Computing Radioimmunoassay Analysis Dose-Response Curves

Gary D. Knott, Ph.D.
Civilized Software, Inc.
12109 Heritage Park Circle
Silver Spring MD 20906
Tel.: (301)-962-3711
email: csi@civilized.com
URL: www.civilized.com

Introduction

The study of how a ligand material, such as a hormone or antigen, binds to one or more kinds of molecular complexes, called sites, is of fundamental importance in biochemistry. Sites are often embedded in cell membranes, and the binding may serve to control the behavior of the cell itself. Typically we are interested in the number of distinct kinds of sites and their frequency of occurrence, and also the equilibrium constants for each ligand-site binding reaction which indicates the absolute strength of each such binding reaction.

For example, quantitative analysis of ligand-receptor binding can be easily performed using appropriate software such as the MLAB system discussed in this paper. MLAB is a computer program whose name is an acronym for "modeling laboratory"; it is an interactive system for mathematical modeling, originally developed at the National Institutes of Health. MLAB can fit multiple non-linear models to data points. We are interested here in *designing* standard displacement assays. Typically such assays involve measuring the competition between radiolabelled and cold ligand in detergent-solubilized membrane preparations or on whole cells. Affinity constants and

limit values of binding protein concentrations for single or multiple sites can be computed by fitting saturation curves in MLAB. Output can include Scatchard plots obtained by a suitable transformation.

One specific elaboration of the study of simple binding equilibrium states arises in radioimmunoassay analysis. Suppose P is a substance (a ligand or antigen) to be assayed which binds with a material Q (possibly, but not necessarily, an antibody) to form a complex B, and suppose further that H is a radioactive material which competes with P in binding to Q molecules. H is called the labeled or "hot" ligand. Thus, we have:

$$\begin{aligned} H + Q &\underset{K_2}{\overset{K_1}{\rightleftharpoons}} L \\ P + Q &\underset{K_4}{\overset{K_3}{\rightleftharpoons}} B. \end{aligned}$$

Let P_0, H_0, and Q_0 be the initial amounts of P, H, and Q. If a number of experiments with different values of P_0, but with H_0 and Q_0 fixed, are done, and the bound labeled material L is measured, we obtain a set of data points (P_0, L) which, except for error, define a so-called dose-response curve. Generally, L falls as P_0 increases, because more P is competing with the hot ligand, so more B and less L are formed. Given such a dose-response curve, we can mix an unknown amount (possibly zero) of P (the dose) with H_0 moles of H and Q_0 moles of Q and measure L (the response) at equilibrium, and then read off from our dose-response curve how much P there must have been. This is an assay for the material P. The radioactivity is merely a device that allows us to measure the amount of L that is formed. Any other method that allows the amount of L to be determined can alternately be used.

Developing an assay entails providing a suitably-accurate dose-response curve. Thus, we wish to refine our dose-response curve by curve-fitting the experimentally-obtained data points (P_0, L).

Let the equilibrium constant $A_1 = K_1/K_2 = L/(HQ)$, and let the equilibrium constant $A_2 = K_3/K_4 = B/(PQ)$, with all species measured at equilibrium. Then $Q_0 = Q + B + L$, $H_0 = H + L$, and $P_0 = P + B$, so with some manipulation, we have:

$L = A_1 H_0 Q/(1+A_1 Q)$, and $Q_0 = Q(1+A_2 P_0/(1+A_2 Q)+A_1 H_0/(1+A_1 Q))$.

We can define this model in MLAB as follows:

```
*FUNCTION L(P0) = LV(Q(P0))
*FUNCTION LV(QV) = A1*H0*QV/(1+A1*QV)
*FUNCTION Q(P0) = ROOT(Z,0,Q0,Z*(1+A1*H0/(1+A1*Z)+A2*P0/(1+A2*Z))-Q0)
```

These commands exemplify the MLAB FUNCTION statement, which is used to define a function or differential equation. Note that arguments of functions must be explicitly specified. Variables, such as /A1/ and /A2/, which appear in the body of a function, but not in its argument list, are called *parameters*. Parameters must be assigned values before an associated function can be evaluated.

ROOT is an operator which is built-in in MLAB. /ROOT(Z,A,B,E)/ is a value between /A/ and /B/ which, when taken as the value of the dummy variable, /Z/, makes the expression, /E/, which involves /Z/, equal to zero. Thus /ROOT(Z,A,B,E)/ is a solution of $E(Z) = 0$. The model given above, involving a so-called *implicit* function, deserves careful study.

Now we can use *MLAB* to fit L to the (P_0, L)-data by adjusting A_1 and A_2 (and even H_0 and Q_0 if desired). Often we assume $A_1 = A_2$, but this is not necessary. This model demands that the data points be taken at equilibrium, and that the reaction is adequately

described by the simple competitive scheme given above. In practice this model is very sensitive to the initial guesses, and constraints are often required. Usually H_0 is fixed at 1 and the L-data is expressed as a percent of total H. The P_0 data should then be in percent of total H units, as well.

The MLAB Mathematical Modeling System

MLAB has hundreds of useful functions, e.g., the discrete Fourier transform function dft and the parametric spline interpolation function splinep. One of the central components of the system is a curve fitting program which will adjust the parameters of a model function to minimize the weighted sum of the errors raised to a specified power. A repertoire of mathematical operators and functions, routines for solving differential equations, a collection of routines for onscreen drawing and for hardcopy plotting, and mechanisms for saving data between sessions provide a powerful and convenient environment for data manipulation, arithmetic calculations, and for building and testing models.

The user communicates with MLAB by typing commands which are executed at once or by providing a script to be executed. Should the user have questions, typing /HELP/ will put the on-line system documentation at his disposal. The MLAB language is defined in the MLAB reference manual.

One of MLAB's main uses is to fit models to data. Curve-fitting is a useful analytical tool in many diverse disciplines. The basic notion is easily described. Given data, say various points in the plane (x_1, y_1), (x_2, y_2), ..., (x_n, y_n), and a function $y = f(x)$ where f involves some parameters, say a and b, as for example $f(x) = ax^b +$

1, we may wish to calculate values for the parameters a and b so that the function f well-predicts the observed data, that is, so that $f(x_i) = y_i$ for $1 \leq i \leq n$. In this case, we say we have fit the model f to the data by estimating the parameters a and b. The end result is merely the values obtained for the initially unknown parameters. The same principles apply in higher dimensions with arbitrarily many parameters. MLAB can simultaneously fit multiple non-linear model functions, some or all of which may be implicit functions, or may even be defined by a system of differential equations.

The curve-fitting and graphics display facilities of MLAB make it an ideal tool for the estimation of equilibrium constants from data, which typically consists of observed amounts of ligand bound for various specified amounts of ligand provided for binding.

Dose-Response Curve Computation in MLAB

Here is an example showing the calculation of the dose-response curve for certain data which is read-in-below.

To begin we introduce the appropriate constraints (which should always be used for this particular model), guess the equilibrium constants A_1 and A_2, and then estimate them, as follows.

```
* CONSTRAINTS N ={A1>0,A2>0}
```

The /CONSTRAINTS/ statement permits the user to specify successive linear inequalities or equations involving the parameters (or potential parameters). Now we may specify values for the parameters, guessing when necessary.

```
A1 = 1; A2 = 2; H0 = 2; Q0 = 4
```

Here the /ASSIGNMENT/ statement is exemplified. In this case all the above assignments are assigning values to scalar variables, but the /ASSIGNMENT/ statement is used to assign values to matrices as well. This can be seen in the next /ASSIGNMENT/ statement which defines a matrix, M.

```
M = READ(assay,21,2)
```

The /READ/ operator takes optional array size arguments; in this case a 21 row by 2 column matrix is specified, and reads in numbers from the specified file to construct an appropriately-dimensioned matrix as the result. This matrix is then, in this case, assigned to /M/.

We have established a model function, L, and entered data, M. We expect that $L(M[i,1]) \approx M[i,2]$ would hold, if only the parameters A_1 and A_2 were set to their ''correct'' values. The following /FIT/ statement requests MLAB to estimate A_1 and A_2 by assigning them values which minimize the sum-of-squares objective function $S(A_1, A_2) = \sum_{i=1}^{8} \left(L(M[i,1]) - M[i,2]\right)^2$.

```
FIT(A1,A2), L TO M WITH WEIGHT EWT(M), CONSTRAINTS N

final parameter values
      value               error                dependency    parameter
   6.1567727039        0.9593487946          0.7470305183    A1
  60.5719353014       12.2219429924          0.7470305183    A2
6 iterations
```

```
CONVERGED
best weighted sum of squares = 9.870322e+01
weighted root mean square error = 2.279234e+00
weighted deviation fraction = 5.480005e-02
R squared = 9.873349e-01
no active constraints
```

The behavior of the /FIT/ statement depends upon the supplied constraints as well as upon the MLAB control variables: /MAXITER/, the maximum number of iterations and /TOLSOS/, the requested convergence factor.

MLAB uses a carefully-tuned version of the Marquardt-Levenberg magnified-diagonal algorithm which is, in turn, a form of the Gauss-Newton procedure for minimizing a function which is in the form of a sum-of-squares. This process estimates the value of the parameter vector ($b = (A_1, A_2)$ in our case) by successive approximations $b^{(0)}$, $b^{(1)}$, ..., $b^{(n)}$, where $b^{(0)}$ is the vector of initial guesses for A_1 and A_2, and $b^{(j+1)} = b^{(j)} + \beta^{(j)}$, where

$$
\begin{aligned}
\beta^{(j)} &= (X'V^{-1}X + \varepsilon G)^{-1}X'V^{-1}(y - (f(x_1; b^{(j)}), \ldots, f(x_8; b^{(j)}))'), \quad \text{with} \\
X_{st} &= \partial f(x_s; b^{(j)})/\partial b_t \quad \text{and} \\
G_{st} &= \text{if } s = t \text{ then } (X'V^{-1}X)_{st} \text{ else } 0 \quad \text{and} \\
x_s &= M[s, 1] \quad \text{for} \quad 1 \leq s \leq 8, \quad \text{and} \\
y &= (M[1, 2], \ldots, M[8, 2])',
\end{aligned}
$$

where V is the estimated covariance matrix of the observations. In our example, $V = I$, the identity matrix. In general V is determined from weight-values supplied by the user.

An iteration consists of computing $b^{(j+1)}$ from $b^{(j)}$. Note that this requires the partial derivatives of the model function with respect to the parameters evaluated at $b^{(j)}$, since these values form the matrix X. In MLAB, these derivatives are automatically computed symbolically and evaluated to form X. The convenience thus obtained is considerable and the parameter estimation process is provided with more accurate derivative values. For example the derivative of L with respect to A_1 can be explicitly displayed in MLAB as follows.

```
* TYPE L'A1, LV'A1, Q'A1
FUNCTION L'A1(P0) = LV'A1(Q(P0))+LV'QV(Q(P0))*Q'A1(P0)
FUNCTION LV'A1(QV) = (H0*QV*(1+A1*QV)-QV*A1*H0*QV)/((1+A1*QV)*(1+A1*QV))
FUNCTION Q'A1(P0) = EVAL(Z,ROOT(Z,0,Q0,
  Z*(1+(A1*H0)/(1+A1*Z)+(A2*P0)/(1+A2*Z))-Q0),
  -(((H0*(1+A1*Z)-Z*A1*H0)/((1+A1*Z)*(1+A1*Z)))*Z)/
  ((1+(A1*H0)/(1+A1*Z)+(A2*P0)/(1+A2*Z))-((A2*A2*P0)/
  ((1+A2*Z)*(1+A2*Z))+(A1*A1*H0)/((1+A1*Z)*(1+A1*Z)))*Z))
```

Indeed derivatives are full-fledged members of the class of functions and can be used in graphics or curve-fitting in MLAB just as can any other user-defined function.

A sub-iteration consists of computing $b^{(j+1)}$ with a particular value of ε which specifies the amount of diagonal magnification. At each iteration, the value ε starts at 10^{-9} and is increased until the corresponding value of $b^{(j+1)}$ results in a smaller sum-of-squares value, whereupon this vector is taken to be the final $b^{(j+1)}$ iterate.

The parameter estimation process stops when the limit of the number of iterations is reached or, more usually, when the decrease in the sum-of-squares value between successive iterations is less than

a specified fractional amount determined by the user-specified convergence factor in /TOLSOS/. For /TOLSOS/ $= .001$, the sum-of-squares must change by less than $.1$ percent for the curve-fitting process to stop based on this criterion.

When the estimation process does stop, the parameters are reset to their computed estimates, and they and their estimated standard deviations are typed out. Associated values called dependency values, which lie between 0 and 1, are also typed out. It suffices here to remark that large dependency values above $.99$ usually indicate a non-unique solution; that is, other parameter estimates exist which would provide a nearly equally-small sum-of-squares.

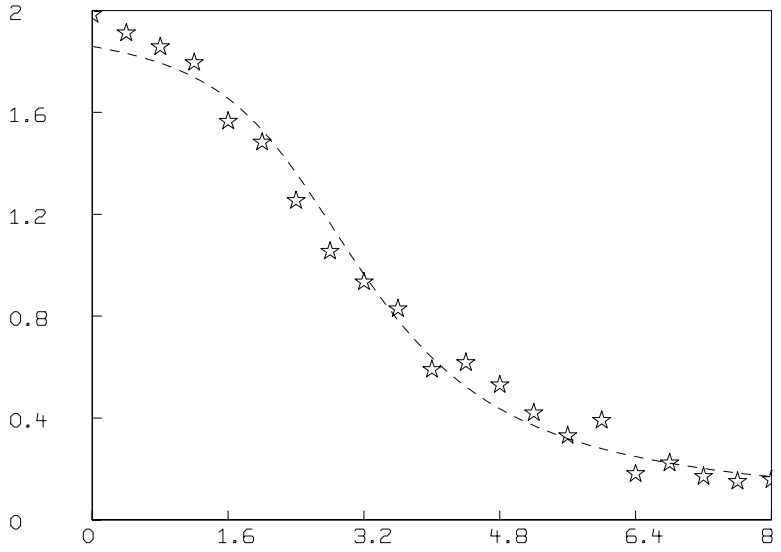
MLAB also types-out the root-mean-square error which is the estimate of the standard deviation in each observation, given that they are identically distributed. This quantity should roughly equal the experimental error in the data. There are, of course, many caveats and restrictions which must hold to insure the validity of the supporting statistics provided.

The material typed out above shows that the vector of parameters (A_1, A_2) has been estimated to be $(6.15677 \pm 0.95935, 60.5719 \pm 12.2219)$, with reasonably small dependency values, and with an RMS error of about $.987$, which should be comparable with the experimental error in our data, M. The sum-of-squares was reduced to 98.7 at the final parameter values.

In order to visually see how our model with its parameters set to their best-estimated values corresponds to the data, we may draw a graph of the data points and the model function. Although we are drawing only the simplest and most direct kind of picture here, it should be noted that MLAB provides facilities for many types

of point-symbols and types of lines, axes with arbitrarily-placed numeric labels in various formats, titles in the form of text strings in arbitrary sizes and various fonts with subscripts and superscripts, color, and a number of other special features. It is quite possible to prepare more or less elaborate publication-quality graphs with a modest amount of effort. Indeed this is one of MLAB's most-used facilities. The desired graph can be constructed as follows.

```
* DRAW M, LINETYPE NONE,POINTTYPE STAR
* DRAW C2 = POINTS(L,0:8:.2)
* VIEW
```



The first /DRAW/ command above plots the data points, while the second /DRAW/ command constructs a curve called /C2/, which is a graph consisting of solid straight-lines connecting the points which

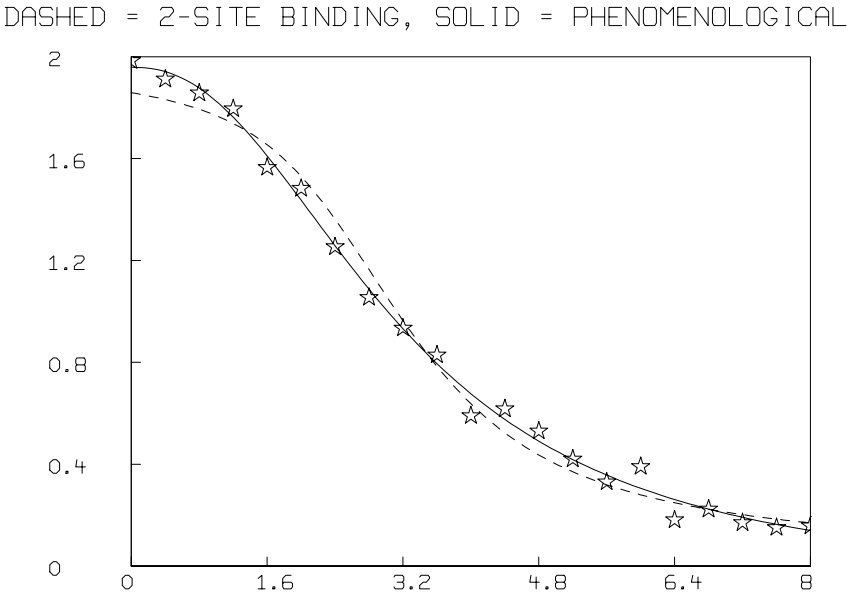
are the rows of the 2-column matrix which is the value of the expression /POINTS(L,0:8:.2)/. This matrix has the values 0 through 8 in steps of .2 in its first column and corresponding values of the function L evaluated at 0 through 8 in steps of .2 in the second column. The /POINTS/ operator is very useful for graphing functions. Both these curves are drawn in the default MLAB 2D-window called /W/ (since no other window is specified) which has predefined labeled axes already present. The picture finally appears when its display is requested with a VIEW statement and a plot can be obtained if desired using the /PLOT/ statement.

Often a ''phenomenological'' model for dose-response curves is preferred. Experience has shown that the function $L(P_0) = (a-d)/(1+(P_0/c)^b)+d$, where b is the ''slope-factor'' approximately equal to 1, a is the zero dose response, d is the infinite dose response, and c is the dose for a 50 percent response, often fits well. In either case, the fitting should be done with weights (usually the MLAB estimated weights, /ewt(m)/, where M is the (P_0, L) data matrix, suffices).

Now let us fit the phenomenological model and draw the comparison below.

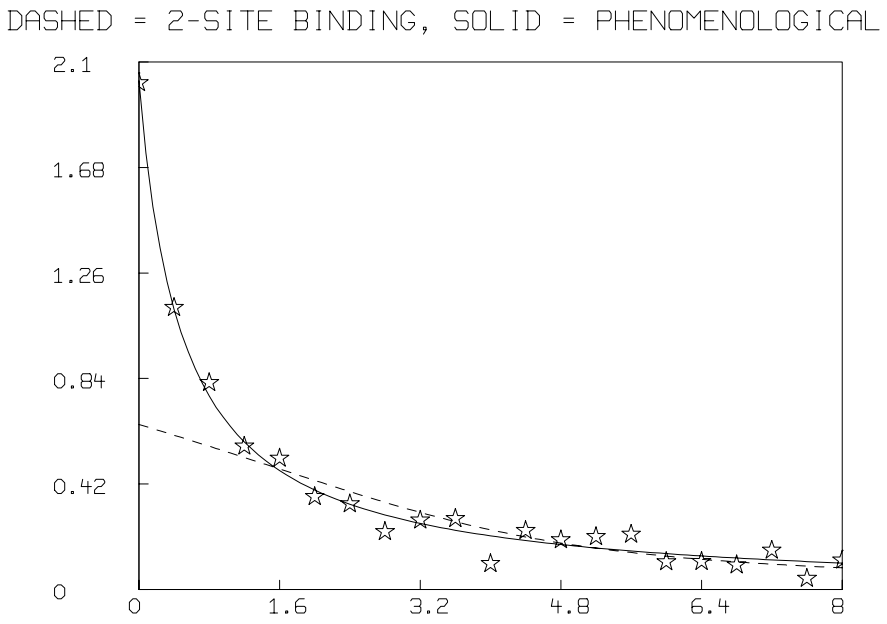
```
FCT EL(p) = (a-d)/(1+(p/c)^b)+d

a = 2; b = 2.5; c = 3.0; d = 0;

FIT(a,b,c,d), EL TO M WITH WEIGHT EWT(M)

final parameter values
      value               error                dependency    parameter
   1.9589377175        0.0253123625          0.6168286269    a
```

```
    2.300705405         0.1574645119          0.8828919509   b
    3.176751857         0.0979352743          0.8541792257   c
  -0.0773479916         0.0539555619          0.9449998881   d
3 iterations
CONVERGED
best weighted sum of squares = 2.767049e+01
weighted root mean square error = 1.275804e+00
weighted deviation fraction = 2.149476e-02
R squared = 9.958668e-01

DRAW M LT NONE, PT STAR
DRAW POINTS(L,0:8!101) LT DASHED
DRAW POINTS(EL,0:8!101)
TOP TITLE "DASHED = 2-SITE BINDING, SOLID = PHENOMENOLOGICAL"
VIEW
```



Both the direct second-order competitive binding model and the phenomenological model fit well here. In other cases, however, the second-order

competitive binding model is inferior to the phenomenological model as is demonstrated in the example below. The reasons for this include the likelyhood that the actual binding reactions are more complex than the simple competitive binding scheme postulated above.

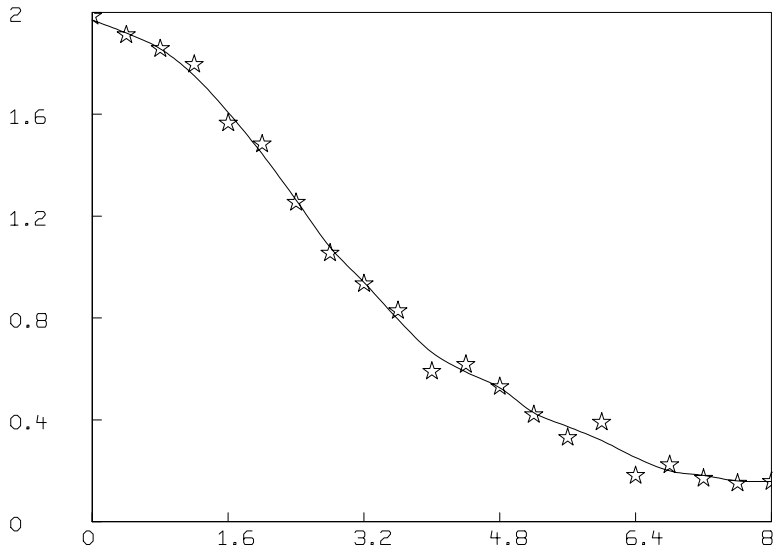


Once the dose-response function L is determined, the assay value for P_0, given an observed amount, L_1, of the material L, is just $V(L_1)$, where in MLAB, we have:

```
*FUNCTION V(L1) = ROOT(X,0,PMAX, L(X)-L1).
```

Finally, as another approach, the function L can be determined entirely empirically from the data matrix M as follows, where the L-unit is fraction of total H.

```
*G = SMOOTH(M,3)
*G = INTERPOLATE(G,0:8:.04)
*DRAW M LT NONE PT STAR
*DRAW G
*VIEW
```



Any of a number of non-parametric regresion schemes can be used; here we have used a weighted moving average smoothing function.

A confidence interval for P_0 can be established, at least approximately, by following the scheme for so-called calibrated estimation. A useful reference in this regard, as well as for other purposes, is: ‘‘The Application of Robust Calibration to Radioimmunoassay’’ by James Tiede and Marcello Pagano in Biometrics, Vol. 35, pp. 567:574, Sept. 1979.