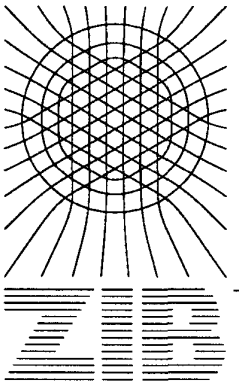
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Heilbronner Str. 10, D-1000 Berlin 31



P. Deuflhard * U. Nowak M. Wulkow

Recent Developments in Chemical Computing

* Invited speaker, European Symposium on Computer Applications in Chemical Industry, Erlangen, April 1989.

Preprint SC 90–7 (July 1990)

The authors wish to thank Erlinda Cadano-Körnig for her excellent TEX-typing of this manuscript.

Contents

Abstract

The paper surveys three aspects of chemical computing, which seem to play a role in recent developments. First, extrapolation methods for the numerical treatment of differential–algebraic equations are introduced. The associated extrapolation code LIMEX has reached a certain level of sophistication, which makes it a real competitor to the elsewhere widely used multi–step code DASSL of Petzold. Second, adaptive methods of lines for partial differential equations such as those arising in combustion problems are treated. Both static and dynamic regridding techniques are discussed in some detail. Finally, some new ideas about the treatment of the kinetic equations arising from polymer reactions are presented. The new feature of the suggested approach is the application of a Galerkin procedure using sets of orthogonal polynomials over a discrete variable (which, of course, in the case of polymer reactions is the polymer degree). The new approach may open the door to a new reliable low dimensional treatment of complex polymer reactions.

0. Introduction

Chemical computing ranges from the simulation of kinetics and polyreactions to the simulation of complete reactors. Numerical methods for these large scale problems have to be carefully adapted to the special situation. The present paper will survey three aspects of chemical computing, which seem to play an important role in recent developments.

The simulation of large reaction systems in chemistry requires not only the treatment of large and stiff systems of ordinary differential equations (ODE's) but also the solution of differential algebraic equations (DAE's). Section 1 will introduce *extrapolation* methods for DAE's, which have reached a high level of sophistication. The associated code LIMEX is a real competitor to the elsewhere widely used multi-step code DASSL. The code LIMEX has been successfully implemented within the simulation packages LARKIN and DIVA.

Extrapolation also plays an important role for combustion problems as described in Section 2. The partial differential equations (PDE's) arising there may be solved by means of a method of lines using *adaptive regridding* methods in space (including moving nodes). Impressive progress is documented in the simulation of chemically reacting flows in 2-D. In the given example, systems of up to 16000 DAE's were solved by LIMEX.

Standard kinetic packages fail, if problems from polymer chemistry have to be considered. As each polymer chain length appearing in a chain length distribution (CLD) leads to one ordinary differential equation, realistic problems lead to prohibitively large ODE systems. In Section 3, a new approach is presented, which allows the treatment of polyreactions up to any polymer degree. The method uses a Galerkin approximation of the CLD by means of orthogonal polynomials of a *discrete* variable, which represents the polymer degree in the present context. A parameter appearing in the orthogonal polynomials can be adapted in such a way, that the method has similarities with a method of lines for PDE's (with moving basis functions instead of moving nodes). An important feature is that a cheap and reliable error estimation is available.

Throughout the paper real life examples are included to illustrate the efficiency of the described methods.

1. Extrapolation Methods for Implicit and Differential–Algebraic Equations

Many problems of computational chemistry arise as implicit systems of differential–algebraic equations (DAE's) in the form:

$$B(y)y' = f(y) \tag{1.1}$$

Such systems are treated in the recent textbooks by BRENAN/CAMPBELL/PETZOLD [2] and HAIRER/LUBICH/ROCHE [11]. Whenever $B(y)$ is *nonsingular*, then the above system is said to be of *index* $= 0$. Such a system is formally equivalent to the standard system of ordinary differential equations (ODE's):

$$y' = B(y)^{-1}f(y) =: g(y) \; . \tag{1.2}$$

However, efficient numerical algorithms will certainly treat (1.1) directly. If $B(y)$ is *singular*, then (1.1) may nevertheless have a *unique* solution — a situation, which is characterized to be of *index* $= 1$. For *index* > 1, solutions may still exist. But, in order to assure uniqueness, further equations are to be added, since such systems are *algebraically incomplete* (see RHEINBOLDT [23]). In special cases, systems with *index* > 1 can be directly treated as well. For general systems, however, *index* > 1 means that the problem is at least *ill–posed.* Hence, as a general advise, any systems arising with *index* > 1 should be preprocessed in such a way that they eventually have *index* $= 0$ or *index* $= 1$. For this reason, the present chapter is restricted to just these two cases. In some problems, system (1.1) arises in the special *separate* form

$$\begin{aligned} &\text{a)} \quad u' = f_1(u,v) \\ &\text{b)} \quad 0 = f_2(u,v) \; . \end{aligned} \tag{1.3}$$

Here *index* $= 1$ means that the algebraic condition $f_2 = 0$ can be solved for v — which, in turn, requires that the derivative matrix

$$C := \frac{\partial f_2}{\partial v} \quad \text{is nonsingular} \; . \tag{1.4}$$

An explicit numerical solution for v (say, in terms of a Newton iteration), however, is only appropriate, if the arising differential system

$$u' = f_1(u, v(u)) \tag{1.5}$$

is *non-stiff* — for a further discussion see e.g. DEUFLHARD/NOWAK [7].

For both *index* = 0 and *index* = 1, the following extension of the *semi-implicit Euler discretization* (see DEUFLHARD [5], DEUFLHARD/HAIRER/ZUGCK [6], DEUFLHARD/NOWAK [8]) has turned out to be extremely useful ($k = 0, 1, \ldots, B_k := B(y_k)$):

$$\begin{aligned} &\text{a)} \quad (B_k - h\ A)\Delta y_k = h\ f(y_k) \\ &\text{b)} \quad y_{k+1} = y_k + \Delta y_k \end{aligned} \tag{1.6}$$

Herein (1.6.a) is a linear system to be solved — typically large and sparse — and the matrix A may be defined either as

$$A := \frac{\partial}{\partial y}(f(y) - B(y)y')|_{y=y_0} \tag{1.7}$$

or as an approximation of this expression, which should represent any strong couplings of the dynamic system (1.1). For $A = 0$, (1.6) is just an *explicit Euler discretization* for system (1.2) — which restricts its applicability to *index* = 0. If, for separate systems (1.3), the matrix A only contains C from (1.4), then the discretization (1.6) — with $B = I$ — is equivalent to an explicit Euler discretization of (1.5) in combination with only one Newton iteration per discretization step to take care of the algebraic conditions (1.3.b). Generally speaking, a sophisticated choice of A may lead to significant reductions of both computing times and storage requirements for a given problem at hand. This additional feature is a clear advantage of the more recent *semi-implicit* discretizations over any *implicit* discretizations that are in quite common use — see [2].

The above discretization (1.6) has been shown to possess a so-called *perturbed asymptotic expansion* — see DEUFLHARD/HAIRER/ZUGCK [6] for the case $B(y)$ = const and LUBICH [15] for the general case. This expansion permits the combination of the extended semi-implicit Euler discretization with *extrapolation* in powers of the stepsize h. As a consequence, well-known techniques of order and stepsize control (cf. [5]) apply with minor modifications. Details of the algorithm are described in [8]. The associated code name LIMEX is a mnemotechnic abbreviation of **L**inearly **IM**plicit **EX**trapolation integrator. A variant of the code, LIMEXS, has been implemented for the case when the arising linear systems (1.6.a) are large and **S**parse. The new extrapolation codes have appeared to be highly robust and efficient in a number of important applications. Rather than going into further algorithmic detail, a few examples from computational chemistry are given.

Large reaction kinetics. The numerical simulation of large chemical reaction systems requires the integration of large ODE systems, which are typically nonlinear and stiff and have a sparse Jacobian matrix — for an extensive treatment of the subject see DEUFLHARD/NOWAK [7] and references therein. The situation treated there is characterized by the kinetic equations

$$c_i' = w_i(c,T) \quad i = 1,\ldots,n \tag{1.8}$$

for the species concentrations c_i and for a *given temperature* $T(t)$. A convenient package to solve such systems is the code LARKIN — mnemotechnically for **LAR**ge chemical **KIN**etics. In quite a number of cases, however, the thermodynamic equations for pressure p, density ρ, and temperature T must be added. These cases have been treated by WALKOWIAK [24].

Assume first that the *density* is explicitly given, say, in the form

$$\text{a)} \quad \frac{\rho'(t)}{\rho(t)} = h_\rho(t)\,. \tag{1.9}$$

Then, with the additional notation $(i = 1,\ldots,n)$

$$\begin{aligned} H_i &: \text{species enthalpy} \\ R &: \text{universal gas constant,} \\ C_{pi} &: \text{specific heat coefficient,} \end{aligned}$$

one ends up with the extended system of $(n+1)$ ODE's

$$\text{b)} \quad c_i' = w_i(c,T) + c_i h_\rho(t)$$

$$\text{c)} \quad T' = \left[\sum_{i=1}^{n}(H_i - RT)w_i(c,T) - h_\rho(t)p\right] \Big/ \left[\sum_{i=1}^{n}(R - C_{pi})c_i\right]$$

$$\text{d)} \quad p = RT\sum_{i=1}^{n} c_i$$

The thus defined system is of standard form and can be treated by a slight modification of LARKIN.

Next, assume that the *pressure* is given in the form

$$\text{a)}\quad p'(t) = h_p(t)\,, \quad p(0) = p_0\,. \tag{1.10}$$

This leads to a system of $(n+2)$ *explicit* ODE's

$$\text{b)}\quad c_i' = w_i(c,T) + c_i \cdot \frac{\rho'}{\rho}$$

$$\text{c)}\quad T' = \left[h_p(t) - \sum_{i=1}^{n} H_i w_i(c,T)\right] \Big/ \sum_{i=1}^{n} c_{pi} c_i$$

$$\text{d)}\quad \rho' = \rho\left[\frac{p'(t)}{p(t)} - \frac{R}{T}\sum_{i=1}^{n} w_i(c,T) - \frac{T'}{T}\right]$$

If one inserts the expression for ρ'/ρ from (1.10.d) into (1.10.b) — as suggested in the chemical standard literature (cf. [10]) — then the associated Jacobian matrix is *full*. As a consequence, computing times needed by any stiff integrator would blow up for sufficiently complex systems! The remedy suggested in [24] is to treat (1.10.b) as *implicit* ODE

$$\text{e)}\quad c_i' - c_i\frac{\rho'}{\rho} = w_i(c,T)$$

and to insert the expression for T' from (1.10.c) into (1.10.d). The thus obtained partly implicit system of $(n+2)$ ODE's then typically leads to a *sparse* linear system (1.6). Moreover, since the arising system has *index* $= 0$, one may formally apply a semi-implicit Euler discretization to system (1.2), which leads to

$$y_{k+1} = y_k + h(B_0 - hA)^{-1} B_0 B_k^{-1} f(y_k) \tag{1.11}$$

wherein $y := (c_1, \ldots, c_n, T, \rho)$ and A from (1.7). Upon exploiting the special structure of (1.10), one ends up with the computationally simple scheme

$$\text{a)}\quad z_k := B_0 B_k^{-1} f(y_k) = \begin{pmatrix} w_i + (B_0 - B_k)_{i,n+2} \\ T' \\ \rho' \end{pmatrix} \tag{1.12}$$

$$\text{b)}\quad (B_0 - hA)\Delta y_k = h z_k$$

$$\text{c)}\quad y_{k+1} = y_k + \Delta y_k$$

The sparse pattern of $B_0 - hA$ inherits the sparse pattern of f_y plus certain entries in the diagonal and the last column. On this algorithmic basis, an efficient simulation of rather complex chemical systems on comparatively small computers is possible. An implementation of the algorithm can be found in the latest revision of LARKIN.

Dynamic process simulation. The above package LIMEXS has been implemented and adapted as a *core routine* within the process simulation package DIVA — compare [12] and references therein. For illustration purposes, consider the two coupled distillation columns represented schematically in Fig. 1.1. The associated mathematical model comprises $n = 561$ differential-algebraic equations (112 differential equations, 449 algebraic equations). The Jacobian matrix has 4020 nonzero entries — which means a sparseness of roughly 1 percent. Within the operator training unit of DIVA, a real sampling time of 4 sec. led to an acceleration factor 2–10 of the simulation over the real time process [14]. In this special problem class, a structural advantage of LIMEX over multi-step codes like DASSL [21] plays an important role: after small discontinuities there is no need to compute new consistent initial conditions, which leads to a rather smooth course of integration.

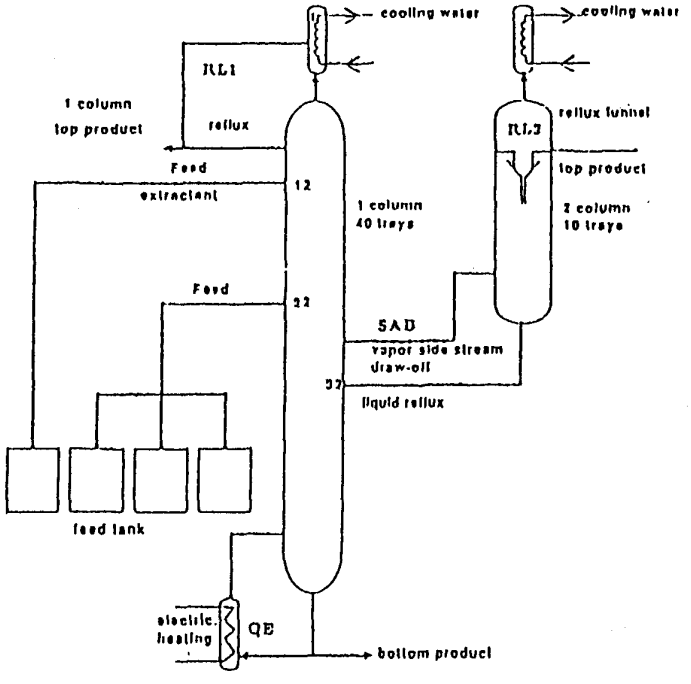


Figure 1.1: Coupled Distillation Columns.

2. Adaptive Methods of Lines for Partial Differential Equations

One rather popular approach to the numerical solution of partial differential equations (PDE's) is the method of lines (MOL). It involves discretization with respect to space variables first — thus generating a *system of ODE's*, which is typically *large and stiff*. The linear systems arising in the course of stiff integration (with respect to the time variable) exhibit a typical block structure. For PDE systems in one space variable only, these linear systems are usually solved by a *direct* method such as a block tridiagonal solver or a band solver. For PDE's in more than one space variable, special *iterative* linear solvers appear to be the exclusive choice. In frequent cases, *algebraic conditions* enter into the problem: examples of such an occurrence are 1–D formulations based on spherical or cylindrical coordinates or incompressible flow problems. In addition, part of the ODE's may be implicit. Apart from any specifications of the applied linear solver, the described MOL approach directly leads to the problem class treated in the preceding Section 1.

In the above approach, the only *adaptive* algorithmic device is the order and stepsize control with respect to time discretization. This turns out to be not enough for a class of challenging problems in chemical engineering (such as combustion problems), which additionally requires a time–dependent adaptation of the space discretization — a so–called *adaptive regridding*. Two principal types of adaptive regridding are in quite common use: the *static* and the *dynamic* regridding, which are now described in some detail.

Static regridding. Assume that within a given time layer the solution is represented as a discrete solution over a given space grid. In this situation one may aim at an *equidistribution of the spatial discretization error*. This error is typically not precisely known and can in most cases only be estimated. Any such error estimation technique can then be used as a heuristic to construct a new space grid, which hopefully equidistributes at least the estimated spatial error. Two typical options are in use:

(a) Given a *fixed* number N of nodes, generate a new grid with also exactly N nodes.

(b) Given a spatial error estimation technique, generate a new grid such that the estimated error is below a given threshold value — which means a *variable* number of nodes.

Static regridding can already be performed at the beginning (interpolation error in comparison with given initial solution) and repeated after an appropriate number of time steps — possibly after each time step. It is clear that any such regridding introduces an element of *discontinuity* into the ODE formulation. Part of this discontinuous effect can be damped by use of *(discrete) Sobolev norms* to be applied in the order and stepsize control of the ODE solver. The basic structure of this kind of adaptive MOL, however, gives a *clear advantage to any one-step ODE solver including extrapolation methods* — as opposed to the multi-step ODE solvers that, at present, are most widely distributed [2]. Rather than further principal elaboration of the topic, an illustrative example is given now.

Example: Chemically Reacting Flows in 2–D (MAAS/WARNATZ [16])
In [16] an adaptive MOL package is presented to solve the compressible Navier–Stokes equations including detailed chemistry and a multi-step transport model. The basic equations are:

$$
\begin{aligned}
&\text{a) Continuity} && \\
&\qquad \frac{\partial \rho}{\partial t} + \operatorname{div}(\rho\, \mathbf{v}) = 0 && \\
&\text{b) Species mass} && \\
&\qquad \rho \frac{\partial w_1}{\partial t} + \rho\, \mathbf{v} \operatorname{grad} w_i + \operatorname{div} \mathbf{j}_i = \dot{\omega}_i M_i && \\
&\text{c) Momentum} && (2.1)\\
&\qquad \frac{\partial \rho\, \mathbf{v}}{\partial t} + \operatorname{grad} P + \operatorname{div}(\rho\, \mathbf{v} \circ \mathbf{v}) = 0 && \\
&\text{d) Energy} && \\
&\qquad \frac{\partial \rho h}{\partial t} - \frac{\partial P}{\partial t} + \operatorname{div}(\rho\, \mathbf{v}\, h) - \mathbf{v} \operatorname{grad} p && \\
&\qquad\quad + \operatorname{div} \mathbf{j}_q + \Pi : \operatorname{grad} \mathbf{v} = \dot{q} &&
\end{aligned}
$$

with P = pressure, T = temperature, n_s = number of species, w_i = mass fraction of species i, M_i = molar mass of species i, $\dot{\omega}_i$ = molar scale rate of formation of species i, h = specific enthalpy, ρ = density, $\mathbf{v}$ = velocity, $\mathbf{j}_q$ = heat flux, $\mathbf{j}_i$ = diffusion flux of species i, Π = stretch tensor, $\dot{q}$ = source term for deposition of energy, and t = time.

Restricting the problem to simple two-dimensional geometries, (2.1) is simplified to a system of PDE's with the independent variables t, r, z (time and two space variables) and the dependent variables ρ, v_r, v_z, T, P, w_i (density, velocities, temperature, pressure, species mass fraction). The final set of equations

consists of 1 algebraic equation and $n_s + 4$ implicit ODE's ($n_s :=$ number of species). Within the MOL treatment, the spatial discretization is done by finite difference approximations on a non-uniform tensor product grid — for an illustration see Fig. 2.1.

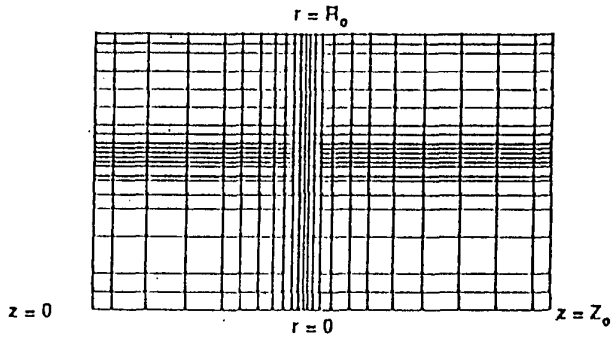


Figure 2.1: Adaptive tensor product grid.

The resulting large system of ODE's is solved by the semi-implicit extrapolation integrator LIMEX in combination with a special iterative linear system solver. To resolve the fronts in the solution profiles, a static regridding procedure of type (a) is applied after each time step. The above mentioned advantage of a one-step method within the adaptive MOL approach shows up clearly. By using the extrapolation code LIMEX instead of the multi-step code DASSL [21], 1-D simulations could be accelerated up to factors of 10–15 [17]. In this situation, comparisons for the 2-D case were not performed. The final MOL package consists of about 30.000 lines of FORTRAN-code. It has been written by two chemical engineers (U. Maas and J. Warnatz) in close and steady cooperation with two of the present authors (P. Deuflhard and U. Nowak). Now, after this fruitful cooperation over years, the package is able to perform internationally outstanding simulations.

As an example, take the simulation of the *thermal ignition of an ozone-oxygen mixture*, a model of 6 elementary reactions for 3 species. Thus one has a system of $n_{\mathrm{pde}} = 8$ PDE's. Due to the adaptive regridding procedure, a comparatively small mesh of size $n_r = 50$, $n_z = 40$ turned out to be sufficient. Nevertheless, one ends up with a system of

$$\bar{n} = n_{\mathrm{pde}} \cdot n_r \cdot n_z = 16.000$$

DAE's to be solved. The simulation of the total chemical process under consideration took about $10h$ CPU-time on an IBM 3090.

Dynamic regridding. The above static regridding means that the space grids are kept fixed during each time discretization step — a technique, which may lead to drastic order and time step restrictions in the presence of fast moving fronts. Moreover, adaptive regridding *after* each time step may come "too late". The effect is especially significant for ODE solvers like extrapolation methods, which would allow "rather large" time steps. This insight leads one to the construction of *moving grid techniques* — sometimes also called *dynamic regridding.* The basic idea is rather simple: one introduces *time dependent spatial nodes* $x(t)$, which, in turn, induce the replacement

$$u(x,t) \;\rightarrow\; \overline{u}(x(t),t) \tag{2.2}$$

for the PDE solution. As a consequence, any time derivative u_t changes according to

$$u_t \;\rightarrow\; \overline{u}_t = u_t + u_x x_t \; . \tag{2.3}$$

Assume that the fixed node MOL had produced some ODE system

$$U' = F(U) \; , \tag{2.4}$$

then the introduction of moving nodes gives rise to some ODE system

$$U' = F(U) + G(U)X' \; . \tag{2.5.a}$$

Note that this ODE system is incomplete, since X' must be determined by some further consideration. The most popular principle is to find X' such that

$$\|U'\|^2 + \alpha\|X'\|^2 = \min \; , \tag{2.5.b}$$

wherein some tuning parameter α needs to be carefully selected. This minimization principle may lead to an explicit expression for X' or to the lacking ODE's for X', which then are *implicit* — see e.g. HYMAN [13] and MILLER/MILLER [18], [19]. In order to make the method really working, one has to avoid the occurrence of *node intersection* — which can be done by adding penalty terms to the objective function in (2.5.b). This introduces at least one further tuning parameter. The task of tuning these parameters may require a considerable amount of interaction for each individual PDE problem.

In principle, one might think of transferring the equidistribution of spatial discretization error, which underlies the above static regridding, to the case of dynamic regridding as well. However, careful examination of this idea leads to the insight that the principle of static regridding cannot be made precise enough to design an efficient and robust dynamic regridding. For this reason, difficult real life problems may require a mix of both strategies.

Example: Moving flame front (PETERS/WARNATZ [20])
The selected problem is test problem A, case 4, from [20], pp. 3– 6. The PDE system reads (T = temperature, Y = chem. species):

$$
\begin{aligned}
&\text{a)}\quad T_t = T_{xx} + R(T,Y) \\
&\qquad Y_t = \frac{1}{\mathrm{Le}} Y_{xx} - R(T,Y) \\
&\qquad R(T,Y) := \frac{\beta^2}{2\,\mathrm{Le}} Y \exp\left[-\frac{\beta(1-T)}{1-\alpha(1-T)}\right] \\
&\qquad \alpha = 0.8\,, \quad \beta = 20\,, \quad \mathrm{Le} = 2 \\
&\text{b)}\quad T_x(\infty) = 1\,, \quad Y_x(\infty) = 0 \\
&\qquad T(-\infty) = 0\,, \quad Y(-\infty) = 1 \\
&\text{c)}\quad t = 0\,, \quad x \le 0 : T = \exp(x) \quad Y = 1 - \exp(\mathrm{Le}\cdot x)\,, \\
&\qquad t = 0\,, \quad x > 0 : T = 1\,, \quad Y = 0\,.
\end{aligned}
\tag{2.6}
$$

The challenge of this problem is the fact that the moving flame front varies its shape and velocity in the course of integration. To solve (2.6), the time–integrator LIMEX is combined with a space discretization by finite differences in the domain [-25, 10]. To adapt the grid during the time integration, a static regridding procedure of type (b) is applied and additionally a dynamic regridding technique is used by introducing a moving grid. Thus, the original system (2.6.a) is transformed to a system of type (2.5.a). To derive equations for X', a special minimization principle is introduced as

$$\|U'\|^2 + \tau\|\Delta X'\| = \min\,, \tag{2.7}$$

where $\Delta X'$ denotes the difference of the velocities of two neighboring nodes. The resulting system to be solved by LIMEX consists of a coupled and implicit ODE system for the dependent variables T_i, Y_i, X_i, $i = 1,\ldots,nx$ (nx: number of nodes). Fig. 2.2 illustrates the strategy mix between static and dynamic regridding for an integration from $t_0 = 0$ to $t_{\mathrm{end}} = 12$. Note that the introduction of moving nodes also simplifies the task of static regridding and allows larger time steps. Moreover, only adaptive methods have a chance to resolve the velocity peak shown in Fig. 2.3. This explains why different authors, having attacked the same problem with non–adaptive methods, produced rather different results [20]. The adaptivity of the mesh may be reflected by the space stepsize ratio

$$\frac{\Delta X_{\max}}{\Delta X_{\min}} \sim 10^4\,. \tag{2.8}$$

These results certainly encourage further developments of *fully* adaptive methods for PDE's.

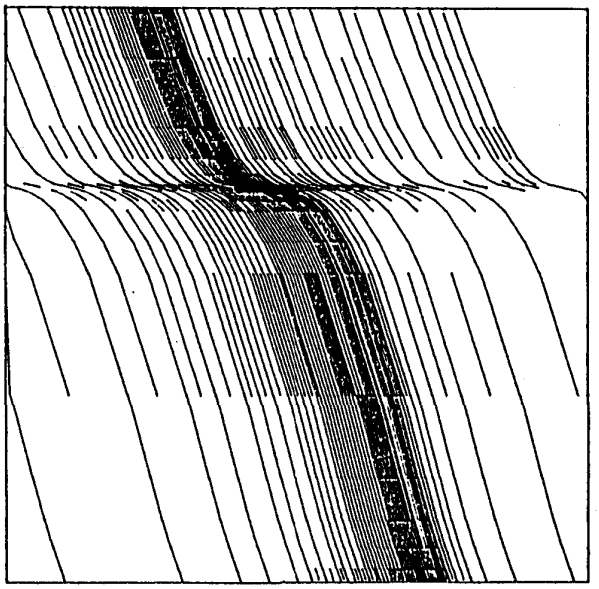


Figure 2.2: Nodal flux in an (x,t)–plane.

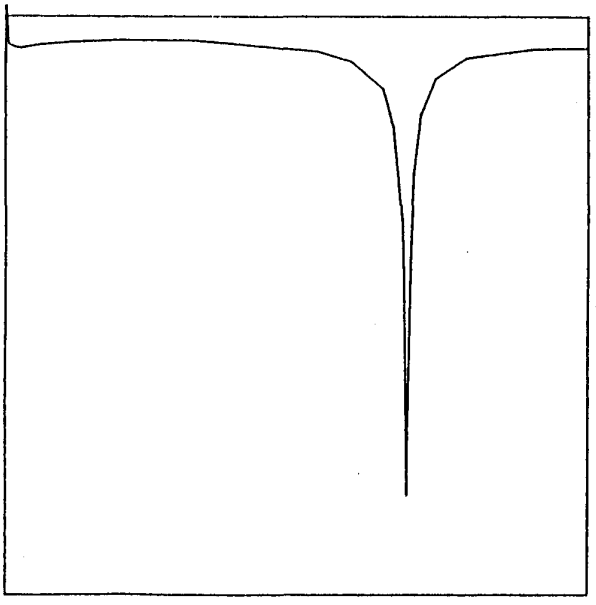


Figure 2.3: Velocity of flame front versus time t.

3. Discrete Galerkin Method for Polyreaction Kinetics

The *discrete Galerkin method* for the computation of the chain length distribution (CLD) arising from polyreaction systems has been suggested first by DEUFLHARD and WULKOW [9]. On the basis of a weight function, which is connected with the type of the chemical process to be modeled, orthogonal polynomials of a discrete variable are constructed. Insertion of an expansion of the CLD by these polynomials into the kinetic equations (the so-called *analytical preprocessing*) leads to differential equations for the expansion coefficients. As it turns out, comparatively few coefficients are needed for a good approximation of the CLD. As an important aspect of the new method, the approximation error can be controlled by an error estimation. Details of the method and many references can be found in [9], [4] and [25].

Basic Approximation Scheme. One assumes that the chain length distribution $P_s(t)$ of a polymer at time t can be expanded in a series

$$P_s(t) = \Psi(s;\rho)\sum_{k=0}^{\infty} a_k(t;\rho)\, l_k(s;\rho)\ ,\ s = 1\ ,\ 2\ ,\ \dots\ , \tag{3.1}$$

where $\Psi(s;\rho)$ is a positive weight function with a (possibly time-dependent) real parameter ρ and $\{l_k(s)\}_{k=0,1,\dots}$ a set of polynomials of a *discrete* variable s, which represents the *polymer degree*. The polynomials are associated with $\Psi(s;\rho)$ by the *orthogonality* relation

$$\sum_{s=1}^{\infty} l_j(s;\rho)\, l_k(s;\rho)\, \Psi(s;\rho) = \gamma_k\, \delta_{jk}\ ,\quad \gamma_k > 0\ ,\qquad j,\, k = 0,\, 1,\, 2,\, \dots\ , \tag{3.2}$$

with δ_{jk} the Kronecker symbol. By use of the orthogonality of the polynomials the coefficients $a_j(t;\rho)$ in (3.1) can be obtained from

$$a_j(t;\rho) = \frac{1}{\gamma_j}\sum_{s=1}^{\infty} P_s(t) l_j(s;\rho)\ ,\quad j = 0, 1, \dots\ . \tag{3.3}$$

Truncation of the expansion (3.1) after $n+1$ terms leads to a *Galerkin approximation*

$$P_{s,\rho}^{(n)}(t) := \Psi(s;\rho)\sum_{k=0}^{n} a_k(t;\rho)\, l_k(s;\rho)\ , \tag{3.4}$$

depending on the parameter ρ and the *truncation index* n. As pointed out in [9] the *truncation error* $\epsilon_n(t)$ of the approximation (3.4), can be efficiently estimated by

$$\epsilon_n(t)^2 = a_{n+1}^2(t)\,\gamma_{n+1}\ . \tag{3.5}$$

It is easily seen that a good choice of the *type* of the weight function Ψ in general and of the *parameter* ρ in particular will help to keep the truncation index n small. The "closer" $\Psi(s;\rho)$ is to $P_s(t)$ the "better" the approximation will be for *small* n. Note that this is not a fitting of a given weight function by its parameters! Such an approach would lead to a distribution of the same type in any case, whereas a Galerkin approximation $P_s^{(n)}(t)$ must not be close to $\Psi(s;\rho)$ in general. For the time being, the choice of Ψ still needs a rough a-priori insight into the chemical process, but the selection of ρ can be done adaptively on the basis of the following requirements:

$$\begin{aligned} &\text{a)}\quad \nu_0(\rho) := \sum_{s=1}^{\infty} \Psi(s;\rho) \;=\; 1\ , \\ &\text{b)}\quad \nu_1(\rho) := \sum_{s=1}^{\infty} s\,\Psi(s;\rho) \;=\; \frac{\mu_1(t)}{\mu_0(t)}\ , \end{aligned} \tag{3.6}$$

with $\mu_0(t)$, $\mu_1(t)$ the statistical moments of $P_s(t)$ at a fixed time t. The normalization (3.6.a) ensures that $\Psi(s;\rho)$ has an interpretation as a probability distribution, condition (3.6.b) gives an implicit definition of $\rho = \rho(t)$ identifying the mean values of $\Psi(s;\rho)$ and $P_s(t)$ and aiming at certain similarities between these distributions. In view of the concept of moving nodes in the PDE context (see Section 2), $\Psi(s;\rho(t))$ is called a *moving weight function.* The relations (3.6) directly imply

$$\begin{aligned} &\text{a)}\quad a_0(t) \;\equiv\; \mu_0(t) \\ &\text{b)}\quad a_1(t;\rho) \;\equiv\; 0 \end{aligned} \tag{3.7}$$

Orthogonal Polynomials of a Discrete Variable. It is well-known that many polymer reactions lead to molecular weight distributions which are similar (in some sense) to the so–called *Schulz-Flory* distribution. For this reason the specialization

$$\Psi(s;\rho) := (1-\rho)\rho^{s-1}\ ,\ 0<\rho<1\ ,\ s=1,2,\ldots\ , \tag{3.8}$$

is made here to illustrate the method (for further choices see [9] and [25]). The orthogonal polynomials associated with $\Psi(s;\rho)$ by means of (3.2) are the

discrete Laguerre polynomials. They are explicitly known by their *three-term recurrence relation.* For the analytical preprocessing of a polyreaction system some properties of discrete Laguerre polynomials are needed, which are listed in [4]. As an example, the treatment of the *propagation step* in a free radical polymerization system requires the relation

$$l_k(s+1;\rho) - l_k(s;\rho) = (\rho-1)\sum_{\nu=0}^{k-1}\rho^{k-1-\nu} l_\nu(s;\rho)\ . \tag{3.9}$$

In [25] *modified discrete Laguerre polynomials* have been introduced, which allow the simulation of certain *heterogeneous* reactions. The essential trick there is the replacement of *fractional* powers (arising from the modeling of surface effects) by *factorial* powers

$$\frac{1}{s^\alpha} \longrightarrow \frac{1}{(s-1)^{(\alpha)}} := \frac{\Gamma(s-\alpha)}{\Gamma(s)}\ , \tag{3.10}$$

in terms of the classical Gamma-function.

Example: Free radical polymerization [4]. The kinetic scheme of free radical polymerization in homogeneous systems is often described by the system [22]

a) Initiation :

$$\begin{aligned} I &\xrightarrow{k_d} 2R \\ R + M &\xrightarrow{k_i} P_1 \end{aligned}$$

b) Propagation :

$$P_s + M \xrightarrow{k_p} P_{s+1}$$

c) Chain Transfer : (3.11)

$$\begin{aligned} P_s + S &\xrightarrow{k_{f_s}} D_s + P_1 \\ P_s + M &\xrightarrow{k_f} D_s + P_1 \end{aligned}$$

d) Termination :

$$\begin{aligned} P_s + P_r &\xrightarrow{k_{t_c}} D_{s+r} \\ P_s + P_r &\xrightarrow{k_{t_d}} D_s + D_r \end{aligned}$$

with I initiator, R initiated radicals, P_s radical polymer of length s, D_s dead polymer of length s, M monomer and S solvent. The modeling of gel effect

and volume contraction has been included due to [3]. The resulting system of differential equations describing the CLD's of radical and dead polymer has been treated by the discrete Galerkin method based on the Schulz-Flory distribution in this case. Analytical preprocessing leads to differential equations for the expansion coefficients $a_k(t)$ from (3.1) , which have been solved by the nonstiff and stiff extrapolation codes DIFEX1 and EULSIM [5].

The interesting maximum polymer degree $s_{\max}$ in this process is around 10^4, so one has to solve a system of double size for the complete model. By means of the Galerkin method this number could be reduced to only 10 – 15 equations for a solution with technical accuracy (10^{-3}) and controlled error. Moreover the representation of the solution by the Galerkin method is *global*, i.e. the concentration of every single species is available – no a-priori selection of fractions and of a maximum polymer degree is required.

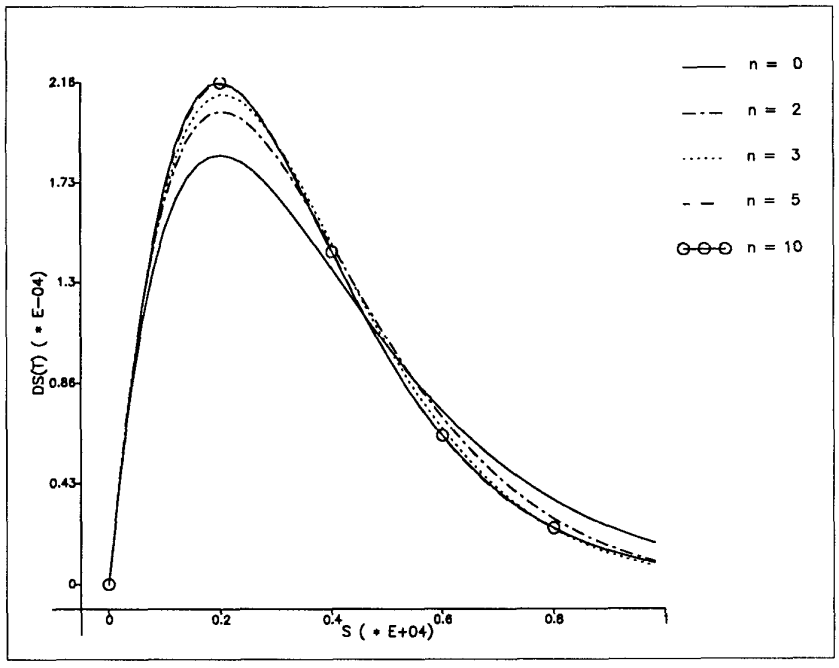


Figure 3.1: Discrete Galerkin approximations of the CLD of dead polymer for $n = 0, 2, 3, 5, 10$, $t = 4.5h$, $s_{max} = 10000$.
Computing Time: 5 sec (CPU) on a SUN 4/60.

The obtained approximations $D_s^{(n)}(t)$ of the dead polymer CLD are shown for $n = 0, 2, 3, 5, 10$ in Fig. 3.1. Final time of integration is $t = 4.5h$, comparable

to the experiments described in [3]. Fig. 3.1 reflects the fast convergence of the method on the basis of a Schulz-Flory distribution ($n = 0$). On the other hand one can see that a pure Schulz-Flory distribution is not sufficient to describe the CLD. Table 3.1 shows the *estimated* relative error $\epsilon_n(t)$ due to (3.5) compared to a *true* error $\bar{\epsilon}_n(t)$, computed by a highly accurate reference solution, for $n = 2, 3, 5, 10, 15$.

truncation index n	estimated error ϵ_n	true error $\bar{\epsilon}_n$
2	$5.1 \cdot 10^{-2}$	$6.0 \cdot 10^{-2}$
3	$2.6 \cdot 10^{-2}$	$3.0 \cdot 10^{-2}$
5	$5.7 \cdot 10^{-3}$	$6.3 \cdot 10^{-3}$
10	$0.9 \cdot 10^{-4}$	$1.1 \cdot 10^{-4}$
15	$1.6 \cdot 10^{-6}$	$1.7 \cdot 10^{-6}$

Table 3.1 Comparison of estimated and true approximation error for dead polymer CLD, $t = 4.5h$, $n = 2, 3, 5, 10, 15$.

In order to illustrate the starting phase of the process, Fig. 3.2 shows the development of the parameter ρ_1 belonging to the polymer radicals in the time interval $[0, 0.1]$. The initial forming of the polymer radicals at the beginning of the reaction, neglected by the QSSA, is reflected. The dense integration points, marked by stars, show the high complexity of the reaction in this phase. An approximately stationary state of ρ_1 is reached after about 10 seconds.

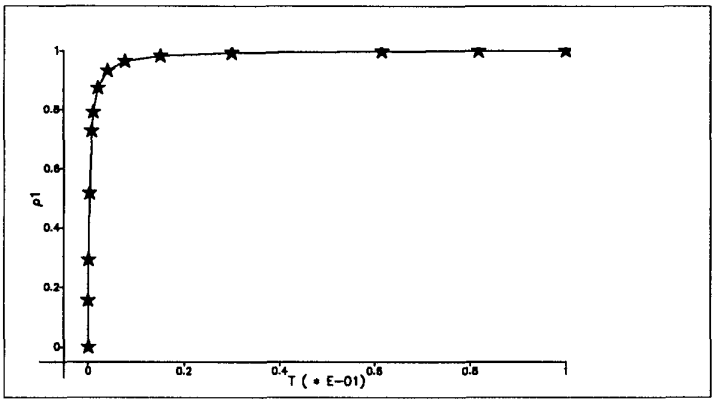


Figure 3.2: Initial phase of the parameter ρ_1.

Note that extensions to other gel effect models which are time- or moment-dependent can be done without changing the analytic preparations. Additional reaction steps e.g. transfer to dead polymer or reactions involving polymer species with a double bond (Vinylacetat reactions) can be treated for itself and added then to the system.

Example: Heterogeneous polymer degradation [1], [25]. In this type of reaction the rate coefficients k_s depend on the degree of the polymer s by $k_s = k_p/s^\alpha$, k_p constant, $\alpha = 1/3$. Here the *modified* discrete Laguerre polynomials are used. The degradation process in [1] starts with an experimental distribution having a number average $\bar{M}_N$ of about 120 000 and a maximum of the WCLD at about $M_{\max} = 100\,000$. The constant reaction coefficient k_p is estimated to be $k_p = 2.11 \cdot 10^{-7}$ in [1]. For the presentation in Fig. 3.3, a maximum polymer degree of $s_{\max} = 400\,000$ is chosen. In case one would try to solve a truncated original system, one may expect $s_{\max} \geq 800\,000$ to be necessary. Thus a direct solution is impossible. In Fig. 3.3, the time evolution of the weight chain length distribution $P_s(t)$ is shown.

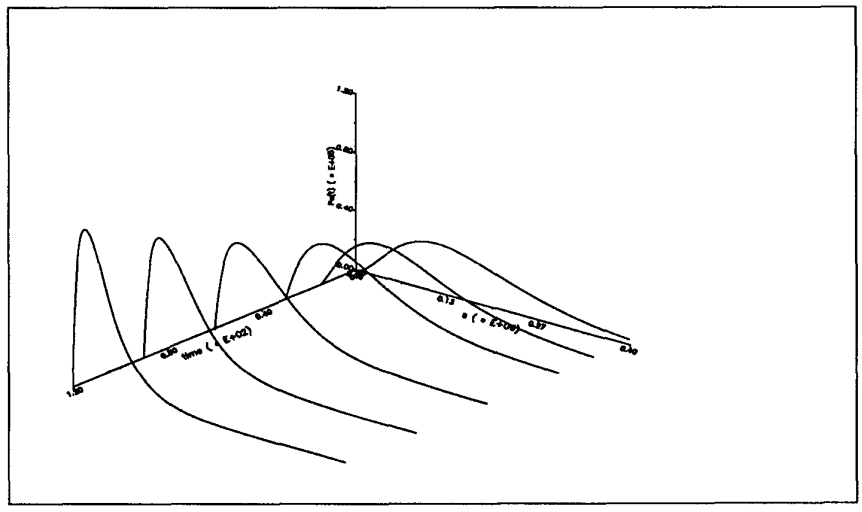


Figure 3.3: Time evolution of the degradation model due to [1], $t = 0, 15, 30, 60, 90, 120$ min, $s_{\max} = 400\,000$. Computing time: 9 sec (CPU) on a SUN 4/60.

References

[1] A. M. Basedow, K. H. Ebert, H. J. Ederer: *Kinetic Studies on the Acid Hydrolysis of Dextran.* Macromolecules, Vol. **11**, p. 774 (1978).

[2] K.E. Brenan, S.L. Campbell, L.R. Petzold: *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations.* North-Holland (1989).

[3] U. Budde: *Zur Reaktionstechnik der radikalischen Lösungspolymerisation in einem automatischen Laborreaktor.* Dissertation, Technische Universität Berlin (1989).

[4] U. Budde, M. Wulkow: *Computation of Molecular Weight Distributions for Free Radical Polymerization Systems.* To appear in Chem. Eng. Sci.

[5] P. Deuflhard: *Recent Progress in Extrapolation Methods for Ordinary Differential Equations.* SIAM Rev. **27**, pp. 505–535 (1985).

[6] P. Deuflhard, E. Hairer, J. Zugck: *One-step and Extrapolation Methods for Differential-Algebraic Systems.* Num. Math. **51**, pp. 501–516 (1987).

[7] P. Deuflhard, U. Nowak: *Efficient Numerical Simulation and Identification of Large Chemical Reaction Systems.* Ber. Bunsenges. Phys. Chem. **90**, pp. 940–946 (1986).

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

[9] P. Deuflhard, M. Wulkow: *Computational Treatment of Polyreaction Kinetics by Orthogonal Polynomials of a Discrete Variable.* IMPACT of Computing in Science and Engineering, Vol. **1**, No. 3, pp. 269–301, Academic Press, Inc. (1989).

[10] W. C. Gardiner, B. F. Walker, C. B. Wakefield: *Mathematical Methods for Modeling Chemical Reactions in Shock Waves.* In A. Lifshitz (ed.): Shock Waves in Chemistry (1981).

[11] E. Hairer, Ch. Lubich, M. Roche: *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods.* Lecture Notes in Mathematics 1409, Springer (1989).

[12] P. Holl, E. Marquardt, E. D. Gilles: *DIVA — A Powerful Tool for Dynamic Process Simulation.* Comp. Chem. Engng., Vol. **12**, No. 5, pp. 421–476 (1988).

[13] J. M. Hyman: *Moving Mesh Methods for Partial Differential Equations.* In J. Goldstein, S. Rosencrans, G. Sod (eds.): Mathematics Applied to Science, pp. 129–153 , Academic Press, Inc. (1988).

[14] A. Kröner: *Private communication* (1989).

[15] Ch. Lubich: *Linearly Implicit Extrapolation Methods for Differential-Algebraic Systems.* Num. Math. **55**, pp. 197–211 (1989).

[16] U. Maas, J. Warnatz: *Simulation of Chemically Reacting Flows in Two-Dimensional Geometries.* IMPACT of Computing in Science and Engineering, Vol. **1**, No. 4, pp. 394–420, Academic Press, Inc. (1989).

[17] U. Maas, J. Warnatz: *Private communication* (1989).

[18] K. Miller, R. N. Miller: *Moving Finite Elements I.* SIAM J. Numer. Anal., Vol. **18**, No. 6, pp. 1019–1032 (1981).

[19] K. Miller, R. N. Miller: *Moving Finite Elements II.* SIAM J. Numer. Anal., Vol. **18**, No. 6, pp. 1033–1057 (1981).

[20] N. Peters, J. Warnatz (eds.): *Numerical Methods in Laminar Flame Propagation.* Notes on Numerical Fluid Dynamics **6**, pp 232–260, Vieweg (1982).

[21] L. Petzold: *A Description of DASSL: A Differential/Algebraic System Solver.* Proc. IMACS World Congress (1982).

[22] W. H. Ray: *On the Mathematical Modeling of Polymerization Reactors.* J. Macromol. Sci.–Revs. Macromol. Chem. C8 (1), pp. 1–56 (1972).

[23] W. C. Rheinboldt: *Differential-Algebraic Systems as Differential Equations on Manifolds.* Math. Comp. **43**, pp. 473–482 (1985).

[24] D. Walkowiak: *Numerische Behandlung großer, adiabater chemischer Reaktionssysteme.* Universität Heidelberg, Diploma Thesis (1986).

[25] M. Wulkow, P. Deuflhard: *Towards an Efficient Computational Treatment of Heterogeneous Polymer Reactions.* Konrad–Zuse–Zentrum Berlin, Preprint SC 90–1 (1990).