A First Course on Kinetics and Reaction Engineering

Example 16.1

Problem Purpose

This problem shows how to use numerical least squares fitting to test a single parameter rate expression using kinetics data from a PFR where the integrated form of the mole balance design equation cannot be properly linearized for the use of linear least squares fitting.

Problem Statement

A rate expression is needed for the decomposition of phosgene, equation (1). A series of experiments were performed using a plug flow reactor with a diameter of 0.25 cm and a length of 10 cm. The feed in all cases was pure COS, and the reactor operated isothermally at 1500K and 1 atm. The feed rate was varied and the conversion of COS was recorded. The results are presented in the table below. Find a rate expression that is consistent with these data.

$$COS \rightarrow CO + S \qquad (1)$$

Inlet Flow Rate $(cm^3\ min^{-1})$	***fractional COS conversion***
10	0.99
20	0.93
30	0.86
40	0.78
60	0.66
75	0.60
100	0.51

Problem Analysis

This is the same problem as Example 15.4. That solution showed that the mole balance design equation could be analytically integrated, but it could not be properly linearized for linear least squares fitting. Here, the problem will be solved using numerical least squares. A further examination of the solution presented in Example 15.4 shows that the integrated design equation cannot be solved to obtain an explicit expression for the response variable (the conversion of COS). Therefore, in the solution presented here, the mole balance design equation will also be solved numerically.

Problem Solution

The data provided for this problem are from a PFR that was operated isothermally, isobarically and at steady state. The only reaction taking place was reaction (1), so the only equation needed to model the reactor is a mole balance on one of the reactants or products. The rate expression to be tested is given in equation (2), and when it is substituted into a mole balance design equation for COS, equation (3) results.

$$r_1 = k_1 P_{COS} \tag{2}$$

$$\frac{d\dot{n}_{COS}}{dz} = \frac{-\pi D^2}{4} k_1 P_{COS} \tag{3}$$

In the experiments described here, there is only one set variable, the inlet volumetric flow rate. The response variable is the fractional conversion of COS. The model-predicted value of the response variable for any one data point is calculated using equation (4) where $\dot{n}_{COS}$ is found by solving equation (3) for that data point.

$$f_{COS} = \frac{\dot{n}_{COS}^0 - \dot{n}_{COS}}{\dot{n}_{COS}^0} \tag{4}$$

Thus, the model we want to fit to the experimental data is a combination of equations (3) and (4), and since we can't use linear least squares, we will use numerical least squares. Supplemental Unit S4 describes how numerical least squares is performed. There are many software packages that will perform least squares fitting numerically. 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 (here just the rate coefficient, k) that appears in the model
- a set of experimental data points, each of which consists of the experimentally measured value of the response variable (here the COS conversion) and corresponding values for each of the set variables (here there is only one, the inlet volumetric flow rate)
- 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 equation (3) numerically; to do so you will need to provide three additional things as input (for each data point):
    - the initial value of z (here 0) and the corresponding initial value of $\dot{n}_{COS}$ (here $\dot{n}_{COS}^0$)
    - the final value of z (here L)
    - code that evaluates the derivative, i. e. using the right hand side of equation (3)
  - the code will then use equation (4) to calculate the model-predicted response

Providing a guess to a numerical least squares routine can be challenging. Typically one has no idea what value to guess, and the possibilities span many orders of magnitude. If the guess is too far off, then the numerical method may fail because it can't make new guesses that are any better (see Supplemental Unit S4). Here, however, we have the benefit of knowing the answer from the solution of Example 15.4 ($1.5 \pm 0.06 \times 10^{-3}$ mol cm^{-3} min^{-1} atm^{-1}). In this problem, I'll use 0.001 as my guess. That isn't the exact answer, so we'll be able to see that the numerical least squares fitting does work and generates the same answer, but it is close enough that the numerical method won't fail due to a poor guess. That takes care of the first bullet item above.

The second item we will need to provide is the data set. When we use numerical least squares fitting, this is trivial because we simply use the set and response variables directly. That is, since we aren't linearizing the model equation, we don't need to define new x and y data. The response variable is the fractional conversion of COS and the experimental set variable is the inlet volumetric flow rate. Their values are given in the table in the problem statement, and we can use them directly.

The next items we need to provide are used in the numerical solution of equation (3). Just as when we analytically integrated that equation, we need to provide the lower limits of integration. In the context of numerical solution of an ordinary differential equation, these are referred to as the initial conditions. As noted in the bullets, the initial condition for every experiment was that the inlet molar flow rate of COS was equal to a known value, $\dot{n}_{COS}^0$, at the inlet to the PFR (that is, at $z = 0$). This is calculated using the ideal gas law as shown in equation (5) since the feed is pure COS. We want to solve equation (3) to find the molar flow rate of COS at the outlet from the PFR, so the final condition for the integration is that $z = L$.

$$\dot{n}_{COS}^0 = \frac{P\dot{V}^0}{RT} \tag{5}$$

The last two items we need to provide are both code. In one case the code must evaluate the right hand side of equation (3). When we do so, we will be provided with a value (the current best guess) for k, so we can treat it as a known quantity; π and the reactor inside diameter, D, are also known constants. Therefore, the only thing we need to calculate is P_{COS}. Doing that is also relatively easy because we also will be provided with the value of $\dot{n}_{COS}$. Thus, P_{COS} can be calculated using the ideal gas law as in equation (6). In order to do so, the total molar flow rate must first be calculated. That, too, is straightforward. Since $\dot{n}_{COS}^0$ and $\dot{n}_{COS}$ are known, the extent of reaction can be calculated using equation (7), and once that is known, the total molar flow rate can be computed using equation (8), as described in Unit 1.

$$P_{COS} = \frac{\dot{n}_{COS}}{\dot{n}_{tot}} P \tag{6}$$

$$\dot{\xi}_1 = \dot{n}_{COS}^0 - \dot{n}_{COS} \tag{7}$$

$$\dot{n}_{tot} = \dot{n}_{COS}^0 + \dot{\xi}_1 \tag{8}$$

The other code we need to provide must calculate the fractional conversion of COS after equation (3) has been solved. At that point, both the inlet and outlet molar flow rates of COS will be known, so equation (4) can be used to calculate the COS fractional conversion.

Upon performing the calculations with whatever software one prefers, one finds the best value of k_1 to equal 1.5 ± 0.02 x 10^{-3} mol cm^{-3} min^{-1} atm^{-1} (95% confidence limits based upon the data given in the problem statement). As expected, this is the same as the value found in Example 15.4. The uncertainty is

smaller, but in Example 15.4 the standard deviation was taken as the uncertainty whereas here it is the 95% confidence limits. Figure 1 shows a model plot. Notice that the model plot is no longer linear since we used *numerical* least squares fitting not *linear* least squares. The correlation coefficient is 0.96, indicating a reasonably good fit. The scatter of the data from the line in Figure 1 is small and random. Therefore, the rate expression can be accepted as sufficiently accurate.

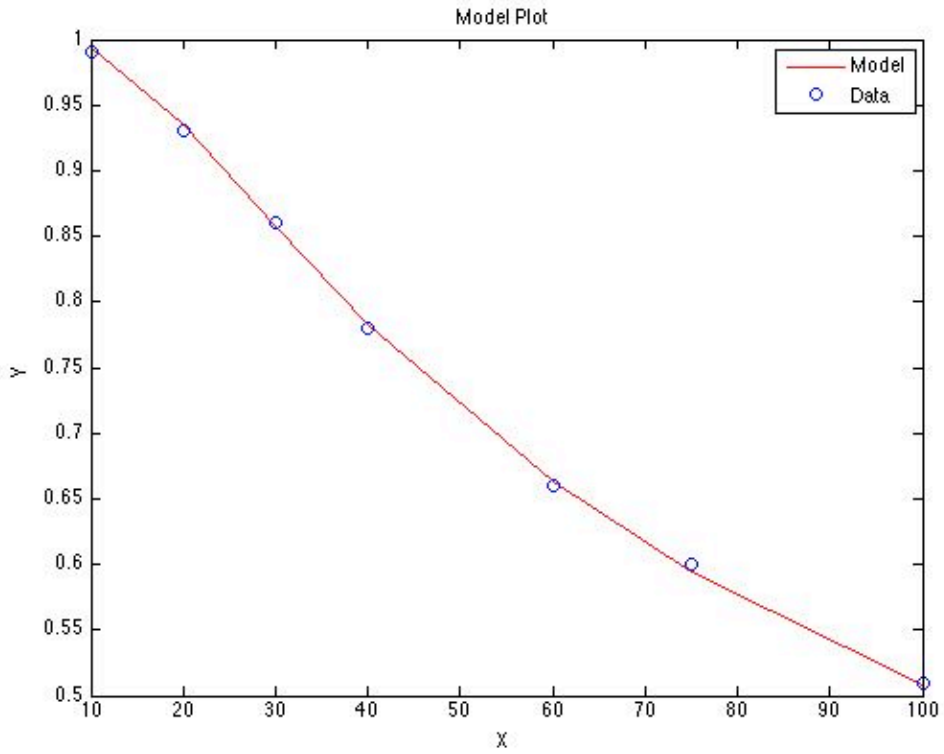

Figure 1. Model plot showing the experimental data as points and the model's predictions as a line.

Calculation Details Using MATLAB

Supplemental Unit S4 describes numerical least squares fitting; you should read it if you haven't already done so. It also provides three MATLAB template files that can be used to perform numerical least squares fitting when (a) an explicit expression is available for the calculation of the model-predicted response, (b) when a set of non-differential model equations must be solved numerically and the result used to calculate the model-predicted response or (c) a set of initial value ordinary differential model equations must be solved numerically and the result used to calculate the model-predicted response. Here we need to perform the latter task, so we will use the corresponding template file, FitNumDifSR.m. Before that file can be used it must be modified in six places. Each location where a modification is

required is marked by a comment that begins "% EDIT HERE." I will make a few additional modifications; hopefully they will make the code easier to follow.

The notation used in the template file, FitNumDifSR.m, is the same as that used in Supplemental Unit S4. It is a general notation, and before discussing the modifications to the template file, it will be worthwhile to relate that general notation to the notation used here in the preceding problem solution. The general notation uses $\hat{y}$ to represent the response variable; it is a column vector with one row for each experimental data point; in the present problem, $\hat{y}$ is equivalent to f_{COS}. In the general notation, there can be n_{set} set variables; each one is represented as x_i (i equals 1 through n_{set}), and each one is a column vector with one row for each experimental data point. In the present problem there is only one set variable, and x_1 (or simply x) is equivalent to $\dot{V}^0$. The general notation uses p to represent the model parameters; it is a column vector with one row per model parameter. In this problem, p has only one row and p_1 is equivalent to k. In the general notation, there is a set of initial value ordinary differential model equations of the form given in equation set (9). The dependent variables in the equation set are denoted as u_i and the independent variable as v. In the present problem there is only one equation in the set, namely equation (3); u_1 (or simply u) is then equivalent to $\dot{n}_{COS}$, v is equivalent to z and the function g_1 (or simply g) is equal to the right hand side of equation (3). Finally, in the general notation, the model-predicted response is represented by y, which is a column vector with one row per experimental data point. Each element of y can be calculated explicitly by evaluation of a function, f, that depends on the set variables, x_i, and the solutions to the differential model equations, u_i, as given in equation (10). In the present problem y is equivalent to the value of f_{COS} calculated using equation (4), and thus the function, f, is equivalent to the right hand side of equation (4).

$$\begin{aligned}
\frac{du_1}{dv} &= g_1\left(v,u_1,\cdots,u_n;p_1,\cdots,p_{n_{par}}\right); \quad u_1\left(v_0\right)=u_1^0 \\
\frac{du_2}{dv} &= g_2\left(v,u_1,\cdots,u_n;p_1,\cdots,p_{n_{par}}\right); \quad u_2\left(v_0\right)=u_2^0 \\
&\vdots \\
\frac{du_n}{dv} &= g_n\left(v,u_1,\cdots,u_n;p_1,\cdots,p_{n_{par}}\right); \quad u_n\left(v_0\right)=u_n^0
\end{aligned} \tag{9}$$

$$y = f\left(x_1,\cdots,x_{n_{set}},u_1,\cdots,u_n\right) \tag{10}$$

With that as background, we can now consider the modification of the template file, FitNumDifSR.m, for use in solving this problem. To begin, I made a copy of FitNumDifSR.m and saved it with the name Example_16_1.m. The template file begins with a long set of comments, after which the function FitNumDifSR 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 FitNumDifSR to Example_16_1. None of these are required modifications. The result of all these changes can be seen in Listing 1.

```
% Modified version of the MATLAB template file FitNumDifSR.m from "A First
% Course on Kinetics and Reaction Engineering" used in the solution of
% Example 16.1 of "A First Course on Kinetics and Reaction Engineering."
%
function Example_16_1(p_guess)
```

Listing 1. Result of the first (non-required) modifications of the template file, FitNumDifSR.m.

The first required modification involves entering all known constants and quantities from the problem statement. In doing so, I made sure that the units used are consistent. This modification follows immediately after the code shown in Listing 1, replacing the first "% EDIT HERE" comment. The resulting code is shown in Listing 2. The second required modification involves combining the vectors containing the set variables into a matrix named `x`. In the present problem, there is only one set variable, so the matrix consists of a single column vector containing the inlet volumetric flow rate for each of the experiments. The third required modification involves entering the experimental responses as a column vector named `y_hat`. The second and third required modifications follow directly after the code shown in Listing 2, replacing the next two "% EDIT HERE" comments. The resulting code is shown in Listing 3.

```
    % Known quantities and constants (in consistent units)
    Diam=0.25; % cm
    L=10.; % cm
    T=1500; % K
    P=1.0; % atm
    Rgas=82.06; % (atm-cm^3)/(mol-K)
```

Listing 2. Result of the first required modification of the template file, FitNumDifSR.m.

```
    x = [10 20 30 40 60 75 100]'; % inlet volume flow (cm^3/min)
    y_hat = [0.99 0.93 0.86 0.78 0.66 0.60 0.51]'; % COS conversion
```

Listing 3. Results of the second and third required modifications of the template file, FitNumDifSR.m.

The fourth required modification appears in the internal function odeqns which follows immediately after Listing 3. The original version of odeqns is shown in Listing 4. Notice that this function receives the independent (`v`) and dependent (`u`) variables as arguments and it return the value of the derivative, `dudv` at those value of `u` and `v`. Though not a required modification, I changed each occurrence of `v` to `z`, each occurrence of `u` to `nCOS` and each occurrence of `dudv` to `dnCOSdz`. Notice also that within this internal function, the set variables are available in a column vector named `x_set`, hence the inlet volumetric flow rate is available as `x_set(1)`. Similarly, the model parameters are available in a column vector named `p`, hence the rate coefficient is available as `p(1)`. I added a statement defining `k` and another defining `VFR0` (the inlet volumetric flow rate). Again, these are not required modifications. The fourth required modification involves adding the code to evaluate the derivative using the right hand of equation (3). In general, there could be a set of equations, and hence a set of derivatives. Each evaluated derivative

would then represent a row in the column vector dudv (or, here, dnCOSdz). In the present problem there is just one row. However, the code in that row requires PCOS. Therefore, in the rows preceding the evaluation of the derivative, I entered code to calculate the inlet COS molar flow rate according to equation (5), the extent of reaction according to equation (7), the total molar flow rate according to equation (8) and the partial pressure of COS according to equation (6). These modifications are required. The code resulting from all of these modifications is shown in Listing 5; it replaced the code in Listing 4.

```
      % 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
% EDIT HERE (Required modification 4 of 6):
        dudv = [
             % Evaluate du1/dv = g1(v,u(1),...,u(n);p(1),...,p(n_par)) here
             % Evaluate du2/dv = g2(v,u(1),...,u(n);p(1),...,p(n_par)) here
             % and so on through gn, one per line
        ];
    end % of internal function odeqns
```

Listing 4. Internal function odeqns as it appears in the template file, FitNumDifSR.m.

```
      % Function that evaluates the ODEs
    function dnCOSdz = odeqns(z,nCOS)
        % Current parameter values are available in column vector p
        k = p(1);
        % Current set variable values are available in column vector x_set
        VFR0 = x_set(1);
        % Calculate quantities needed to evaluate the derivative
        n0COS = (P/Rgas/T)*VFR0;
        extent = n0COS - nCOS;
        ntot = n0COS + extent;
        PCOS = P*nCOS/ntot;
        % Evaluate the derivative
        dnCOSdz = [
             -pi()*Diam^2/4*k*PCOS
        ];
    end % of internal function odeqns
```

Listing 5. Internal function odeqns after the fourth required (red) and other non-required (blue) modifications of the template file, FitNumDifSR.m.

The remaining modifications occur within the internal function nlmodel. As before, the values of the parameters are available in this internal function in a vector named p and the set variables in a vector named x_set, and as before, I added a statement defining VFR0 so the notation is more consistent with the solution set up, above. (I didn't enter a statement defining k because k is not used anywhere in this internal function.) For the same reason, I also again changed each occurrence of v (or v0 or vf) to z (or z0 or zf), each occurrence of u to nCOS and each occurrence of u0 to n0COS. (Note, nlmodel also contains a variable named uu; I did not rename this quantity.) None of these modifications were required. The fifth *required* modification involved entering the initial and final values, for z0, n0COS and zf (after

renaming as just described). In general, there might be several differential equations, and consequently a column vector of initial values of the dependent variable. Since the present problem only involves one equation in the set of differential equations, I entered the initial value for n0COS as a scalar, not a one-element vector, though either would work. The sixth and final required modification involves adding code to calculate the response using equation (4). Listing 6 shows internal function nlmodel after all of the modifications were made; the required modifications are highlighted in red while the other, non-required modifications are highlighted in blue.

```
    function y = nlmodel(p_current,x)
        % Declare y
        y = zeros(n_data,1);
        % 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
            VFR0 = x_set(1);

            % Initial and final values
            z0 = 0;
            n0COS = (P/Rgas/T)*VFR0;
            zf = L;

            % Solve the set of ordinary differential model equations
            [z,uu] = ode45(@odeqns,[z0 zf],n0COS);
% NOTE: For stiff equations replace ode45 with ode15s
            last_value = length(z);
            nCOS = uu(last_value,:);

            % calculate the response
            y(i) = (n0COS-nCOS)/n0COS;
        end
    end % of internal function nlmodel
```

Listing 6. Internal function nlmodel after the fifth and sixth required (red) and other non-required (blue) modifications of the template file, FitNumDifSR.m.

After all of the modifications were completed, the code was executed by typing Example_16_1(0.001) at the MATLAB command prompt. The value in parentheses at the end of the function name is the guess for the parameter value discussed earlier. Upon entering that command, the output shown in Listing 7 was produced along with the model plot shown previously as Figure 1.

```
>> Example_16_1(0.001)

r_squared =

    0.9994

Best Values for the Parameters:

pf =

    0.0015

95% Confidence Intervals for the Parameters:

pf_u =

   2.3473e-05
```

Listing 7. Output from Example_16_1.m showing correlation coefficient, parameter value and its uncertainty.