# Parameter Estimation in Systems of Nonlinear Equations

**K. Schittkowski** [1]

**Abstract:** A numerical method is presented to determine parameters in a system of nonlinear equations in the following sense: Proceeding from given experimental data, e.g., observation times and measurements, the minimum least-squares distance of the measured data from a fitting criterion depending on the solution of a system of nonlinear equations is to be computed. Specifically coupled mass equilibrium models are described in more detail that are used in pharmaceutical applications for receptor-ligand binding studies. They are used for instance for the radioimmunological determination of Fenoterol or related substances. Numerical results based on laboratory data are included to test the robustness of the algorithm implemented.

[1]Mathematisches Institut, Universität Bayreuth, D - 95440 Bayreuth

## 1. Introduction

Parameter estimation plays an important role in many natural science and other disciplines. The key idea is to estimate unknown parameters in the mathematical model that describes the real-life situation, by minimizing the distance of some known experimental data from the computed model values. Thus also model parameters that cannot be measured directly, can be estimated by a least-squares fit and analysed subsequently.

The model to be considered now consists of a system of nonlinear equations that depend on some parameters to be estimated, and an independent *time* variable. Only for illustration purposes we denote the independent model variable as the *time* variable of the system and the dependent data as measurement values of an experiment. On the other hand, both terms may have any other meaning within a model depending on the underlying application.

It is assumed that the system of nonlinear equations is solvable and differentiable with respect to the system variables and the parameters to be estimated on the whole domain. To be able to solve the system uniquely and to get derivatives in an efficient and numerically stable way, also regularity of the system will be assumed.

The numerical solution of the problem is then achieved in two steps. First the system of nonlinear equations is solved to get model function values and, in particular, the corresponding derivatives. A fitting criterion depending on the system variables and the independent optimization parameters is formulated subsequently. The resulting data are then inserted into a standard parameter-estimation code to compute the least-squares fit. Upper and lower bounds for the parameters to be estimated can be taken into account.

One typical application, to be described in more detail, is the coupled mass equilibrium model consisting of antibodies with one binding site or with several mutually independent binding sites of equal intrinsic affinity, and monovalent antigens, see, e.g., Bürgisser, Hancock, Lefkowitz, De Lean (1980) or De Lean, Hancock, Lefkowitz (1981) for more details about the model. The regularity of this system is easily verified. A possible application is the radioimmunological determination of Fenoterol. From this application several test cases are derived to show that the numerical solution is by no means trivial.

In Section 2 the general mathematical least-squares problem is outlined, in particular the evaluation of gradients. Some details on the numerical implementation are summarized in Section 3. The coupled mass equilibrium model is described in Section 4 and numerical results are presented in Section 5.

## 2. The Mathematical Model

The basic mathematical model is the least-squares problem, i.e., the problem of minimization of a sum of squares of nonlinear functions of the following form:

$$
\begin{array}{rl} & \min \sum_{i=1}^{l} f_i(x)^2 \\ x \in \mathsf{R}^n : & x_l \le x \le x_u \end{array} \quad . \tag{1}
$$

Here we assume that the parameter vector $x$ is $n$-dimensional and that all nonlinear functions $f_i(x)$ are continuously differentiable with respect to $x$ on the feasible domain defined by upper and lower bounds on the variables.

In the following we restrict all investigations to parameter estimation problems, where one model function is available with one additional variable $t$ called *time*. Proceeding now from $l$ measurements of the form

$$
(t_i, y_i) \ , \ i = 1, ..., l \ \ ,
$$

where $l$ *time* values and $l$ corresponding measurement values are defined, and a model function $h(x,t)$, we get the above problem formulation by setting

$$
f_i(x) = h(x, t_i) - y_i \ \ .
$$

Then the underlying idea is to minimize the distance between the model function at certain *time* points and the corresponding measurement values. This distance is denoted the residual of the problem. In the ideal case the residuals are zero indicating a perfect fit of the model function by the measurements. Again we have to assume that function $h(x,t)$ is differentiable for all $x$ within the rectangle $x_l \le x \le x_u$.

Since for practical applications additional weighting factors are highly useful, we instead use the notation

$$
f_i(x) = w_i(h(x, t_i) - y_i) \tag{2}
$$

with suitable weighting factors $w_i$, $i = 1, ..., l$.

Our special goal is to estimate parameters in systems of nonlinear equations, which may depend also on the *time* variable. In this case the model function $h(x,t)$ depends in addition on the solution vector of a system of nonlinear equations, i.e.,

$$
h(x,t) = h^s(x, z(x,t), t) \ \ , \tag{3}
$$

where $z(x,t) \in \mathsf{R}^m$ is implicitely defined by the solution of the system

$$
z \in \mathsf{R}^m : \quad \begin{array}{rcl} s_1(x,z,t) & = & 0 \\ & \cdots & \\ s_m(x,z,t) & = & 0 \end{array} \quad . \tag{4}
$$

The system functions $s_1, \ldots, s_m$ are assumed to be continuously differentiable with respect to variables $x$ and $z$. Moreover we require the regularity of the system, i.e., that the system is solvable and that the derivative matrix

$$\left( \frac{\partial s_i(x,z,t)}{\partial z_j} \right)$$

has full rank for all $x$ with $x_l \le x \le x_u$ and for all $z$, for which a solution $z(x,t)$ of (4) exists. Consequently the function $z(x,t)$ is differentiable with respect to all $x$ in the feasible domain. Now let $t$ be fixed and let $z(x,t)$ a solution of (4). If we denote $S(x,z) := (s_1(x,z,t), \ldots, s_m(x,z,t))^T$ for all $x$ and $z$, we get from the identity $S(x, z(x,t)) = 0$, which is to be satisfied for all $x$, the derivative

$$\nabla_x S(x, z(x,t)) + \nabla_z S(x, z(x,t)) \nabla z(x,t) = 0 \ .$$

Here $\nabla_x S(x,z)$ and $\nabla_z S(x,z)$ denote the partial derivatives of $S(x,z)$ with respect to the parameters $x$ and $z$, respectively. In other words, the desired functional matrix $\nabla z(x,t)$ is obtained from the linear system

$$\nabla_x S(x, z(x,t)) + \nabla_z S(x, z(x,t)) V = 0 \tag{5}$$

where $V$ is an $m \times n$-matrix.

Note that we describe here the implicit function theorem. Since $\nabla_z S(x, z(x))$ is nonsingular, the above system is uniquely solvable.

Given any parameter vector $x$ as provided by the used optimization algorithm, and an experimental *time* value $t_i$, system (4) is solved yielding a solution vector $z_i(x) := z(x, t_i)$ that is then inserted into the function $h^s(x,z,t)$ to evaluate a model function value

$$f_i(x) = w_i(h^s(x, z_i(x), t_i) - y_i) \tag{6}$$

where $y_i$ denotes experimental data, $i = 1, \ldots, l$, The corresponding gradient is computed from

$$\nabla f_i(x) = w_i(\nabla_x h^s(x, z_i(x), t_i) + V_i \nabla_z h^s(x, z_i(x), t_i)) \tag{7}$$

where $V_i$ solves the linear matrix system

$$\nabla_x S(x, z_i(x)) + \nabla_z S(x, z_i(x)) V = 0 \tag{8}$$

for all $i = 1, ..., l$.

## 3. Numerical Implementation

First let us consider the nonlinear least-squares problem (1) in its general notation. Because of the standard formulation, many different numerical algorithms and computer codes are available and can be used. It was decided to implement two different subroutines giving the user the opportunity to switch from one code to the other whenever an algorithm is unable to find a solution in a proper way:

1. Subroutine DN2GB was developed by Dennis, Gay and Welsch (1981a, 1981b). As part of the PORT library, the code is widely used in practice, well tested and very efficient for the type of application we are considering. The algorithm is a variation of Newton's method and capable to handle upper and lower bounds on the variables. Part of the Hessian is computed directly and part is approximated by a quasi-Newton update method. In certain situations the algorithm reduces to a Gauss-Newton or a Levenberg-Marquardt algorithm. The method is stabilized by a trust-region technique along with an adaptive choice of a model Hessian to achieve convergence.

2. The second algorithm implemented, is the code DFNLP developed by Schittkowski (1988). The basic idea is to introduce additional variables and equality constraints, and to solve the resulting constrained nonlinear programming problem by the sequential quadratic programming algorithm NLPQL of Schittkowski (1985/86). It can be shown, that typical features of a special purpose method are retained, i.e., the combination of a Gauss-Newton and a quasi-Newton search direction, see Schittkowski (1988) for details. The additional variables and constraints are substituted in the quadratic programming subproblem, so that calculation time is not increased by this approach.

Any subroutine for solving smooth nonlinear least-squares problems requires the evaluation of model function values and of gradient values. Whenever an iterate $x_k$ of the least-squares algorithm is given, we first have to solve $l$ nonlinear systems of equations

$$z \in \mathsf{R}^m : \quad \begin{array}{rcl} s_1(x_k, z, t_i) & = & 0 \\ & \cdots & \\ s_m(x_k, z, t_i) & = & 0 \end{array} \tag{9}$$

for $i = 1, \ldots, l$. By inserting the solution $z(x_k, t_i)$ into (3) and (2), we then get the required model function values $f_i(x_k)$, $i = 1, \ldots, l$. The starting value $x_0$ must be provided by the user.

The nonlinear systems of equations of above form are treated as general optimization problems, where we minimize an artificial objective function

$$\frac{1}{\tau_i} \|z\|^2$$

subject to the constraints (9). The scaling parameter $\tau_i := \max(1, t_i)$ is introduced based on numerical experience, since for the application under consideration the *time* parameters that are also part of the system equation differ in their oder of magnitude drastically. The formulation of a nonlinear programming problem is more flexible and allows to detect situations, where (9) is contradictious or where (9) does not possess a unique solution.

The resulting nonlinear programming problem is solved by the FORTRAN code NLPQL, see Schittkowski (1985/86). The algorithm proceeds from a successive quadratic approximation of the Lagrangian function and linearization of constraints. To get a search direction, a quadratic programming problem must be solved in each iteration step. A subsquent line search stabilizes the algorithm. The function and gradient values with respect to each iteration variable are part of the problem formulation and must be provided by the user. Also the initial values required for the start of an optimization cycle must be predetermined by the user in a suitable way. They may depend on the parameters of the outer least-squares problem.

As mentioned above, the parameter-estimation algorithm DFNLP does also require the execution of subroutine NLPQL for solving the modified nonlinear programming problem. To avoid execution conflicts, DFNLP uses reverse communication for evaluating function and gradient values, cf. the program documentation for details.

The evaluation of gradients of the model functions $\nabla f_i(x_k)$ at certain iterates $x_k$ is required by both least-squares algorithms. If a solution $z(x_k, t_i)$ of (9) is not known from a previous function call, system (9) must be solved again by NLPQL. Subsequently the linear system (8) is solved in a numerically stable way by the code H12 as published in Lawson and Hanson (1974). The code is based on a numerically stable Householder transformation of $\nabla_z S(x_k, z(x_k, t_i))$, cf. (8). By inserting the solution into (7) we obtain the desired gradient value.

The numerical algorithm to estimate parameters in systems of nonlinear equations as described above, is implemented in double precision FORTRAN in form of two different programs that can be executed completely independently of each other. The corresponding input files differ only in a few parameters that control certain options of the algorithms. Nonlinear model functions $s_1(x, z, t)$, ..., $s_m(x, z, t)$, starting values $z_0(x)$ for solving (4) and the fitting criterion $h^s(x, z, t)$ must be provided by the user in form of a separate subroutine. Moreover all gradients of the model functions with respect to $x$ and $z$ are to be evaluated within a user written subroutine.

The usage of the codes is described in Schittkowski (1992). Since, however, the model part of both codes is identical, we use the notation 'SYSFIT' throughout this paper to identify both approaches. The codes possess the following additional features:

1. **Scaling:** For all practical parameter-estimation problems, scaling of model function values is very important because of the variation of the experimental numerical data. Thus SYSFIT proceeds from scaling parameters

$$w_i := \underline{w}_i \overline{w}_i \ , \quad i = 1, \ldots, l$$

see (6), where $\underline{w}_i$ is an individual scaling value for the $i$-th experiment and $\overline{w}_i := {y_i}^e$ a computed scaling factor based on an exponent $e$ given by the user, where $e$ may be any real number. Alternatively a user may require that the constant scaling factor

$$\overline{w}_i := \left( \sum_{j=1}^{l} y_j^2 \right)^{-1/2}$$

is to be taken into account for the whole measurement set.

2. **Statistical data:** Proceeding from the assumption that the model is sufficiently linear in a neighbourhood of an optimal solution vector and that all experimental data are Gaussian and independent, some statistical data can be evaluated:

   - Variance/covariance matrix of the problem data
   - Correlation matrix of the problem data
   - Estimated variance of residuals
   - Confidence intervals for the individual parameters subject to the significance levels 1%, 5% or 10%, respectively

3. **Plot data:** After obtaining a numerical solution that satisfies the termination conditions, all measurement data and a large number of model function values at equidistant *time* points are written on a file together with some additional plot information. The data can be exploited to compute plots directly on a suitable device. Logarithmic *time* values are stored on request.

## 4. The Coupled Mass Equilibrium Model

The relationship between antibody and antigen or antibody and antigen determinant, i.e., ligand, can be described by a simple mass equilibrium if the following assumptions are satisfied, see, e.g., Rominger and Albert (1985):

- The high affinity between antibodies belong to the immunoglobulin (Ig)G class.
- The two binding sites of these antibodies have the same intrinsic affinity and are independent of each other.
- For concentrations in the picomolar range the activity of the reactants is equal to their concentration.
- The antibody population is uniform.

Under these assumptions, the coupling of $m_e$ antibodies with one binding site and $n_e$ monovalent antigens is described by the equation

$$x_{ij} = K_{i,j} r_i l_j \tag{10}$$

for $i = 1, \ldots, m_e$ and $j = 1, \ldots, n_e$, where $r_i$ and $l_j$ are given by

$$r_i := R_i - \sum_{k=1}^{n_e} x_{ik} \tag{11}$$

and

$$l_j := L_j - \sum_{k=1}^{m_e} x_{kj} \quad , \tag{12}$$

see also Feldman (1972). Here we proceed from the mass conservation law and use the notation

| | | |
|---|---|---|
| $R_i$ | - | total concentration of $i$-th antibody |
| $L_j$ | - | total concentration of $j$-th antigen |
| $x_{ij}$ | - | concentration of the $i$-th antibody and the $j$-th antigen in equilibrium |
| $K_{ij}$ | - | association constant between the $i$-th antibody and the $j$-th antigen |

By insertion, the above system is easily transformed into the equivalent system of equations

$$\begin{aligned} r_i(1 + \sum_{k=1}^{n_e} K_{ik} l_k) - R_i &= 0, \; i = 1, \ldots, m_e \quad , \\ l_j(1 + \sum_{k=1}^{m_e} K_{kj} r_k) - L_j &= 0, \; j = 1, \ldots, n_e \quad . \end{aligned} \tag{13}$$

The system is a set of $m_e + n_e$ nonlinear equations and $m_e + n_e$ unknowns $r_i$, $i = 1, \ldots, m_e$ and $l_j$, $j = 1, \ldots, n_e$. If we set $m := m_e + n_e$,

$$z = (r_1, \ldots, r_{m_e}, l_1, \ldots, l_{n_e})^T \ ,$$

and if we denote the functions $s_1, \ldots, s_m$ in the corresponding way, we obtain system (4).

To be able to solve (4) uniquely and to compute the gradient values of $f_i(x)$ for $i = 1, \ldots, l$, we need the regularity of system (4), i.e., that the derivative matrix of (13) is nonsingular. The functional matrix with respect to the variables $r_i$, $i = 1, \ldots, m_e$, and $l_j$, $j = 1, \ldots, n_e$, is given by

$$\begin{pmatrix} D_{K,l} & : & D_r K \\ D_l K^T & : & D_{K,r} \end{pmatrix} \ , \tag{14}$$

where $K$ is an $m_e \times n_e$ matrix with elements $K_{ij}$, $i = 1, \ldots, m_e$, $j = 1, \ldots, n_e$, and $D_{K,l}$, $D_{K,r}$, $D_l$, $D_r$ are the diagonal matrices

$$D_{K,l} := diag(1 + \sum_{k=1}^{n_e} K_{1k} l_k, \ldots, 1 + \sum_{k=1}^{n_e} K_{m_e k} l_k) \ ,$$

$$D_{K,r} := diag(1 + \sum_{k=1}^{m_e} K_{k1} r_k, \ldots, 1 + \sum_{k=1}^{m_e} K_{k n_e} r_k) \ ,$$

$$D_r := diag(r_1, \ldots, r_{m_e}) \ ,$$

$$D_l := diag(l_1, \ldots, l_{n_e}) \ .$$

If we compare now the sums of the first $m_e$ columns of (14) with the corresponding diagonal element, we get the estimate

$$1 + \sum_{k=1}^{n_e} K_{ik} l_k > \sum_{k=1}^{n_e} K_{ik} l_k \tag{15}$$

for $i = 1, \ldots, m_e$. In the same way we obtain for the column sums of the second part

$$1 + \sum_{k=1}^{m_e} K_{kj} r_k > \sum_{k=1}^{m_e} K_{kj} r_k \tag{16}$$

for $j = 1, \ldots, n_e$. Both (15) and (16) show that (14) is a diagonal dominant matrix provided that all data are nonnegative, and we get the following theorem.

**Theorem:** *Assume that the data $K_{ij}$, $l_j$ and $r_j$ are nonnegative for $i = 1, \ldots, m_e$, $j = 1, \ldots, n_e$. Then the functional matrix of the left hand side of (13) is nonsingular.*

The so-called *time* variable is $t := L_{n_e}$. The unknown parameter $x$ to be estimated consists of the association constants $K_{ij}$, $i = 1, \ldots, m_e$, $j = 1, \ldots, n_e$, and the total concentrations $R_i$ for $i = 1, \ldots, m_e$ and $L_j$ for $j = 1, \ldots, n_e - 1$. Thus we get $n := m_e n_e + m_e + n_e - 1$ parameters that must be computed by a least-squares fit.

However when applying the model just described to practical situations, e.g., the radioimmunological determination of Fenoterol, it turns out that the *time* parameters $L_{n_e}$ varies in the order of magnitude drastically. To prevent numerical instabilities, it is recommended to scale the last equation of (13) by $L_{n_e}$, yielding

$$\frac{l_{n_e}}{L_{n_e}}(1 + \sum_{k=1}^{m_e} K_{kn_e} r_k) - 1 = 0 \quad , \tag{17}$$

which is used instead of the original equation whenever $L_{n_e} > 1$.

For the same reason it is not recommended to use the result of a previous numerical solution of (13) whenever $L_{n_e}$ is changed. Instead it is proposed to insert

$$(R_1, \ldots, R_{m_e}, L_1, \ldots, L_{n_e-1}, 1)^T \tag{18}$$

in the numerical algorithm solving system (13) or (4), respectively.

The fitting criterion, i.e., the model function $h^s(x, t)$ in (3) is given by the expression

$$h^s(x, z, t) = L_1 - l_1 = \sum_{k=1}^{m_e} x_{k1} \quad . \tag{19}$$

We consider two examples in form of coupled mass equilibrium problems, that are frequently formulated in practice. The first model consists of one receptor and two ligands, i.e., $m_e = 1$ and $n_e = 2$. Proceeding from (13) we obtain the system

$$\begin{aligned} r_1(1 + K_{11} l_1 + K_{12} L_2 l_2^\star) - R_1 &= 0 \ , \\ l_1(1 + K_{11} r_1) - L_1 &= 0 \ , \\ l_2^\star(1 + K_{12} r_1) - 1 &= 0 \ . \end{aligned}$$

The system parameters are $r_1$, $l_1$ and $l_2^\star$, and the parameters to be estimated, are $K_{11}$, $K_{12}$, $R_1$ and $L_1$. $L_2$ is the independent model or *time* variable to be filled with experimental data. The fitting criterion is $L_1 - l_1$ and we use the starting values $r_1 = R_1$, $l_1 = L_1$ and $l_2^\star = 1.0$ for solving the system of nonlinear equations. The last equation is scaled according to (17) by $l_2^\star := l_2/L_2$.

The second example proceeds from two receptors and two ligands, i.e. $m_e = 2$ and $n_e = 2$. We use (13) and get the system

$$r_1(1 + K_{11} l_1 + K_{12} L_2 l_2^\star) - R_1 = 0 \ ,$$

$$\begin{aligned} r_2(1 + K_{21}l_1 + K_{22}L_2l_2^\star) - R_2 &= 0 \ , \\ l_1(1 + K_{11}r_1 + K_{21}r_2) - L_1 &= 0 \ , \\ l_2^\star(1 + K_{12}r_1 + K_{22}r_2) - 1 &= 0 \ , \end{aligned}$$

where the last equation is scaled (17) and where we define $l_2^\star := l_2/L_2$. The system parameters are $r_1$, $r_2$, $l_1$ and $l_2^\star$, the optimization parameters $K_{11}$, $K_{12}$, $K_{21}$, $K_{22}$, $R_1$, $R_2$ and $L_1$. $L_2$ is the independent *time* variable, now combined through $l_2^\star$ with $l_2$. Fitting criterion is $L_1 - l_1$ and starting values for solving the nonlinear system are $(R_1, R_2, L_1, 1)^T$.

In particular these small-scale models have been studied in the literature, see Bürgisser, Hancock, Lefkowitz, De Lean (1980), De Lean, Hancock, Lefkowitz (1981) and Rominger, Albert (1985), and are regularly used in testing new substances. Also some software is available for the special application, e.g., the LIGAND system of Munson and Rodbard (1980).

## 5. Numerical Results

Both versions of the SYSFIT program differing by the two optimization methods implemented, are designed to solve parameter-estimation problems in arbitrary systems of nonlinear equations. Thus the user has to provide his model functions in form of a separate FORTRAN subroutine.

On the other hand the special mass equlilibrium model describing receptor-ligand binding studies as outlined in the previous section, can be implemented easily and the resulting executable program is running daily in a chemical laboratory. Because of this special type of application that motivated the development of SYSFIT all test cases are coupled mass equilibrium problems mostly with real life experimental data.

To facilitate the input of data and the formulation of the nonlinear system of equations, an *easy-to-use* interface was developed under the name EASY_SYS, see Schittkowski (1992) for details. The user interface is running under MS-DOS, although the numerical programs executed, can be transferred to any other system as well. Besides of the internal binding models that are completely described by the number $m_e$ of receptors and the number $n_e$ of ligands also other nonlinear systems of equations and fitting criteria can be specified through the PCOMP-language, see Dobmann, Liepelt, and Schittkowski (1994), that interpretes statements in a FORTRAN-like syntax und evaluates gradients automatically. Consequently we do not need to compile and link the program when changing the model. EASY_SYS was used for preliminary analysis and the preparation of plot data.

All numerical test results have been obtained on a HP-Apollo workstation running under HP-UX. The codes are implemented in double precision FORTRAN.

Table 1 summarizes all test cases identified by a name and the number of receptors and ligands. Moreover the number of experiments, the number of multiple measurements and the range for the $L_{n_e}$-values is reported. Note that $L_{n_e}$ plays the role of the *time* variable, and the table shows that their order of magnitude varies drastically. By the relatively low number of receptors and ligands, respectively, the most frequent practical situations are reflected.

The measurement data are in most cases real-life experiments. In some situations the data are generated from predetermined parameter sets, to get also least-squares problems with small residuals. All residuals are scaled by the inverse of the sum of squares of all observation values $y_i$, a frequently used scaling technique in practical applications.

The goal of the numerical tests is to get an impression on the degree of difficulty of the least-squares problems and the robustness of the optimization algorithms. The number of test runs is by far too small to derive any conlusions about the relative superiority of one code over the other. Therefore all numerical tests proceed from the same starting point $x_0 = (1, \ldots, 1)^T$, which is in some cases far away from a solution of the least-squares problems. Note however, that whenever a code like SYSFIT is running routinely in a practical environment, the users know a lot about the internal structure of the model and the experiment leading to the data to be fitted. Thus they are able to provide much better starting points so that the inefficiencies we observe, are somewhat *academic*.

In Table 2 we collect the numerical results achieved. For each data set the least-squares algorithms DN2GB of Dennis, Gay, and Welsch (1981b) and DFNLP of Schittkowski (1988) are executed. For both codes we define one set of tolerances that is not changed within the test series. The relative stopping tolerance for function and variable values was set to 1.0E-5 for DN2GB and the final accuracy for satisfying an optimality condition was set to 1.0E-8 for DFNLP. The maximum number of iterations, i.e., gradient evaluations, is 300 for both algorithms. The SQP-code NLPQL of Schittkowski (1985/86) implemented to solve the systems of nonlinear equations, uses the termination tolerance 1.0E-11 for satisfying a Kuhn-Tucker condition, and is not allowed to make more than 50 iterations. However the numerical tests show that the solution of systems of nonlinear equations usually requires not more than 6 iterations with respect to the very small final accuracy.

In Table 2 we list the name of the test case, the least-squares code DN2GB or DFNLP and the final weighted residual. The subsequent two columns list the number of function and gradient evaluations of the outer least-squares algorithm until a termination condition is satisfied, i.e., evaluations of function sets $f_1(x)$, ..., $f_l(x)$ and derivatives $\nabla f_1(x)$, ..., $\nabla f_l(x)$. Finally the last column reports the termination message as generated by the least-squares algorithm. The meaning of these parameters is listed in the following table and more information can be retrieved from the corresponding program documentations.

| *code* | $i$ | *meaning* |
|---|---|---|
| DN2GB | 4 | relative function convergence |
| | 7 | singular convergence |
| | 8 | false convergence |
| | 9 | function evaluation limit |
| | 10 | iteration limit |
| DFNLP | 0 | optimality conditions satisfied |
| | 1 | iteration limit |
| | 4 | function evaluation limit in line search |

First we observe that the numerical results differ drastically even if we have the same underlying mathematical model and even if the experimental data seem to have more or less the same structure. In quite a few test cases, the least-squares algorithms are unable to reach a solution.

To get an overview about the achieved results, we summarize them in the subsequent table. We list the average number of function and gradient evaluations for both algorithms under the condition that the relative difference of the computed residuals is less than some tolerance $\epsilon$. Moreover the number of test runs for which the one or other algorithm got a higher residual value, is also reported under the column $n_{err}$.

| $\epsilon$ | DN2GB | | | DFNLP | | |
|---|---|---|---|---|---|---|
| | $n_f$ | $n_{\nabla f}$ | $n_{err}$ | $n_f$ | $n_{\nabla f}$ | $n_{err}$ |
| 10.0 | 132.6 | 72.6 | 6 | 150.0 | 66.6 | 1 |
| 1.0 | 120.0 | 61.9 | 7 | 135.9 | 62.2 | 4 |
| 0.1 | 141.6 | 70.8 | 8 | 107.1 | 48.8 | 8 |
| 0.01 | 146.6 | 73.4 | 8 | 112.9 | 50.1 | 10 |
| 0.001 | 34.0 | 25.9 | 13 | 67.3 | 36.0 | 11 |
| 0.0001 | 14.5 | 11.5 | 15 | 13.7 | 11.2 | 11 |

To understand the interpretation of the results, consider the first row of the table. It says that in 6 test runs, DN2GB computed a residual that is ten times bigger than the corresponding one computed by DFNLP. On the other hand, DFNLP found only one residual being ten times larger than the residual of DN2GB. For all other numerical solutions, mean values are computed and listed indicating e.g. that the average number of function evalutions of DN2GB is 72.6 against 66.6 of DFNLP.

In some of the failure situations, one algorithm satisfies its internal optimality conditions earlier than the other. It is either possible that another local solution is approximated, or that the curvature of the objective function is very flat in some region. In the first case, the code terminates successfully from the viewpoint of the solution method. In the second case, the model could be very ill-conditioned preventing the algorithm from making significant progress close to a solution.

Also the average number of iterations is relatively large indicating that the starting point is far away from a solution and that the least squares problems are by no means trivial. There is no significant difference between the number of function or gradient evaluations between the two optimization algorithms, when both approximate the same optimal solution very closely. These are, however, the simple test cases and lead to problems which can be solved in relatively few iterations. On the other hand the more difficult problems identified by a larger effort to reach a solution, are also more unstable, i.e. the algorithms stop quite often at different solution points.

Again we have to indicate the importance of initial parameter values. By exploiting additional knowledge about the experiment and numerical experience from the past, all these problems can be solved much more efficiently in practice.

| *problem* | $m_e$ | $n_e$ | $l$ | $v$ | $L_{n_e}^{min}$ | $L_{n_e}^{max}$ |
|---|---|---|---|---|---|---|
| E11-1 | 1 | 1 | 33 | 3 | 0.3 | 3.0E+4 |
| E11-2 | 1 | 1 | 27 | 3 | 1.0 | 1.0E+9 |
| E11-3 | 1 | 1 | 7 | 1 | 100.0 | 3.0E+3 |
| E12-1 | 1 | 2 | 27 | 3 | 0.1 | 3.0E+5 |
| E12-2 | 1 | 2 | 44 | 3-4 | 0.1 | 1.0E+7 |
| E12-3 | 1 | 2 | 10 | 1 | 0.001 | 1.0E+6 |
| E12-4 | 1 | 2 | 12 | 1 | 1.0 | 1.0E+10 |
| E12-5 | 1 | 2 | 10 | 1 | 5.2 | 5.2E+3 |
| E12-6 | 1 | 2 | 21 | 1 | 10.0 | 1.0E+7 |
| E12-7 | 1 | 2 | 18 | 3 | 100.0 | 1.0E+7 |
| E12-8 | 1 | 2 | 21 | 3 | 10.0 | 1.0E+7 |
| E12-9 | 1 | 2 | 21 | 3 | 10.0 | 1.0E+7 |
| E12-10 | 1 | 2 | 42 | 3 | 3.0 | 1.0E+6 |
| E12-11 | 1 | 2 | 45 | 3 | 3.0 | 3.0E+6 |
| E12-12 | 1 | 2 | 16 | 1 | 1.0 | 5.2E+3 |
| E12-13 | 1 | 2 | 12 | 1 | 1.0 | 2.9E+3 |
| E12-14 | 1 | 2 | 15 | 1 | 0.01 | 5.0E+4 |
| E13-1 | 1 | 3 | 26 | 1 | 0.01 | 1.0E+3 |
| E21-1 | 2 | 1 | 20 | 1 | 100.0 | 1.3E+5 |
| E21-2 | 2 | 1 | 10 | 1 | 0.1 | 1.0E+2 |
| E21-3 | 2 | 1 | 22 | 2 | 3.0 | 3.0E+5 |
| E21-4 | 2 | 1 | 33 | 3 | 0.3 | 3.0E+4 |
| E22-1 | 2 | 2 | 16 | 1 | 0.1 | 1.0E+5 |
| E22-2 | 2 | 2 | 75 | 3 | 0.1 | 1.0E+5 |
| E22-3 | 2 | 2 | 26 | 1 | 0.01 | 1.0E+5 |
| E22-4 | 2 | 2 | 22 | 1 | 0.01 | 1.0E+4 |
| E22-5 | 2 | 2 | 66 | 3 | 60.0 | 1.0E+7 |
| E23-1 | 2 | 3 | 18 | 1 | 1.0 | 1.0E+11 |
| E31-1 | 3 | 1 | 11 | 1 | 0.1 | 1.0E+5 |
| E32-1 | 3 | 2 | 47 | 3-4 | 0.1 | 1.0E+7 |
| E32-2 | 3 | 2 | 20 | 1 | 0.1 | 6.0E+3 |
| E32-3 | 3 | 2 | 31 | 1 | 0.1 | 1.0E+11 |

Table 1: Test examples

| *problem* | *code* | $r^\star$ | $n_f$ | $n_{\nabla f}$ | $i$ |
|---|---|---|---|---|---|
| E11-1 | DN2GB | .31999556E-2 | 14 | 13 | 4 |
| | DFNLP | .31999557E-2 | 25 | 20 | 0 |
| E11-2 | DN2GB | .73164932E-1 | 4 | 4 | 4 |
| | DFNLP | .73164932E-1 | 5 | 5 | 0 |
| E11-3 | DN2GB | .13881922E-4 | 20 | 17 | 4 |
| | DFNLP | .13881924E-4 | 10 | 9 | 0 |
| E12-1 | DN2GB | .18746759E-2 | 25 | 18 | 7 |
| | DFNLP | .18738550E-2 | 282 | 124 | 0 |
| E12-2 | DN2GB | .18411076E-2 | 110 | 69 | 4 |
| | DFNLP | .18780923E-2 | 119 | 65 | 0 |
| E12-3 | DN2GB | .81951850E-4 | 153 | 72 | 7 |
| | DFNLP | .40521965E-3 | 21 | 15 | 0 |
| E12-4 | DN2GB | .22565179E-3 | 20 | 14 | 7 |
| | DFNLP | .35322154E-3 | 55 | 28 | 0 |
| E12-5 | DN2GB | .21838945E-3 | 246 | 140 | 7 |
| | DFNLP | .17882418E-2 | 816 | 300 | 1 |
| E12-6 | DN2GB | .38360135E-2 | 160 | 120 | 7 |
| | DFNLP | .38355137E-2 | 174 | 97 | 0 |
| E12-6 | DN2GB | .24333253E-2 | 43 | 22 | 4 |
| | DFNLP | .24184895E-2 | 67 | 35 | 4 |
| E12-8 | DN2GB | .43556653E-2 | 104 | 37 | 8 |
| | DFNLP | .45256660E-2 | 15 | 14 | 0 |
| E12-9 | DN2GB | .17543020E-2 | 68 | 47 | 4 |
| | DFNLP | .29484979E-2 | 39 | 25 | 0 |
| E12-10 | DN2GB | .33184711E-2 | 90 | 54 | 7 |
| | DFNLP | .41217194E-2 | 231 | 148 | 0 |
| E12-11 | DN2GB | .20182247E-2 | 38 | 26 | 7 |
| | DFNLP | .20182096E-2 | 34 | 25 | 0 |
| E12-12 | DN2GB | .11793656E-3 | 4 | 4 | 4 |
| | DFNLP | .11793658E-3 | 3 | 3 | 0 |
| E12-13 | DN2GB | .10896416E-2 | 385 | 300 | 10 |
| | DFNLP | .61315567E-2 | 38 | 26 | 0 |
| E12-14 | DN2GB | .20214247E-3 | 68 | 43 | 4 |
| | DFNLP | .24250829E-3 | 780 | 300 | 1 |
| E13-1 | DN2GB | .35496344E-2 | 9 | 7 | 9 |
| | DFNLP | .83539111E-6 | 74 | 48 | 0 |

Table 2: Numerical results (continued)

| *problem* | *code* | $r^\star$ | $n_f$ | $n_{\nabla f}$ | $i$ |
|---|---|---|---|---|---|
| E21-1 | DN2GB | .95882426E-2 | 7 | 5 | 7 |
| | DFNLP | .95881311E-2 | 5 | 5 | 0 |
| E21-2 | DN2GB | .22928501E-2 | 9 | 7 | 9 |
| | DFNLP | .22860469E-2 | 18 | 14 | 0 |
| E21-3 | DN2GB | .29177488E-1 | 9 | 5 | 9 |
| | DFNLP | .71931171E-2 | 20 | 17 | 0 |
| E21-4 | DN2GB | .38118137E-2 | 9 | 8 | 9 |
| | DFNLP | .33262345E-2 | 35 | 25 | 0 |
| E22-1 | DN2GB | .21259883E-6 | 556 | 300 | 10 |
| | DFNLP | .21844094E-3 | 795 | 300 | 1 |
| E22-2 | DN2GB | .19120880E-2 | 521 | 240 | 7 |
| | DFNLP | .18993610E-2 | 222 | 97 | 0 |
| E22-3 | DN2GB | .86277200E-3 | 599 | 250 | 9 |
| | DFNLP | .85717898E-3 | 63 | 38 | 0 |
| E22-4 | DN2GB | .10273025E-2 | 599 | 297 | 9 |
| | DFNLP | .10364061E-2 | 664 | 222 | 0 |
| E22-5 | DN2GB | .80301533E-1 | 9 | 6 | 9 |
| | DFNLP | .87379733E-3 | 72 | 50 | 1 |
| E23-1 | DN2GB | .53578544E-4 | 9 | 4 | 9 |
| | DFNLP | .53485176E-4 | 8 | 8 | 0 |
| E31-1 | DN2GB | .56380461E-3 | 9 | 6 | 9 |
| | DFNLP | .21771691E-5 | 79 | 69 | 0 |
| E32-1 | DN2GB | .20098739E-0 | 57 | 13 | 8 |
| | DFNLP | .34415227E-2 | 79 | 43 | 0 |
| E32-2 | DN2GB | .51722003E-1 | 43 | 7 | 8 |
| | DFNLP | .47775840E-3 | 77 | 39 | 0 |
| E32-3 | DN2GB | .74446523E-1 | 9 | 6 | 9 |
| | DFNLP | .84562137E-4 | 300 | 183 | 0 |

Table 2: Numerical results

## Acknowledgement

The author would like to thank Dr. Rominger, Dr. Speck and Dr. Wolf from Boehringer Ingelheim KG for the motivation to begin this study, and the cooperation, particularly the provision of binding models and experimental data.

## References

Bürgisser E., Hancock A.A., Lefkowitz R.J., De Lean A. (1980): *Anomalous equilibrium binding properties of high-affinity racemic radioligands,* Molecular Pharmacology, Vol. 19, 205-216

De Lean A., Hancock A.A., Lefkowitz R.J. (1981): *Validation and statistical analysis of a computer modeling method for quantitative analysis of radioligand binding data for mixtures of pharmacological receptor subtypes,* Molecular Pharmacology, Vol. 21, 5-16

Dennis J.E.jr., Gay D.M., Welsch R.E. (1981a): *An adaptive nonlinear least-squares algorithm,* ACM Transactions on Mathematical Software, Vol. 7, No. 3, 348-368

Dennis J.E.jr., Gay D.M., Welsch R.E. (1981b): *Algorithm 573: NL2SOL-An adaptive nonlinear least-squares algorithm,* ACM Transactions on Mathematical Software, Vol. 7, No. 3, 369-383

Feldman, H.A. (1972): *Mathematical theory of complex ligand-binding systems of equilibrium,* Mathematical Biochemistry, Vol. 48, 317-338

Kranz D.M., Herron J.N., Voss Jr. E.W. (1982): *Mechanisms of ligand binding by monoclonal anti-fluorescyl antibodies,* The Journal of Biological Chemistry, Vol. 257, No. 12, 6987-6995

Lawson C.L., Hanson R.J. (1974): *Solving Least Squares Problems,* Prentice Hall, Englewood Cliffs, N.J.

Dobmann M., Liepelt M., Schittkowski K. (1994): *PCOMP: A FORTRAN code for automatic differentiation,* to appear: ACM Transactions on Mathematical Software

Munson P.J., Rodbard D. (1980): *LIGAND: A versatile computerized approach for characterization of ligand-binding systems,* Analytical Biochemistry, Vol. 107, 220-239

ROMINGER K.L., ALBERT H.J. (1985): *Radioimmunological determination of Fenoterol. Part I: Theoretical fundamentals,* Arzneimittel-Forschung/Drug Research, Vol. 35, No. 1a, 415-420

SCHITTKOWSKI K. (1985/86): *NLPQL: A FORTRAN subroutine solving constrained nonlinear programming problems,* Annals of Operations Research, Vol. 5, 485-500

SCHITTKOWSKI K. (1988): *Solving nonlinear least-squares problems by a general purpose SQP-method,* in: Trends in Mathematical Optimization, K.-H. Hoffmann, J.-B. Hiriart-Urruty, C. Lemarechal, J. Zowe eds., International Series of Numerical Mathematics, Vol. 84, Birkhäuser

SCHITTKOWSKI K. (1992): *SYSFIT: A FORTRAN code for estimating parameters in systems of nonlinear equations and in coupled mass equilibrium models,* Report, Mathematisches Institut, Universität Bayreuth