A First Course on Kinetics and Reaction Engineering
Supplemental Unit S4. Numerically Fitting Models to Data

Defining the Problem

Many software routines for fitting a model to experimental data can only be used when the model is of a pre-defined mathematical form. The more common pre-defined forms are linear (see Supplemental Unit S3), polynomial, exponential and logarithmic. This supplemental unit describes how to fit models that do not have one of the pre-determined mathematical forms to experimental data. Specifically, it considers fitting five kinds of models to corresponding data sets.

1. The single response variable is an explicit function of the set variables.
2. The single response variable is computed after first solving a set of non-linear equations.
3. The single response variable is computed after first solving a set of initial value ordinary differential equations.
4. The multiple response variables are computed after first solving a set of non-linear equations, and a complete data set[1] is used for fitting.
5. The multiple response variables are computed after first solving a set of initial value ordinary differential equations, and a complete data set[1] is used for fitting.

Information and Data Required for Numerical Solution

Fewer mathematics programs have routines for the type of fitting considered in this supplemental unit. If a routine is available for fitting a user-provided single response model to experimental data, it will most likely require you, at the minimum, to provide the following input information and data:

- a set of experimental data points, each of which consists of a value for the response variable, $\hat{y}$, and corresponding values for each of the set variables x_1 through $x_{n_{set}}$
- a guess for the value of each unknown parameter that appears in the model
- code that calculates the model-predicted value of the response variable, given values for the parameters and values for the set variables

If the calculation of the model-predicted value of the response variable requires the prior numerical solution of a set of non-linear equations, the code provided above will need to call additional numerical routines for solving those equations (see Supplemental Unit S2). In that case, you will additionally need to provide the following:

- code that evaluates the set of non-linear equations to be solved, given values for the unknowns
- code that provides a guess for the values of the unknowns

[1] For present purposes, a complete multiple response data set is one where the value of every response variable was measured in every experiment.

If the calculation of the model-predicted value of the response variable requires the prior numerical solution of a set of initial value ordinary differential equations, the code provided above will need to call additional numerical routines for solving initial value ordinary differential equations (see Supplemental Unit S5). In that case, you will additionally need to provide the following:

- code that evaluates the derivatives in the set of initial value ordinary differential equations given values of the independent and dependent variables
- code that provides the initial values of the independent and dependent variables
- code that provides the final value of the independent variable

Once the required input has been provided, the software can be expected to return the best value for each parameter in the model, the value of the correlation coefficient, r^2, and some measure of the uncertainty in the value of each parameter (typically either the standard error or the 95% confidence interval). If the software does not generate them, the results can be used to generate a model plot (when there is only one set variable, x) or a parity plot and residuals plots (when there are two or more set variables) that can be used to assess how accurately the model represents the data.

I am not aware of mathematics software for fitting multiple response models to data. For the two situations considered in this supplemental unit (4 and 5 in the list under "Defining the Problem," above), template files are provided that require the same input as the corresponding single response models. However, the template files only return the best values for the model parameters and the plots; the uncertainties and the correlation coefficient are not computed.

Overview of the Numerical Method

As discussed in Supplemental Unit S3, when a model for an experiment is used to calculate the response variable, y_l, for data point l by substitution of the corresponding $x_{i,l}$ values for that data point, it is not expected to yield the exact same value as the experimentally measured response, $\hat{y}_l$. Instead, due to experimental uncertainty, etc., it is expected that there will be an error (also called a residual), ε_l, as defined in equation (1).

$$\varepsilon_l = \hat{y}_l - y_l \tag{1}$$

The model is fit to the data by adjusting the values of the parameters so as to minimize the aggregate error over all of the data points. The *overall* error or objective function, Φ, *for single response data* is taken to equal the sum over all of the experimental data points of the squares of the errors, equation (2), where g represents the model equation and the θ_i represent the model parameters. Using the squares of the errors prevents positive and negative errors from offsetting each other.

$$\Phi = \sum_{l=1}^{n_e} \varepsilon_l^2 = \sum_{l=1}^{n_e} \left(\hat{y}_l - y_l\right)^2 = \sum_{l=1}^{n_e} \left(\hat{y}_l - g\left(x_{1,l}, x_{2,l}, \cdots, x_{n_s,l}, \theta_1, \theta_2, \cdots, \theta_{n_p}\right)\right)^2 \tag{2}$$

The parameters, $\underline{\theta}$, in equation (2) are to be chosen so that the resulting value of Φ is a minimum. When that is true, the partial derivative of Φ with respect to each of the parameters will be equal to zero. Equation (3) expresses this requirement for any one parameter θ_k. By applying this criterion to each parameter, a set of n_p equations can be generated, and those equations can be solved simultaneously to find the "best" values of the n_p parameters.

$$\frac{\partial \Phi}{\partial \theta_k} = 0 \tag{3}$$

That is the approach described and used in Unit S3 for the situation where the function g is linear. It can be applied equally well to explicit non-linear models. Indeed, for some common non-linear equations (exponentials, logarithms, polynomials, power series, etc.), expressions for the best values of the parameters have been derived in that way and programmed into software such as MATLAB and Excel. For the non-linear equations where that has been done, the regression analysis is performed exactly as described in Unit S3, except that a pre-programmed function for the desired non-linear model is called instead of a function for a linear model. This situation will ***not*** be considered here.

If equations for the parameters have not been derived as described above and pre-programmed into available software, you have two options for performing the regression analysis. The first is to manually take the derivatives in equation (3) for every parameter in the model you are considering and then solve the resulting set of equations for the parameter values. That is a tedious process; a new set of derivatives must be generated for each model that is considered. A less-demanding alternative approach is to minimize equation (2) numerically. This is the approach that is utilized in this supplemental unit.

With this approach, the model does not need to be an explicit expression for the value of the response variable, because it is not necessary to substitute it into the objective function and analytically take the derivative of the result. As a consequence, the model can take the form of a set of differential or non-differential equations that are solved numerically (see Supplemental Units S2 and S5). This can be quite advantageous for the analysis of kinetics data because the reactor design equations are typically sets of initial value ordinary differential equations or sets of non-linear equations.

There are a number of different methods for the numerical minimization of a function. Most of them are variations on a scheme where one begins with a guess for the values of parameters. Using that guess, the objective function is evaluated to find the value of Φ. The guessed parameter values are then perturbed somehow, and the value of Φ corresponding to the perturbed parameters is calculated. The two values of Φ are then compared, and whichever of the two parameter sets gave the lower value of Φ is taken as the new guess. This process is repeated over and over, typically until one of three things happens: a maximum number of iterations occurs, the value of the objective function is decreasing by less than some minimum amount or the values of the parameters are changing by less than some minimum amount. The maximum number of iterations and the minimum change amounts can typically be specified as options, but reasonable default values are supplied when values are not specified. The main

difference between the different methods for numerical minimization involve the logic used to perturb the parameter guesses.

One thing to note about numerical minimization is that the minimization process might stop before the function has been fully minimized. For example, the maximum number of iterations might be reached before the minimum had been located. Another way to say this is that the solution might not be fully converged. For this reason, it is always a good idea to take the parameters that are found by numerical minimization and restart the minimization process using them as the new guess. If the new minimization generates different parameter values, then the original result probably was not fully converged. Thus, the parameters found by numerical minimization should be used as guesses and the minimization process should be repeated until the values of the parameters stop changing. This is a good indication that the minimization process has converged.

There is one other significant difference between numerical minimization and the analytical approach. It is possible that the objective function, equation (2), may have *multiple local minima and maxima*. For example, Figure 1 shows an objective function that displays a local minimum and a global minimum. The analytical approach can be implemented so it locates the global minimum and returns the corresponding parameter values. In contrast, numerical minimization will converge to only one minimum. That minimum could be the desired global minimum, but it could also be a local minimum. For this reason, it is usually advisable, when using numerical minimization methods, to perform the *converged* minimization a number of times, supplying significantly different initial guesses for the parameters each time. Then, if more than one converged minimum is located you will be able to determine which has the lowest value of the objective function.

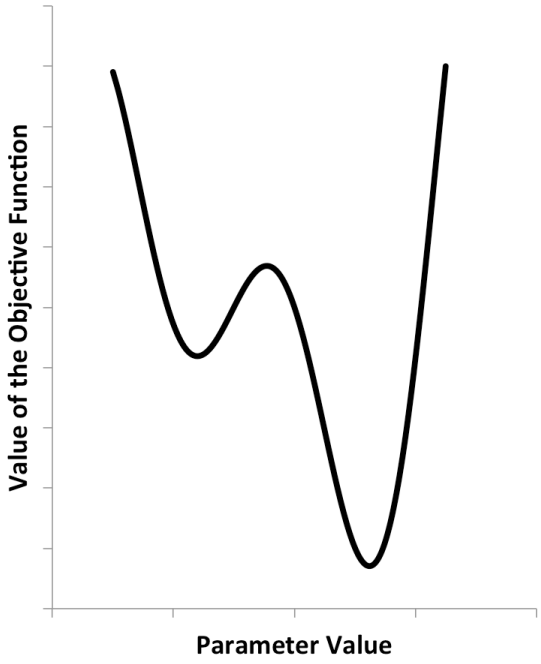

Figure S3.1. Example of an objective function with a local minimum and a global minimum.

In the scenario just described, you will never be absolutely certain that you have identified the global minimum of the objective function. At some point you need to examine the best set of parameters that you have found and decide whether they are physically reasonable. Then you need to examine how well the model fits the data. If the parameter values are physically reasonable and the model fits the data very well, it probably isn't advisable to continue looking for solutions where the objective function is smaller still.

Finally, as noted in Unit 16, equation (2) cannot be used as the objective function when fitting multiple response data, and the proper objective function to use is not always obvious. The template files provided with this supplemental unit for fitting multiple response models to data use the determinant

objective function presented in equation (4), where the errors are defined by equation (5). As noted in Unit 16, this objective function is only appropriate if a complete data set is being used, i. e. if every response variable was measured in every experiment. The statistics associated with calculating uncertainties and correlation coefficients for multiple response models are more involved and have not been incorporated in the template files provided with this supplemental unit for fitting multiple response data.

$$\Phi = \begin{vmatrix} \sum_{\text{all j}} \left(\varepsilon_{1j}\right)^2 & \sum_{\text{all j}} \varepsilon_{1j}\varepsilon_{2j} & \cdots & \sum_{\text{all j}} \varepsilon_{1j}\varepsilon_{nj} \\ \sum_{\text{all j}} \varepsilon_{1j}\varepsilon_{2j} & \sum_{\text{all j}} \left(\varepsilon_{2j}\right)^2 & \cdots & \sum_{\text{all j}} \varepsilon_{2j}\varepsilon_{nj} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{\text{all j}} \varepsilon_{1j}\varepsilon_{nj} & \sum_{\text{all j}} \varepsilon_{2j}\varepsilon_{nj} & \cdots & \sum_{\text{all j}} \left(\varepsilon_{nj}\right)^2 \end{vmatrix} \tag{4}$$

$$\varepsilon_{ij} = \left(y_{i,\text{model}} - y_{i,\text{expt.}}\right)_j \tag{5}$$

MATLAB Template Files

A function named `nlinfit` is available in MATLAB for performing non-linear equation fitting by numerical minimization of the sum of the squares of the errors *for single response data*. In other words, the function `nlinfit` uses equation (2) as the objective function and minimizes it numerically. When you use `nlinfit`, you must pass to it, as arguments, a ***guess*** for the value of each of the parameters in the model, the experimental values of the set variables for all of the data points, the corresponding experimentally measured values of the response variable for all of the data points and the name of a function (technically a function handle). You also must provide that function separately. It must take a set of parameter values and the experimental values of the set variables as arguments, and it must return the corresponding values of the response variable that are predicted by the model for the given input. If the calculation of the response variable involves solving a set of ordinary differential equations or a set of non-linear equations, additional MATLAB functions will need to be called. Those functions are identified and discussed in Supplemental Units S2 and S5.

A second function named `nlparci` is available in MATLAB for computing the confidence intervals for the model parameters that are returned by `nlinfit`. Several other functions, most notably the function named `plot`, are available for creating model plots, parity plots and residuals plots. If you are going to use MATLAB to fit non-linear model equations to experimental data, you are strongly encouraged to read the MATLAB documentation for the functions mentioned; no further details will be presented here.

When a multiple response model is being fit to experimental data, `nlinfit` cannot be used because equation (2) is not the proper objective function. MATLAB does provide a function named `fminsearch` that can be used, though. The difference is that `fminsearch` simply minimizes a function

with respect to variables that appear in that function. The user is responsible for providing the function to be minimized and for identifying the variables to be used in minimizing it. Hence, to use it in the present situation a subroutine that calculates Φ using equation (4) must be provided, with the model parameters as the variables to be used in minimizing it. That is, `fminsearch` will receive as arguments a guess for the parameters and the name of a function that evaluates that objective function. That function can have only one argument, namely a column vector containing a set of model parameters. As a result, some additional coding is required in order to provide the values of the set and experimental response variables to that subroutine.

This course is about kinetics and reaction engineering, not computer programming. Therefore MATLAB template files have been created to automate as much of the fitting process as possible. However, it isn't possible to ***completely*** remove the need for programming. Consequently, each of the template files to be introduced here does require some modification each time it is used. The required modifications are clearly indicated in the template files, they are described in How-To files that accompany this unit and they are illustrated in Examples S4.1 through S4.5.

FitNonlinSR.m can be used to fit an explicit, non-linear model to single response data where arbitrary data point l is of the form $\left(\left(\hat{y}_l, x_{1,l}, x_{2,l}, \cdots, x_{n_s,l}\right); l = 1 \ldots n_{data}\right)$. In particular, it must be possible to solve the model equation explicitly for the response variable, y, as in equation (6), so that f is a function of the set variables, x_i, and parameters, θ_k, but not of the response variable. The function f is assumed to be non-linear. Though the approach described here will work in situations where f is linear, there are better methods available for linear models (see Supplemental Unit S3).

$$y = f\left(x_1, x_2, \cdots, x_{n_s}, \theta_1, \theta_2, \cdots, \theta_{n_p}\right) \tag{6}$$

FitNumAlgSR.m can be used to fit a model of the form given in equation (7) to a set of ***single response*** experimental data points $\left(\left(\hat{y}_l, x_{1,l}, x_{2,l}, \cdots, x_{n_s,l}\right); l = 1 \ldots n_{data}\right)$, by first solving a set of non-linear equations of the form given in equations (8) for the unknowns $\underline{u}$.

$$y = f\left(u_1, u_2, \cdots, u_{n_u}, x_1, x_2, \cdots, x_{n_s}\right) \tag{7}$$

$$\left.\begin{aligned} g_1\left(u_1, u_2, \cdots, u_{n_u}, \theta_1, \theta_2, \cdots, \theta_{n_p}; x_1, x_2, \cdots x_{n_s}\right) &= 0 \\ g_2\left(u_1, u_2, \cdots, u_{n_u}, \theta_1, \theta_2, \cdots, \theta_{n_p}; x_1, x_2, \cdots x_{n_s}\right) &= 0 \\ \vdots & \\ g_{n_u}\left(u_1, u_2, \cdots, u_{n_u}, \theta_1, \theta_2, \cdots, \theta_{n_p}; x_1, x_2, \cdots x_{n_s}\right) &= 0 \end{aligned}\right\} \tag{8}$$

FitNumDifSR.m can be used to fit a model of the form given in equation (7) to a set of ***single response*** experimental data points $\left(\left(\hat{y}_l, x_{1,l}, x_{2,l}, \cdots, x_{n_s,l}\right); l = 1 \ldots n_{data}\right)$, by first solving a set of initial value ordinary differential equations of the form given in equations (9) for the unknowns $\underline{u}$.

$$\left.\begin{aligned}
\frac{du_1}{dv} &= g_1\left(v,u_1,u_2,\cdots,u_{n_u},\theta_1,\theta_2,\cdots,\theta_{n_p};x_1,x_2,\cdots x_{n_s}\right); \quad u_1(0)=u_1^0\left(x_1,x_2,\cdots x_{n_s}\right)\\
\frac{du_2}{dv} &= g_2\left(v,u_1,u_2,\cdots,u_{n_u},\theta_1,\theta_2,\cdots,\theta_{n_p};x_1,x_2,\cdots x_{n_s}\right); \quad u_2(0)=u_2^0\left(x_1,x_2,\cdots x_{n_s}\right)\\
&\vdots\\
\frac{du_{n_u}}{dv} &= g_{n_u}\left(v,u_1,u_2,\cdots,u_{n_u},\theta_1,\theta_2,\cdots,\theta_{n_p};x_1,x_2,\cdots x_{n_s}\right); \quad u_{n_u}(0)=u_{n_u}^0\left(x_1,x_2,\cdots x_{n_s}\right)
\end{aligned}\right\} \quad (9)$$

FitNumAlgMR.m can be used to fit a model of the form given in equations (10) to a set of *multiple response* experimental data points $\left(\left(x_{1,l},x_{2,l},\cdots,x_{n_s,l},\hat{y}_{1,l},\hat{y}_{2,l},\cdots,\hat{y}_{n_r,l}\right);l=1\ldots n_{data}\right)$, by first solving a set of non-linear equations of the form given in equations (8) for the unknowns $\underline{u}$. *FitNumAlgMR.m can only be used if every response variable has been measured for every data point*; a so-called *complete data set*.

$$\begin{aligned}
y_1 &= f_1\left(u_1,u_2,\cdots,u_{n_u},x_1,x_2,\cdots,x_{n_s}\right)\\
&\vdots\\
y_{n_r} &= f_{n_r}\left(u_1,u_2,\cdots,u_{n_u},x_1,x_2,\cdots,x_{n_s}\right)
\end{aligned} \quad (10)$$

Finally, **FitNumDifMR.m** can be used to fit a model of the form given in equations (10) to a set of *multiple response* experimental data points $\left(\left(x_{1,l},x_{2,l},\cdots,x_{n_s,l},\hat{y}_{1,l},\hat{y}_{2,l},\cdots,\hat{y}_{n_r,l}\right);l=1\ldots n_{data}\right)$, by first solving a set of initial value ordinary differential equations of the form given in equations (9) for the unknowns $\underline{u}$. *FitNumDifMR.m can only be used if every response variable has been measured for every data point*; a so-called *complete data set*.