Proc. Natl. Acad. Sci. USA
Vol. 78, No. 11, pp. 6739–6743, November 1981
Biochemistry

General solution to the inverse problem of the differential equation of the ultracentrifuge

(nonlinear least-squares algorithm/finite-element method)

G. PETER TODD AND RUDY H. HASCHEMEYER

Department of Biochemistry, Cornell University Medical College, 1300 York Avenue, New York, New York 10021

Communicated by Alton Meister, July 27, 1981

ABSTRACT **Whenever experimental data can be simulated according to a model of the physical process, values of physical parameters in the model can be determined from experimental data by use of a nonlinear least-squares algorithm. We have used this principle to obtain a general procedure for evaluating molecular parameters of solutes redistributing in the ultracentrifuge that uses time-dependent concentration, concentration-difference, or concentration-gradient data. The method gives the parameter values that minimize the sum of the squared differences between experimental data and simulated data calculated from numerical solutions to the differential equation of the ultracentrifuge.**

Equations that are part of a model thought to describe a physical process may incorporate constants representing unknown physical parameters characteristic of the system under study. Solution of the inverse problem—determination of unknown parameters from experimental data—can be a difficult task if the model is complex. Whenever there exists a simulation algorithm incorporating a model, initial conditions, and estimates of unknown parameters to simulate experimental data, the inverse problem can be solved by using a nonlinear algorithm to determine the values of the unknown parameters that minimize in some (e.g., least squares) sense the difference between experimental and simulated data.

We have used this principle to obtain a generalized least-squares solution to the inverse problem of the ultracentrifuge. Time-dependent ultracentrifuge data for normally considered models can be simulated by using the finite-element solution of systems of differential equations of the ultracentrifuge presented by Claverie, Dreux, and Cohen (1–3). We report here that this simulation algorithm can be combined with a modified Gauss–Newton nonlinear least-squares algorithm (4, 5) to provide a general method for determination of least-square estimates of unknown parameters from time-dependent experimental ultracentrifuge data. Either concentration, concentration-difference, or concentration-gradient data may be used.

The general procedure can be summarized as follows. Let **P** be a vector whose elements are estimates of the unknown parameters, and let experimental data be expressed as a vector **Y** whose elements are the data values at various times and radial positions. A corresponding vector of simulated data **Z(P)** can be calculated by using the simulation algorithm. The goal of the nonlinear least-squares algorithm is to find $\hat{\mathbf{P}}$, the value of **P** that minimizes the residual sum of squares (RSS),

$$\mathrm{RSS} = \sum_{i=1}^{l} \mathrm{w}_i\,[\mathrm{Y}_i - \mathrm{Z}_i(\mathbf{P})]^2,$$

where l is the length of **Y** (the number of experimental observations) and **w** is an optional vector of case weights. Once $\hat{\mathbf{P}}$ has been found, the applicability of the chosen model can be evaluated from the SD of the simulated data from the experimental data.* If the SD is significantly larger than what can be attributed to experimental error, then the model is inappropriate. Alternative physically reasonable models often all fit the data within experimental error, and the correct choice would entail further study. Discussion of this likely possibility is beyond the scope of this paper.

METHODS

Model Equations for Simulation of Data. The sedimentation of a broad range of systems normally encountered in ultracentrifugal analysis can be adequately described by systems of partial differential equations of the form (6)

$$\frac{\partial C_k}{\partial t} + \frac{1}{r}\frac{\partial (rJ_k)}{\partial r} = f_k, \qquad J_k = s_k\omega^2 rC_k - D_k\frac{\partial C_k}{\partial r}, \qquad [1]$$

where the subscript k refers to solute k, C is the solute concentration, t is the time, r is the radius from the center of rotation, J is the solute flux, f is a "source term", s is the sedimentation coefficient, ω is the angular velocity, and D is the diffusion coefficient. The source term f represents a standard kinetic expression for the rate of change of C due to any chemical reactions.

The sedimentation and diffusion coefficients are generally nonconstant and can be represented in the forms $s = s_0(1 - \varepsilon)$ and $D = D_0(1 - \eta)$, where ε and η are functions that describe the variation of s and D from their "basic" values s_0 and D_0 (3). This formulation is convenient for the finite-element numerical solution. Once the functional forms of s and D are specified, ε and η are readily calculated from $\varepsilon = 1 - s/s_0$ and $\eta = 1 - D/D_0$. Sedimentation and diffusion coefficients may be nonconstant due to such effects as nonideality, changes in solution density and viscosity because of solute redistribution, or changes in the effective partial specific volume or frictional ratio of a solute due to changes in pressure or solution composition. Any such effects, singly or in combination, may be incorporated into the functions ε and η. Rate constants appearing in f may also be nonconstant. We will assume that ε, η, and f are functions of the concentrations of one or more of the solutes or of ω and r due to pressure effects.

A differential equation in the form of Eq. 1 applies to each solute present in the ultracentrifuge cell. However, solutes that are not of interest (e.g., low molecular weight solutes that do not contribute to the experimentally observable data) can be

* SD = $[\mathrm{RSS}/(l - p)\bar{w}]^{1/2}$, where p is the number of unknown parameters and $\bar{w} = (1/l)\,\Sigma_{i=1}^{l}\,\mathrm{w}_i$, which equals 1 if no weighting is used.

ignored if they do not affect the redistribution of solutes that are of interest through f, ε, or η terms. A system of such equations that is thought to describe an experimental system can be considered a model of that system.

Solutions of such systems of differential equations are the concentrations of each solute as a function of radius and time,

$$C(r,t);\ r_m \leq r \leq r_b \text{ and } t \geq t_0,$$

where r_m and r_b are the radii of the meniscus and the bottom of the ultracentrifuge cell, respectively. To be uniquely determined, such solutions must conform to certain boundary and initial conditions imposed by the experimental design. The boundary conditions are $J(r_m,t) = J(r_b,t) = 0$, describing the fact that no solute can cross the meniscus or cell bottom. The initial condition is $C(r,t_0) = C_0(r)$, the initial concentration distribution. Such solutions, calculated by using a specified set of parameters and initial conditions, can be used to obtain simulated experimental concentration, concentration-difference, or concentration-gradient data at specific times and radii, which can be compared to actual experimental data.

An analytical solution of Eq. 1 has been found only for the simplest case [i.e., $f = \varepsilon = \eta = 0$ (7, 8)], and that solution is so complex that it is seldom used. A number of methods to calculate numerical solutions have been developed in the past two decades. Some of these methods are applicable only to some of the experimental situations that can be described by Eq. 1 (9–12), while others appear capable of handling the general case (1–3, 13). [For a discussion of simulation procedures other than the finite-element method, see Cox (12)]. Any method that is applicable to the chosen model of the experimental system can in principle be used in the simulation algorithm. Desirable features of a numerical method are accuracy, efficiency, generality, ease of application, and ease of adaptability to different model systems. On the basis of these criteria, we have chosen to use the finite-element method of Claverie *et al.* (1–3). This method is capable of simulating data for any model system based on Eq. 1.

Because initial conditions must be specified, the experiment to be simulated must be set up such that the initial concentration distributions for each solute are either known or can be estimated from experimental data measured at time t_0 and the current estimates of any necessary unknown parameters. If the t_0 data are concentration difference or gradient, conversion to concentration data requires knowledge of the concentration at some point in the cell, which would have to be considered an additional unknown parameter if it has not been determined separately. The initial time t_0 may be taken as any convenient time—when the rotor starts to spin, when the rotor attains constant speed, or when a layering occurs—as long as the initial conditions can be specified. For example, with a kinetically controlled self-associating system (i.e., interactions too slow to maintain chemical equilibrium as solutes redistribute), the concentration of each solute when the rotor starts spinning can be calculated from the total concentration and the equilibrium constants for the associations. However, any subsequent redistribution of solutes disturbs the equilibrium, and so concentrations of individual solutes can no longer be calculated from the total concentration distribution.

An associating system in rapid equilibrium (i.e., interactions are sufficiently rapid to maintain chemical equilibrium as solutes redistribute) may be conveniently described by a single equation in the form of Eq. 1, obtained by adding the individual differential equations for the associating solutes. Then C, f, and J in Eq. 1 are replaced by their aggregate values C_T, f_T, and J_T. For example, for a monomer–dimer–. . .–m-mer system

$$C_T = \sum_{j=1}^{m} C_j = \sum_{j=1}^{m} K_j C_1^j,\ f_T = \sum_{j=1}^{m} f_j = 0, \qquad [2]$$

and

$$J_T = \bar{s}\omega^2 r C_T - \frac{\bar{D}\partial C_T}{\partial r},$$

where K_j is the monomer–j-mer equilibrium constant ($K_1 = 1$) and $\bar{s}$ and $\bar{D}$ are the average sedimentation and diffusion coefficients and defined as

$$\bar{s} = \frac{\sum_{j=1}^{m} s_j C_j}{C_T} = \frac{\sum_{j=1}^{m} s_j K_j C_1^j}{C_T} \qquad [3]$$

and

$$\bar{D} = \frac{\sum_{j=1}^{m} D_j(\partial C_j/\partial r)}{\sum_{j=1}^{m} (\partial C_j/\partial r)} = \frac{\sum_{j=1}^{m} jD_j K_j C_1^{j-1}}{\sum_{j=1}^{m} jK_j C_1^{j-1}}. \qquad [4]$$

The second equality of Eq. 4 is valid only when each $\partial K_j/\partial r = 0$ (12), and thus an alternative formulation is required if the equilibrium constants are pressure dependent. Although $\bar{s}$ is the weight-average sedimentation coefficient at C_T, $\bar{D}$ has no such simple physical interpretation. Because C_1 can be determined from C_T by finding the positive root of Eq. 2, it is apparent that $\bar{s}$ and $\bar{D}$ are functions of C_T and thus a pressure-independent self-associating system in rapid equilibrium is mathematically equivalent to a single noninteracting solute that has a complex concentration dependence of s and D (6).

The Nonlinear Least-Squares Algorithm. A number of algorithms to find $\hat{\mathbf{P}}$ are available, all of which must be supplied initially with estimates of the unknown parameters that are iteratively corrected until they converge to $\hat{\mathbf{P}}$. All must repeatedly evaluate the vector function $\mathbf{Z}(\mathbf{P})$, which requires significant computer time, and so an algorithm that converges with as few function evaluations as possible is to be preferred. However, some algorithms that normally converge rapidly are more likely to fail to converge, particularly for poor parameter estimates, than some of the slower algorithms that offer stronger guarantees of convergence. In addition, an algorithm that allows the setting of bounds on the values that $\mathbf{P}$ may assume should be used so that nonsensical parameter values, such as negative diffusion or rate constants, can be avoided. Besides being an aid in imposing physical realities on the model, setting of bounds precludes the evaluation of $\mathbf{Z}(\mathbf{P})$ when, due to overcorrections, $\mathbf{P}$ contains parameter values that would cause computational errors and possibly terminate program execution.

The modified Gauss–Newton algorithm we are using achieves a reasonable compromise in meeting the above demands. It incorporates limited step halving to improve convergence and stepwise regression to constrain parameters within bounds (4). The required derivatives of $\mathbf{Z}$ with respect to each unknown parameter are calculated by increasing the jth element of $\mathbf{P}$ by a small quantity $\Delta\mathbf{P}_j$[†] to obtain $\mathbf{P}'$ and then using Newton's forward difference rule,

$$\frac{\partial\mathbf{Z}}{\partial\mathbf{P}_j} \approx \frac{\mathbf{Z}(\mathbf{P}') - \mathbf{Z}(\mathbf{P})}{\Delta\mathbf{P}_j}.$$

† $\Delta\mathbf{P}_j$ should not be so small that significant accuracy is lost due to round-off error.

Therefore, $p + 1$ evaluations of $\mathbf{Z}$ per iteration are required (not counting any step halvings), where p is the number of unknown parameters. Although this algorithm has performed well, we are investigating other nonlinear least-squares algorithms, such as DUD (Doesn't Use Derivatives), which requires only one evaluation of $\mathbf{Z}$ per iteration (14).

The Finite-Element Numerical Solution. Claverie and his associates have presented in some detail the derivation of the finite-element numerical solution of systems of Eq. 1 (1–3). Basically, the finite-element solution uses a "variational formulation" of Eq. 1 that allows implicit incorporation of boundary conditions and elimination of second derivatives. Functions of r (C, f, ε, and η) are approximated by continuous piecewise-linear interpolate functions that equal the original functions at $N + 1$ equally spaced radial positions, where N is the number of elements in the space discretization. Time discretization is accomplished by a finite-difference approximation. This leads to a system of $N + 1$ linear equations in matrix form for each solute (Eq. 5 below). These equations can be used to calculate the concentrations of each solute at the $N + 1$ radial positions at time $t + \Delta t$ from those at time t. Partly as a convenience to our readers and partly because of subscripting errors in one of the original references (3), we briefly present below the finite-element equations and matrix elements.

The time coordinate t is divided into intervals of length Δt, and the space coordinate r is divided into intervals of length $h = (r_b - r_m)/N$. For each solute, the functions C, f, ε, and η are then represented by the vectors $\mathbf{C}_n$, $\mathbf{f}_n$, $\boldsymbol{\varepsilon}_n$ and $\boldsymbol{\eta}_n$ whose elements are the values of the functions at radii r_m, $r_m + h$, $r_m + 2h, \ldots, r_b$ at time $t_0 + n\Delta t$, where n is an integer. Starting with $n = 0$ and the initial conditions ($\mathbf{C}_0$s for each solute), $\mathbf{C}_{n+1}$s are recursively calculated from the $\mathbf{C}_n$s by solving, for each solute, a matrix equation of the form

$$(\boldsymbol{B} + \Delta t D_0 \boldsymbol{A}^1 - \Delta t s_0 \omega^2 \boldsymbol{A}^2)\mathbf{C}_{n+1} = \boldsymbol{B}(\mathbf{C}_n + \Delta t \mathbf{f}_n) + \Delta t D_0(\boldsymbol{U}\mathbf{C}^U + \boldsymbol{V}\mathbf{C}^V + \boldsymbol{W}\mathbf{C}^W) - \Delta t s_0 \omega^2 \boldsymbol{A}^2 \mathbf{C}^A \quad [5]$$

where $\boldsymbol{B}$, $\boldsymbol{A}^1$, $\boldsymbol{A}^2$, $\boldsymbol{U}$, $\boldsymbol{V}$, and $\boldsymbol{W}$ are matrices whose elements depend only on r_m, r_b, and h and $\mathbf{C}^U$, $\mathbf{C}^V$, $\mathbf{C}^W$, and $\mathbf{C}^A$ are vectors that are calculated at each iteration as follows:

$$\mathrm{C}_i^U = (\boldsymbol{\eta}_n)_i \cdot (\mathbf{C}_n)_{i+1} \qquad (i = 1, \ldots, \mathrm{N})$$

$$\mathrm{C}_i^V = (\boldsymbol{\eta}_n)_i \cdot (\mathbf{C}_n)_i \qquad (i = 1, \ldots, N + 1)$$

$$\mathrm{C}_i^W = (\boldsymbol{\eta}_n)_{i+1} \cdot (\mathbf{C}_n)_i \qquad (i = 1, \ldots, N)$$

$$\mathrm{C}_i^A = (\boldsymbol{\varepsilon}_n)_i \cdot (\mathbf{C}_n)_i \qquad (i = 1, \ldots, N + 1).$$

The vectors $\mathbf{f}_n$, $\boldsymbol{\varepsilon}_n$, and $\boldsymbol{\eta}_n$ are calculated at each iteration from the $\mathbf{C}_n$s.

$\boldsymbol{A}^1$, $\boldsymbol{A}^2$, $\boldsymbol{B}$, and $\boldsymbol{V}$ are tridiagonal $N + 1$ by $N + 1$ matrices. $\boldsymbol{U}$ and $\boldsymbol{W}$ are $N + 1$ by N matrices that contain only $2N$ nonzero elements. Let $r_i = r_m + (i - 1)h$ for $i = 1, \ldots, N + 1$. Then the matrix elements are calculated as follows:‡

$$A_{1,1}^1 = (r_1/h) + 1/2; \; A_{i,i}^1 = 2r_i/h \qquad (i = 2, \ldots, N)$$

$$A_{N+1,N+1}^1 = (r_{N+1}/h) - 1/2$$

$$A_{i,i-1}^1 = A_{i-1,i}^1 = (-r_i/h) + 1/2 \qquad (i = 2, \ldots, N + 1)$$

$$A_{1,1}^2 = (-r_1^2/2) - (r_1 h/3) - h^2/12;$$

$$A_{i,i}^2 = -2r_i h/3 \qquad (i = 2, \ldots, N)$$

$$A_{N+1,N+1}^2 = (r_{N+1}^2/2) - (r_{N+1}h/3) + h^2/12$$

$$A_{i,i-1}^2 = (r_i^2/2) - (2r_i h/3) + h^2/4 \qquad (i = 2, \ldots, N + 1)$$

$$A_{i,i+1}^2 = (-r_i^2/2) - (2r_i h/3) - h^2/4 \qquad (i = 1, \ldots, N)$$

$$B_{1,1} = (r_1 h/3) + h^2/12; \; B_{i,i} = 2r_i h/3 \qquad (i = 2, \ldots, N)$$

$$B_{N+1,N+1} = (r_{N+1}h/3) - h^2/12$$

$$B_{i,i-1} = B_{i-1,i} = (r_i h/6) - h^2/12 \qquad (i = 2, \ldots, N + 1)$$

$$V_{1,1} = (r_i/2h) + 1/6; \; V_{i,i} = r_i/h \qquad (i = 2, \ldots, N)$$

$$V_{N+1,N+1} = (r_{N+1}/2h) - 1/6$$

$$U_{i+1,i} = -U_{i,i} = -V_{i+1,i} = (r_i/2h) + 1/6 \qquad (i=1, \ldots, N)$$

$$W_{i,i} = -W_{i+1,i} = -V_{i,i+1} = (r_i/2h) + 1/3 \qquad (i = 1, \ldots, N).$$

All other matrix elements are zero, allowing great saving in computer storage space.

The right side of Eq. 5 can be reduced to a single vector $\mathbf{R}_n$ by performing the indicated operations. The expression in parentheses on the left side of the Eq. 5 reduces to a tridiagonal matrix $\boldsymbol{M}$, which needs to be recalculated only if Δt or ω change. The resulting equation, $\boldsymbol{M}\mathbf{C}_{n+1} = \mathbf{R}_n$, is conveniently solved for $\mathbf{C}_{n+1}$ by a gaussian elimination procedure that requires only $3(N + 1)$ multiplications and $2(N + 1)$ additions (15). Note that use of an "operator" matrix (1–3) to solve the matrix equation is much less efficient because $(N + 1)^2$ multiplications and $N(N + 1)$ additions are required to multiply an operator and a vector, and storage of the operator requires $(N + 1)^2$ locations.§ If f, ε, and η are all zero, then the right side of Eq. 5 simplifies to $\boldsymbol{B}\mathbf{C}_n$. One of the convenient features of the finite-element algorithm is that the additional complications of nonzero f, ε, or η are handled simply by additional terms on the right side of Eq. 5.

The effect of varying the rotor speed can be handled in either of two ways. The current value of ω can be substituted into Eq. 5 at each iteration, requiring the recalculation of $\boldsymbol{M}$. Alternately, $s\omega^2$ can be treated as a single function. Then, $\varepsilon = 1 - (s\omega^2/s_0\omega_0^2)$ and ω in Eq. 5 is replaced by ω_0, a constant. The latter method is computationally more rapid if ε is non-zero anyway, otherwise the two methods are approximately equally efficient.

The accuracy of the calculations depends on h and Δt, and calculated solutions converge to the true solutions as h and Δt approach zero (neglecting roundoff error). In general, large $\partial C/\partial r$ values require small h, and large $\partial C/\partial t$ values require small Δt. The values of h and Δt can be selected independently and can easily be changed during the simulation in response to changes in the magnitudes of the gradients. Checks on the correctness of our simulation algorithm have been reported previously (15). We have added checks for nonzero f, ε, and η, including obtaining the correct concentration distribution at sedimentation equilibrium and determining the weight average s from sedimentation velocity runs.

RESULTS AND DISCUSSION

Our method is illustrated by the following applications, in which experimental concentration data have been simulated and provided as input to the Gauss–Newton algorithm. We simulated the effects of experimental error by adding normally distributed

‡ An apparent subscript error leads to incorrect specification of the $\boldsymbol{A}^2$, $\boldsymbol{U}$, $\boldsymbol{V}$, and $\boldsymbol{W}$, matrix elements by Calverie (3).

§ With the APL computer language, the operator is more efficient for $N \leq 400$ if storage space is not a problem because of the relative efficiency of matrix multiplication in this language compared with the looping required to do the gaussian elimination.

Table 1. Accuracy of estimates of s and D of an ideal protein obtained from simulated experimental concentration data with superimposed random error of various SDs

SD of error in data, mg/ml	Percent error in parameters		Error set*
	s	D	
0.0025	−0.1	−0.4	1
0.010	−0.5	−1.4	1
0.025	−1.3	−3.4	1
0.050	−2.5	−6.6	1
0.025	+1.1	−0.3	2
0.025	−0.4	−1.2	3
0.025	−0.2	−0.3	4

Concentration distributions consisting of 51 equally spaced readings taken at 2 and 4 hr were calculated by using the following parameters: s = 5.73 S, $D = 5.46 \times 10^{-7}$ cm^2/sec, rotor speed = 16,150 rpm, r_m = 6.5 cm, r_b = 6.8 cm, C_0 = 0.8 mg/ml, h = 60 μm, and Δt = 20 sec.
* Different random error sets give different results.

random number sets with various SDs to the simulated data. Table 1 shows the accuracy of estimates of s and D obtained for an ideal protein from concentration distributions measured at 2 and 4 hr during a moderate speed ($\sigma = s\omega^2/D = 3$ cm^{-2}) run. The initial protein concentration of 0.8 mg/ml was chosen to give the maximum readable fringe density (≈400 fringes per cm) at the cell bottom at sedimentation equilibrium using interference optics. A SD of 0.0025 mg/ml roughly corresponds to the level of accuracy attainable with a well-aligned interference optical system (0.01 fringes). In our laboratory, we have found that the SD for absorption optics is typically 0.005–0.01. If the initial concentration had been chosen to obtain $A = 1$ at the cell bottom at equilibrium, then this level of error would yield parameter estimates of about the same accuracy as those given in Table 1 for SDs of 0.025–0.05 mg/ml. For initial estimates within a factor of 2 of the correct values, convergence to the least-square values of s and D to four significant figures required, at most, five iterations of the Gauss–Newton algorithm. Even when initial estimates were off by a factor of 10, convergence required at most 10 iterations.

Table 2 shows the accuracy of estimates of four unknown parameters for an associating monomer–dimer system in rapid equilibrium with constant sedimentation and diffusion coefficients. C_0 was again selected to give the maximum readable fringe density at the cell bottom at sedimentation equilibrium. Note that σ_2 ($s_2\omega^2/D_2$) is not considered an additional unknown because it is assumed to be twice σ_1. As intuitively expected, K_2 and σ_1 are highly negatively correlated (asymptotic correlation coefficient of −0.996). As the number of unknown parameters increases, the accuracy of parameter estimates can be expected to decrease, especially for highly correlated parameters, and it is therefore helpful to make reasonable assumptions that reduce the number of unknown parameters.

When solutes are in rapid equilibrium, determination of individual association and dissociation rate constants is meaningless and only equilibrium constants can be determined. Equilibrium constants do not appear in f terms but can be introduced as unknown parameters by either of two methods. The differential equations for each solute in the equilibrium can be summed, as previously discussed, so that equilibrium constants appear directly in Eqs. 3 and 4 and rate constants appearing in f terms drop out. Alternatively, for each equilibrium constant, either the association or dissociation rate constant can be fixed at some value large enough to be compatible with the assumption of rapid equilibrium but not so large that an extremely small Δt would be required for sufficient accuracy in the simulation.

Table 2. Accuracy of estimates of four unknown parameters for an ideal monomer–dimer system in rapid equilibrium obtained from simulated data with various error levels

Data used; hr*	SD of error in data, mg/ml	Percent error in parameter			
		σ_1	D_1	D_2	K_2
0, 4	0.0025	+1.3	+1.6	−3.5	−9
0, 4	0.0025†	+2.9	−1.3	−1.4	−23
0, 6	0.0025	+4.2	−4.0	+6.0	−36
0, 4, 8, 12	0.0025	+1.0	−0.6	+0.2	−9
0, 2, 4, 6‡	0.0025	−0.6	+1.1	−0.9	+8
0, 2, 4, 6	0.005	−1.2	+2.2	−1.8	+17
0, 2, 4, 6	0.01	−2.4	+4.4	−3.3	+36
0, 2, 4, 6, 8, 10	0.01	+3.2	+1.8	−4.2	−25
0, 2, 4, 6, 8, 10	0.01†	+2.5	+1.1	−3.7	−21

Concentration distributions for various times consisting of 51 equally spaced readings were calculated by using the following parameters: $\sigma_1 = \sigma_2/2 = 2$ cm^{-2} (13,187 rpm), $D_1 = 5.46 \times 10^{-7}$ cm^2/sec, $D_2 = 3.76 \times 10^{-7}$ cm^2/sec, K_2 = 1.6 (mg/ml)$^{-1}$, r_m = 6.5 cm, r_b = 6.8 cm, C_0 = 0.7 mg/ml, h = 60 μm, Δt = 20 sec.
* Times to reach 90 and 99% of sedimentation equilibrium, as determined by calculation of K_2 from the meniscus concentrations, were 21 and 29 hr, respectively.
† A different random error set was used than for the above line.
‡ When these data and initial parameter estimates of σ_1 = 1.5 cm^{-2}, $D_1 = 6.01 \times 10^{-7}$ cm^2/sec, $D_2 = 4.77 \times 10^{-7}$ cm^2/sec, and K_2 = 0.4 (mg/ml)$^{-1}$ were used, the parameters converged to their final values to four significant figures in 15 iterations of the Gauss–Newton algorithm with a total of 20 step halvings.

The other rate constant is then considered an unknown parameter, and the equilibrium constant is calculated as the appropriate ratio of the fixed rate constant and the value of the unknown rate constant determined by the nonlinear algorithm.

Computer costs are dictated primarily by the number of repeated simulations. The total number of simulations is equal to $I(p + 1) + H + 1$, where I is the number of iterations of the Gauss-Newton algorithm, p is the number of unknown parameters, and H is the total number of step halvings. The cost per simulation is proportional to N, to the number of Δt time periods simulated, and to the number of solutes considered in the model. Each simulation for the problem in Table 1 required ≈1.0 sec of central processing unit time.¶ Simulations that include effects such as solute interactions or nonconstant s, D, or ω^2 result in a slight to moderate increase in the cost per simulation.

In conclusion, we emphasize that the solution of the inverse problem presented here has the very desirable property of fitting parameters to experimental data in a manner such that they theoretically converge to the exact least-square values as Δt and h become small. Of additional importance is the fact that the finite-element method is ideally suited to simulating data for many other processes that involve solute fluxes, such as electrophoresis and chromatography (3). With only slight modification, the program developed here for least-squares estimation of physical parameters from ultracentrifuge data can be used to obtain parameters from such processes as well.

This work was supported by the National Institutes of Health Grant HL-11802 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.

¶ This refers to the time for double-precision FORTRAN calculations on the IBM VM-370/168 system at Cornell University, Ithaca, NY. A majority of the computations were actually performed on a Floating Point Systems 190L Array Processor that uses a precision intermediate between single and double.

1. Claverie, J. M., Druex, H. & Cohen, R. (1975) *Biopolymers* **14**, 1685–1700.
2. Cohen, R. & Claverie, J. M. (1975) *Biopolymers* **14**, 1701–1716.
3. Claverie, J. M. (1976) *Biopolymers* **15**, 843–857.
4. Jennrich, R. I. & Sampson, P. F. (1968) *Technometrics* **10**, 63–72.
5. Jennrich, R. I. & Raltson, M. L. (1979) *Annu. Rev. Biophys. Bioeng.* **8**, 195–238.
6. Fujita, H. (1975) *Foundations of Ultracentrifugal Analysis* (Wiley, New York).
7. Archibald, W. J. (1938) *Phys. Rev.* **52**, 746–752.
8. Archibald, W. J. (1938) *Phys. Rev.* **54**, 371–374.
9. Bethune, J. L. & Kegeles, G. (1961) *J. Phys. Chem.* **65**, 1761–1764.
10. Bethune, J. L. (1970) *J. Phys. Chem.* **74**, 3837–3845.
11. Goad, W. B. (1970) in *Interacting Macromolecules*, ed. Cann, J. R. (Academic, New York), pp. 207–223.
12. Cox, D. J. (1978) **48**, 212–242.
13. Dishon, M., Weiss, G. H. & Yphantis, D. A. (1966) *Biopolymers* **4**, 449–455.
14. Raltson, M. L. & Jennrich, R. I. (1978) *Technometrics* **20**, 7–14.
15. Todd, G. P. & Haschemeyer, R. H. (1981) *Biopolymers*, in press.