A First Course on Kinetics and Reaction Engineering

Example 16.2

Problem Purpose

This problem illustrates the analysis of multiple response kinetics data using numerical fitting.

Problem Statement

A batch reactor was used to study the dehydration of glucose using sulfuric acid as a catalyst. The initial concentration of glucose, $C_G{}^0$, and the length of time that the reagents were allowed to react, t, were set for each data point and the final concentrations of glucose (C_G), HMF (C_H) and levulinic acid (C_L) were measured. Determine whether the data can be described accurately by the first order reaction network given in equations (1) through (3). If the network model is accurate, find the best values of the three first order rate coefficients. The data are given in Table 1.

$$G \rightarrow H \tag{1}$$

$$H \rightarrow L \tag{2}$$

$$H \rightarrow T \tag{3}$$

Table 1. Data for Example 16.2.

$C_G{}^0$ (M)	t (min)	C_G (M)	C_H (M)	C_L (M)
0.4809	36.0	0.4188	0.0063	0.0298
0.4809	143.0	0.2882	0.0046	0.1200
0.4809	296.0	0.2188	0.0027	0.1682
0.4809	561.0	0.0868	0.0011	0.2578
0.4725	25.0	0.4163	0.0067	0.0279
0.4725	185.0	0.2629	0.0043	0.1334
0.4725	335.5	0.1598	0.0021	0.2056
0.4725	462.0	0.1074	0.0014	0.2326

Problem Analysis

This problem provides data from a batch reactor; there are two set variables, the initial glucose concentration and the reaction time and three response variables, the final concentrations of glucose, HMF and levulinic acid. Three reactions occurred during the experiments; we are asked to test a kinetic model where each of the three reactions is first order. Since the problem involves multiple response data,

least squares fitting cannot be used. However, the data set is complete (every response variable measured for every experiment), so the equation (16.2) from the informational reading, reproduced here as equation (4) can be used as the objective function to be minimized. The errors that appear in equation (4) are defined in equation (5). This will be done numerically as described in Supplemental Unit S4. While the initial value ordinary differential equations that make up the model could be solved analytically, here they will be solved numerically. Finally, it should be noted that the data set here is quite small; this problem will serve to illustrate the data analysis process, but in a real kinetics study, many more data should be used.

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

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

Problem Solution

The three reactions taking place are mathematically independent. Therefore mole balance design equations for three reactants or products will be needed to model the system. In this case (isothermal, isobaric, multiple reactions), the mole balance design equations will take the form shown in equation (6). Choosing glucose (G), HMF (H) and levulinic acid (L) leads to the mole balances given in equations (7) through (9) after the rate expressions are substituted. In this case, the unknowns that are found by solving the differential model equations (the final values of C_G, C_H and C_L) are the response variables, so no additional expressions are needed to compute the model-predicted responses.

$$\frac{dn_i}{dt} = V \sum_{\substack{j = \text{all} \\ \text{reactions}}} r_{i,j} \quad \Rightarrow \quad \frac{d\left(\frac{n_i}{V}\right)}{dt} = \sum_{\substack{j = \text{all} \\ \text{reactions}}} r_{i,j} \quad \Rightarrow \quad \frac{dC_i}{dt} = \sum_{\substack{j = \text{all} \\ \text{reactions}}} r_{i,j} \tag{6}$$

$$\frac{dC_G}{dt} = -k_1 C_G \; ; \quad C_G(0) = C_G^0 \tag{7}$$

$$\frac{dC_H}{dt} = k_1 C_G - \left(k_2 + k_3\right) C_H \; ; \quad C_H(0) = 0 \tag{8}$$

$$\frac{dC_L}{dt} = k_2 C_H \; ; \quad C_L(0) = 0 \tag{9}$$

Thus, the model we want to fit to the experimental data is given by equations (7) through (9), but as noted in the problem analysis, we cannot use a numeric least squares method. Instead, we will have to use a general numerical minimization routine. There are many software packages that provide numeric routines for the minimization of a function. You should choose software that you are familiar with for this purpose. No matter which software you select, you will need to provide the following information/data as input:

- a set of guesses, one for each of the unknown parameters that appears in the model
- a set of experimental data points, each of which consists of the experimentally measured value of the response variables and corresponding values for each of the set variables
- code that calculates the model-predicted value of the response variable for a data point, given the value of each set variable for that data point along with the value of each model parameter
  - here the code must first solve equations (7) through (9) numerically; to do so you will need to provide three additional things as input (for each data point):
    - the initial value of the independent variable, t, and the corresponding initial values of the dependent variables, C_G, C_H and C_L.
    - the final value of the independent variable, t.
    - code that evaluates the derivatives, i. e. using the right hand side of equations (7) through (9)
- code that evaluates the appropriate objective function, in this case equation (5) using the model-predicted responses from the preceding bullet and the experimentally measured responses

Guesses are needed for each of the three rate coefficients, and if the guesses are too far off, the numerical minimization method will fail. There are some strategies that can be used to generate a guess, but I won't go into them here. Instead, I will simply note that the following guesses were used here: k_1 = 0.01, k_2 = 0.1, k_3 = 0.1.

The experimental set variables and response variables are used exactly as they appear in the problem statement; no calculations are needed. For each experiment, the initial time is t = 0. The corresponding initial concentration of glucose is one of the set variables and the initial concentrations of HMF and levulinic acid presumably were zero. The final time for each experiment is the other set variable. No quantities other than the dependent variables and the model parameters appear on the right hand sides of equations (7) through (9), so no additional calculations are needed to implement the code for the third main bullet item. Similarly, writing code to evaluate the right hand side of equation (4) is straightforward and does not require any additional quantities to be computed. Therefore, we have all the information and data we need to perform the fitting.

Upon performing the calculations with whatever software one prefers, one finds the best values of k_1 through k_3, respectively, are 0.0032, 0.1405 and 0.0765 min^{-1}. Since we did not use least squares, values for the correlation coefficient and uncertainties are not computed. We could perform the calculations to do so, but that goes beyond the scope of this course. The results can be used to prepare a parity plot for each response variable and two residuals plots for each response variable. The former parity plots are shown as Figures 1 through 3. They suggest a fairly good fit: the data fall near the

diagonal line. The parity plots are not shown here, but they show the deviations of the data from the model to be random, not systematic. Thus, for a preliminary study, we can accept the kinetic model and the rate coefficients found here. As noted, in a real kinetics study, many more data should be used.

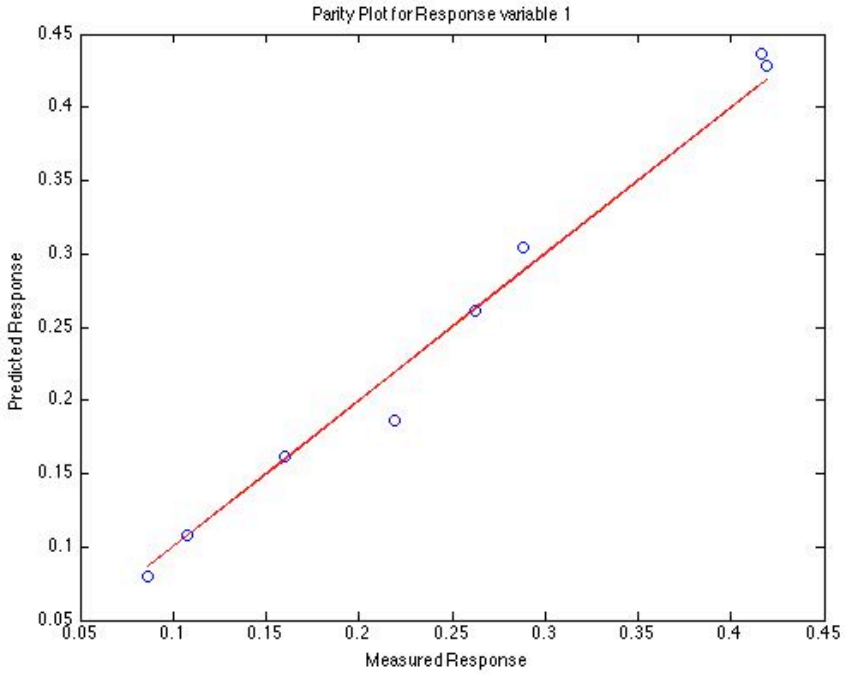

Figure 1. Parity plot for the first response variable, C_G.

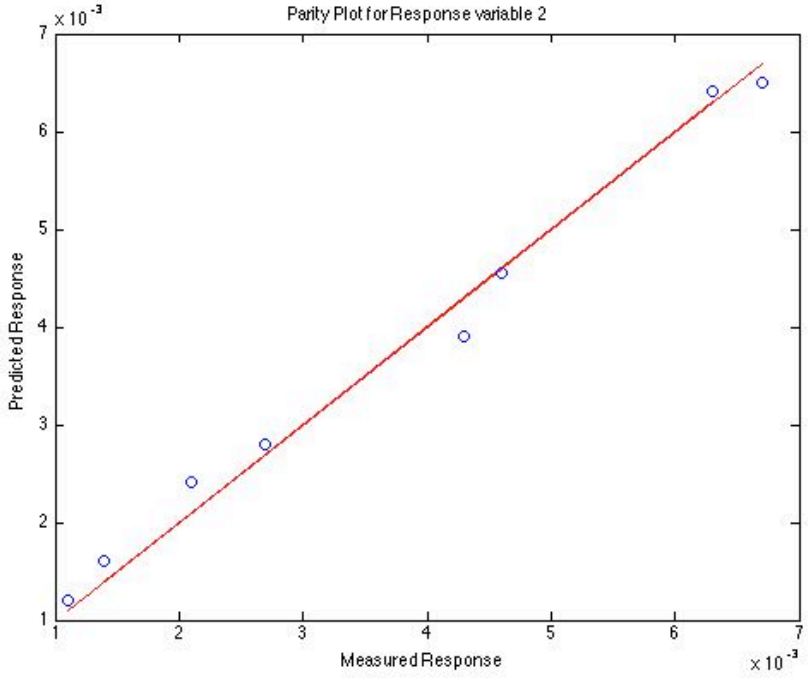

Figure 2. Parity plot for the first response variable, C_H.

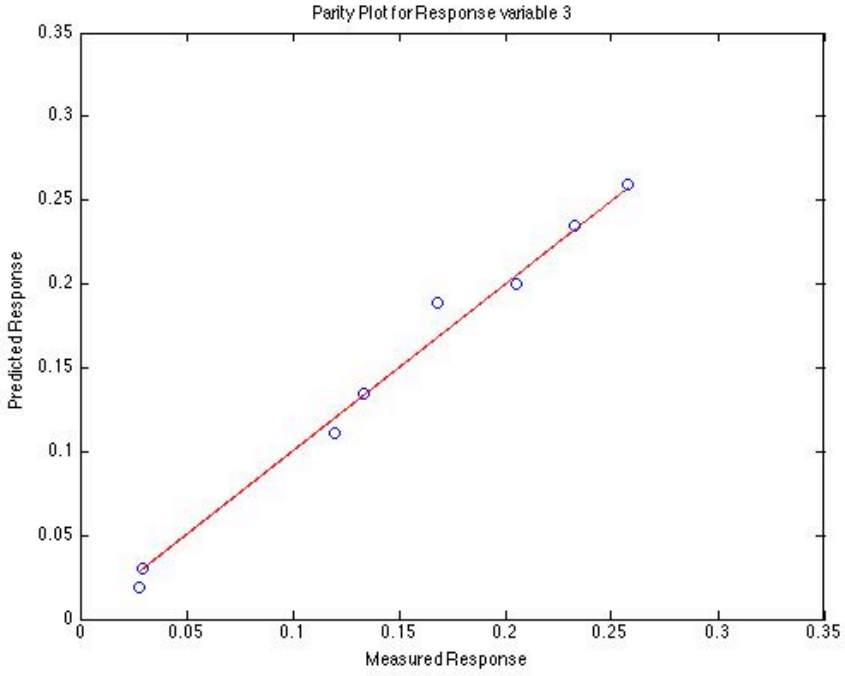

Figure 3. Parity plot for the first response variable, C_L.

Calculation Details Using MATLAB

Supplemental Unit S4 describes how to numerically fit a multiple response model to experimental data when the model is either a set of non-differential equations or a set of initial value ordinary differential equations. It also provides template codes for each of the two situations. In this problem the model is a set of initial value ordinary differential equations (ODEs). The corresponding MATLAB template file is named FitNumDifMR.m. That file needs to be modified in six places in order to apply it to solve a particular problem. Each location where a modification is required is marked by a comment that begins "% EDIT HERE."

I started by making a copy of FitNumDifMR.m and saved it with the name Example_16_2.m. The template file begins with a long set of comments, after which the function FitNumDifMR begins. I replaced these comments with a comment stating the origin and purpose of the modified file. Since the template file contains a MATLAB function, the function name must match the filename, so I changed the function name from FitNumDifMR to Example_16_2. None of these are required modifications. The first required modification involves entering all known constants and quantities from the problem statement. In the present problem the only such data are the experimental set and response variables given in the table in the problem statement. Each column in the table was entered as a column vector with a representative variable name. In doing so, I made sure that the units used are consistent. The next required modification involves combining the vectors containing the set variables into a matrix named `x`. I created such a matrix where the first column contains the initial glucose concentrations and the second column contains the reaction times. The third required modification similarly involves combining the vectors containing the response variables into a matrix named `y_hat`. I created such a matrix where the first column contains the final concentrations of glucose, the second column contains the final concentrations of HMF and the third column contains the final concentrations of levulinic acid. Listing 1 shows the first part of the modified file where all of these non-required and required modifications were performed.

```
% Modified version of the MATLAB template file FitNumDifMR.m from "A First
% Course on Kinetics and Reaction Engineering" used in the solution of
% Example 16.2 of "A First Course on Kinetics and Reaction Engineering."
%
function Example_16_2(p_guess)
    % Data from the problem statement (in consistent units)
    CG0 = [0.4809 0.4809 0.4809 0.4809 0.4725 0.4725 0.4725 0.4725]';
    t = [36.0 143.0 296.0 561.0 25.0 185.0 335.5 462.0]';
    CG = [0.4188 0.2882 0.2188 0.0868 0.4163 0.2629 0.1598 0.1074]';
    CH = [0.0063 0.0046 0.0027 0.0011 0.0067 0.0043 0.0021 0.0014]';
    CL = [0.0298 0.1200 0.1682 0.2578 0.0279 0.1334 0.2056 0.2326]';

    x = [CG0 t];
    y_hat = [CG CH CL];
```

Listing 1. Results of making the first three required modifications to FitNumDifMR.m and also modifying the initial comments and the function name.

The fourth required modification appears in the internal function odeqns which follows immediately after Listing 3. Within this function the independent variables in the set of ODEs is represented as `v` and

the dependent variables are represented as the vector u. Thus, for the present problem v corresponds to the reaction time, t, and u is a vector with three elements: C_G, C_H and C_L. Within this internal function the model parameters are available as a vector p which, in this problem, will contain three elements: k_1, k_2 and k_3. The required modification involves evaluating the right hand sides of the model equations (7) through (9) and saving the results as the rows of the column vector dudv, which, in this problem, represents the set of derivatives of the concentrations with respect to time. For this problem, no additional quantities are needed in order to evaluate the derivatives, and the coding, shown in Listing 2, is straightforward.

```
% Function that evaluates the ODEs
function dudv = odeqns(v,u)
    % Current parameter values are available in column vector p
    % Current set variable values are available in column vector x_set
    dudv = [
        -p(1)*u(1)
        p(1)*u(1) - (p(2) + p(3))*u(2)
        p(2)*u(2)
    ];
end % of internal function odeqns
```

Listing 2. Internal function odeqns after the fourth required modification of the template file, FitNumDifMR.m.

The final two required modifications occur within the internal function named mrmodel. In this internal function, the parameter values are available in the vector p, as before, and the set variables are available in a vector named `x_set` that contains the initial glucose concentration for the current experiment as its first element and the corresponding reaction time as its second element. The first of the two modifications to mrmodel involve providing the initial values of the independent and dependent variables of the ODEs along with the final value of the independent variable for the current data point. As before, u and v represent the concentrations and reaction time, respectively. Each experiment begins with the reaction time equal to zero, so `v0` is set equal to 0. As before, `u0` is a vector (only for initial values) where the first element is the concentration of glucose, the second element is the concentration of HMF and the third column is the concentration of levulinic acid. The initial concentration of glucose for the present data point is available as the first element of the vector `x_set`. Since only glucose was present at the start of each experiment, the initial concentrations of HMF and levulinic acid are always equal to zero. The final reaction time is available as the second element of the vector `x_set`, so `vf` is set equal to that element of `x_set`. Thus, this required modification was straightforward. In many cases, the experimentally measured responses are related to the final values of the dependent variables of the ODEs. The sixth and final required modification is to calculate those responses from the final values of the dependent variables, in the present problem, C_G, C_H and C_L. In this particular problem, however, the final values of the dependent variables ***are*** the response variables. Thus all that is required is to set the

responses equal to these final values. Listing 3 shows the results of making the fifth and sixth required modifications to the template file.

```
    % Function that solves the model equations and calculates the model-
    % predicted responses
    function y = mrmodel(p_current)
        % Declare y
        y = zeros(n_data,n_resp);
        % Make the current parameters available to the model equations
        p = p_current;

        for i = 1:n_data
            % Make the set variables available to the model equations
            for j = 1:n_set
                x_set(j) = x(i,j);
            end

            % Initial and final values
            v0 = 0;
            u0 = [
                x_set(1)
                0
                0
            ];
            vf = x_set(2);

            % Solve the set of ordinary differential model equations
            [v,uu] = ode45(@odeqns,[v0 vf],u0);
% NOTE: For stiff equations replace ode45 with ode15s
            last_value = length(v);
            u = uu(last_value,:);

            % Use the solution to calculate the responses, y(1), y(2), ...
            % by evaluating f1(u(1),...,u(n),x_set(1),...,x_set(n_set)),
            % f2(u(1),...,u(n),x_set(1),...,x_set(n_set)), etc.
            y(i,:) = [
                u(1)
                u(2)
                u(3)
            ];
        end
    end % of internal function mrmodel
```

Listing 3. Internal function mrmodel after the final required modifications (highlighted) of the template file, FitNumDifMR.m.

After all of the modifications were completed, a vector named `guess` was created at the MATLAB command prompt containing the three guesses for the rate coefficients. The code then was executed by typing Example_16_2(guess). The code did find a minimum for the objective function. It is possible that the numerical method may not have fully converged, so the result was entered as the vector named `guess` and the code was re-run. The same result was obtained, suggesting that the minimization routine had converged. It is possible that the resulting values of the parameters correspond to a local minimum, and not the global minimum, of the objective function. To test that, the code should be re-run using very different guesses to see if it converges to the same result or a different one. For the present problem, every guess that was used either led to these same results or else the numerical minimization failed.

Listing 4 shows the command line input and output from running the code; the parity plots shown previously and six residuals plots were also generated.

```
>> guess = [10^-2 10^-1 10^-1];
>> Example_16_2(guess)
Best Values for the Parameters:

pf =

    0.0032    0.1406    0.0765

>> guess = [0.0032    0.1406    0.0765];
>> Example_16_2(guess)
Best Values for the Parameters:

pf =

    0.0032    0.1405    0.0765
```

Listing 4. Command line input and output from running Example_16_2.m.