Biophysical Chemistry 17 (1983) 321–336
Elsevier

GENERALIZED FINITE ELEMENT SOLUTION TO ONE-DIMENSIONAL FLUX PROBLEMS

G. Peter TODD and Rudy H. HASCHEMEYER

Department of Biochemistry, Cornell University Medical College, 1300 York Avenue, New York, NY 10021, U.S.A.

Received 5th October 1982
Accepted 10th January 1983

Key words: Finite element method; Flow simulation; Ultracentrifugation; Electrophoresis; Chromatography

A finite element numerical solution to the general one-dimensional flow equation is derived in a form that provides a convenient and general means to simulate a wide variety of one-dimensional flow techniques of interest to biological scientists, e.g., ultracentrifugation, electrophoresis, chromatography, etc. Diverse physical models defined in terms of column geometry, solute interactions, and the dependence of transport parameters on column position, time, or concentrations of one or more solutes, can be accommodated. A particularly useful aspect of the formulation is that a wide variety of boundary conditions can be simply applied to the end result, without rederivation of the solution for each new case. The numerical solution is expressed as matrix equations that are sufficiently general so that incorporation of particular models can be effected by substitution of appropriate quantities into the final result.

1. Introduction

The principles underlying the finite element method for solution of differential equations were first presented by Courant in 1943. The method was developed in its initial stages primarily by engineers who applied it to problems in structural mechanics and elasticity. Over the past two decades applications of the finite element method have been made with increasing frequency in areas of engineering and physics where simulation of model behavior is paramount (e.g., see refs. 1 and 2).

The finite element method was introduced to biological scientists through a series of three papers by Claverie, Dreux and Cohen [3–5]. They presented equations to calculate concentration distributions of solutes redistributing with time in the ultracentrifuge. Their formulation included chemical interactions among solutes, and the functional dependence of a solute's sedimentation and diffusion coefficients on its own concentration, or on the concentrations of other solutes (thus allowing incorporation of such effects as thermodynamic nonideality or changes in solution density or viscosity due to solute redistribution). Their equations were expressed in relatively simple matrix form appropriate for computer solution. In the last of this series, Claverie [5] suggested that extension of the method to other flow techniques could result in a similarly convenient solution.

Further impetus to generalize their method resulted from our recent demonstration that its 'thought-wise obvious' incorporation to solve the inverse problem of the ultracentrifuge was indeed practical [6]. Thus, given a model of the experimental system, it is possible to determine the values of unknown parameters in the model that minimize the least-square difference between calculated and experimental data.

It should be recognized that several numerical methods for the simulation of flow techniques of interest to biological scientists have appeared over the last two decades (including ultracentrifugation [7–10], electrophoresis [10], isoelectric focusing [11,12] and molecular exclusion chromatography [13,14]). Some of the simulation techniques have been applied to more than a single flow method, and most appear capable of extension to a variety of flow techniques. However, significant limitations often exist, in particular, because the ability to incorporate realistically a variety of boundary conditions has not been demonstrated. It is beyond the intention of this manuscript to discuss these simulation methods. A brief discussion of most of them has been given by Cox [8].

It is the purpose of this manuscript to present a general finite element solution to one-dimensional flow problems such that a physical model of the process defined in terms of column geometry, boundary conditions, forces or flows acting to effect solute redistribution, diffusion, interactions between solutes, thermodynamic nonideality, etc., can be incorporated into the generalized solution without requiring separate derivation for each case. Diverse models can be accommodated with a few substitutions, minor rearrangements of the final matrix equation, and straightforward computations of matrix elements. An intuitive corollary suggests that a general computer program can be written such that each new model for each new process of interest can be accommodated by supplying relatively simple subroutines.

In summary, we present a formulation whose end result can be transformed with relative ease into a solution for any model of one-dimensional flow irrespective (to the best of our knowledge) of the complexity of the model. We wish to point out, however, that a one-dimensional model cannot rigorously apply to a number of commonly encountered flow processes that are frequently considered to be one-dimensional (e.g., chromatography). Nonetheless, for true one-dimensional problems, and to the degree that other problems can be rendered one-dimensional by approximation, we suggest that the approach presented here is unrivaled in generality and convenience of application.

2. The one-dimensional flow equation

2.1 General formulation

Simulation of experimental data for a one-dimensional flow process involving solutes numbered from 1 to s requires calculating values of the functions

$C_k(x,t)$; $a \le x \le b$, $t \ge t_0$, $k = 1,\ldots,s$,

where C_k is the concentration (mass/volume) of solute k, x the spatial coordinate along which flow can occur, a and b the positions of the boundaries, t the time, and t_0 the initial time. Such functions represent the solution of an appropriate system of s continuity equations, subject to initial and boundary conditions determined by the experimental design. Extending the general one-dimensional continuity equation presented by Tanford [15] to include solute interconversions results in

$$\frac{\partial C_k}{\partial t} + \frac{1}{A_k}\frac{\partial (A_k J_k)}{\partial x} = Q_k, \tag{1}$$

where $A_k(x)$ is the cross-sectional area of the column that is accessible to solute k, $J_k(x,t)$ the flux of solute k per unit of accessible cross-sectional area, and $Q_k(x,t)$ represents a standard kinetic expression for the rate of change of C_k due to any associations, dissociations, or other reactions of the solutes. For example, for an associating monomer-dimer system, we would have $Q_1 = k_d C_2 - k_a C_1^2$ and $Q_2 = -Q_1$, where k_a and k_d are the association and dissociation rate constants, and the subscripts 1 and 2 refer to the monomer and dimer, respectively. Q_k can always be evaluated from the concentrations of reactants, and appropriate rate constants. Q_k could include reactions with sites on the column if the distribution of such sites was sufficiently uniform throughout the column cross-section that the one-dimensionality of the problem was maintained.

The flux J_k can be expressed in general form as

$$J_k = F_k C_k - \sum_{l=1}^{s} D_{kl}\frac{\partial C_l}{\partial x}, \tag{2}$$

where $F_k(x,t)$ is the average flow rate (cm/s) solute k would have in the absence of concentration gradients (when there is no solute redistribution due to diffusion), $D_{kk}(x,t)$ the main diffusion coefficient of solute k in the column medium, and $D_{kl}(x,t)$, for $l \neq k$, cross-term diffusion coefficients [16,17]. F and D terms can be functions of x and t directly, or functions of the concentrations of one or more of the solutes.

Experimental conditions are usually such that cross-term diffusion coefficients are sufficiently small to be ignored, in which case eq. 2 simplifies to

$$J_k = F_k C_k - D_k \frac{\partial C_k}{\partial x}, \tag{3}$$

where $D_k = D_{kk}$. This more convenient expression for the flux will be used until section 6, where the finite element solution will be expanded to include cross-terms.

Two or more interacting solutes where associations and dissociations are sufficiently rapid to maintain equilibrium as solutes redistribute may be conveniently described by a single equation of the form of eq. 1 obtained by adding the equations for the individual solutes that are in equilibrium. Then C_k, Q_k and J_k in eq. 1 are replaced by their aggregate values C_T, Q_T and J_T. For example, for a rapidly self-associating monomer-dimer-... n-mer system, we have

$$C_T = \sum_{j=1}^{n} C_j = \sum_{j=1}^{n} K_j C_1^j, \tag{4}$$

$$Q_T = \sum_{j=1}^{n} Q_j,$$

$$J_T = \bar{F} C_T - \bar{D} \frac{\partial C_T}{\partial x},$$

where K_j is the monomer-j-mer equilibrium constant ($K_1 = 1$), and $\bar{F}$ and $\bar{D}$ average values of F and D defined as

$$\bar{F} = \frac{\sum_{j=1}^{n} F_j C_j}{C_T} = \frac{\sum_{j=1}^{n} F_j K_j C_1^j}{C_T}$$

and

$$\bar{D} = \frac{\sum_{j=1}^{n} D_j (\partial C_j / \partial x)}{\sum_{j=1}^{n} (\partial C_j / \partial x)} = \frac{\sum_{j=1}^{n} j D_j K_j C_1^{j-1}}{\sum_{j=1}^{n} j K_j C_1^{j-1}} \tag{5}$$

The second equality of eq. 5 is valid only when each $\partial K_j / \partial x = 0$ [8], and thus an alternative formulation is required if this is not the case. Although $\bar{F}$ is the weight-average flow rate, $\bar{D}$ has no such simple physical interpretation. Because C_1 can be determined from C_T by finding the positive root of eq. 4, it is apparent that $\bar{F}$ and $\bar{D}$ are functions of C_T. Thus, such an associating system where equilibrium is maintained is mathematically equivalent to a single solute that has a complex concentration dependence of F and D [16].

2.2. *Some selected examples*

We now present some examples of the application of eqs. 1 and 3. Electrophoresis illustrates flow techniques in a rectangular coordinate system, ultracentrifugation illustrates cylindrical coordinates, and diffusion into a sphere illustrates spherical coordinates.

2.2.1. *Electrophoresis*

Let x be the distance from the top of an electrophoresis column; then $a = 0$ and $b = l$, the length of the column. Let $A_k = \xi_k A$, where A is the column cross-sectional area and ξ_k the fraction of A that is accessible to solute k. ξ_k depends on the properties of any supporting matrix (e.g., polyacrylamide gel) and the solute. Since A would normally be constant it may be factored out of the $\partial(A_k J_k)/\partial x$ term in eq. 1, and cancelled. Thus, eqs. 1 and 3 become

$$\frac{\partial C_k}{\partial t} + \frac{1}{\xi_k} \frac{\partial(\xi_k J_k)}{\partial x} = Q_k \quad \text{and} \quad J_k = \mu_k E C_k - D_k \frac{\partial C_k}{\partial x},$$

where μ is the electrophoretic mobility, and E the electric field strength [10]. Note that in polyacrylamide gel electrophoresis, for example, a gradient in gel concentration would result in ξ_k being a function of x. The top of the column could be selected as some point above the top of the gel (where ξ_k would be 1), so that the initially layered solute band is included in the column. The boundary conditions would depend on the experimental design. Whenever the experiment is designed so that no solute reaches the boundaries, the particularly simple boundary condition of zero flux ($J_k(0, t) = 0$ or $J_k(l, t) = 0$) can be used.

2.2.2. *Ultracentrifugation*

In ultracentrifugation it is conventional to use the radius from the axis of rotation (r) as the spatial coordinate. Then a and b become r_a and r_b, the radii of the meniscus and bottom of the ultracentrifuge cell. The cross-sectional area is $\theta r d$, where θ is the sector angle of the ultracentrifuge cell (in radians) and d the cell thickness. Since θd is constant, eqs. 1 and 3 become

$$\frac{\partial C_k}{\partial t} + \frac{1}{r}\frac{\partial (rJ_k)}{\partial r} = Q_k, \quad r_a \le r \le r_b,$$

and

$$J_k = s_k\omega^2 rC_k - D_k\frac{\partial C_k}{\partial r},$$

where s_k is the sedimentation coefficient of solute k, and ω the angular velocity [16]. The boundary conditions are $J_k(r_a, t) = J_k(r_b, t) = 0$, reflecting the fact that no solute can cross the meniscus or bottom of the centrifuge cell.

2.2.3. Diffusion in a sphere

The diffusion of solute into or out of a spherical object (e.g., a gel bead or an idealized cell) is a one-dimensional problem where eq. 1 applies if the solute is symmetrically distributed about the center of the sphere. If r is the distance from the center of the sphere, then $a = 0$, $b = r_b$, the sphere's radius, and $A_k = 4\pi r^2\xi_k$. For constant ξ_k, eq. 1 becomes

$$\frac{\partial C_k}{\partial t} = \frac{1}{r^2}\frac{\partial (r^2 J_k)}{\partial r} = Q_k, \quad 0 \le r \le r_b,$$

If solute flow occurs by diffusion only the flux is

$$J_k = -D_k\frac{\partial C_k}{\partial r}.$$

The boundary condition at the center of the sphere would be $J_k(0, t) = 0$. Any of several boundary conditions might apply at the surface of the sphere.

2.3. One-dimensional approximations to multi-dimensional problems

Techniques involving bulk flow of solvent or multiple phases (e.g., chromatography) are not truly one-dimensional processes, since generally neither the solvent flow rate nor the solute concentration is constant throughout a given column cross-section. Solvent flows more slowly near solid interfaces (solid support material or column walls) than away from such interfaces, and of course there is no flow at all in stationary phases. Thus, solute molecules are transported at a rate that depends on their position within the column cross-section. If solute molecules are not completely equilibrated throughout the accessible column cross-section the one-dimensional flow equation does not rigorously apply.

Nonetheless, since rigorous mathematical description of such processes would be extremely complex, equations of the form of eqs. 1 and 3 are frequently applied as an approximation. Then C_k and F_k represent the average concentration and flow rate of solute k throughout the portion of the column cross-section that is accessible to solute k. Since the flow rate is not constant throughout the column cross-section, a process known as dispersion occurs, which results in spreading of solute bands beyond that expected due to diffusion. Dispersion is often approximated by replacing D_k in eq. 3 with L_k, the coefficient of axial dispersion, which empirically reflects band spreading due to dispersion as well as diffusion (e.g., see ref. 18). However, dispersion differs mathematically from diffusion and this approximate treatment does not reproduce the skewed peaks that are actually observed.

3. Finite element space discretization

Analytical solutions to systems of differential equations of the form of eqs. 1 and 2 are available for only very few of the most simple cases, and therefore in general it is necessary to resort to numerical solutions. In the following sections we present a finite element formulation for calculation of numerical solutions to any system of such equations, provided only that D_{kj}, F_k and Q_k terms are constants or functions of solute concentrations, x and t; and A_k is constant or a function of x only. In this section we perform a finite-element space discretization of eq. 1, a partial differential equation in the variables x and t, to obtain a series of differential equations in the variable t only that apply at discrete x positions on the interval a to b.

For economy of notation we now drop the subscript k with the implicit understanding that an equation of the form of eq. 1 applies to each solute present. If these solutes interact in any way (including effects on D and F) then the equations must be solved simultaneously. As a further convenience we delay consideration of diffusion cross-terms until section 6. Thus, eqs. 1 and 3 simplify to

$$\frac{\partial C}{\partial t}+\frac{1}{A}\frac{\partial(AJ)}{\partial x}=Q, \tag{6}$$

and

$$J=FC-D\frac{\partial C}{\partial x}. \tag{7}$$

We will present a quite general derivation, the result of which can readily be applied to specific models by making appropriate post facto substitutions and simplifications. In order to accomplish this we will express D, F and A as the composite functions

$$D=(D^0+D^c)D^x,\quad F=(F^0+F^c)F^x,\quad A=A^0A^x \tag{8}$$

where D^0, F^0 and A^0 are constants with respect to x (D^0 and F^0 might vary with t), D^c and F^c can be functions of solute concentrations (including other solutes) or other functions of x and t, and D^x, F^x and A^x are functions of x only. D^0, F^0 and A^0 may be regarded as the 'basic' values of the parameters; D^x, F^x and A^x should incorporate any direct dependence of the parameters on x (e.g., as would arise in electrophoresis in a gradient of gel concentration); and D^c and F^c incorporate functional dependences of D and F on other factors. For example, in ultracentrifugation we have $F=s\omega^2r$ and $A=\theta rd$ (r is used here in place of x). Suppose the solute is nonideal and s and D depend on the solute concentration according to $s=s_0(1-k_sC)$ and $D=D_0(1+k_DC)$, where s_0, k_s, D_0 and k_D are constants [16]. Then we could choose $D^0=D_0$, $D^c=D_0k_DC$, $D^x=1$, $F^0=s_0\omega^2$, $F^c=-s_0\omega^2k_sC$, $F^x=r$, $A^0=\theta d$ and $A^x=r$.

Substituting eq. 8 into eqs. 6 and 7, and cancelling A^0 we get

$$\frac{\partial C}{\partial t}+\frac{1}{A^x}\frac{\partial(A^xJ)}{\partial x}=Q, \tag{9}$$

and

$$J=(F^0+F^c)F^xC-(D^0+D^c)D^x\frac{\partial C}{\partial x}. \tag{10}$$

The space discretization then proceeds as follows (many variations are possible, for a general reference see ref. 2). Multiplying eq. 9 by A^xv, where v is a function of x to be specified later, and integrating over x from a to b yields

$$\int_a^b\frac{\partial C}{\partial t}A^xv\,\mathrm{d}x+\int_a^b\frac{\partial(A^xJ)}{\partial x}v\,\mathrm{d}x=\int_a^bQA^xv\,\mathrm{d}x.$$

[Multiplication by A^x here results in weighting the solution according to the cross-sectional area at each point on the interval a to b. Multiplication by alternate functions would effect both the weighting and the difficulty of calculating the matrix elements.] Integrating the second term of this equation by parts gives

$$\int_a^b\frac{\partial C}{\partial t}A^xv\,\mathrm{d}x+J(b,t)A^x(b)v(b)-J(a,t)A^x(a)v(a)$$
$$-\int_a^bA^xJ\frac{\mathrm{d}v}{\mathrm{d}x}\mathrm{d}x=\int_a^bQA^xv\,\mathrm{d}x. \tag{11}$$

We now divide the interval a to b into N subintervals or 'elements' of lengths $h_1, h_2,\ldots,h_N$. The h_i terms are most frequently chosen all the same so that $h_i=h=(b-a)/N$ for $i=1,\ldots,N$. Let $x_1, x_2,\ldots,x_{N+1}$ be the positions of the endpoints of the elements, which are called nodes ($x_1=a$ and $x_{N+1}=b$).

Next we defined a set of basis functions $P_i(x)$, $i=1,\ldots,N+1$, that we can use to approximate continuous functions of x by specifying the function values at the nodes. A convenient choice for these functions is the piecewise-linear set used by Claverie et al. [3], sometimes called 'hat' functions because of their shape, which are defined as follows:

$$P_i=(x-x_i)/h_{i-1}+1,\quad x_{i-1}\le x\le x_i$$
$$P_i=1-(x-x_i)/h_i,\quad x_i\le x\le x_{i+1}$$
$$P_i=0,\quad x<x_{i-1}\text{ or }x>x_{i+1}$$

Linear combinations of these functions form continuous piecewise-linear interpolates of continuous functions. For example, the function C is approximated as $\sum_{j=1}^{N+1}C_jP_j$, where $C_j=C(x_j,t)$. This approximate function equals C at the nodes, and is a linear interpolation between nodes.

By approximating the functions Q, F^c and D^cC in the same manner, and substituting these approximations into eqs. 11 and 10 we obtain

$$\int_a^b\left(\sum_{j=1}^{N+1}\frac{\partial C_j}{\partial t}P_j\right)A^x v\,\mathrm{d}x + J_{N+1}A_{N+1}^x v(b) - J_1A_1^x v(a)$$

$$-\int_a^b A^x J\frac{\mathrm{d}v}{\mathrm{d}x}\mathrm{d}x = \int_a^b\left(\sum_{j=1}^{N+1}Q_jP_j\right)A^x v\,\mathrm{d}x, \qquad (12)$$

and

$$J = F^0F^x\left(\sum_{j=1}^{N+1}C_jP_j\right) + F^x\left(\sum_{j=1}^{N+1}F_j^cC_jP_j\right)$$

$$-D^0D^x\left(\sum_{j=1}^{N+1}C_j\frac{\mathrm{d}P_j}{\mathrm{d}x}\right) - D^x\left(\sum_{k=1}^{N+1}D_k^cP_k\right)\left(\sum_{j=1}^{N+1}C_j\frac{\mathrm{d}P_j}{\mathrm{d}x}\right) \qquad (13)$$

where $Q_j = Q(x_j, t)$, $D_k^c = D^c(x_k, t)$, $F_j^c = F^c(x_j, t)$, $J_1 = J(a, t)$, $J_{N+1} = J(b, t)$, $A_1^x = A^x(a)$ and $A_{N+1}^x = A^x(b)$.

The next step is to replace the function v by a set of $N + 1$ weighting functions to obtain a system of $N + 1$ equations. For this purpose we will use the same functions that were used as the basis functions, which is known as the Galerkin method [2]. Substituting P_i, $i = 1,\dots, N + 1$, for v in eq. 12 yields the system of equations

$$\int_a^b\left(\sum_{j=1}^{N+1}\frac{\partial C_j}{\partial t}P_j\right)A^xP_i\,\mathrm{d}x + J_{N+1}A_1^xP_i(b) - J_1A_{N+1}^xP_i(a)$$

$$-\int_a^b A^x J\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x$$

$$= \int_a^b\left(\sum_{j=1}^{N+1}Q_jP_j\right)A^xP_i\,\mathrm{d}x;\ i = 1,\dots,N+1 \qquad (14)$$

Note that $P_1(a) = 1$ and $P_i(a) = 0$ for $i \neq 1$; and $P_{N+1}(b) = 1$ and $P_i(b) = 0$ for $i \neq N + 1$. Substituting for J in eq. 14 and rearranging we obtain

$$\int_a^b\left(\sum_{j=1}^{N+1}\frac{\partial C_j}{\partial t}P_j\right)A^xP_i\,\mathrm{d}x + J_{N+1}A_{N+1}^xP_i(b) - J_1A_1^xP_i(a)$$

$$+ D^0\int_a^b A^xD^x\left(\sum_{j=1}^{N+1}C_j\frac{\mathrm{d}P_j}{\mathrm{d}x}\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x$$

$$- F^0\int_a^b A^xF^x\left(\sum_{j=1}^{N+1}C_jP_j\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x = \int_a^b\left(\sum_{j=1}^{N+1}Q_jP_j\right)A^xP_i\,\mathrm{d}x$$

$$\int_a^b A^xD^x\left(\sum_{k=1}^{N+1}D_k^cP_k\right)\left(\sum_{j=1}^{N+1}C_j\frac{\mathrm{d}P_j}{\mathrm{d}x}\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x$$

$$+\int_a^b A^xF^x\left(\sum_{j=1}^{N+1}F_j^cC_jP_j\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x;\ i = 1,\dots,N+1. \qquad (15)$$

Rearrangements, and in the case of the term involving D_k^c terms, use of the property of the hat functions that $P_kP_j = 0$ whenever j and k differ by more than 1, show that the terms of this system of equations may be expressed in matrix notation as follows [3–5]:

$$\int_a^b\left(\sum_{j=1}^{N+1}\frac{\partial C_j}{\partial t}P_j\right)A^xP_i\,\mathrm{d}x;\ i = 1,\dots,N+1 \rightarrow \boldsymbol{B}\frac{\mathrm{d}\boldsymbol{C}}{\mathrm{d}t}$$

$$J_{N+1}A_{N+1}^xP_i(b) - J_1A_1^xP_i(a);\ i = 1,\dots,N+1 \rightarrow \boldsymbol{j}$$

$$D^0\int_a^b A^xD^x\left(\sum_{j=1}^{N+1}C_j\frac{\mathrm{d}P_j}{\mathrm{d}x}\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x;\ i = 1,\dots,N+1 \rightarrow D^0\boldsymbol{A}^1\boldsymbol{C}$$

$$F^0\int_a^b A^xF^x\left(\sum_{j=1}^{N+1}C_jP_j\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x;\ i = 1,\dots,N+1 \rightarrow F^0\boldsymbol{A}^2\boldsymbol{C}$$

$$\int_a^b\left(\sum_{j=1}^{N+1}Q_jP_j\right)A^xP_i\,\mathrm{d}x;\ i = 1,\dots,N+1 \rightarrow \boldsymbol{BQ}$$

$$\int_a^b A^xD^x\left(\sum_{k=1}^{N+1}D_k^cP_k\right)\left(\sum_{j=1}^{N+1}C_j\frac{\mathrm{d}P_j}{\mathrm{d}x}\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x;$$

$$i = 1,\dots,N+1 \rightarrow \boldsymbol{UC}^u + \boldsymbol{VC}^v + \boldsymbol{WC}^w$$

$$\int_a^b A^xF^x\left(\sum_{j=1}^{N+1}F_j^cC_jP_j\right)\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x;\ i = 1,\dots,N+1 \rightarrow \boldsymbol{A}^2\boldsymbol{C}^A$$

where $\boldsymbol{A}^1$, $\boldsymbol{A}^2$, $\boldsymbol{B}$ and $\boldsymbol{V}$ are $N + 1$ by $N + 1$ matrices, and $\boldsymbol{U}$ and $\boldsymbol{W}$ $N + 1$ by N matrices whose elements are defined as follows:

$$A_{i,j}^1 = \int_a^b A^xD^x\frac{\mathrm{d}P_j}{\mathrm{d}x}\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x,\quad A_{i,j}^2 = \int_a^b A^xF^xP_j\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x$$

$$B_{i,j} = \int_a^b P_jP_iA^x\mathrm{d}x,\quad V_{i,j} = \int_a^b A^xD^xP_j\frac{\mathrm{d}P_j}{\mathrm{d}x}\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x$$

$$U_{i,j} = \int_a^b A^xD^xP_j\frac{\mathrm{d}P_{j+1}}{\mathrm{d}x}\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x,$$

$$W_{i,j} = \int_a^b A^xD^xP_{j+1}\frac{\mathrm{d}P_j}{\mathrm{d}x}\frac{\mathrm{d}P_i}{\mathrm{d}x}\mathrm{d}x \qquad (16)$$

$\boldsymbol{C}$, $\boldsymbol{Q}$, $\boldsymbol{C}^A$, $\boldsymbol{C}^v$ and $\boldsymbol{j}$ are column vectors of length $N + 1$, and $\boldsymbol{C}^u$ and $\boldsymbol{C}^w$ column vectors of length N:

$$\boldsymbol{C} = (C_1, C_2, \dots, C_{N+1})^{\mathrm{T}},\quad \boldsymbol{Q} = (Q_1, Q_2, \dots, Q_{N+1})^{\mathrm{T}}$$

$$\boldsymbol{C}^A = (F_1^cC_1, F_2^cC_2, \dots, F_{N+1}^cC_{N+1})^{\mathrm{T}}.$$

$$C^v = (D_1^c C_1, D_2^c C_2, \ldots, D_{N+1}^c C_{N+1})^{\mathrm{T}}$$
$$C^u = (D_1^c C_2, D_2^c C_3, \ldots, D_N^c C_{N+1})^{\mathrm{T}},$$
$$C^w = (D_2^c C_1, D_3^c C_2, \ldots, D_{N+1}^c C_N)^{\mathrm{T}}$$
$$j = (-J_1 A_1^x, 0, 0, \ldots, 0, J_{N+1} A_{N+1}^x)^{\mathrm{T}} \tag{17}$$

(T indicates transposition.)

With the additional substitution $D^0 A^1 - F^0 A^2 = A$, eq. 15 now becomes

$$B\frac{\mathrm{d}C}{\mathrm{d}t} + j + AC = BQ - (UC^u + VC^v + WC^w) + A^2 C^A \tag{18}$$

This completes the finite element space discretization. We obtain a matrix equation of the form of eq. 18 for each solute $1, \ldots, s$. These equations represent a simultaneous system of first-order differential equations with independent variable t, and contain no functions of x. We now seek the solution to this system of equations, i.e., the solute concentrations at the nodes (the C vectors) as a function of t, subject to the boundary and initial conditions. (Application of boundary conditions and time discretization will be covered in the following sections.)

Solutions of eq. 18 represent approximate solutions of eq. 1. (Strictly speaking the functions $\Sigma_{j=1}^{N+1} C_j P_j$ are the approximate solutions to eq. 1.) The accuracy of these solutions depends on the size of h (or h_i terms). Solutions of eq. 18 converge to exact solutions of eq. 1 as h approaches zero. Convergence is second order – the error is approximately proportional to h^2. This relation between the error and h becomes increasingly exact as h becomes smaller. Of course additional error is introduced in the time discretization.

The vectors Q, C^A, C^u, C^v and C^w are all functions of the C terms for one or more solutes, and possibly t. If any of these vectors are nonzero then the system of equations will in general be nonlinear. All matrices except A depend only on $x_1, \ldots, x_{N+1}$ (constants), D^x, F^x and A^x (functions of x), and therefore are independent of time and need to be calculated only once. A also depends on D^0 and F^0, which in some cases might change with time. Since $P_i P_j = 0$ whenever i and j differ by more than unity, the matrices A, A^1, A^2, B and V are all tridiagonal – having only $3N + 1$ nonzero elements. U and W contain only $2N$ nonzero elements, but actually need not be calculated and stored at all since their elements are all the same as, or additive inverses of, elements of V. Matrix elements for several common situations are given in appendix A.

Eq. 18 was formulated with generality in mind, and considerable simplification frequently occurs. If D does not depend on solute concentration then D^c in eq. 8 can be chosen as zero, and $UC^u + VC^v + WC^w = 0$. Similarly, if $\Gamma^c = 0$, then $A^2 C^A = 0$. If solute interconversions do not occur then $BQ = 0$.

4. Boundary conditions

In this section we illustrate how different boundary conditions can be incorporated into the finite element solution. The general procedure is to derive an appropriate expression for the flux at the boundary (J_1 or J_{N+1}), substitute it into j in eq. 18, and make convenient rearrangements. In all cases, an equation is obtained that incorporates the desired boundary conditions, and can be expressed in the form

$$B'\frac{\mathrm{d}C}{\mathrm{d}t} + A'C = B'Q - (UC^u + V'C^v + WC^w) + A^{2\prime}C^A + k \tag{19}$$

The vector k contains all zeros except possibly the first and last elements. The matrices in this equation correspond to those in eq. 18, but the prime symbols (′) denote that certain elements may be modified. Boundary conditions at $x = a$ are incorporated by modifications to the first element of k, and element (1, 1) of the primed matrices. Boundary conditions at $x = b$ are incorporated by modifications to the last element of k, and element $(N + 1, N + 1)$ of the primed matrices.

We will assume that J_1 and J_{N+1} can be expressed in the forms

$$J_1 = \alpha_1\left(\frac{\mathrm{d}C_1}{\mathrm{d}t} - Q_1\right) + (\alpha_2 F_1 + \alpha_3 D_1 + \alpha_4)C_1 + \alpha_5 \tag{20}$$

and

$$J_{N+1} = \omega_1\left(\frac{\mathrm{d}C_{N+1}}{\mathrm{d}t} - Q_{N+1}\right) + (\omega_2 F_{N+1} + \omega_3 D_{N+1} + \omega_4)C_{N+1} + \omega_5 \tag{21}$$

where the α and ω terms are either constants or functions of time. these expressions are very general. and probably encompass all practical boundary conditions. Some examples of the applications of these expressions are presented later in this section.

Let the notation $|y/z|$ represent an $N+1$ by $N+1$ matrix with element (1, 1) equal to y, element ($N+1$, $N+1$) equal to z, and zeros elsewhere. Then substituting eqs. 20 and 21 into j in eq. 18, and rearranging we can obtain

$$\left(B + \left|\frac{-\alpha_1 A_1^x}{\omega_1 A_{N+1}^x}\right|\right)\frac{\mathrm{d}C}{\mathrm{d}t}$$
$$+\left[D^0\left(A^1 + \left|\frac{-\alpha_3 D_1^x A_1^x}{\omega_3 D_{N+1}^x A_{N+1}^x}\right|\right)\right.$$
$$\left.F^0\left(A^2 - \left|\frac{-\alpha_2 F_1^x A_1^x}{\omega_2 F_{N+1}^x A_{N+1}^x}\right|\right) + \left|\frac{-\alpha_4 A_1^x}{\omega_4 A_{N+1}^x}\right|\right]C$$
$$-\left(B + \left|\frac{-\alpha_1 A_1^x}{\omega_1 A_{N+1}^x}\right|\right)Q$$
$$\left[UC'' + \left(V + \left|\frac{-\alpha_3 D_1^x A_1^x}{\omega_3 D_{N+1}^x A_{N+1}^x}\right|\right)C' + WC''\right]$$
$$+\left(A^2 - \left|\frac{-\alpha_2 F_1^x A_1^x}{\omega_2 F_{N+1}^x A_{N+1}^x}\right|\right)C^A$$
$$+\left(\alpha_5 A_1^x, 0, 0, \ldots, 0, -\omega_5 A_{N+1}^x\right)^\mathrm{T} \tag{22}$$

where $D_1^x = D^x(a)$, $D_{N+1}^x = D^x(b)$, $F_1^x = F^x(a)$ and $F_{N+1}^x = F^x(b)$. This equation illustrates the changes to elements of matrices and the vector k that are necessary to incorporate particular boundary conditions. For example, if α_1 is nonzero, $\alpha_1 A_1^x$ is subtracted from element (1, 1) of B in eq. 18 to obtain B' in eq. 19. If ω_2 is nonzero $\omega_2 F_{N+1}^x A_{N+1}^x$ is subtracted from element ($N+1$, $N+1$) of A^2.

To illustrate the details involved in deriving eq. 22, let us consider the case where α_2 and ω_2 are nonzero, and all other α and ω terms are zero. Then

$$J_1 = -\alpha_2 F_1 C_1 \text{ and } J_{N+1} = \omega_2 F_{N+1} C_{N+1},$$

$$j = (-J_1 A_1^x, 0, 0, \ldots, 0, J_{N+1} A_{N+1}^x)^\mathrm{T},$$

$$= (-\alpha_2 F_1 C_1 A_1^x, 0, \ldots, 0, \omega_2 F_{N+1} C_{N+1} A_{N+1}^x)^\mathrm{T}.$$

Since $F = (F^0 + F^c)F^x$

$$j = (-\alpha_2 F^0 F_1^x C_1 A_1^x, 0, \ldots, 0, \omega_2 F^0 F_{N+1}^x C_{N+1} A_{N+1}^x)^\mathrm{T}$$
$$+ (-\alpha_2 F_1^c F_1^x C_1 A_1^x, 0, \ldots, 0, \omega_2 F_{N+1}^c F_{N+1}^x C_{N+1} A_{N+1}^x)^\mathrm{T}$$

Since C_1 and C_{N+1} are the first and last elements of C, and $F_1^c C_1$ and $F_{N+1}^c C_{N+1}$ are the first and last elements of C^A, we can obtain

$$j = F^0\left|\frac{-\alpha_2 F_1^x A_1^x}{\omega_2 F_{N+1}^x A_{N+1}^x}\right|C + \left|\frac{-\alpha_2 F_1^x A_1^x}{\omega_2 F_{N+1}^x A_{N+1}^x}\right|C^A$$

Substitution of this expression for j into eq. 18 and rearrangement to combine the coefficient matrices of C and C^A then leads to the modifications of the A^2 matrix shown in eq. 22. The modifications for the other α and ω terms can be derived in a similar manner.

4.1. *Known flux at the boundaries*

The flux at a boundary may be known as a consequence of the experimental design. This includes the case of zero flux at the column ends. Note that the boundary flux could be a predetermined function of time. Then α_5 or ω_5 is the value of the flux, and all other α or ω terms are zero.

4.2. *flux equals F times the concentration at a boundary*

For many flow processes the flux of solute leaving a column is equal to the product of the solute concentration at the boundary and the flow rate out of the column. If $J_{N+1} = F_{N+1}C_{N+1}$ then $\omega_2 = 1$ and all other ω terms are zero. Thus, this boundary condition can be incorporated by subtracting $F_{N+1}^x A_{N+1}^x$ from element ($N+1$, $N+1$) of A^2. The analogous boundary condition $J_1 = F_1 C_1$ is satisfied by adding $F_1^x A_1^x$ to element (1, 1) of A^2.

4.3. *The boundary concentration equals the concentration of an adjoining compartment*

If a uniformly mixed reservoir of solution (compartment) freely exchanges solute with the column across a boundary, then its solute concentration and rate of change of C due to reactions would be the same as for the boundary, i.e., C_1 and Q_1, or C_{N+1} and Q_{N+1}. Let m be the mass of solute in a compartment adjoining a column at $x = a$, and let V be the compartment volume. Then

$$\frac{\mathrm{d}C_1}{\mathrm{d}t} = \frac{\mathrm{d}(m/V)}{\mathrm{d}t} = \frac{1}{V}\frac{\mathrm{d}m}{\mathrm{d}t} - \frac{m}{V^2}\frac{\mathrm{d}V}{\mathrm{d}t} = \frac{1}{V}\frac{\mathrm{d}m}{\mathrm{d}t} - \frac{C_1}{V}\frac{\mathrm{d}V}{\mathrm{d}t} \tag{23}$$

But

$$\frac{\mathrm{d}m}{\mathrm{d}t} = -A_1J_1 + VQ_1 + k_+ - k_-C_1,$$

where $A_1 = A(x_1)$, and k_+ and k_-C_1 are the rates of gain and loss of solute mass in the compartment due to exchange with external sources. Substituting this into eq. 23 gives

$$\frac{\mathrm{d}C_1}{\mathrm{d}t} = \frac{1}{V}(-A_1J_1 + VQ_1 + k_+ - k_-C_1) - \frac{C_1}{V}\frac{\mathrm{d}V}{\mathrm{d}t}.$$

Solving for J_1 yields

$$J_1 = -\frac{V}{A_1}\frac{\mathrm{d}C_1}{\mathrm{d}t} + \frac{V}{A_1}Q_1 - \left(\frac{k_-}{A_1} - \frac{1}{A_1}\frac{\mathrm{d}V}{\mathrm{d}t}\right)C_1 + \frac{k_+}{A_1}.$$

Thus

$$\alpha_1 = -\frac{V}{A_1},\ \alpha_4 = -\left(\frac{k_-}{A_1} + \frac{1}{A_1}\frac{\mathrm{d}V}{\mathrm{d}t}\right),\ \text{and}\ \alpha_5 = \frac{k_+}{A_1}.$$

This boundary condition can be applied at $x = b$ in the same manner. Then

$$\omega_1 = \frac{V}{A_{N+1}},\ \omega_4 = \left(\frac{k_-}{A_{N+1}} + \frac{1}{A_{N+1}}\frac{\mathrm{d}V}{\mathrm{d}t}\right),\ \text{and}\ \omega_5 = -\frac{k_+}{A_{N+1}}.$$

4.4. Known concentration at a boundary

The concentration at a boundary as a function of time may be predetermined (known) as a consequence of the experimental design. This could occur, for example, if the boundary adjoins a reservoir that can give up or accept solute without affecting its concentration. An adjoining compartment like that described in the preceding example, but with a very large volume, would act as such a reservoir – which suggests an extremely simple mechanism to incorporate this boundary condition. The results of the preceding example show that this boundary condition can be applied simply by addition of a very large number (e.g., 10^{10}) to element (1, 1) or ($N+1$, $N+1$) of $\boldsymbol{B}$, and substitution of the known value for C_1 or C_{N+1}.

For completeness we note an alternative method of incorporating this boundary condition. If the solute concentration at a boundary is known then the number of unknown variables (C_i terms) in the system of equations represented by eq. 18 is reduced by 1. If C_1 is known then eq. 2–$N+1$ comprise a system of N equations in N unknown variables ($C_2, C_3, \ldots, C_{N+1}$) that can be solved independently of the first equation, and thus the first equation can be dropped from the system. Dropping the first or last equation from the system represented by eq. 18 can be accomplished by deleting the first or last rows of all matrices, and the first or last element of $\boldsymbol{j}$. If desired, the dropped equation can be used to calculate the flux at the boundary so that, for example, the net solute mass that has entered or exited the column can be calculated (this approach has the advantage over the application of eq. 3 that conservation of total solute mass is assured). However, since this method leads to modifications of eq. 18 beyond those of eq. 19, it is much less convenient than the method presented above, and we will not consider it further.

5. Time discretization

A matrix equation of the form of eq. 19 is obtained for each solute $1, \ldots, s$. These equations are a matrix representation of a simultaneous system of first-order differential equations. Our goal is to calculate concentration distributions ($\boldsymbol{C}$ vectors) at times t_1, t_2, etc., given a specified set of initial conditions in the form of concentration distributions for each solute at an initial time t_0. This type of initial value problem can in principle be solved by a variety of numerical techniques, but numerical instability can be a serious problem with a number of commonly used methods such as the Adams predictor-corrector or Runge-Kutta methods [19]. With the Adams method corrector convergence could not be achieved in a reasonable number of iterations unless an excessively small Δt was used. With the Runge-Kutta method very high accuracy can be obtained, but a fairly small Δt must still be used to prevent instability due to propagation of error. Both methods require smaller Δt values as h_i terms become smaller. It has been our experience that the procedure presented bellow is usually a better compromise between considerations of accuracy, stability efficiency, and simplicity.

For convenience, let

$$d = B'Q - (U'C^u + V'C^v + W'C^w) + A^{2\prime}C^A. \tag{24}$$

The terms comprising the vector $\boldsymbol{d}$ are in general nonlinear with respect to the concentration distributions. Eq. 19 now becomes

$$B'\frac{dC}{dt} + A'C = d + k \tag{25}$$

The time discretization of this equation can be derived in a finite-element context [2], but we will use a simpler finite-difference approach since the same final result is obtained. In order to calculate concentration distributions at time t_{n+1} from those at time t_n, we will assume that all vectors are constant over the time interval as follows: $\mathrm{d}C/\mathrm{d}t = (C_{n+1} - C_n)/\Delta t$, where $\Delta t = t_{n+1} - t_n$; $C = (1 - \theta)C_n + \theta C_{n+1}$, where $0 \le \theta \le 1$; and $\boldsymbol{d}$ and $\boldsymbol{k}$ equal some average values over the time interval, $\bar{\boldsymbol{d}}$ and $\bar{\boldsymbol{k}}$. By making these substitutions into eq. 19 and rearranging we obtain the final recurrence equation

$$(B' + \Delta t\theta A')C_{n+1} = [B' - \Delta t(1 - \theta)A']C_n + \Delta t(\bar{d} + \bar{k}) \tag{26}$$

If we choose $\bar{\boldsymbol{d}} = \boldsymbol{d}_n$, then $\bar{\boldsymbol{d}}$ can be calculated from the C_n terms for each solute. Since $\boldsymbol{k}$ contains no unknown values, $\bar{\boldsymbol{k}}$ can always be evaluated. The right-hand side of eq. 26 can thus be reduced to a single vector by performing the indicated operations. The expression in parentheses on the left-hand side of eq. 26 reduces to a tridiagonal matrix that needs to be recalculated only if Δt, D^0, or F^0 change. The resulting equation is conveniently solved for C_{n+1} by a Gaussian elimination procedure that requires only $3N + 1$ multiplications and $2N$ additions (see appendix B). Thus, C_{n+1} terms can be calculated from C_n terms by solving this equation for each solute. Starting with the initial conditions (C_0 terms), concentration distributions for all subsequent times can be calculated by recursively solving this equation.

The accuracy of calculated solutions depends on the choice of Δt, θ, h_i terms, and the nature of the particular problem being solved. Calculated solutions converge to the true (exact) solutions as Δt and h_i terms approach zero (neglecting the effect of roundoff errors in computations). Table 1 illustrates the effect of varying Δt and θ on accuracy for sedimentation velocity-type ultracentrifuge simulation.

Table 1

Comparison of accuracy obtained with different values of θ and Δt for a sedimentation velocity simulation

Accuracy is expressed as the standard deviation of 30 min concentration distributions (excluding the bottom 0.1 cm) from a fully converged solution ($\Delta t = 0.009$, $\theta = 0.5$). The following parameters were used in the simulations: $r_a = 6.4$ cm, $r_b = 6.9$ cm, $s = 5.727 \times 10^{-13}$ s, $D = 5.461 \times 10^{-7}$ cm²/s, rpm = 50000, $h = 0.01$ cm, initial concentration of 1 mg/ml.

Δt	Standard deviation (mg/ml) (× 100)			
	$\theta = 1$	$\theta = 2/3$	$\theta = 1/2$	$\theta = 0$
1.17	0.0691	0.0231	0.00004	0.0697
2.34	0.1375	0.0461	0.00016	0.1400
4.69	0.2725	0.0918	0.00064	0.2827
9.38	0.5356	0.1824	0.00256	0.5765
18.75	1.0363	0.3600	0.01024	1.2018
37.5	1.9507	0.7021	0.04099	[a]
75	3.5161	1.3417	0.16477	[a]
150	5.9706	2.4905	0.67641	[a]
300	9.4508	4.4933	3.23980	[a]

[a] Solution is unstable.

Setting $\theta = 1$ results in the implicit scheme used by Claverie et al. [3–5]. If $\boldsymbol{d} = 0$ (i.e., $Q = F^c = D^c = 0$, as in table 1) this scheme is stable for any choice of Δt. With $\theta = 0$ we have an explicit scheme where stability depends on the choice of Δt. In fact, with $\boldsymbol{d} = 0$ the solution is unconditionally stable for $\theta \ge 1/2$, although oscillations can occur near $\theta = 1/2$ [2]. Except for values of θ in the vicinity of 1/2, convergence with respect to Δt is linear (first order) – the error is approximately proportional to Δt. For $\theta = 1/2$ convergence is second order – the error is approximately proportional to Δt^2. These relations between Δt and the error become exact as Δt approaches zero. As table 1 illustrates, use of $\theta = 1/2$ can dramatically increase accuracy. However, if $\boldsymbol{d} \neq 0$ the accuracy and convergence properties become more like the $\theta = 0$ case.

If we use the same scheme for $\boldsymbol{d}$ as was used for C, i.e., $\bar{\boldsymbol{d}} = (1 - \theta)\boldsymbol{d}_n + \theta\boldsymbol{d}_{n+1}$, accuracy and convergence properties similar to the $\boldsymbol{d} = 0$ case result. However, because of the nonlinearity of $\boldsymbol{d}$, eq. 26 cannot be solved explicitly for C_{n+1} (since $\bar{\boldsymbol{d}}$ depends on C_{n+1} terms for each solute), and so an iterative solution procedure is required [2]. For the first iteration $\bar{d}$ can be extrapolated from past values, or d_n can be used. For subsequent iterations d_{n+1} can be calculated from the C_{n+1} terms, and $\bar{d}$ evaluated from the above formula. A few iterations should normally significantly improve the accuracy. In order to make the nonlinear terms as small as possible it is always best to choose D^0, D^c, F^0 and F^c so that D^c and F^c are as small as possible compared to D^0 and F^0.

6. Diffusion cross-terms

In the preceding sections we made the usual assumption that diffusion cross-terms were negligible, and indeed it is most often possible to choose experimental conditions so that this is the case (e.g., by minimizing charge interactions among solute molecules by introducing high salt concentrations). Nonetheless, considerable theoretical effort has gone into this area (e.g., see refs. 17 and 20), and a numerical method for calculating solutions to flow equations that include cross-terms may be of value to some investigators. Fortunately, inclusion of cross-terms results in only slight additional complication.

The solution already presented encompasses cases in which flow of a solute depends on the concentrations of other solutes through D^c and F^c terms. Inclusion of diffusion cross-terms introduces the dependence of solute flow on the concentration gradients of other solutes.

We will retrace steps in the space discretization outlined in section 3 to point out changes that are necessary to utilize eq. 2 to describe the flux instead of eq. 3. Again we drop the subscript k. Instead of eq. 7 we thus obtain

$$J = FC - \sum_{l=1}^{s} D_l \frac{\partial C_l}{\partial x}.$$

Note that $C = C_k$ (i.e., C_l for $l = k$) and $D = D_k$. Expressing D_l, for $l \neq k$, as the composite function $D_l^c D^x$, eq. 10 becomes

$$J = (F^0 + F^c) F^x C - D^0 D^x \frac{\partial C}{\partial x} - D^x \sum_{l=1}^{s} D_l^c \frac{\partial C_l}{\partial x}$$

Approximating the functions D_l^c and C_l as usual, the last term of eq. 13 is transformed to

$$-D^x \sum_{l=1}^{s} \left[\left(\sum_{k=1}^{N+1} D_{l,k}^c P_k \right) \left(\sum_{j=1}^{N+1} C_{l,j} \frac{\mathrm{d}P_j}{\mathrm{d}x} \right) \right].$$

where $D_{l,k}^c = D_l^c(x_k, t)$ and $C_{l,j} = C_l(x_j, t)$. The next to last term in eq. 15 then becomes

$$-\int_a^b A^x D^x \sum_{l=1}^{s} \left[\left(\sum_{k=1}^{N+1} D_{l,k}^c P_k \right) \left(\sum_{j=1}^{N+1} C_{l,j} \frac{\mathrm{d}P_j}{\mathrm{d}x} \right) \right] \frac{\mathrm{d}P_i}{\mathrm{d}x} \mathrm{d}x$$

Again making use of the property of the hat functions that $P_j P_k = 0$ when j and k differ by more than unity, this term can be written in matrix notation as

$$-\left[U\left(\sum_{l=1}^{s} C_l^u \right) + V\left(\sum_{l=1}^{s} C_l^v \right) + W\left(\sum_{l=1}^{s} C_l^w \right) \right] \tag{27}$$

where

$$C_l^u = \left(D_{l,1}^c C_{l,2}, D_{l,2}^c C_{l,3}, \ldots, D_{l,N}^c C_{l,N+1} \right)^{\mathrm{T}}$$

$$C_l^v = \left(D_{l,1}^c C_{l,1}, D_{l,2}^c C_{l,2}, \ldots, D_{l,N+1}^c C_{l,N+1} \right)^{\mathrm{T}}$$

$$C_l^w = \left(D_{l,2}^c C_{l,1}, D_{l,3}^c C_{l,2}, \ldots, D_{l,N+1}^c C_{l,N} \right)^{\mathrm{T}}$$

Eq. 27 then replaces $(UC^u + VC^v + WC^w)$ in eq. 18 and corresponding expressions in subsequent equations. Thus, the only change necessary to incorporate diffusion cross-terms is to calculate the vectors C_l^u, C_l^v and C_l^w, and to sum over all solutes ($l = 1, \ldots, s$). The $D_{l,k}^c$ terms ($l = 1, \ldots, s$, $k = 1, \ldots, N+1$) are calculated according to the functional form for the dependence of the cross-term diffusion coefficients on solute concentrations of the chosen model. Otherwise the simulation can proceed as before.

7. Alternate basis functions

In section 3 the basis and weighting functions (P_i, $i = 1, \ldots, N+1$) were chosen as the piecewise-linear hat functions. With these functions convergence with respect to h (or h_i terms) is second order. Many other choices for the basis and weighting functions with different convergence properties are possible [2]. With piecewise-quadratic basis functions (defined below) conver-

Table 2

Comparison of accuracy for linear, quadratic, and cubic basis functions

The standard deviation of calculated equilibrium concentration distributions from the exact solution is given for various values of h using the following parameters: $D^0 = 5\times10^{-7}$ cm^2/s, $F^0 = 2\times10^{-5}$ cm/s, $b-a=0.5$ cm, $A^x = D^x = F^x = 1$, $D^c = F^c = J_a = J_b = 0$, uniform initial concentration of 1 mg/ml.

N	h (cm)	Standard deviation (mg/ml)		
		Linear	Quadratic	Cubic
6	0.08333	1.8967	1.5916	1.2507
12	0.04167	0.5666	0.3982	0.2439
18	0.02778	0.2625	0.1509	0.0742
24	0.02083	0.1410	0.0710	0.0288
30	0.01667	0.0967	0.0385	0.0132
36	0.01389	0.0674	0.0230	0.0068
42	0.01190	0.0496	0.0148	0.0038
48	0.01042	0.0381	0.0100	0.0023
60	0.00833	0.0244	0.0052	0.0010

gence is third order – the error is approximately proportional to h^3. With piecewise-cubic functions convergence is fourth order.

Table 2 compares the accuracy obtained with these basis functions for an equilibrium solution (similar to sedimentation equilibrium) where there is no error from time discretization, and where an analytical solution can be obtained so that the error can be precisely calculated. Use of these higher-order basis functions can significantly increase accuracy. Although the error is more sensitive to the choice of h with these functions, in our experience the quadratic and especially the cubic functions nearly always give significantly better results in practical situations. A notable exception occurs when simulating sedimentation velocity experiments, where the extremely rapid change in the concentration gradient at the bottom of the cell can give rise to oscillations of the solution that are worse for the quadratic and cubic functions, unless very small h_i terms are used near the bottom. If the concentration distribution at the bottom is not of interest, these oscillations can be prevented by the artificial application of the boundary condition $J(b) = F(b)C(b)$, which allows solute to flow out the bottom of the cell so that the plateau region extends right to the cell bottom.

With the quadratic functions we again divide the interval a to b into subintervals of lengths $h_1,\ldots,h_N$, but now we must stipulate that N is divisible by 2 and $h_1 = h_2$, $h_3 = h_4$, etc. Actually two of these subintervals correspond to a single 'element', but this distinction need not concern us here. The present system maintains the same numbering of P_i, x_i, h_i, C_i terms, etc., as has already been presented. The functions may then be defined as follows:

Let $y = (x - x_i)/2h_i$, and $z = (x_i - x)/2h_{i-1}$

For odd i:

$P_i = 2z^2 - 3z + 1,\ x_{i-2} \le x \le x_i$

$P_i = 2y^2 - 3y + 1,\ x_i \le x \le x_{i+2}$

$P_i = 0,\ x < x_{i-2}$ or $x > x_{i+2}$

For even i:

$P_i = -4y^2 + 1,\ x_{i-1} \le x \le x_{i+1}$

$P_i = 0,\ x < x_{i-1}$ or $x > x_{i+1}$

With the cubic functions N must be divisible by 3, $h_1 = h_2 = h_3$, $h_4 = h_5 = h_6$, etc., and three subintervals correspond to each element. The cubic functions are then defined as follows:

Let $y = (x - x_i)/3h_i$, and $z = (x_i - x)/3h_{i-1}$.

For $i = 2, 5, 8, \ldots, N-1$:

$P_i = 27y^3/2 - 9y^2 - 3y/2 + 1,\ x_{i-1} \le x \le x_{i+2}$

$P_i = 0,\ x < x_{i-1}$ or $x > x_{i+2}$

For $i = 3, 6, 9, \ldots, N$:

$P_i = -27y^3/2 - 9y^2 + 3y/2 + 1,\ x_{i-2} \le x \le x_{i+1}$

$P_i = 0,\ x < x_{i-2}$ or $x > x_{i+1}$

For $i = 1, 4, 7, 10, \ldots, N+1$:

$P_i = -9z^3/2 + 9z^2 - 11z/2 + 1,\ x_{i-3} \le x \le x_i$

$P_i = -9y^3/2 + 9y^2 - 11y/2 + 1,\ x_i \le x \le x_{i+3}$

$P_i = 0,\ x < x_{i-3}$ or $x > x_{i+3}$

To obtain the term $(UC'' + VC' + WC'')$ in eq. 18 we made use of the property of the linear basis functions that $P_iP_j = 0$ when i and j differ by more than unity. Now $P_iP_j = 0$ when i and j differ by more than 2 (for quadratics) or 3 (for cubics), and so instead we obtain a term containing 5 or 7 matrices. To avoid this proliferation of matrices it seems advisable to restrict use of these higher-order basis functions to problems where $D^c = 0$ so that this troublesome term drops out. The elements of the remaining matrices in eq. 18 are modified, and the matrix bandwidths are increased from 3 to 5 for the quadratic functions, and to 7 for the cubics. Therefore, there may be a modest increase in the computations necessary to solve the matrix equation, depending on the algorithm employed. Otherwise the solution procedure is unchanged from what has already been presented.

8. Summary

We can summarize the steps required to simulate a flow technique as follows:

(1) A model of the experimental process must be formulated in terms of the functional forms of A_k and Q_k in eq. 1, F_k and D_k in eq. 3, and boundary conditions for each solute $k = 1, \ldots, s$.

(2) The functions for D, F and A for each solute are expressed as the composite function of eq. 8.

(3) The interval a to b is divided into N elements of lengths $h_1, \ldots, h_{N+1}$, which define the x_i terms and the hat functions (P_i terms).

(4) The matrix elements are computed using eq. 16. The U, V and W matrices are needed only if D^c is nonzero. Matrix elements for several common situations are given in appendix A.

(5) Boundary conditions are applied by writing the expressions for the flux at the boundaries in the form of eqs. 20 and 21, and then making the necessary modifications to matrix elements according to eq. 22.

(6) Starting with initial concentration vectors for each solute (C_0 terms) the concentration vectors at subsequent times are calculated by repeatedly solving eq. 26 for each solute.

Eq. 26 is applicable to an extremely wide variety of experimental situations. Relatively simple substitutions are all that is necessary to adapt this equation to different systems. Geometrical factors; the dependence of D, F and A terms on x are taken into account simply by substituting the appropriate functions for D^x, F^x and A^x in the equations for calculating the matrix elements, eq. 16. Inclusion of such complexities as arbitrary gel concentration gradients in gel electrophoresis is thus straightforward. Discontinuous functions for D^x, F^x and A^x present no special problems.

Complex solute interactions are handled with ease by making substitutions into the expression for d, eq. 24. The dependence of D and F terms on solute concentrations is easily included by substituting the appropriate functions for D^c and F^c terms in the equations to calculate C^A, C^u, C^v and C^w, eq. 17. Solute associations, dissociations, or other reactions, are incorporated through Q.

Solutions of any desired accuracy can be obtained by using sufficiently small values for h (or h_i terms) and Δt. Required computer time is inversely proportional to Δt and h, and memory requirements are also inversely proportional to h. (If an operator matrix is used to solve the matrix equation as suggested in refs. 3–5, rather than Gaussian elimination, then computer time and memory requirements are inversely proportional to h^2.)

We have written a computer program in FORTRAN to simulate one-dimensional flow processes that incorporates most of the ideas presented here. Because of the difficulty of writing a completely self-contained program that can handle the diversity of possible experimental situations (e.g., multiple nonideal solutes interacting in various ways, or unusual column geometries), for any unusual model our program requires certain subroutines to be written by the user. This program can be obtained by to writing the authors.

Appendix A

Since computation of matrix elements can be a tedious and error-prone exercise in integration, formulas for the calculation of matrix elements for three common situations, corresponding to the examples in section 2.2, are presented here. Linear-basis functions are used, and a homogeneous medium is assumed in all cases. For 'i, i' subscripts, i ranges from 2 to N. For '$i, i+1$' and '$i+1, i$' subscripts, i ranges from 1 to N.

For more complicated situations – quadratic

	Rectangular coordinates (electrophoresis)	Cylindrical coordinates (ultracentrifuge)	Spherical coordinates (diffusion in a sphere)
A^x	1	x	x^2
D^x	1	1	1
F^x	1	x	–
$A^1_{1,1}$	$\frac{1}{h_1}$	$\frac{x_1}{h_1}+\frac{1}{2}$	$\frac{x_1^2}{h_1}+x_1+\frac{h_1}{3}$
$A^1_{N+1,N+1}$	$\frac{1}{h_N}$	$\frac{x_{N+1}}{h_N}-\frac{1}{2}$	$\frac{x_{N+1}^2}{h_N}-x_{N+1}+\frac{h_N}{3}$
$A^1_{i,i}$	$\frac{1}{h_{i-1}}+\frac{1}{h_i}$	$\frac{x_i}{h_{i-1}}+\frac{x_i}{h_i}$	$\frac{x_i^2}{h_{i-1}}+\frac{x_i^2}{h_i}+\frac{h_{i-1}+h_i}{3}$
$A^1_{i,i+1}, A^1_{i+1,i}$	$-\frac{1}{h_i}$	$-\frac{x_i}{h_i}-\frac{1}{2}$	$-\frac{x_i^2}{h_i}-x_i-\frac{h_i}{3}$
$A^2_{1,1}$	$-\frac{1}{2}$	$-\frac{x_1^2}{2}-\frac{x_1h_1}{3}-\frac{h_1^2}{12}$	
$A^2_{N+1,N+1}$	$\frac{1}{2}$	$\frac{x_{N+1}^2}{2}-\frac{x_{N+1}h_N}{3}+\frac{h_N^2}{12}$	
$A^2_{i,i}$	0	$-\frac{x_i(h_{i-1}+h_i)}{3}+\frac{h_{i-1}^2-h_i^2}{12}$	
$A^2_{i+1,i}$	$\frac{1}{2}$	$\frac{x_i^2}{2}+\frac{x_ih_i}{3}+\frac{h_i^2}{12}$	
$A^2_{i,i+1}$	$-\frac{1}{2}$	$-\frac{x_i^2}{2}-\frac{2x_ih_i}{3}-\frac{h_i^2}{4}$	
$B_{1,1}$	$\frac{h_1}{3}$	$\frac{x_1h_1}{3}+\frac{h_1^2}{12}$	$\frac{x_1^2h_1}{3}+\frac{x_1h_1^2}{6}+\frac{h_1^3}{30}$
$B_{N+1,N+1}$	$\frac{h_N}{3}$	$\frac{x_{N+1}h_N}{3}-\frac{h_N^2}{12}$	$\frac{x_{N+1}^2h_N}{3}-\frac{x_{N+1}h_N^2}{6}+\frac{h_N^3}{30}$
$B_{i,i}$	$\frac{h_{i-1}+h_i}{3}$	$\frac{x_i(h_{i-1}+h_i)}{3}+\frac{h_{i-1}^2-h_i^2}{12}$	$\frac{x_i^2(h_{i-1}+h_i)}{3}-\frac{x_i(h_{i-1}^2-h_i^2)}{6}+\frac{h_{i-1}^3+h_i^3}{30}$
$B_{i,i+1}, B_{i+1,i}$	$\frac{h_i}{6}$	$\frac{x_ih_i}{6}+\frac{h_i^2}{12}$	$\frac{x_i^2h_i}{6}+\frac{x_ih_i^2}{6}+\frac{h_i^3}{20}$
$V_{1,1}$	$\frac{1}{2h_1}$	$\frac{x_1}{2h_1}+\frac{1}{6}$	$\frac{x_1^2}{2h_1}+\frac{x_1}{3}+\frac{h_1}{12}$
$V_{N+1,N+1}$	$\frac{1}{2h_N}$	$\frac{x_{N+1}}{2h_N}-\frac{1}{6}$	$\frac{x_{N+1}^2}{2h_N}-\frac{x_{N+1}}{3}+\frac{h_N}{12}$
$V_{i,i}$	$\frac{1}{2h_{i-1}}+\frac{1}{2h_i}$	$\frac{x_i}{2h_{i-1}}+\frac{x_i}{2h_i}$	$\frac{x_i^2}{2h_{i-1}}+\frac{x_i^2}{2h_i}+\frac{h_{i-1}}{12}+\frac{h_i}{12}$
$V_{i+1,i}, U_{i,i}, -U_{i+1,i}$	$-\frac{1}{2h_i}$	$-\frac{x_i}{2h_i}-\frac{1}{6}$	$-\frac{x_i^2}{2h_i}-\frac{x_i}{3}-\frac{h_i}{12}$
$V_{i,i+1}, -W_{i,i}, W_{i+1,i}$	$-\frac{1}{2h_i}$	$-\frac{x_i}{2h_i}-\frac{1}{3}$	$-\frac{x_i^2}{2h_i}-\frac{2x_i}{3}-\frac{h_i}{4}$

and cubic basis functions or more complicated formulas for A^x, D^x and F^x – the matrix elements are most conveniently computed by numerical integration. We use a Gauss-Legendre method [21] of sufficient order to give exact values (within roundoff error) when A^x, D^x, F^x and P_i terms are polynomials, as is the usual case.

Appendix B

The overall efficiency of the finite element solution depends greatly on the algorithm used to solve eq. 26. In particular, use of general algorithms (or subroutines) for the matrix-vector multiplications and to solve for C_{n+1} is inefficient in terms of the number of arithmetic operations required. Therefore, in this appendix we present a highly efficient algorithm to solve repeatedly eq. 26.

To solve eq. 26 we must evaluate the right-hand side to obtain a vector, and then solve for $\boldsymbol{C}_{n+1}$ by Gaussian elimination. We will use $\bar{\boldsymbol{d}} = \boldsymbol{d}_n$ so that $\bar{\boldsymbol{d}}$ terms for each solute can be evaluated explicitly from the $\boldsymbol{C}_n$ terms. To evaluate the right-hand side, D_i^c, F_i^c and Q_i $(i = 1, \ldots, N+1)$ are computed from the C_i terms according to the the model of the experimental system; the vectors $\boldsymbol{C}^A$, $\boldsymbol{C}^u$, $\boldsymbol{C}^v$ and $\boldsymbol{C}^w$ are computed according to eq. 17; $\boldsymbol{C}_n$, $\boldsymbol{Q}$, $\boldsymbol{C}^A$, $\boldsymbol{C}^u$, $\boldsymbol{C}^v$ and $\boldsymbol{C}^w$ are multiplied by matrices according to eqs. 24 and 26; and the resulting vectors are all added together to obtain a single vector. The algorithm presented below combines the operations in the evaluation of the right-hand side and the first part of the Gaussian elimination into a single 'loop'; fully exploits the tridiagonality of the matrices and the fact that the elements of the $\boldsymbol{U}$ and $\boldsymbol{W}$ matrices are the same as or additive inverses of elements of $\boldsymbol{V}$; and reduces the number of arithmetic operations per iteration by performing as many operations as possible once at the beginning.

The nonzero elements of the tridiagonal $N+1$ by $N+1$ matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{V}$ can be stored efficiently as $N+1$ by three matrices. Let $\boldsymbol{M}$ represent a full $N+1$ by $N+1$ matrix. The corresponding $N+1$ by 3 matrix, $\boldsymbol{M}'$, is then defined as follows:

$$M'_{1,1} = M'_{N+1,3} = 0$$
$$M'_{i,1} = M_{i,i-1},\quad i = 2, \ldots, N+1$$
$$M'_{i,2} = M_{i,i},\quad i = 1, \ldots, N+1$$
$$M'_{i,3} = M_{i,i+1},\quad i = 1, \ldots, N.$$

Let $\boldsymbol{G}$, $\boldsymbol{R}$, $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ be the matrices $(\boldsymbol{B}' + \Delta t\theta \boldsymbol{A}')$, $[\boldsymbol{B}' - \Delta t(1-\theta)\boldsymbol{A}']$, $\Delta t\boldsymbol{B}'$, $\Delta t\boldsymbol{A}^{2\prime}$, and $-\Delta t\boldsymbol{V}'$, respectively, all stored in $N+1$ by 3 form. Note that $\boldsymbol{G}$ and $\boldsymbol{R}$ normally differ for each solute. $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ also differ unless A^x, D^x, F^x and boundary conditions for the different solutes are the same. The number of arithmetic operations required in the Gaussian elimination per iteration can be reduced by modifying $\boldsymbol{G}$ in a process similar to 'triangular decomposition' as follows:

For $i = 2, 3, 4, \ldots, N+1$:

$$G_{i,1} = -G_{i,1}/G_{i-1,2}$$
$$G_{i,2} = G_{i,2} + (G_{i,1}G_{i-1,3})$$
$$G_{i-1,2} = 1/G_{i-1,2}$$
$$G_{i-1,3} = -G_{i-1,3}$$
$$G_{N+1,2} = 1/G_{N+1,2}$$

$\boldsymbol{G}$ and $\boldsymbol{R}$ need to be recomputed if D^0, F^0, Δt, or θ change. $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ must be recomputed if Δt changes. Also, if any boundary conditions change (except for α_5 or ω_5) then $\boldsymbol{G}$ must be recomputed, but only element (1, 1) or $(N+1, N+1)$ of the other matrices must be changed.

Let C_i be the ith element of $\boldsymbol{C}_n$, and let C_i^* be the ith element of $\boldsymbol{C}_{n+1}$. Using the C_i terms for each solute, Q_i, F_i^c and D_i^c terms for each solute are computed according to the model of the experimental system. Then C_i^* terms (i.e., $\boldsymbol{C}_{n+1}$) are computed at each iteration as follows:

$$C_1^A = C_1F_1^c,\quad C_2^A = C_2F_2^c,\quad C_1^u = C_2D_1^c$$
$$C_1^v = C_1D_1^c,\quad C_2^v = C_2D_2^c,\quad C_1^w = C_1D_2^c$$
$$C_1^* = R_{1,2}C_1 + R_{1,3}C_2 + X_{1,2}Q_1 + X_{1,3}Q_2 + Y_{1,2}C_1^A + Y_{1,3}C_2^A$$
$$+ Z_{2,1}C_1^u + Z_{1,2}C_1^v + Z_{1,3}(C_2^v - C_1^w) + \Delta t\alpha_5 A_1^x$$

For $i = 2, 3, \ldots, N$:

$$C_{i+1}^A = C_{i+1}F_{i+1}^c,\quad C_i^u = C_{i+1}D_i^c,\quad C_{i+1}^v = C_{i+1}D_{i+1}^c,\quad C_i^w = C_iD_{i+1}^c$$
$$C_i^* = R_{i,1}C_{i-1} + R_{i,2}C_i + R_{i,3}C_{i+1} + X_{i,1}Q_{i-1} + X_{i,2}Q_i$$
$$+ X_{i,3}Q_{i+1} + Y_{i,1}C_{i-1}^A + Y_{i,2}C_i^A + Y_{i,3}C_{i+1}^A + Z_{i+1,1}C_i^u$$
$$+ Z_{i,1}(C_{i-1}^v - C_{i-1}^u) + Z_{i,2}C_i^v + Z_{i,3}(C_{i+1}^v - C_i^w)$$
$$+ Z_{i-1,3}C_{i-1}^w - G_{i,1}C_{i-1}^*$$
$$C_{N+1}^* = [R_{N+1,1}C_N + R_{N+1,2}C_{N+1} + X_{N+1,1}Q_N + X_{N+1,2}Q_{N+1}$$
$$+ Y_{N-1,1}C_N^A + Y_{N+1,2}C_{N+1}^A + Z_{N+1,1}(C_N^v - C_N^u)$$
$$+ Z_{N+1,2}C_{N+1}^v + Z_{N,3}C_N^w - \Delta t\omega_5 A_{N+1}^x$$
$$+ G_{N+1,1}C_N^*]G_{N+1,2}$$

For $i = N, N-1, \ldots, 1$:

$$C_i^* = (C_i^* + G_{i,3}C_{i+1}^*)G_{i,2}$$

If Q, F^c, and D^c equal zero then corresponding terms in the above algorithm may be eliminated. For this most simple case multiplication of $[\boldsymbol{B}' - \Delta t(1-\theta)\boldsymbol{A}']$ by $\boldsymbol{C}_n$, and the Gaussian elimination each require $2N$ additions and $3N+1$ multiplications, for a total of $4N$ additions and $6N+2$

multiplications per iteration. If Q is nonzero then an additional $3N+1$ additions and $3N+1$ multiplications are required (not counting computations necessary to calculate Q_i terms). If F^c is nonzero then an additional $3N+1$ additions and $4N+2$ multiplications are required. If D^c is nonzero then an additional $5N+1$ additions, $8N+2$ multiplications, and $2N$ subtractions are required. Similarly, efficient algorithms can be devised for the case of quadratic or cubic basis functions. Details are available from the authors.

Acknowledgements

This work was supported by the National Institutes of Health Grant GM-30282 and Biomedical Research Support Grant RR-05396 to Cornell University Medical College, and by Predoctoral Fellowship SP 179-22299 from the National Science Foundation to G.P.T.

References

1 G.B. Kolata, Science 184 (1974) 887.
2 O.C. Zienkiewicz, The finite element method, 3rd edn., (McGraw-Hill, London, 1977).
3 J.M. Claverie, M. Dreux and R. Cohen, Biopolymers 14 (1975) 1685.
4 R. Cohen and J.M. Claverie, Biopolymers 14 (1975) 1701.
5 J.M. Claverie, Biopolymers 15 (1976) 843.
6 G.P. Todd and R.H. Haschemeyer, Proc. Natl. Acad. Sci. U.S.A. 78 (1981) 6739.
7 M. Dishon, G.H. Weiss and D.A. Yphantis, Biopolymers 4 (1966) 449.
8 D.J. Cox, Methods Enzymol. 48 (1978) 212.
9 L.M. Gilbert and G.A. Gilbert, Methods Enzymol. 48 (1978) 195.
10 J.R. Cann, Interacting macromolecules (Academic Press, New York, 1970).
11 J.R. Cann and D.I. Stimpson, Biophys. Chem. 7 (1977) 103.
12 O.A. Palusinski, M. Bier and D.A. Saville, Biophys. Chem. 14 (1981) 389.
13 J.K. Zimmerman, D.J. Cox and G.K. Ackers, J. Biol. Chem. 246 (1971) 4242.
14 P.W. Chun and M.C.K. Yang, Biophys. Chem. 7 (1978) 347.
15 C. Tanford, Physical chemistry of macromolecules (Wiley, New York, 1961).
16 H. Fujita, Foundations of ultracentrifugal analysis (Wiley, New York, 1975).
17 L.J. Gosting, Adv. Protein Chem. 11 (1956) 429.
18 H.R. Halvorson and G.K. Ackers, J. Polymer Sci. A-2 9 (1971) 245.
19 F. Scheid, Schaums outline of theory and problems of numerical analysis (McGraw-Hill, New York, 1968).
20 H. Kim, J. Phys. Chem. 78 (1966) 562.
21 T.R. McCalla, Introduction to numerical methods and FORTRAN programming (Wiley, New York, 1967).