A First Course on Kinetics and Reaction Engineering

Example S4.5

Problem Purpose

The purpose of this example is to illustrate how to use the MATLAB template file FitNumDifMR.m to perform a regression analysis for multiple response data with a model that consists of a set of initial value ordinary differential equations that must be solved numerically.

Problem Statement

Suppose that 8 experiments were performed wherein the values of two variables, x_1 and x_2 were set and then the values of variables y_1, y_2 and y_3 were measured. The results of the experiments are shown in Table 1. Additionally suppose that equations (1) through (3) constitute a model of the experiments. The goal here is to fit the model to the data and assess the quality of the fit.

Table 1. Data for Example 1.

x_1	x_2	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$
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

$$\frac{dy_1}{dv} = -\theta_1 y_1 \; ; \quad y_1(0) = x_1 \tag{1}$$

$$\frac{dy_2}{dv} = \theta_1 y_1 - (\theta_2 + \theta_3) y_2 \; ; \quad y_2(0) = 0 \tag{2}$$

$$\frac{dy_3}{dv} = \theta_2 y_2 \; ; \quad y_3(0) = 0 \tag{3}$$

Problem Analysis

The data points in this problem have three response variables, and the value of all three responses was measured for every data point. In other words, the data set in this problem is a complete, multiple

response data set. The model consists of a set of initial value ordinary differential equations. These facts indicate that the model can be fit to the data using the MATLAB template file FitNumDifMR.m.

Note: The solution presented here will read very much like the solution to Example S4.4. The reasons are threefold. First, this is the exact same problem. In Example S4.4 the model equations used are the analytical solution of equations (1) through (3) here; that's the only difference. Second, the use of the template file FitNumDifMR.m is very, very similar to the use of FitNumAlgMR.m and FitNumDifSR.m. Third, I'm lazy; it just didn't seem worth the effort to rewrite the solution so it says essentially the same thing as Example S4.4, but using different words.

Problem Solution

Notice, the model equations do not contain unknowns represented by $\underline{u}$. That is because in this problem, the responses are equal to the unknowns in the model equations. You may recall that the template file is set up to solve the model equations for $\underline{u}$ and then use the result to calculate $\underline{y}$. Here it is trivial to recast the equations in that form, as shown in equations (4) through (9). As was done in Example S4.4, the base-10 logarithms of the parameters will be used as the guesses to shrink the range of possible values.

$$\frac{du_1}{dv} = -\theta_1 u_1 \; ; \quad u_1(0) = x_1 \tag{4}$$

$$\frac{du_2}{dv} = \theta_1 u_1 - (\theta_2 + \theta_3) u_2 \; ; \quad u_2(0) = 0 \tag{5}$$

$$\frac{du_3}{dv} = \theta_2 u_2 \; ; \quad u_3(0) = 0 \tag{6}$$

$$y_1 = u_1(x_2) \tag{7}$$

$$y_2 = u_2(x_2) \tag{8}$$

$$y_3 = u_3(x_2) \tag{9}$$

The responses won't always be equal to the unknowns (see Example S4.2). Therefore, from this point forward, equations (4) through (6) will be taken to be the model equations and equations (7) through (9) will be taken to represent the functions $\underline{f}$ referred to in the informational reading.

Following the step-by-step instructions for the use of FitNumDifMR.m, a copy of that file was saved as S4_Example_5.m and is provided with this supplemental unit. The introductory comment and the function name were then changed as in all the other examples in this supplemental unit. There are six required modifications that must be made each time FitNumDifMR.m is used. They are indicated in the code by a comment that begins "% EDIT HERE". The first three are exactly the same as in Example S4.4 and won't be shown here.

The fourth required modification appears within the internal function, `odeqns`. It involves evaluating the three derivatives that constitute the model, namely equations (4) through (6), and returning the results as a column vector, `dudv`. You can see that the only argument received by `odeqns` is the values of u_1, u_2 and u_3, in the form of a column vector named `u`. Looking at the model equations (4) through (6), you can see that they also contain the model parameters, θ_1, θ_2 and θ_3. The parameters are available in the column vector `p`. Recalling that the guesses I'll provide for the parameters are the base 10 logs of the actual parameters, θ_1 is equal to `10^p(1)`, θ_2 is `10^p(2)`, and θ_3 is `10^p(3)`. With this knowledge, the modification of internal function odeqns is straightforward, as shown In Listing 1.

```
  % 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 = [
        -10^p(1)*u(1);
        10^p(1)*u(1) - (10^p(2)+10^p(3))*u(2);
        10^p(2)*u(2);
    ];
end % of internal function odeqns
```

Listing 1. Internal function odeqns after modification.

The other two required modifications takes place within the internal function, `mrmodel`. They occur within a loop through each of the experimental data points, i. The first is where initial values for the model unknowns, u_1, u_2 and u_3, for data point i need to be entered along with the initial and final values of the independent variable, v. It can be seen in equations (4) through (6) that the initial value of v is zero and the initial values of u_1, u_2 and u_3 are x_1, 0 and 0, respectively. Equations (7) through (9) show that the final value of v is equal to x_2. The values of the set variables for data point i are available at this point in the column vector named `x_set`; hence x_2 is available as `x_set(2)`. The corresponding modification of the template file is shown in Listing 2. Just a few lines after that modification, still within the loop through the data points within the internal function, `mrmodel`, the final required modification must be made. This modification is located just after the model equations have been solved for the values of $\underline{u}$ at the specified final v, and the comment indicates that the values of the response variables need to be calculated. In this problem, the response variables are calculated using equations (7) through (9), so this modification simply involves adding the code to calculate y_1, y_2 and y_3 using those equations, as also shown in Listing 2.

```
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 = 0.0;
        u0 = [
            x_set(1);
            0.0;
            0.0;
        ];
        vf = x_set(2);

        % Solve the set of ordinary differential model equations
        [v,uu] = ode45(@odeqns,[v0 vf],u0);
        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 2. Internal function mrmodel after modification.

That completes the required modification of S4_Example_5.m, and it can be saved in the current MATLAB working directory or in a directory that is part of the MATLAB search path. When S4_Example_5 is executed, it must be passed a column vector that contains guesses for each of the base-ten logs of the three parameters in the model, as discussed previously. From this point on, the process is exactly the same as in Example S4.4. Guesses were provided, S4_Example_5 was executed and the process of copying the results (`pf`) into `p_guess` and re-executing S4_Example_5.m was repeated until the returned parameter values were the same as the values guessed, indicating that the minimization had converged. Listing 3 shows the final parameter copy and the text output from running S4_Example_5.m. In addition to the text output, nine figures were generated, including Figures 1 through 4 included here. By

comparison, you can see that the results are essentially the same as those presented in Example S4.4, which is expected since the models are equivalent and the data are identical.

```
>> p_guess = [
   -2.4943
   -0.8521
   -1.1163
];
>> S4_Example_5(p_guess)
Best Values for the Parameters:

pf =
   -2.4943
   -0.8521
   -1.1163
```

Listing 3. Output from the final re-execution of S5_Example_1.m.

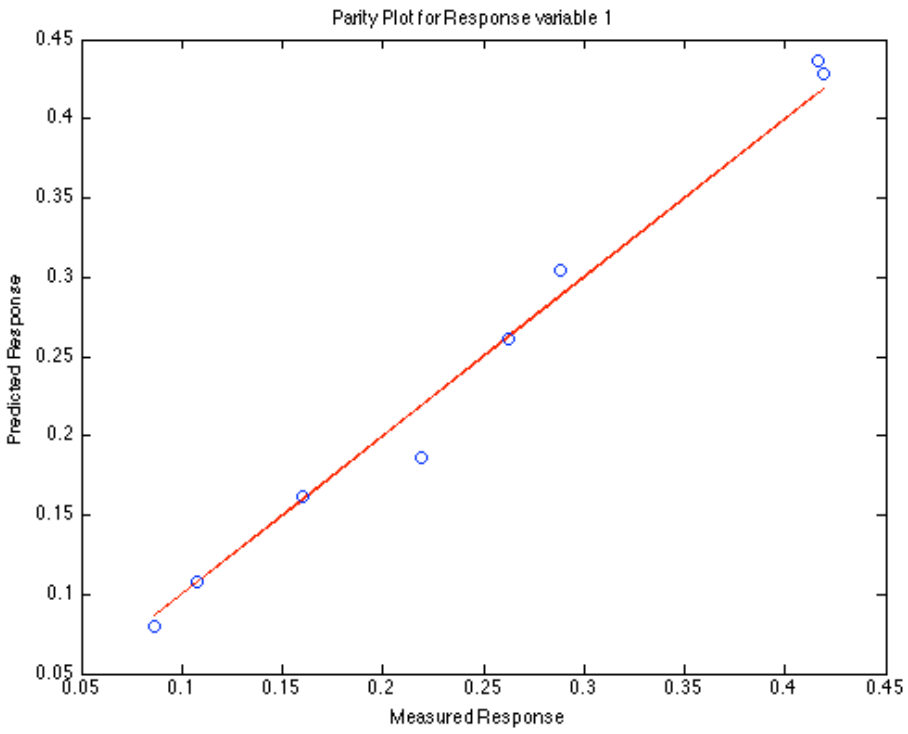

Figure 1. Parity plot showing the predicted versus measured values of response variable y_1.

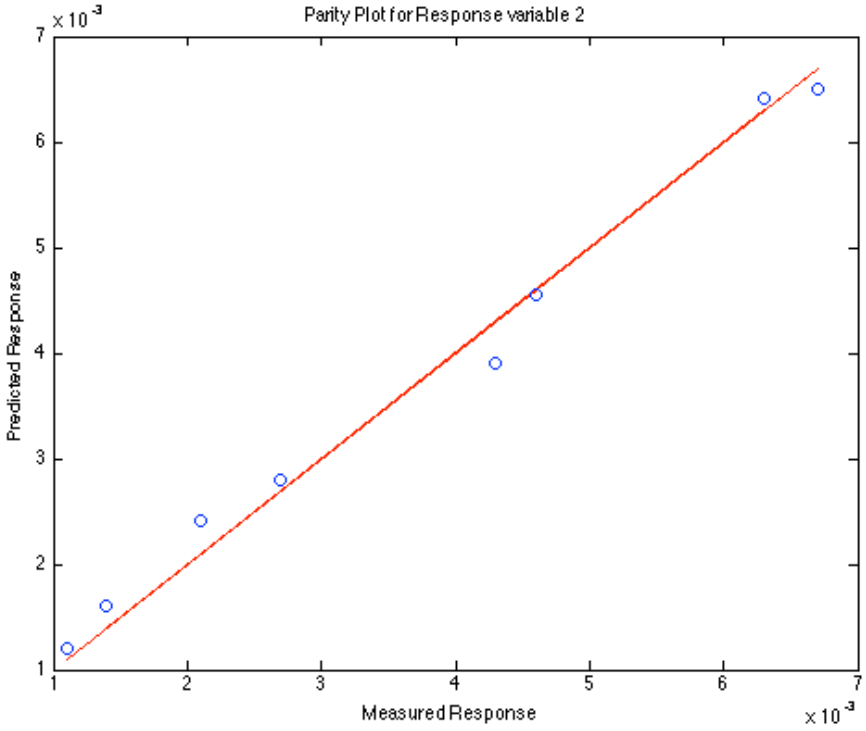

Figure 2. Parity plot showing the predicted versus measured values of response variable y_2.

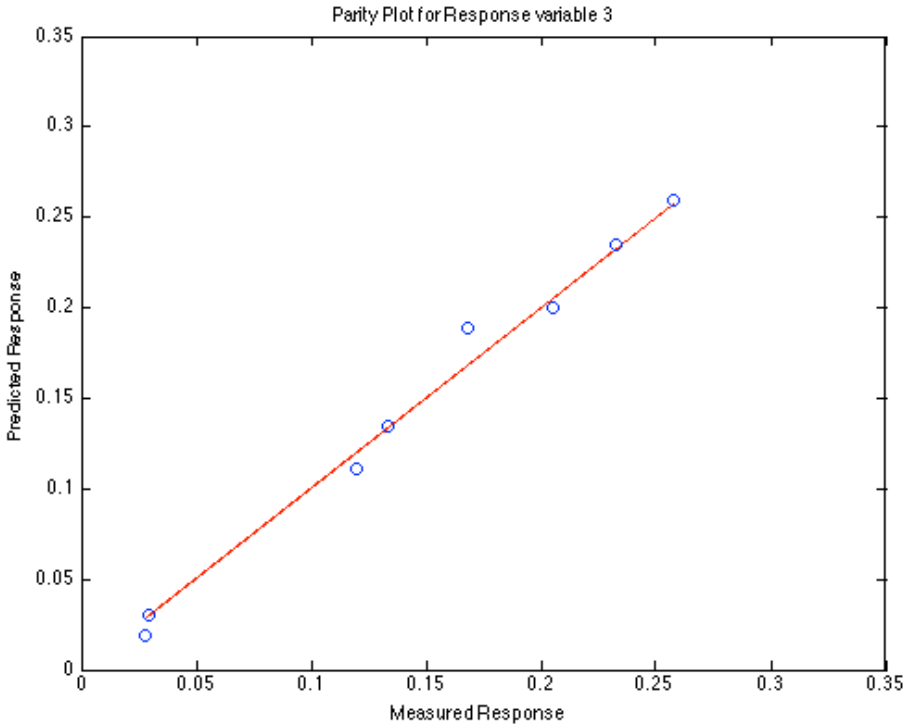

Figure 3. Parity plot showing the predicted versus measured values of response variable y_3.

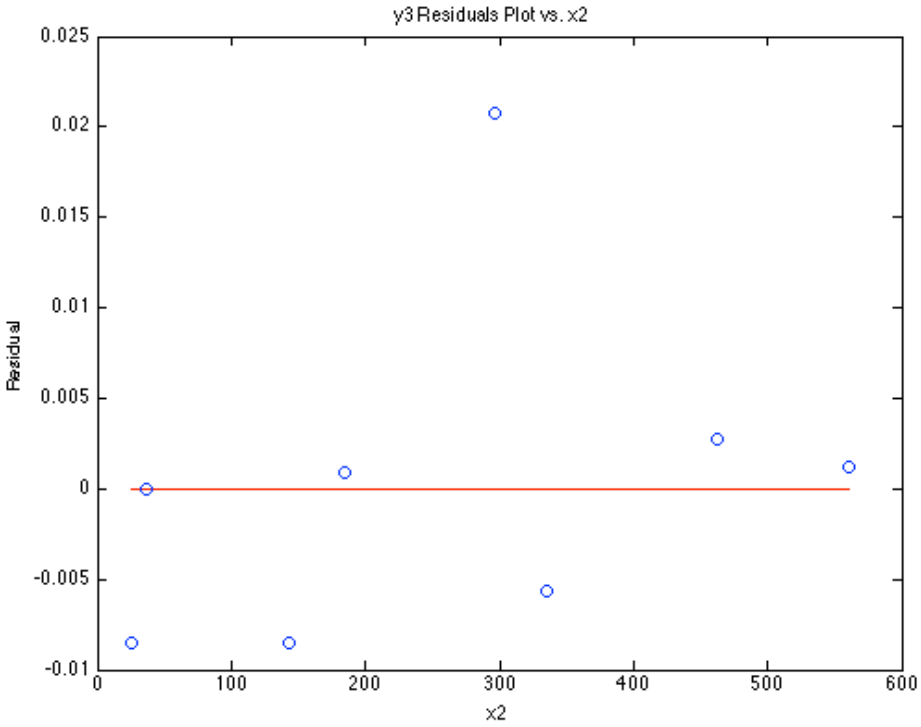

Figure 4. Residuals plot for response variable y_3 versus set variable x_2.