 0.$$

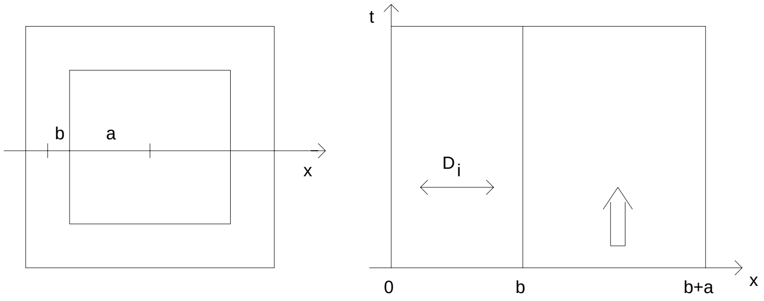

Figure 5.2: *2–phase model*

Material balances in the wall:

$$-D_i\frac{\partial^2 u_i}{\partial x^2} = -c_3\frac{u_2}{u_2+c_4}u_1\ , \quad x\in(0,b), \quad t>0.$$

Boundary and initial conditions:

$$\begin{aligned}\frac{\partial u_i}{\partial x}(0) &= 0,\\ u_i(b^-)-\frac{D_i}{\beta_i}\frac{\partial u_i}{\partial x}(b^-) &= u_i(b^+),\\ u_i(x,0) &= u_i^0.\end{aligned}$$

Here, D_i denotes the diffusion coefficient of the i–th component, β_i is its coupling coefficient, which has to be determined empirically. The coefficients c_i stand for special constants depending on the used chemicals. Since we start with a constant initial value for the domain $\Omega_2=(b,a+b)$, the solution of the first balance equation is again a constant. To set up this equation as part of a boundary value problem, the Cauchy boundary conditions

$$\frac{\partial u_i}{\partial x}(b^+) = \frac{\partial u_i}{\partial x}(a+b) = 0$$

will be added. Clearly, a fixed 3–point spatial discretization of a restricted Ω_2 can be used to approximate the chemical components in the channel exactly. We observe that the model consists of two partial systems – one reaction–diffusion system (wall) and one system described by ordinary differential equations (channel) –, which are connected through an internal coupling condition. Note that the above boundary conditions will generally lead to a solution which is discontinuous at $x = \xi_2$.

Figure 5.3 shows the temporal evolution of a typical solution and its corresponding spatial discretization for a real–life data set. At each time level an initial 9–point discretization of the computational domain $\Omega = (0.0, 0.0005) \cup (0.0005, 0.0006)$ was used. The tolerances were set to be $tol_x = 0.001$ and $tol_t = 0.05$.

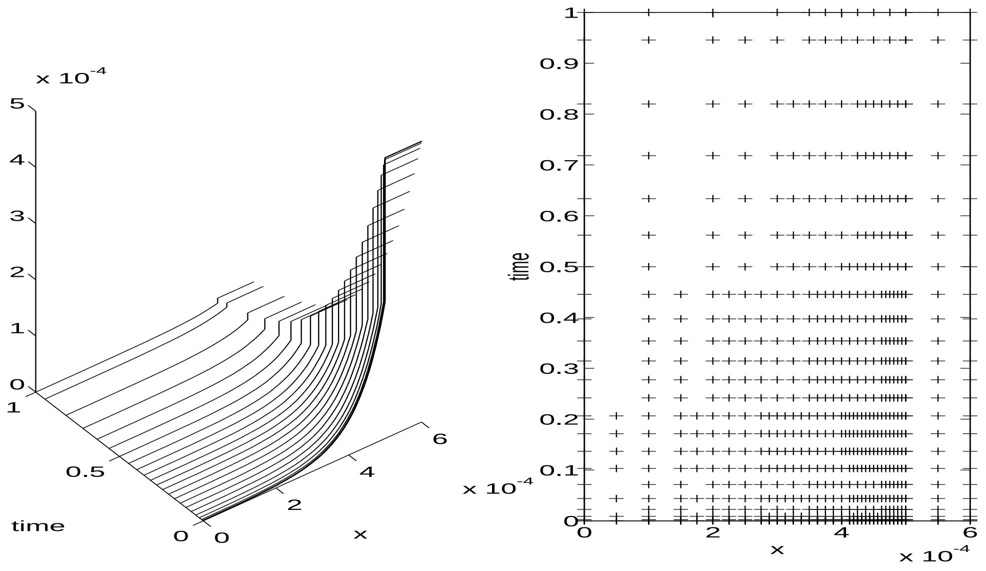

Figure 5.3: *2–phase model: Temporal evolution of the first chemical species and the grid*

The time–dependent internal boundary condition models a material transport, which generates an initial front. With decreasing concentration of the chemical species in the channel and continuing reaction process in the wall, the solution becomes smoother and smoother. The main point here is that the solution is guaranteed to be correctly resolved near the internal boundary layer. This is important since the coupling condition is based on the values in that region.

Next we consider the whole quadrilateral channel and replace it by a round one having the same hydraulic diameter. The model extended in such a way can be used to study the influence of the flow and diffusion in the channel on the reaction in the porous catalyst. In Figure 5.4 the new situation is shown. The corresponding equations of the cylinder–symmetric model look as follows:

Material balances in the channel:

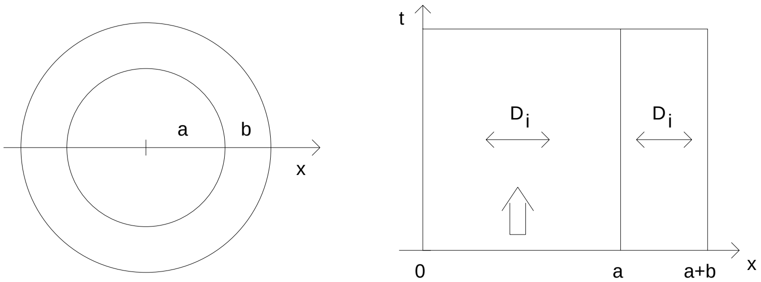

Figure 5.4: *Full model*

$$c_1 \frac{a^2 - x^2}{a^2} x \frac{\partial u_i}{\partial t} - \frac{\partial}{\partial x}\left(x D_i \frac{\partial u_i}{\partial x} \right) = 0, \quad x \in (0, a), t > 0.$$

Material balances in the wall:

$$-\frac{\partial}{\partial x}\left(x D_i \frac{\partial u_i}{\partial x} \right) = -c_2 x \frac{u_2}{u_2 + c_3} u_1 \;, \quad x \in (a, a+b), \quad t > 0.$$

Boundary and initial conditions:

$$\begin{aligned} \frac{\partial u_i}{\partial x}(0) &= \frac{\partial u_i}{\partial x}(a+b) \;=\; 0, \\ D_i \frac{\partial u_i}{\partial x}(a^-) &= D_i \frac{\partial u_i}{\partial x}(a^+), \\ u_i(a^+) &= u_i(a^-), \\ u_i(x,0) &= u_i^0 \;. \end{aligned}$$

The special coupling conditions of this second problem gives us the possibility to use a continuous version of our method due to results of the previous section. In Figure 5.5 the evolution of one component to the steady state is pictured for the same chemical system as used in the previous computation. In contrast to the first problem now the real extension of the channel has to be taken into account, that means $\Omega = (0.0, 0.0025) \cup (0.0025, 0.003)$. The tolerances were set to be $tol_t = 0.0005$ and $tol_x = 0.001$. After the formation of an internal boundary layer, the time step can be enlarged significantly. A uniform 13–point start grid provides a difficult test of the error estimator in space, since the method must rapidly refine near the internal boundary in order to resolve the fast reaction in the catalyst leading to a steep initial front. One can see that the maximum density of points exactly follows the front, ensuring good resolution and high economy.

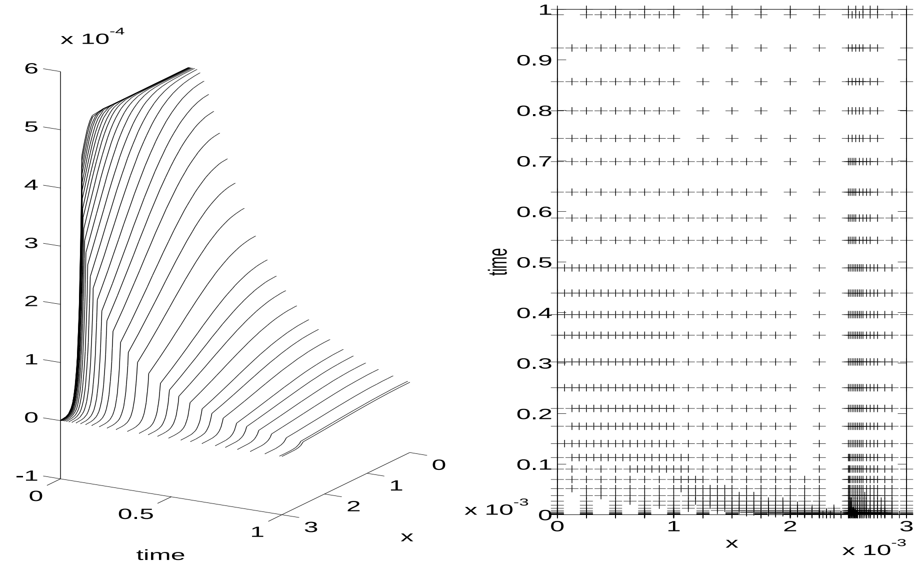

Figure 5.5: *Full model: Temporal evolution of the first chemical species and the grid*

Problem II: Vertical bubble reactor.
In contrast to the above class of problems characterized by a spatially independent phase boundary, gas–fluid–systems are generally more sophisticated. Here, the phase boundaries change their shape and size in time. Figure 5.6 shows a vertical and cylindrical bubble reactor in section.
Different gaseous chemicals stream in at the lower end of the reactor filled with a fluid. The bubbles rise to the top dissolving and reacting with each other. Clearly, with decreasing bubble diameter there is a permanent decrease of the exchange surfaces leading to certain nonlinearities in the corresponding models. To describe a synthesis process of two chemicals A and B in such a bubble reactor, we use a 2–film model shown in Figure 5.7.
In the first film F_1 the chemical A dissolves in the reactor fluid. Then it is very fast transported through the bulk, and reacts with the second chemical B in the other film F_2, see Figure 5.8. As a result new chemicals are produced in F_2, causing further reactions.
Let us consider as an example a special 7–component system where the reactor length is taken as time axis:

Diffusive process in F_1 only for the chemical A:

$$-D_1\frac{\partial^2 u_1}{\partial x^2} = 0, \quad x \in (\xi_1, \xi_2).$$

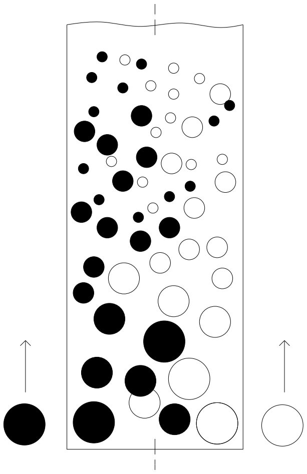

Figure 5.6: *Vertical bubble reactor in section*

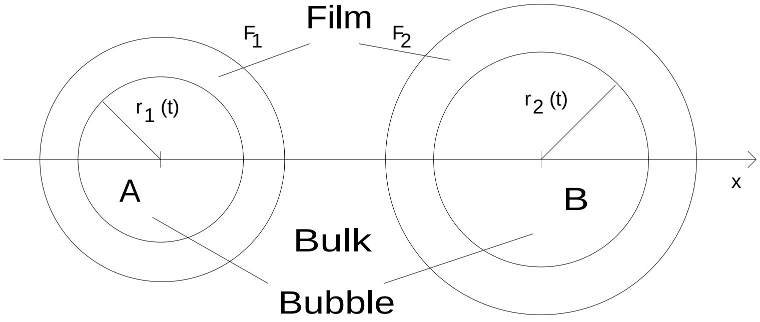

Figure 5.7: *2–film model*

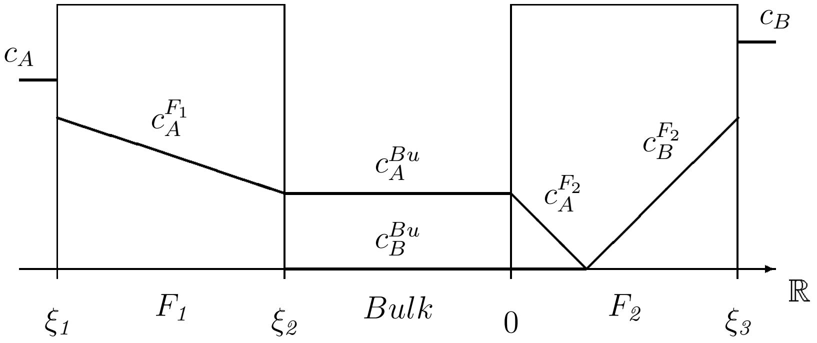

Figure 5.8: *Behaviour of the chemicals A and B on the computational domain*

Transport of all the chemicals through the bulk:

$$c_1 \frac{\partial u_i}{\partial t} = S_1(t) D_i \frac{\partial u_i}{\partial x}(\xi_2^-) + S_2(t) D_i \frac{\partial u_i}{\partial x}(0^+), \quad x \in (\xi_2, 0), \quad t > 0.$$

Reaction and diffusion in F_2:

$$-D_i \frac{\partial^2 u_i}{\partial x^2} = \sum_j k_{i,j} u_j u_i, \quad x \in (0, \xi_3), \quad t > 0.$$

Boundary and initial conditions:

$$\begin{array}{lrcl}
F1: & \frac{\beta_1}{\alpha_1 D_1} u_1(\xi_1) - \frac{\partial u_1}{\partial x}(\xi_1) & = & c_2 \frac{\beta_1}{D_1}, \\
 & u_1(\xi_2^-) & = & u_1(\xi_2^+), \\
Bulk: & u_1(\xi_2^+) & = & u_1(\xi_2^-), \\
 & \frac{\partial u_i}{\partial x}(\xi_2^+) & = & 0, \ i \neq 1, \\
 & u_i(0^-) & = & u_i(0^+), \\
F2: & u_i(0^+) & = & u_i(0^-), \\
 & \frac{\beta_2}{\alpha_2 D_2} u_2(\xi_3) + \frac{\partial u_2}{\partial x}(\xi_3) & = & c_3 \frac{\beta_2}{D_2}, \\
 & \frac{\partial u_i}{\partial x}(\xi_3) & = & 0, \ i \neq 2, \\
 & u_i(x,0) & = & u_i^0.
\end{array}$$

Here, D_i and β_i denote again the diffusion and the coupling coefficient of the i–th component. Additionally, α_i represents the Henry coefficient. The specific exchange areas S_1 and S_2 are strongly time–dependent because they are directly influenced by the bubble radius. These radii have to be computed in a complicated nonlinear process. A further description would go beyond our scope.

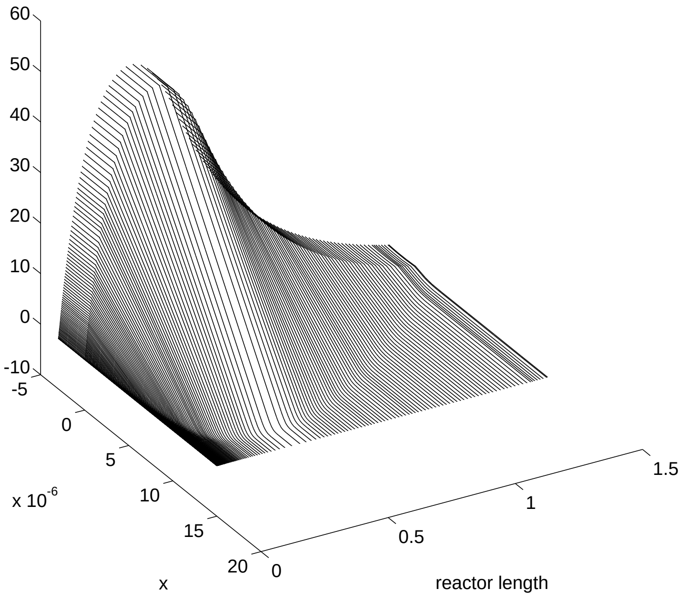

Figure 5.9: *Temporal evolution of the chemical component A*

Realizing the special reaction mechanism

$$\begin{aligned} A+B &\longrightarrow C+D, \\ C+D &\longrightarrow E, \\ C &\longrightarrow F+G, \end{aligned}$$

for a concrete numerical computation, the corresponding reaction coefficients $k_{i,j}$ can be derived by the mass action law.

The resulting system of partial differential equations has various internal boundary conditions. It is thus a challenging example for the proposed numerical algorithm.

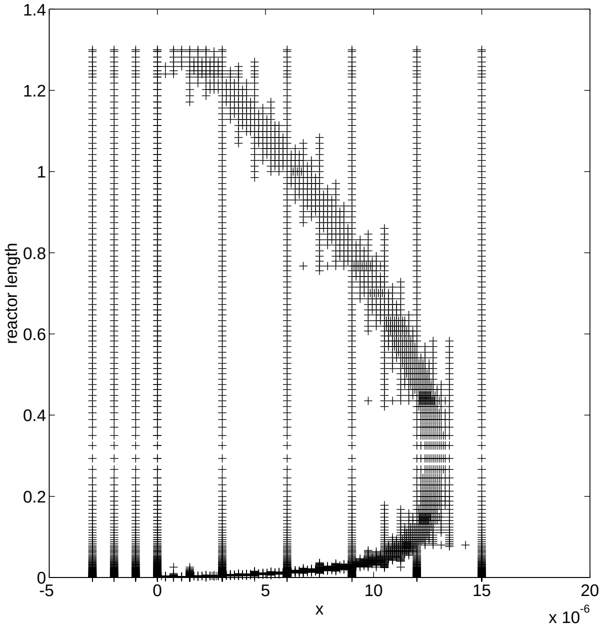

Figure 5.10: *Temporal grid–evolution of the bubble reactor*

In Figure 5.9 and Figure 5.10 the solution of the component A and its grid is pictured for the bulk $(-0.000005, 0.0)$ and for the film $F_2 = (0.0, 0.000015)$. The computation was done for the tolerances $tol_t = 0.001$ and $tol_x = 0.0008$. We observe that the grid is able to follow the dynamics of the problem. The speed of the reaction front varies drastically in time. The crucial point here is to get the correct speed in particular at the beginning of the reaction. In this problem the benefits of full adaptivity in time and space are evident.

Chapter 6

Conclusion

The aim of this paper was to develop a reliable and efficient method that can be used to solve reaction–diffusion problems with internal boundaries in a fully adaptive way. The spatial adaption is connected with an equidistribution of the estimated spatial error. The second adaptive feature is the use of variable time stepsize in the numerical integration. The steplength is determined in such a way that the local error per timestep is below some tolerance.

The method was applied to several problems involving steep travelling fronts and internal layers. In each example we have obtained high resolutions of the solution in both space and time. The adaptive selection of grid points led to drastic reduction of the number of points required by a uniform mesh. The method presented in this paper could also be applied to a number of other chemical problems.

Acknowledgements: This work has been carried out as part of the joint project “Development of adaptive multilevel methods for the reaction technology” between BASF and ZIB. I would like to thank Dr. W. Ruppel and Prof. P. Deuflhard, who encouraged me to do interdisciplinary work in Scientific Computing and supervised the above mentioned project. I also thank Dr. U. Nowak for supplying me with reference solutions obtained by his finite difference method of lines on the bubble reactor. He and Dr. J. Fröhlich read the entire manuscript at various stages and offered constructive suggestions.

Bibliography

[1] S. Adjerid, J.E. Flaherty : *A Moving Finite Element Method with Error Estimation and Refinement for One–Dimensional Time Dependent Partial Differential Equations.* SIAM J. Numer. Anal. **23**, pp. 778–796 (1986)

[2] I. Babuška, W.C. Rheinboldt : *Error Estimates for Adaptive Finite Element Computations.* SIAM J. Num. Anal. **15**, pp. 736–754 (1978)

[3] M. Bieterman, I. Babuška : *An Adaptive Method of Lines with Error Control for Parabolic Equations of the Reaction–Diffusion Type.* J. Comp. Phys. **63**, pp. 33–66 (1986)

[4] F.A. Bornemann : *An Adaptive Multilevel Approach to Parabolic Equations I. General Theory and 1D–Implementation.* IMPACT Comput. Sci. Engrg. **2**, pp. 279–317 (1990)

[5] P. Deuflhard, P. Leinen, H. Yserentant : *Concepts of an Adaptive Hierarchical Finite Element Code.* IMPACT Comput. Sci. Engrg. **1**, pp. 3–35 (1989)

[6] P. Deuflhard, U. Nowak : *Extrapolation Integrators for Quasilinear Implicit ODEs.* In: P. Deuflhard, B. Engquist (eds.): Large Scale Scientific Computing. Progress in Scientific Computing **7**, pp. 37–50. Birkhäuser (1987)

[7] P. Deuflhard, U. Nowak, M. Wulkow : *Recent Developments In Chemical Computing.* Computers Chem. Engrg. **14**, No. 11, pp. 1249–1258 (1990)

[8] E. Hairer, G. Wanner : *Solving Ordinary Differential Equations II, Stiff and Differential–Algebraic Problems.* Springer Series in Computational Mathematics 14. Springer, Berlin–Heidelberg–New York (1991)

[9] A.C. Hindmarsh : *LSODE and LSODI, Two New Initial Value Ordinary Differential Equation Solvers.* ACM SIGNUM Newsletter **15**, pp. 10 (1980)

[10] J. Lang : *KARDOS – KAskade Reaction Diffusion One–dimensional System.* Technical Report TR 93–9 (1993), Konrad–Zuse–Zentrum für Informationstechnik Berlin, Germany

[11] J. Lang, A. Walter : *A Finite Element Method Adaptive in Space and Time for Nonlinear Reaction–Diffusion Systems.* IMPACT Comput. Sci. Engrg. **4**, pp. 269–314 (1992)

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

[13] K. Miller, R.N. Miller : *Moving finite elements I.* SIAM J. Numer. Anal. **18**, pp. 1019–1032 (1981)

[14] N.M. Nachtigal, S.C. Reddy, L.N. Trefethen : *How fast are nonsymmetric matrix iterations?* SIAM J. Matrix Anal. Appl. **13**, No. 3, pp. 778–795 (1992)

[15] U. Nowak : *Adaptive Linienmethoden für nichtlineare parabolische Systeme in einer Raumdimension.* Dissertation, FU Berlin, Fachbereich Mathematik, Technical Report TR 93–14, Konrad–Zuse–Zentrum Berlin (ZIB) (1993)

[16] L.R. Petzold : *A Description of DASSL: a differential–algebraic system solver.* Proc. IMACS World Congress, Montreal, Canada (1982)

[17] M. Roche : *Rosenbrock Methods for Differential Algebraic Equations.* Numer. Math. **52**, pp. 45–63 (1988)

[18] E. Rothe : *Zweidimensionale parabolische Randwertaufgaben als Grenzfall eindimensionaler Randwertaufgaben.* Math. Ann. **102**, pp. 650–670 (1930)

[19] K. Strehmel, R. Weiner : *Linear–implizite Runge–Kutta–Methoden und ihre Anwendungen.* B.G. Teubner Verlagsgesellschaft Stuttgart–Leipzig (1992)

[20] 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. Stat. Comp. **13**, No. 2, pp. 631–644 (1992)

[21] P.A. Zegeling : *Moving–Finite–Element Solution of Time–Dependent Partial Differential Equations in Two Space Dimensions.* Report NM–R9206 (1992), Centrum voor Wiskunde en Informatica (Netherlands)