Computer Physics Communications 27 (1982) 213–227
North-Holland Publishing Company

A CONSTRAINED REGULARIZATION METHOD FOR INVERTING DATA REPRESENTED BY LINEAR ALGEBRAIC OR INTEGRAL EQUATIONS

Stephen W. PROVENCHER

EMBL, Postfach 10.2209, D-6900 Heidelberg, Fed. Rep. Germany

Received 15 December 1981

CONTIN is a portable Fortran IV package for inverting noisy linear operator equations. These problems occur in the analysis of data from a wide variety of experiments. They are generally ill-posed problems, which means that errors in an unregularized inversion are unbounded. Instead, CONTIN seeks the optimal solution by incorporating parsimony and any statistical prior knowledge into the regularizor and absolute prior knowledge into equality and inequality constraints. This can greatly increase the resolution and accuracy of the solution. CONTIN is very flexible, consisting of a core of about 50 subprograms plus 13 small "USER" subprograms, which the user can easily modify to specify special-purpose constraints, regularizors, operator equations, simulations, statistical weighting, etc. Special collections of USER subprograms are available for photon correlation spectroscopy, multicomponent spectra, and Fourier–Bessel, Fourier and Laplace transforms. Numerically stable algorithms are used throughout CONTIN. A fairly precise definition of information content in terms of degrees of freedom is given. The regularization parameter can be automatically chosen on the basis of an F-test and confidence region. The interpretation of the latter and of error estimates based on the covariance matrix of the constrained regularized solution are discussed. The strategies, methods and options in CONTIN are outlined. The program itself is described in the following paper.

1. Introduction

Most experiments in the natural sciences are indirect. That is, the observed data, y_k, are related to the desired function or vector $\boldsymbol{x}$ by operators O_k,

$$y_k = O_k \boldsymbol{x} + \epsilon_k, \quad k = 1,\ldots,N_y, \tag{1.1}$$

where the ϵ_k are unknown noise components. One is then faced with the inverse problem of estimating $\boldsymbol{x}$ from the noisy measurements y_k. Often the O_k are approximated by linear integral operators, and eq. (1.1) can be written

$$y_k = \int_a^b F_k(\lambda)\, s(\lambda)\, \mathrm{d}\lambda + \sum_{i=1}^{N_L} L_{ki}\beta_i + \epsilon_k, \tag{1.2}$$

where the O_k have been replaced by the known functions $F_k(\lambda)$ and $\boldsymbol{x}$ has been replaced by the function $s(\lambda)$, which is to be estimated. The extra optional sum over the known L_{ki} and N_L unknown β_i permits, for example, a constant background term, β_1, to be included by setting $N_L = 1$ and all the $L_{k1} = 1$.

CONTIN is a general program package for solving eq. (1.2) and systems of linear algebraic equations. Eq. (1.2) includes Fredholm and Volterra integral equations of the first kind, and it occurs in far too many kinds of experiments to attempt to list here. However, five general types of causes of eq. (1.2) are: (1) *imperfect input* into the system being studied, e.g., a spread in energy, space or time of an input that should be perfectly sharp, as in polychromaticity and slit width and length effects in X-ray and neutron scattering [1]; (2) *imperfect detection* of the output of the system being studied, e.g., when the impulse response of the detection system has significant spread due to optical, mechanical, electronic of physical limitations, as in convolutions of energy spectra with detector responses [2]; (3) *imperfect systems* being studied, e.g., due to practical requirements of mass or geometry, as in the use of thick (rather than infinitely thin) targets in cross section and decay

0010-4655/82/0000–0000/$02.75

studies [3]; (4) *indirect measurements* – even if causes (1)–(3) were negligible, the function of interest is still often related to the data by eq. (1.2), e.g., by a Fourier transform in diffraction experiments or a Laplace transform in relaxation experiments [4]; (5) *multicomponent systems* can be a mixture of a large number of independent components, each with its characteristic response to the input, and the object is to estimate the concentration distribution of the components, e.g., with photon correlation [5] or circular dichroic spectroscopy [6]. Often several of the above causes are present in a single experiment. However, if they are all described by linear integral operators, then they can be combined into a single operator to form eq. (1.2), as with low angle scattering data requiring correction for slit width and height, polychromaticity and Fourier transformation [1].

For most $F_k(\lambda)$, estimating $s(\lambda)$ in eq. (1.2) is an *ill-posed problem*. This is best illustrated by an example [7,8] using the version of the Riemann–Lebesgue lemma that says that, in the usual case that the $F_k(\lambda)$ are absolutely integrable,

$$\lim_{\omega\to\infty}\int_a^b F_k(\lambda)\sin(\omega\lambda)\,d\lambda = 0. \tag{1.3}$$

This means that, even for arbitrarily small $\epsilon_k \neq 0$ and an arbitrarily large amplitude A, there still exists an ω such that $s(\lambda) + A\sin(\omega\lambda)$ still satisfies eq. (1.2) to within the experimental errors, ϵ_k. Thus there exists a large (generally infinite) set, Ω, of possible solutions, all satisfying eq. (1.2) to within experimental error. Even worse, the members of Ω can have arbitrarily large deviations from each other and therefore from the true solution (as illustrated with the arbitrary A above); i.e., the errors are unbounded.

A common special case of eq. (1.2) is

$$y(t_k) = \int_a^b F(\lambda, t_k)\, s(\lambda)\, d\lambda + \epsilon_k. \tag{1.4}$$

In some cases there are exact (when $\epsilon_k = 0$) analytic inversion formulas for $s(\lambda)$. However, any such "exact" inversion ignoring the ϵ_k will select from Ω one member, which will depend on the ϵ_k. In view of the unboundedness of the errors, it is almost certain that this member will be a very poor estimate of the true $s(\lambda)$. Therefore exact inversion formulas or iterative algorithms converging to them cannot be directly applied to experimental data. Even with popular modifications like low-pass filtering the data or solution, the following difficulties cause loss in accuracy: (1) truncation, extrapolation and interpolation errors can occur in evaluating integrals in the inversion formulas; (2) the data are not properly statistically weighted, since the contribution of each data point to the solution is determined solely by such things as integration formulas, spacing between the t_k, etc.; (3) prior knowledge, such as $s(\lambda) \geq 0$, cannot be easily imposed.

Another common strategy for stabilizing an ill-posed problem has been to reduce the number of degrees of freedom by fitting a parameterized model to the data or to use a coarse histogram or grid to represent the solution. Here there is the dilemma that serious errors can result if the number of degrees of freedom is too small (because of an inadequate model) or too large (because of instabilty of the solution to noise). The correct number of degrees of freedom is usually very difficult to specify beforehand. CONTIN attempts to automatically select from Ω the best compromise between a stable solution and an adequate model. The number of degrees of freedom is not fixed beforehand. It is automatically determined during the analysis by the constraints and the regularizor, which can adapt itself to the noise level and amount of data.

CONTIN permits combinations of three types of strategy for doing this: (1) *Absolute prior knowledge* can often greatly increase the accuracy and resolution of a solution. For example, if it is known that $s(\lambda) \geq 0$, this constraint is very powerful at eliminating the many oscillating members of Ω that occur because of eq. (1.3). (2) *Statistical prior knowledge* of the mean and covariance of the solution and the ϵ_k permits the optimal (i.e., minimum mean-square error for any linear unbiased estimate) solution to be directly obtained [9]. While such prior knowledge is seldom precise, in some cases the lack of prior knowledge can lead to a natural covariance and mean for the solution via the principle of exchangeability [10], which has been very useful in analyzing complex multicomponent spectra [6]. (3) The principle of *parsimony*

says that, of all the members of Ω that have not been eliminated by prior knowledge, choose the simplest one, i.e., the one that reveals the least amount of detail or information that was not already known or expected. While this solution may not have all the detail of the true solution, the detail that it does have is necessary to fit the data and therefore less likely to be artifact.

The details of how these strategies are best applied can obviously vary greatly from one problem to the next. Therefore, CONTIN has been designed to be very flexible. Detailed instructions and examples on the use of CONTIN are available in the following paper and in the Users Manual [11]. In this paper, we only outline the design, capabilities, usage and computational methods of CONTIN.

2. Design of CONTIN

CONTIN is a self-contained program with about 5000 lines and 66 subprograms. Every attempt was made to adhere to 1966 ANSI Fortran IV standards, and CONTIN is meant to be fully portable. The only necessary changes should be specifying four machine dependent variables and perhaps opening input and output files. It has been tested on DEC 1090, IBM 360/91, Amdahl 470 and VAX 11/780 machines and has been distributed to about 100 laboratories with various computers.

There are two versions, 1SP and 1DP, the latter with more parts in DOUBLE PRECISION. Version 1DP is recommended, except for machines like CDC with 60-bit REAL representations. Updated versions 2SP and 2DP are now being published in the CPC Library. The main new features are automatic internal scaling and more detailed error estimates. Otherwise they are basically the same as versions 1SP and 1DP, and the discussions in this paper apply to all versions. The high-speed storage requirements and running times are problem dependent but are typically 50000 36-bit words and 60 s on a DEC 1090 KL.

CONTIN has been designed to be very flexible, but still easy to use. There are more than 40 "control variables" that control the options in CONTIN. They are all set to commonly used default values in the BLOCK DATA subprogram, and the user need only input the ones that are to be changed. Similarly, there are 13 short, fully documented "USER" subprograms that usually do not need to be changed. However, they can be easily modified by the user, and this makes CONTIN very flexible because they define most aspects of the problem and strategies described in section 1.

There are also "applications packages", which are collections of USER subprograms for a specific problem together with a set of test data. Applications packages are documented in ref. [11] for the inversion of Laplace transforms [12], Fourier transforms in low-angle scattering, and Fourier–Bessel transforms in fiber diffraction [13] and for the analysis of multicomponent systems with dynamic light scattering [5] and circular dichroism [6].

Part of the computation time is proportional to the cube of the number of parameters used to represent the solution, and it becomes unreasonably expensive when there are more than about 100 parameters. This is usually more than adequate when the integral over λ in eq. (1.2) is one-dimensional, but not when the domain of $s(\lambda)$ is two- or three-dimensional. In these cases, a more specialized regularization technique, such as maximum entropy with a very efficient optimization algorithm [14] would have to be used.

3. Methods of solution

3.1. Formation of linear equations

The first step is to convert eq. (1.2) to a system of linear algebraic equations. CONTIN can automatically do this by numerical integration of eq. (1.2),

$$y_k = \sum_{m=1}^{N_g} c_m F_k(\lambda_m)\, s(\lambda_m) + \sum_{i=1}^{N_L} L_{ki}\beta_i + \epsilon_k, \quad (3.1)$$

where c_m are the weights of the quadrature formula. The solution, $s(\lambda)$, is then determined at the

N_g grid points λ_m. Eq. (3.1) can be rewritten as

$$y_k = \sum_{j=1}^{N_x} A_{kj} x_j + \epsilon_k, \tag{3.2}$$

where

$$N_x = N_g + N_L, \tag{3.3}$$

and the $N_x \times 1$ vector $\boldsymbol{x}$ contains all the unknowns, $s(\lambda_m)$ and β_i, and the $N_y \times N_x$ matrix A contains the $c_m F_k(\lambda_m)$ and L_{ki}. CONTIN can, of course, handle systems of linear algebraic equations, which are already in the form of eq. (3.2). They can be ill-conditioned, linearly dependent, or even underdetermined with $N_y < N_x$.

As an alternative to numerical integration, CONTIN also permits the solution to be represented as

$$s(\lambda) = \sum_{j=1}^{N_g} x_j B_j(\lambda), \tag{3.4}$$

where the $B_j(\lambda)$ are a convenient basis set of functions specified by the user. Substitution of eq. (3.4) into (1.2) yields eq. (3.2) with

$$A_{kj} = \int_a^b F_k(\lambda)\, B_j(\lambda)\, d\lambda, \quad j = 1,\ldots,N_g. \tag{3.5}$$

With either representation, it is assumed that the errors in going from eq. (1.2) to (3.2) are much less than the ϵ_k. This is usually no problem, even with numerical integration and $N_g \approx 40$. The best verification is to repeat the analysis with a larger N_g and see that the results do not change. When this is the case, numerical integration is usually more convenient, because only the $F_k(\lambda_m)$ have to be evaluated and not the integrals in eq. (3.5). CONTIN also has many options that make numerical integration very easy to use.

Sometimes, however, it might be necessary to use the representation in eq. (3.4) to avoid excessive numerical integration errors, e.g., because $F_k(\lambda)$ varies very rapidly with λ. Eq. (3.5) can then be evaluated with sufficient accuracy, either analytically or numerically with a fine grid. This need only be done once; the matrix A can be stored on a file for later use (see IUNIT in section 4.2). Therefore complicated $F_k(\lambda)$ can be handled economically. As we shall see below, there are important advantages when the $B_j(\lambda)$ are B-splines with equally spaced knots [15,16].

3.2. Constraints

CONTIN permits the following general linear inequality and equality constraints to be imposed on the solution vector, $\boldsymbol{x}$:

$$\sum_{j=1}^{N_x} D_{ij} x_j \geq d_i, \quad i = 1,\ldots,N_{ineq}, \tag{3.6}$$

$$\sum_{j=1}^{N_x} E_{ij} x_j = e_i, \quad i = 1,\ldots,N_{eq}, \tag{3.7}$$

where N_{ineq}, N_{eq}, and the arrays D, E, $\boldsymbol{d}$ and $\boldsymbol{e}$ can be specified by the user. These can be very useful for imposing absolute prior knowledge, strategy (1) of section 1.

Imposing prior knowledge of the nonnegativity of $s(\lambda)$, for example, can greatly increase the accuracy and resolution of a solution. When numerical integration is used, we simply have from eqs. (3.1) and (3.2) $\boldsymbol{x} = \boldsymbol{s}$, where $\boldsymbol{s}$ is the $N_g \times 1$ vector with elements $s(\lambda_m)$ (and we have ignored the last N_L elements of $\boldsymbol{x}$ to simplify the notation). Therefore $\boldsymbol{s} \geq \boldsymbol{0}$ (where $\boldsymbol{0}$ will denote a vector with all components zero) is imposed by setting $N_{ineq} = N_g$, $\boldsymbol{d} = \boldsymbol{0}$, and $D = I$ in eq. (3.6), where I will denote an identity matrix. CONTIN has a control variable (NONNEG in section 4.3) for automatically doing this.

When eq. (3.4) is used instead of numerical integration, then from eq. (3.4) we have

$$\boldsymbol{s} = B\boldsymbol{x}, \tag{3.8}$$

where B is the $N_g \times N_g$ matrix with elements $B_{ij} = B_j(\lambda_i)$. Therefore $\boldsymbol{s} \geq \boldsymbol{0}$ is imposed as above, except with $D = B$ instead of $D = I$. When the $B_j(\lambda)$ are normalized cubic B-splines with N_g knots of equal spacing, Δ, from $\lambda = a$ to $\lambda = b$, then B is simply a tridiagonal Töplitz matrix with nonzero elements $B_{jj} = 2/3$ and $B_{j,j-1} = 1/6$ [15,16].

In order to represent the solution to within its inherent resolution, N_g must be large enough so that Δ is always substantially less than the point spread function of the solution (see section 3.7). In

this case, imposing a constraint like $\boldsymbol{s} \geq \boldsymbol{0}$ is nearly equivalent to imposing $s(\lambda) \geq 0$ on the entire interval $a \leq \lambda \leq b$. When B-splines are used, constraints like $s(\lambda) \geq 0$ can be *exactly* imposed by constraining $\boldsymbol{x} \geq \boldsymbol{0}$. Similarly constraints on the shape of $s(\lambda)$ can be exactly imposed by using $\boldsymbol{x}$ instead of $\boldsymbol{s}$. These nice properties are related to Schoenberg's variation diminishing spline approximation [16]. However, because this is a lower order approximation, a larger N_g might have to be used. The constraints could also be made more accurate by imposing them at more λ values than just the N_g grid points; i.e., with $N_{ineq} > N_g$. Usually, neither of these two measures are necessary in practice.

3.3. Regularization

In going from eq. (1.2) to (3.2), we go from an ill-posed problem to an ill-conditioned one. That is, there will generally be a large set, Ω', of vectors $\boldsymbol{x}$ all of which satisfy eq. (3.2) to within experimental error, ϵ_k. The constraints in eqs. (3.6) and (3.7) can eliminate many unacceptable members of Ω', but there are usually many remaining with large variations from each other.

We could take the ordinary constrained weighted least-squares solution to eq. (3.2), i.e., the $\boldsymbol{x}$ that satisfies

$$V(0) = \| M_\epsilon^{-1/2}(\boldsymbol{y} - A\boldsymbol{x}) \|^2 = \text{minimum} \qquad (3.9)$$

subject to the constraints in eqs. (3.6) and (3.7), where $\| \cdot \|$ is the Euclidean norm, M_ϵ is the (positive definite) covariance matrix of the ϵ_k, and $\boldsymbol{y}$ is the $N_y \times 1$ vector with elements y_k. However, this solution is just one member of Ω', usually strongly influenced by the ϵ_k. Considering the large variation among members of Ω', it is very unlikely that this solution will be close to the true solution.

We therefore need to impose statistical prior knowledge and parsimony, strategies (2) and (3) of section 1. These can often be imposed in such a way that the optimal solution satisfies

$$V(\alpha) = \| M_\epsilon^{-1/2}(\boldsymbol{y} - A\boldsymbol{x}) \|^2 + \alpha^2 \| \boldsymbol{r} - R\boldsymbol{x} \|^2 = \text{minimum} \qquad (3.10)$$

subject to the constraints in eqs. (3.6) and (3.7). The second term on the right is called the *regularizor*. Its form is determined by the $N_{reg} \times 1$ and $N_{reg} \times N_x$ arrays $\boldsymbol{r}$ and R, which (along with N_{reg}) can be specified by the user. The regularizor penalizes an $\boldsymbol{x}$ for deviations from behavior expected on the basis of statistical prior knowledge or parsimony. The relative strength of the regularizor is determined by α, the *regularization parameter*. The way that CONTIN helps in the choice of α is discussed in section 3.6. We now outline some common types of regularizors.

3.3.1. Statistical prior knowledge

Sometimes $\bar{\boldsymbol{x}}$ and M_x, the mean and covariance matrix of $\boldsymbol{x}$, can be specified. The solution obtained by putting $R = M_x^{-1/2}$, $\boldsymbol{r} = R\bar{\boldsymbol{x}}$ and $\alpha = 1$ in eq. (3.10) is optimal in that it has the minimum expected mean-square error of all linear unbiased estimates [9]. This solution is optimal even if $\boldsymbol{\epsilon}$ and $\boldsymbol{x}$ are not normally distributed. If they are normally distributed, then the solution is the maximum likelihood one and has the minimum expected mean-square error of all unbiased estimates (not just linear ones).

Priors for $\bar{\boldsymbol{x}}$ and M_x may come from previous experiments, or one may wish to test if the solution deviates significantly from the expected one. Even when precise information on $\bar{\boldsymbol{x}}$ and M_x is not available, the *lack* of such knowledge can dictate reasonable priors for $\bar{\boldsymbol{x}}$ and M_x. For example, if all the x_j are the same type of variable, e.g., fractions of library spectra [6], then the principle of exchangeability [10] suggests $\bar{x}_j = 1/N_x$ for $j = 1, \ldots, N_x$ and $M_x = I$, an identity matrix. This is similar to ridge regression [17], which is widely used, even when this Bayesian justification does not apply.

3.3.2. Parsimony

Often statistical prior knowledge is insufficient, and parsimony [strategy (3) of section 1] must be used to select the "simplest" member of Ω'. The definition of "simplest" obviously depends on the problem. Often, however, sudden sharp changes or extra peaks in $s(\lambda)$ would yield significant unexpected information. "Smoothness with minimum number of peaks" would then be a good definition of parsimony, since it would tend to guard against

significant artifacts. CONTIN has options for restricting the number of extrema (see section 4.7). A good measure of the lack of smoothness, and therefore a good regularizor to impose smoothness, is [7,16]:

$$\|\boldsymbol{r} - R\boldsymbol{x}\|^2 = \int_a^b [s''(\lambda)]^2 \, d\lambda, \tag{3.11}$$

where prime denotes differentiation.

When numerical integration with equally spaced λ_m in eq. (3.1) is used, eq. (3.11) can be approximated by the sum of the squares of the second differences, $s(\lambda_{m-1}) - 2s(\lambda_m) + s(\lambda_{m+1})$. This is done by setting $\boldsymbol{r} = \boldsymbol{0}$ and $R = P$, where P is the $(N_g + 2) \times N_g$ matrix of bandwidth three,

$$P = \begin{bmatrix} 1 & & & & & & \\ -2 & 1 & & & & & 0 \\ 1 & -2 & 1 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & 1 & -2 & 1 & & \\ 0 & & & 1 & -2 & & \\ & & & & 1 & & \end{bmatrix}. \tag{3.12}$$

[The constant factor, $\Delta^{-2} = (\lambda_m - \lambda_{m-1})^{-2}$, has been left out of eq. (3.12) because it can be absorbed into α, the regularization parameter, which is usually left as a parameter to be determined by the methods of section 3.6.] The first two rows are formed by taking second differences at two external points, $\lambda_0 = a - \Delta$ and $\lambda_{-1} = a - 2\Delta$. The first row imposes the condition $s(\lambda_{-1}) = s(\lambda_0) = 0$ and the second row $s(\lambda_0) = 0$. These two rows should be deleted unless there is prior knowledge that $s(\lambda) = 0$ in this region. Otherwise they will tend to incorrectly bias $s(\lambda)$ to smoothly approach zero near $\lambda = a$. Similarly, the last two rows should be deleted unless there is prior knowledge that $s(b + \Delta) = s(b + 2\Delta) = 0$. Section 4.5 explains how the user can specify which rows to delete and have CONTIN automatically set up the regularizor.

When eq. (3.4) is used instead of numerical integration, one can also apply the second-difference operator to eq. (3.8), simply obtaining $R = PB$ instead of $R = P$. However, eq. (3.11) can be specified exactly by setting $\boldsymbol{r} = \boldsymbol{0}$ and $R = G^{1/2}$, where G is the $N_x \times N_x$ matrix with elements

$$G_{ij} = \int_a^b B_i''(\lambda) \, B_j''(\lambda) \, d\lambda. \tag{3.13}$$

When the $B_j(\lambda)$ are cubic B-splines with equally spaced knots between $\lambda = a$ and $\lambda = b$, G is a symmetric heptadiagonal matrix with the elements along each diagonal equal except for the first three and the last three, $6G_{jj} = 2, 8, 14, 16, \ldots, 16, 14, 8, 2$; $6G_{j,j\pm1} = -3, -6, -9, \ldots, -9, -6, -3$; $G_{j,j\pm2} = 0$; $6G_{j,j\pm3} = 1$, for $j = 1, \ldots, N_x$. (We have again absorbed a normalization constant into α.) If it is known that $s(\lambda) = 0$ outside of the region covered by the splines, then the integration limits in eq. (3.13) can be extended to $\pm\infty$. There are then no more truncation effects at the boundaries and all elements of each diagonal are equal; i.e., G becomes a Töplitz matrix with $6G_{ji} = 16, -9, 0, 1$, for $i = j$, $j \pm 1$, $j \pm 2$, $j \pm 3$. This is eq. (15) of ref. [18], except for typographical errors. An integration limit in eq. (3.13) should not be extended to $\pm\infty$ unless there is prior knowledge that $s(\lambda) = 0$ outside the range of the splines at that boundary; otherwise $s(\lambda)$ is biased toward zero at that boundary.

Parsimony may often be considered a vague type of statistical prior knowledge. For example, experience has shown that many (continuous approximations to) polymer molecular weight distributions are well represented by smooth functions, and there are often good chemical kinetic grounds to expect this. More generally, $s(\lambda)$ is often a macroscopic composite of a large number of microscopic contributions and, for example, Jaynes' principle of maximum degeneracy for prior probability laws can naturally lead to the maximum entropy regularizor [19]. This regularizor is nonlinear and cannot be put in the form of eq. (3.10), but it also tends to smooth the solution. We have found eq. (3.11) to be a very useful general-purpose smoothing regularizor. It is probably more effective at suppressing extra isolated peaks in $s(\lambda)$ than maximum entropy, which takes no direct account of correlation or connectivity between neighboring grid points of $s(\lambda)$.

3.4. Computational methods

Once the regularizor and constraints are specified, CONTIN finds the unique solution to eq. (3.10) subject to the constraints in eq. (3.6) and (3.7). The computational methods are outlined in the appendix, and the steps are only summarized here. First the (possibly very long) arrays y and A are orthogonally transformed sequentially, row-by-row, so that the $N_y \times N_x$ array A need never be in high-speed storage. Then a series of orthogonal transformations and changes of variables are applied, whereby the equality constraints are eliminated and the regularizor diagonalized, yielding a constrained ridge regression problem. This is transformed into a least distance programming problem, whose unique solution is obtained in a finite number of steps. Numerically stable procedures [20] are used throughout.

3.5. Information content and degrees of freedom

The solution x typically has 40 or more components. However, it should be clear that most inverse problems are so unstable to noise that 40 independent parameters could never be reliably determined in practice. The regularizor and constraints make the x_j correlated. It is therefore important to know how many degrees of freedom (i.e., truly independent parameters or pieces of information) there are in a solution.

Several methods of determining N_{DF}, the number of degrees of freedom, have been proposed. One of the earliest uses a sampling theorem for a solution of compact support [21] and another the eigenvalues of the kernel of the integral operator [22]. However, these are not appropriate for digital data, which are discrete rather than continuous and often have nonstationary noise and nonuniform spacing. Twomey and Howell [23] eliminated all of these problems by using the eigenvalues of the $N_y \times N_y$ Grammian matrix of the $F_k(\lambda)$. This could be applied to eq. (1.2) when N_y is small, $N_L = 0$, and there are no constraints.

One area where N_{DF} is defined very naturally and precisely is in ordinary least squares. For example, if there are enough data and parameters to represent the solution, then the expected value of $V(0)$ in eq. (3.9) is simply $N_y - N_{DF}$. The regularizor complicates matters because an additional bias term, B, occurs,

$$E\{V(\alpha)\} = N_y - N_{DF} + B^2, \tag{3.14}$$

as explained for eq. (A.35). However, we have transformed the problem to ridge regression in eq. (A.13), and Mallows [24] has derived useful statistical properties for this problem when there are no binding inequality constraints. We therefore define N_{DF} so that the precise expressions, such as eq. (3.14) for ordinary least squares and least squares on a subset of regression coefficients still hold when $\alpha > 0$. When this is done with Mallows' C_L statistic [24], which is an estimate of the scaled sum of squared errors in the x_j, we obtain

$$N_{DF} = \sum_{j=1}^{N_{xye}} \delta_j, \tag{3.15}$$

where

$$\delta_j \equiv s_j^2 / \left(s_j^2 + \alpha^2\right) \tag{3.16}$$

and s_j and N_{xye} are defined by eqs. (A.23) and (A.24). If this were done with Mallows' V_L and V_L^*, the variance terms in the scaled sum of squared errors and in $V(\alpha)$, N_{DF} would be

$$\sum_{j=1}^{N_{xye}} \delta_j^2, \tag{3.17}$$

$$\sum_{j=1}^{N_{xye}} \left(2\delta_j - \delta_j^2\right), \tag{3.18}$$

respectively.

Fortunately, these three N_{DF} values are usually quite close. The differences between the terms in eqs. (3.15) and (3.17) and between (3.18) and (3.15) are both $\delta_j(1 - \delta_j)$. These are usually small because the eigenvalues s_j^2 generally decrease rapidly, with few near α^2; δ_j is therefore usually close to zero or one. CONTIN arbitrarily uses N_{DF} in eq. (3.15), which lies between those of eqs. (3.17) and (3.18). It seldom varies from the other two by more than 0.5 for unstable problems (when the s_j^2 decrease rapidly) or 1.0 for more stable problems such as the analysis of multicomponent circular dichroism spectra.

We can obtain an instructive heuristic interpretation of eq. (3.15) by considering the filtering effect of the regularizor. For the simple case in section A.2 we can use eq. (A.19) to write eq. (A.31) as

$$\boldsymbol{x} = K_2 Z H_1^{-1} W \boldsymbol{g}(\alpha), \tag{3.19}$$

where the $N_{xe} \times 1$ vector $\boldsymbol{g}(\alpha)$ has elements

$$g_j(\alpha) = s_j \gamma_j / \left(s_j^2 + \alpha^2\right), \tag{3.20}$$

and contains all the α-dependence of $\boldsymbol{x}$. Thus the sole effect of the regularizor is to multiply the contribution of each of the N_{xye} components γ_j by the factor

$$g_j(\alpha)/g_j(0) = \delta_j, \tag{3.21}$$

and it is natural to define N_{DF} in eq. (3.15) as just the sum of these fractional contributions.

Note from eqs. (A.15) and (A.23) that the s_j are the singular values of $CK_2ZH_1^{-1}$. Thus the s_j (and N_{DF}) contain information on the equality constraints (in K_2) and regularizor (in ZS_1^{-1}) as well as on the properly weighted discrete design matrix A (in C). The s_j are determined by the structure of the problem alone; they are independent of the data. They can therefore be useful in evaluating and comparing experimental designs. In contrast, α is determined by the data (as described in section 3.6). As the noise in the data increases, so does α. When $s_j \neq 0$, δ_j in eq. (3.16) decreases monotonically from 1.0 when $\alpha = 0$ toward 0.0 as $\alpha \sim \infty$.

CONTIN prints the s_j in decreasing order under the heading "SINGULAR VALUES". Problems with smooth $F_k(\lambda)$ tend to be more difficult because the s_j rapidly decrease. For example, with Laplace transforms, N_{DF} can be surprisingly small, even with relatively accurate data. Thus N_{DF} (and other information discussed in section 3.6) can help prevent overinterpreting a solution.

It is important to note the meaning of N_{DF} when there are active inequality constraints. As outlined in section A.2, each binding inequality constraint is eliminated as an equality constraint and does not contribute to N_{DF}. For example, if the true $s(\lambda)$ were approximately a δ-function, accurate data and nonnegativity constraints might result in a solution with $s(\lambda_m) = 0$ for all but two grid points. We would then have $N_{DF} \leq 2$, which means that the solution can be described by two parameters. It does *not* mean that the data are only capable of determining at most two parameters. A lower limit for this information content would be given by the N_{DF} obtained from an analysis without the inequality constraints. This limit neglects the superresolution due to the inequality constraints, and this can be very significant for smooth $F_k(\lambda)$ such as in Laplace transforms.

3.6. Choosing the regularization parameter

There are some cases when the choice of α is clear. For example, with a regularizor constructed from prior statistical knowledge of M_x and $\bar{x}$, as discussed in section 3.3, $\alpha = 1$ would be appropriate.

In most cases, however, it is best to let CONTIN perform a series of solutions where α is gradually increased from a very small value. For each solution, CONTIN outputs α, α/s_1, $V(\alpha)$, $\| M_\epsilon^{-1/2}(\boldsymbol{y} - A\boldsymbol{x}) \|^2$, N_{DF} and

$$\hat{\sigma} \equiv \| M_\epsilon^{-1/2}(\boldsymbol{y} - A\boldsymbol{x}) \| / \left(N_y - N_{DF}\right)^{1/2}, \tag{3.22}$$

as well as several other quantities and optional plots, which will be described below. The principle of parsimony says to take the largest α that is consistent with the data. Considering the way in which N_{DF} was defined in section 3.5, the expected value of $\hat{\sigma}$ is approximately 1.0 as long as α is small enough so that the bias term due to the regularizor is not significant. Therefore, a reasonable choice is the largest α such that $\hat{\sigma} \approx 1.0$.

Unfortunately, there is often an unknown scale factor in M_ϵ; i.e., we know the relative uncertainties and correlations in the data points fairly well, but not their absolute values. In this case the expected value of $\hat{\sigma}$ is unknown, and α must be chosen with another criterion. For this purpose CONTIN also outputs

$$\mathrm{PROB1}(\alpha) \equiv P\left[F_1(\alpha), N_{DF}(\alpha_0), N_y - N_{DF}(\alpha_0)\right], \tag{3.23}$$

where $P(F, n_1, n_2)$ is Fisher's F-distribution with

n_1 and n_2 degrees of freedom,

$$F_1(\alpha) \equiv \frac{V(\alpha) - V(\alpha_0)}{V(\alpha_0)} \frac{N_y - N_{DF}(\alpha_0)}{N_{DF}(\alpha_0)} \qquad (3.24)$$

and α_0 is the α for which the weighted sum of squared residuals, $\|M_\epsilon^{-1/2}(y - Ax)\|^2$, is minimum. Normally α_0 will be the smallest in the series of α values used, but numerical instabilities may require a slightly larger α_0. In either case, the solution with α_0 will be approximately the ordinary least squares solution, which corresponds to $\alpha = 0$. Eqs. (3.23) and (3.24) then define the usual confidence regions for ordinary least squares. That is, the fractional increase of $V(\alpha)$ over $V(\alpha_0)$ will occur $100[1 - \text{PROB1}(\alpha)]\%$ of the time due to chance alone (i.e., due to the random sample of noisy data). Therefore, only when $\text{PROB1}(\alpha)$ is greater than, say, about 0.9 are there significant grounds to suspect that α may be too large and biasing the solution. At the other extreme, parsimony dictates to suspect any α with $\text{PROB1}(\alpha) \lesssim 0.1$ as being too small; with such a weakly weighted regularizor, artifacts could very well be present. The recommended region is between these limits. CONTIN outputs the solution with $\text{PROB1}(\alpha)$ nearest 0.5 once again at the end of the analysis. $\text{PROB1}(\alpha)$ increases monotonically with increasing α.

We have consistently found this criterion to be useful and reliable over a wide range of applications and have given a brief heuristic justification [12]. However, Obenchain [25] had independently proposed the same criterion and given a detailed justification. It is implicitly assumed that the noise is multivariate normal with zero means. However, slight deviations from normality are not nearly as critical as unaccounted for systematic errors or presmoothed data. In these latter cases, one must use other criteria for choosing α, such as visual inspection of plots of the weighted residuals or fits to the data (section 4.8).

3.7. Error estimates and resolving power

Care must be exercised in making statements about error estimates and confidence regions, since the errors in solutions to ill-posed problems are generally unbounded. Version 2 of CONTIN computes Σ_x, the covariance matrix (derived in section A.2) of the solution, and plots the $(\Sigma_x)_{jj}^{1/2}$ as "error bars" with the solution. However, this neglects the bias term in eq. (A.35); i.e., it assumes that the regularizor is not biasing the solution and that there are enough parameters to represent the solution adequately. As $\alpha \sim \infty$ the error bars lose all meaning and approach zero.

Similarly, as discussed in section 3.6, the range of solutions with $0.1 \lesssim \text{PROB1}(\alpha) \lesssim 0.9$ would define something analogous to a confidence region only on the assumption that the regularizor is not significantly biasing the solution and that the parameterization is adequate. For example, if the regularizor is imposing smoothness, then one could say, "the smoothest solution consistent with the data probably lies within this range". The qualifier "smoothest" must be there. Nevertheless, these analogs to error estimates and confidence regions can be very useful at indicating which regions or features of the solution are well determined by the data and which are uncertain.

Another useful way of assessing the accuracy of a solution or the potential of an experimental method is to use simulations (section 4.6) to determine resolving power and point spread functions. For example, suppose that in the analysis of a data set the criteria of section 3.6 yielded a certain α. CONTIN has the option to repeat the analysis with α fixed at this value and everything else the same except that the data are replaced with noise-free simulated data corresponding to $s(\lambda) = \delta(\lambda - \lambda_0)$, a Dirac δ-function at $\lambda = \lambda_0$. The resultant solution tells how much a δ-function is spread by the regularization. Repeating the analysis at different λ_0 values tells where the spread is large and the resolution poor (due to insufficient information in the data on that region of the λ-axis) and where the resolution is high. One can also investigate pairs of δ-functions and see how close they can come on the λ-axis and still be resolvable into two peaks.

With linear systems, these point spreads apply directly to the solution with the original data set. When there are binding inequality constraints, the correspondence is no longer exact because the

solution is nonlinear. However, it is precisely this nonlinear action of the inequality constraints that can be so effective at increasing the resolution and the stability of the solution to noise.

CONTIN also allows more conventional simulations using $s(\lambda)$ that are likely to occur in practice and superposed pseudorandom noise. These simulated data are then analyzed in the usual way with α chosen with the criteria of section 3.6.

4. Control variables and USER subprograms

This section gives a brief overview of the most important options available in CONTIN. Some of the control variables and all 13 USER subprograms are mentioned. Full descriptions are given in the following paper and in ref. [11]. In this section all variable names in all capitals are control variables. The names of USER subprograms are uniquely identified because their names all begin with the characters "USER".

4.1. Input control

Several analyses can be performed in one run. If LAST = .FALSE., then CONTIN will read in and analyze another data set after the present analysis is finished. Values of control variables are preserved from one analysis to the next. Only the values that are to be changed need to be input.

DOUSIN = .TRUE. will call USERIN after the input data has been read but before the analysis is started. Therefore USERIN can be used to perform any necessary preprocessing of the input data. All input data and control variables are output right after the optional call to USERIN, just before the analysis is started. Extensive tests for inconsistencies and error conditions are made during input and analysis. There are more than 50 diagnostics with likely causes and remedies given in the Users Manual [11].

Other control variable arrays specify the input format for arrays such as the y_k. There are also three control variable arrays, RUSER, IUSER, LUSER, of type REAL, INTEGER and LOGICAL and with length 100, 50 and 30, respectively. They (as well as all other control variables) are available in COMMON blocks in all USER subprograms. They can be used to input data for use in USER subprograms or to store results produced there.

4.2. Specifying the problem

The conversion of eq. (1.2) to (3.1) is specified as follows. IQUAD = 1 if the problem is already in the form of linear algebraic equations; the c_m in eq. (3.1) will all be set to 1.0. When IQUAD = 2 or 3, the c_m will be automatically computed for the trapezoidal or Simpson's rule, respectively.

GMNMX(1) = a and GMNMX(2) = b in eq. (1.2). IGRID = 1 will put the quadrature grid in equal intervals of λ from a to b. IGRID = 2 will put the grid in equal intervals of the monotonic function $h(\lambda)$, which is specified in USERTR. The default version of USERTR sets $h(\lambda) = \ln(\lambda)$. IGRID = 3 will call USERGR to define special-purpose c_m and λ_m in eq. (3.1). The default version of USERGR simply reads the λ_m in and computes the c_m for the trapezoidal rule. NG = N_g in eq. (3.1).

USERK evaluates the $F_k(\lambda_m)$ in eq. (3.1). If NLINF = N_L is positive, then USERLF is called to evaluate the L_{ki} in eq. (3.1). IUNIT ≥ 0 specifies that the $F_k(\lambda_m)$ and L_{ki} be written on the storage device identified by the integer IUNIT. They are then only evaluated once and are read later when they are needed (twice for each α value).

4.3. Constraints

DOUSNQ = .TRUE. will call USERNQ to set N_{ineq}, D and $\boldsymbol{d}$ in eq. (3.6). For the special case of nonnegativity, it is better to simply set NONNEG = .TRUE., which will automatically constrain $x_j \geq 0$ for $j = 1,\ldots,N_g$. NEQ = $N_{eq} > 0$ will call USEREQ to set E and $\boldsymbol{e}$ in eq. (3.7).

4.4. Weighting the data

The most common assumption is that the covariance matrix M_ϵ in eq. (3.10) is diagonal, i.e., that the experimental errors are uncorrelated. In this case, CONTIN has convenient options for

specifying M_ϵ in terms of the least-squares weights, $W_k = (M_\epsilon)^{-1}_{kk} = 1/\sigma^2(y_k)$, where $\sigma^2(y_k)$ is the variance of the random variable y_k. (If M_ϵ is not diagonal, the user could modify USERIN and USERK to account for this.)

IWT = 1 will set the $W_j = 1$, which is appropriate if the noise level is the same for all data points. IWT = 2 assumes $\sigma^2(y_k) = \bar{y}_k$, where $\bar{y}_k$ is the expectation value of y_k; this is appropriate if the data follow Poisson statistics. IWT = 3 assumes $\sigma^2(y_k) = \bar{y}_k^2$. IWT = 4 simply reads the W_k in. IWT = 5 will call USERWT to compute the W_k for special cases not covered by IWT = 1,...,4. There are default versions of USERWT appropriate for photon correlation spectroscopy and fibre diffraction.

It is often dangerous to use the y_k in place of the unknown $\bar{y}_k$. For example with Poisson statistics this would cause the weights, $W_k = 1/y_k$, to be too large for those y_k with $\epsilon_k < 0$ and would thus bias the analysis toward these data points. Therefore, when IWT = 2, 3 or 5, CONTIN performs a preliminary unweighted analysis (i.e., with all $W_k = 1.0$). The resultant "fit values", $\hat{y}_k$, computed from the right-hand side of eq. (3.2), are used as a better estimate of $\bar{y}_k$ to compute the W_k. Price [26] has pointed out that, for Poisson statistics, this produces approximately a maximum-likelihood analysis. When NERFIT > 0, an added precaution against very small $\hat{y}_k$ causing very large W_k is taken by estimating the noise level in the y_k near the minimum of $|\hat{y}_k|$ [11].

4.5. Regularization

NORDER = n, $n = 0, 1, \ldots, 5$ will set $\|\boldsymbol{r} - R\boldsymbol{x}\|^2$ in eq. (3.10) to the sum of the squares of the nth differences of successive values of the x_j for $j = 1, \ldots, N_g$. For example, NORDER = 2 automatically produces the regularizor in eq. (3.12), and NORDER = 0 sets R to the $N_g \times N_g$ identity matrix. Thus NORDER = 2 is good for imposing smoothness, and NORDER = 0 performs ridge regression [17]. Other positive values of NORDER are generally inappropriate.

NENDZ(J) with J = 1 or 2 specifies the external boundary conditions for $s(\lambda)$ when NORDER > 0 and the grid of λ_m points is in equal intervals, Δ, of λ or $h(\lambda)$. [See section 4.2 and the discussion following eq. (3.12).] NENDZ(1) is the number of external grid points, $s(a - \Delta)$, $s(a - 2\Delta), \ldots$, specified to be zero. NENDZ(2) is the number of external grid points, $s(b + \Delta)$, $s(b + 2\Delta), \ldots$, specified to be zero. Thus NORDER = 2 and NENDZ(1) = NENDZ(2) = 2 would result in the regularizor in eq. (3.12); NENDZ(1) = 0 would eliminate the first two rows and NENDZ(2) = 0 the last two rows.

NORDER < 0 will call USERRG to set a special-purpose N_{reg}, $\boldsymbol{r}$ and R in eq. (3.10). There are default versions of USERRG to impose statistical prior knowledge of the expectation value of $\boldsymbol{x}$ [27,6].

There are other control variables to specify the number and range of α values. The default settings insure that the range is automatically set wide enough.

4.6. Simulation

SIMULA = .TRUE. will call USERSI and USEREX to produce simulated data. USEREX produces noise-free values for the y_k. USERSI then adds pseudorandom variables to these noise-free values. The default version of USEREX and USERSI simulate a δ-function distribution for $s(\lambda)$ with noisy data corresponding to a second-order correlation function in dynamic light scattering [12], with the noise level specified by the user. These versions can be easily modified for other types of simulations.

Simulations can be very useful in assessing the potential of an experimental method or design, even before the instrument is built or the experiment performed. Section 3.7 discusses some applications of simulations.

4.7. Peak-constrained solutions

Often the presence of extra extrema or peaks in a solution, $s(\lambda)$, would be important new information. The principle of parsimony would then dictate that, in addition to smoothness, the solution with the fewest number of extrema that is consistent with the data should be sought.

There are six control variables, described in

detail elsewhere [11], that can limit the number of extrema in $s(\lambda)$. This is done by dividing the λ-grid into monotonic regions where inequality constraints make $s(\lambda)$ either nonincreasing or nondecreasing. The boundaries of these regions are then systematically moved, one grid point at a time, until a local minimum in $V(\alpha)$ is found. Thus, in contrast to the usual case in CONTIN, peak-constrained solutions are not guaranteed to be global optima, only local. However, the most common application is constraining $s(\lambda)$ to have only one peak. A global solution can then be obtained easily by exhaustively covering all the grid points. As the allowed number of extrema increases, the inefficiency of the computation rapidly increaes. Therefore no more than five extrema (i.e., a bimodal solution) are allowed.

Peak-constrained solutions can be useful in guarding against artifacts. If a solution with several peaks is obtained, a peak-constrained analysis can be performed to see if a solution with fewer peaks, but still consistent with the data, can be found. If not, then one can say with more confidence that the data *require* the original number of peaks, and therefore that they are less likely to be artifacts.

4.8. Output control

There are 12 control variables for specifying the quantity and spacing of the output. There are options for producing line-printer plots of the solution, the fit to the data and the weighted residuals of the fit. There are also options for computing various moments of the solution.

DOUSOU = .TRUE. will call USEROU right after each solution is plotted. This can be useful if further output or computations with the solution are desired. The default version of USEROU multiplies $\boldsymbol{x}$ by a matrix to compute the fraction of each structural class present from the fractional contribution of library circular dichroic spectra to the data spectrum [6].

ONLY1 = .FALSE. will cause a second curve (that has been computed in USERSX) to be plotted with the solution. This is most often used with simulations to plot the exact solutions.

Acknowledgements

I thank Jürgen Glöckner for much help in testing and preparing CONTIN for distribution, Prof. Leo De Maeyer and Robert Vogel for helpful discussions, and Ines Benner for typing the manuscript. Prof. Gerson Kegeles introduced me to this field (and several others) twenty years ago, and I dedicate this paper to him on the occasion of his 65th birthday.

Appendix. Computational methods

A.1. Constrained solution of eq. (3.10)

In this appendix we outline the quadratic programming solution to eq. (3.10) subject to the constraints in eqs. (3.6) and (3.7). Priorities are placed on saving computation time and high-speed storage and in exploiting the fact that CONTIN performs a series of solutions only varying α or the inequality constraints. Version 2 of CONTIN performs automatic internal scaling of the arrays, but to avoid complicating the notation, this is not explicitly shown.

The matrix A in eq. (3.10) is $N_y \times N_x$, and N_y can be 1000 or more. Therefore, to avoid keeping a large A in high-speed storage, if $N_y > N_x$, $M_\epsilon^{-1/2}\boldsymbol{y}$ and $M_\epsilon^{-1/2}A$ are orthogonally transformed, row-by-row, to $\boldsymbol{\eta}$ and C with sequential Householder transformations [20]. This produces a problem with the identical solution, $\boldsymbol{x}$,

$$V(\alpha) = \|\boldsymbol{\eta} - C\boldsymbol{x}\|^2 + V_1 + \alpha^2 \|\boldsymbol{r} - R\boldsymbol{x}\|^2 = \text{minimum}, \tag{A.1}$$

but where $\boldsymbol{\eta}$ and C are only $N_{xy} \times 1$ and $N_{xy} \times N_x$, respectively, where

$$N_{xy} \equiv \min(N_x, N_y).$$

The constant V_1 can be ignored, since it will not affect the solution. CONTIN assumes that M_ϵ is diagonal so that $M_\epsilon^{-1/2}$ simply multiplies each row of $\boldsymbol{y}$ and A by a consant during the processing. The user could modify USERIN and USERK to handle a nondiagonal M_ϵ (i.e., correlated noise) if necessary.

The rank of E in eq. (3.7) must be N_{eq}; i.e., the equality constraints must be linearly independent and consistent with each other. The equality constraints are eliminated using an orthogonal basis of the null space of E [20]. That is, an $N_x \times N_x$ orthogonal matrix, K, is computed with Householder transformations such that EK_1 is lower triangular and EK_2 has all zero elements when K is partitioned into

$$K = [K_1 | K_2], \tag{A.2}$$

where K_1 is $N_x \times N_{eq}$, K_2 is $N_x \times N_{xe}$ and $N_{xe} \equiv N_x - N_{eq}$.

We make the change in variables

$$\boldsymbol{x} = K \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = K_1 \boldsymbol{x}_1 + K_2 \boldsymbol{x}_2, \tag{A.3}$$

where we have partitioned the vector of new variables into the $N_{eq} \times 1$ and $N_{xe} \times 1$ vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Eq. (3.7) then reduces to $EK_1 \boldsymbol{x}_1 = \boldsymbol{e}$, which yields part of the solution,

$$\boldsymbol{x}_1 = (EK_1)^{-1} \boldsymbol{e}. \tag{A.4}$$

Substituting eq. (A.3) into (A.1) and (3.6), we obtain a problem with reduced dimensions,

$$\| (\boldsymbol{\eta} - CK_1 \boldsymbol{x}_1) - CK_2 \boldsymbol{x}_2 \|^2 + \alpha^2 \| (\boldsymbol{r} - RK_1 \boldsymbol{x}_1) - RK_2 \boldsymbol{x}_2 \|^2 = \text{minimum} \tag{A.5}$$

subject only to

$$DK_2 \boldsymbol{x}_2 \geqslant \boldsymbol{d} - DK_1 \boldsymbol{x}_1, \tag{A.6}$$

where only $\boldsymbol{x}_2$ is unknown. If there are no equality constraints, there is no K_1 or $\boldsymbol{x}_1$ and K_2 is simply the $N_x \times N_x$ identity matrix.

If $N_{reg} < N_{xe}$, then zero rows are added to R and $\boldsymbol{r}$ so that $N_{reg} = N_{xe}$. The singular value decomposition of RK_2 is then computed.

$$RK_2 = U \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} Z^T, \tag{A.7}$$

where the superscript T denotes matrix transposition, $U(N_{reg} \times N_{reg})$ and $Z(N_{xe} \times N_{xe})$ are orthogonal matrices, H_1 is an $N_{xe} \times N_{xe}$ diagonal matrix, and H_2 is $(N_{reg} - N_{xe}) \times N_{xe}$ with all zero elements. If necessary, to make H_1 and RK_2 full rank, the diagonal elements (singular values) of H_1 are increased to at least a small fraction of the largest singular value. The size of the fraction is automatically set according to the relative machine precision. We then make the change in variables

$$\boldsymbol{x}_3 = Z^T \boldsymbol{x}_2, \quad \boldsymbol{x}_2 = Z \boldsymbol{x}_3, \tag{A.8}$$

and multiply the vector inside the second $\| \cdot \|^2$ in eq. (A.5) by U^T (which does not change the value of $\| \cdot \|^2$, since U^T is orthogonal). Eqs. (A.5) and (A.6) then become

$$\| (\boldsymbol{\eta} - CK_1 \boldsymbol{x}_1) - CK_2 Z \boldsymbol{x}_3 \|^2 + \alpha^2 \| \boldsymbol{r}_1 - H_1 \boldsymbol{x}_3 \|^2 + \alpha^2 \| \boldsymbol{r}_2 \|^2 = \text{minimum}, \tag{A.9}$$

$$DK_2 Z \boldsymbol{x}_3 \geqslant \boldsymbol{d} - DK_1 \boldsymbol{x}_1, \tag{A.10}$$

where the $N_{xe} \times 1$ and $(N_{reg} - N_{xe}) \times 1$ vectors $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are defined by

$$U^T (\boldsymbol{r} - RK_1 \boldsymbol{x}_1) = \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \end{bmatrix}. \tag{A.11}$$

In eq. (A.9), the term $\alpha^2 \| \boldsymbol{r}_2 \|^2$ will be discarded since it is independent of the solution $\boldsymbol{x}_3$ and therefore does not affect it.

We can now transform the problem to a constrained ridge regression problem with the change of variables

$$\boldsymbol{x}_4 = H_1 \boldsymbol{x}_3 - \boldsymbol{r}_1, \quad \boldsymbol{x}_3 = H_1^{-1} (\boldsymbol{x}_4 + \boldsymbol{r}_1). \tag{A.12}$$

Eqs. (A.9) and (A.10) then become

$$\| (\boldsymbol{\eta} - CK_1 \boldsymbol{x}_1 - CK_2 Z H_1^{-1} \boldsymbol{r}_1) - CK_2 Z H_1^{-1} \boldsymbol{x}_4 \|^2 + \alpha^2 \| \boldsymbol{x}_4 \|^2 = \text{minimum}, \tag{A.13}$$

$$DK_2 Z H_1^{-1} \boldsymbol{x}_4 \geqslant \boldsymbol{d} - DK_1 \boldsymbol{x}_1 - DK_2 Z H_1^{-1} \boldsymbol{r}_1. \tag{A.14}$$

We perform the singular value decomposition

$$CK_2 Z H_1^{-1} = QSW^T, \tag{A.15}$$

where $Q(N_{xy} \times N_{xy})$ and $W(N_{xe} \times N_{xe})$ are orthogonal and $S(N_{xy} \times N_{xe})$ is diagonal. Making the change in variables

$$\boldsymbol{x}_5 = W^T \boldsymbol{x}_4, \quad \boldsymbol{x}_4 = W \boldsymbol{x}_5, \tag{A.16}$$

and multiplying the vector in the left $\| \cdot \|^2$ in eq. (A.13) by Q^T, we can write eqs. (A.13) and (A.14) as

$$\| \boldsymbol{\gamma} - S \boldsymbol{x}_5 \|^2 + \alpha^2 \| \boldsymbol{x}_5 \|^2 = \text{minimum}, \tag{A.17}$$

$$DK_2ZH_1^{-1}Wx_5 \geq d - DK_1x_1 - DK_2ZH_1^{-1}r_1, \tag{A.18}$$

where the $N_{xy} \times 1$ vector γ is defined by

$$\gamma = Q^T(\eta - CK_1x_1 - CK_2ZH_1^{-1}r_1). \tag{A.19}$$

The following equation is identical to eq. (A.17) except for some constants, independent of x_5, which do not affect x_5:

$$\|\tilde{\gamma} - \tilde{S}x_5\|^2 = \text{minimum}, \tag{A.20}$$

where $\tilde{\gamma}$ is the $N_{xe} \times 1$ vector with components

$$\tilde{\gamma}_j = \gamma_j s_j (s_j^2 + \alpha^2)^{-1/2}, \quad j = 1, \ldots, N_{xye} \tag{A.21}$$

and $\tilde{S}(N_{xe} \times N_{xe})$ is diagonal with diagonal elements

$$\tilde{S}_{jj} = (s_j^2 + \alpha^2)^{1/2}, \tag{A.22}$$

where

$$s_j \equiv S_{jj}, \quad j = 1, \ldots, N_{xye}, \tag{A.23}$$

$$N_{xye} \equiv \min(N_{xy}, N_{xe}) = \min(N_y, N_x - N_{eq}) \tag{A.24}$$

and, if $N_y < N_{xe}$,

$$s_j \equiv 0, \quad \tilde{\gamma}_j \equiv 0, \quad j = N_{xye} + 1, \ldots, N_{xe}. \tag{A.25}$$

The equivalence of problems (A.17) and (A.20) can be easily verified by comparing the two explicit expansions of the $\|\cdot\|^2$, which are simple sums of squared binomials.

We obtain the final form of the problem by making the change of variables

$$\xi = \tilde{S}x_5 - \tilde{\gamma}, \quad x_5 = \tilde{S}^{-1}(\xi + \tilde{\gamma}), \tag{A.26}$$

whereby eqs. (A.20) and (A.18) become

$$\|\xi\|^2 = \text{minimum}, \tag{A.27}$$

$$DK_2ZH_1^{-1}W\tilde{S}^{-1}\xi \geq d - DK_1x_1 - DK_2ZH_1^{-1}(r_1 + W\tilde{S}^{-1}\tilde{\gamma}). \tag{A.28}$$

Eqs. (A.27) and (A.28) are solved with a least distance programming procedure [20] that checks the consistency of eq. (A.28) and finds the unique solution, ξ, in a finite number of steps. From ξ and eqs. (A.3), (A.8), (A.12), (A.16) and (A.26), we obtain our solution,

$$x = K_1x_1 + K_2ZH_1^{-1}[W\tilde{S}^{-1}(\xi + \tilde{\gamma}) + r_1]. \tag{A.29}$$

CONTIN performs a series of solutions keeping everything constant except either α or the inequality constraints (during peak-constrained solutions). In either case, most of the time-consuming computations, e.g., the singular value decompositions in eqs. (A.7) and (A.15), are done only once for the whole series. In eqs. (A.28) and (A.29), only $\tilde{\gamma}$ and the diagonal matrix $\tilde{S}^{-1}$ depend on α, and only D depends on the inequality constraints that change during the peak-constrained solution. Therefore, during either of these series, the only time-consuming part is the least distance programming solution of eqs. (A.27) and (A.28). This could perhaps be speeded up by using a parametric programming approach, where the preceding solution provides a starting point for the next solution. However, the Lawson–Hanson algorithms [20] have proven to be reliable, numerically stable, and efficient in time and storage.

A.2. Error estimates

For simplicity, we first consider the special case where (1) there are no binding inequality constraints, and hence $\xi = 0$ in eq. (A.29); (2) there are no equality constraints, and hence no x_1 in eq. (A.29); (3) $N_x \leq N_y$; and (4) $N_{reg} = N_x$ and $r_1 = 0$ in eq. (A.11). This last condition is satisfied when $N_{eq} = 0$ and $r_1 = 0$, as with smoothing regularizors. These conditions mean that $N_{xy} = N_{xye} = N_{xe} = N_x$.

Under the above conditions, eq. (A.29) reduces to

$$x = K_2ZH_1^{-1}W\tilde{S}^{-1}\tilde{\gamma}. \tag{A.30}$$

Substituting eqs. (A.19), (A.21) and (A.22) into (A.30), we can write

$$x = K_2ZH_1^{-1}WG(\alpha)\,Q^T\eta, \tag{A.31}$$

where $G(\alpha)$ is the $N_{xe} \times N_{xe}$ diagonal matrix with elements

$$G_{jj} = s_j/(s_j^2 + \alpha^2). \tag{A.32}$$

Thus, x is just a linear transformation of η, and

Σ_x, the covariance matrix of $\boldsymbol{x}$, is simply

$$\Sigma_x = \left(K_2 Z H_1^{-1} W G Q^T\right) M_\eta \left(K_2 Z H_1^{-1} W G Q^T\right)^T. \quad \text{(A.33)}$$

$M_\eta = I$, since $\boldsymbol{\eta}$ was obtained from orthogonal transformations of $M_\epsilon^{-1/2}\boldsymbol{y}$, which has an identity covariance matrix. Since Q is also orthogonal, eq. (A.33) reduces to

$$\Sigma_x = \left(K_2 Z H_1^{-1} W\right) G^2 \left(K_2 Z H_1^{-1} W\right)^T. \quad \text{(A.34)}$$

The expected squared error in any x_j is

$$E\left\{\left(x_j - x_j^0\right)^2\right\} = (\Sigma_x)_{jj} + \left[E\left\{x_j - x_j^0\right\}\right]^2, \quad \text{(A.35)}$$

where the x_j^0 are the true values. The second term on the right, the (generally unknown) squared bias, will only be zero when α is zero or when the regularizor does not bias the solution. From eqs. (A.32) and (A.34), it is clear that the $(\Sigma_x)_{jj}$ decrease monotonically with increasing α, approaching zero as $\alpha \sim \infty$. The squared bias term increases monotonically with increasing α [17]. Therefore care must be taken in discussing expected errors solely on the basis of Σ_x.

This special case can be generalized by removing assumptions (2)–(4) above. The algebra is more complicated because the matrices are no longer necessarily square or full rank, and the constraints and regularizor imply that $\boldsymbol{x}$ is to be replaced by $\boldsymbol{x} - K_1\boldsymbol{x}_1 - K_2 Z H_1^{-1}\boldsymbol{r}_1$.

Inequality constraints present a fundamental problem because the estimate is no longer linear, although some progress in this area has been made [28]. CONTIN simply takes the binding inequality constraints in a solution and eliminates them as equality constraints using the procedure in eqs. (A.2)–(A.6). The covariance matrix is then recomputed for the reduced unconstrained problem, as described above. This will underestimate the variances.

References

[1] O. Glatter, J. Appl. Cryst. 10 (1977) 415.
[2] K. Shizuma, Nucl. Instr. and Meth. 173 (1980) 395.
[3] M.L. Johnson, J.L. Romero, T.S. Subramanian and F.P. Brady, Nucl. Instr. and Meth. 169 (1980) 179.
[4] S.W. Provencher and V.G. Dovi, J. Biochem. Biophys. Meth. 1 (1979) 313.
[5] S.W. Provencher, J. Hendrix, L. De Maeyer and N. Paulussen, J. Chem. Phys. 69 (1978) 4273.
[6] S.W. Provencher and J. Glöckner, Biochem. 20 (1981) 33.
[7] D.L. Phillips, J. Ass. Comput. Mach. 9 (1962) 84.
[8] P.H. Merz, J. Comput. Phys. 38 (1980) 64.
[9] O.N. Strand and E.R. Westwater, J. Ass. Comput. Mach. 15 (1968) 100.
[10] D.V. Lindley and A.F.M. Smith, J. Roy. Stat. Soc. B34 (1972) 1.
[11] S.W. Provencher, CONTIN Users Manual, EMBL Technical Report DA05, European Molecular Biology Laboratory (1982).
[12] S.W. Provencher, Makromol. Chem. 180 (1979) 201.
[13] C. Nave, R.S. Brown, A.G. Fowler, J.E. Ladner, D.A. Marvin, S.W. Provencher, A. Tsugita, J. Armstrong and R.N. Perham, J. Mol. Biol. 149 (1981) 675.
[14] J. Skilling and R.K. Bryan, Monthly Notices Roy. Astron. Soc. (submitted).
[15] I.J. Schoenberg, Cardinal Spline Interpolation (SIAM, Philadelphia, 1973).
[16] C. de Boor, A Practical Guide to Splines (Springer, New York, 1978).
[17] A.E. Hoerl and R.W. Kennard, Technometrics 12 (1970) 55.
[18] H.S. Hou and H.C. Andrews, IEEE Trans. Comput. C-26 (1977) 856.
[19] B.R. Frieden, Comput. Graphics Image Processing 12 (1980) 40.
[20] C.L. Lawson and R.J. Hanson, Solving Least Squares Problems (Prentice-Hall, Englewood Cliffs, 1974).
[21] G. Toraldo di Francia, J. Opt. Soc. Am. 45 (1955) 497.
[22] G. Toraldo di Francia, J. Opt. Soc. Am. 59 (1969) 799.
[23] S. Twomey and H.B. Howell, Appl. Opt. 6 (1967) 2125.
[24] C.L. Mallows, Technometrics 15 (1973) 661.
[25] R.L. Obenchain, Technometrics 19 (1977) 429.
[26] P.F. Price, Acta Cryst. A35 (1979) 57.
[27] S. Twomey, J. Ass. Comput. Mach. 10 (1963) 97.
[28] W.J. Hemmerle and T.F. Brantle, Technometrics 20 (1978) 109.