A First Course on Kinetics and Reaction Engineering

Example 31.1

Problem Purpose

This problem will help you determine whether you have mastered the learning objectives for this unit. It also illustrates the approach described in the informational reading for simultaneously solving the PFR design equations and the mixing point design equations.

Problem Statement

A certain type of cell grows at a rate given by the equation (1) below where [S] is the substrate concentration and [X] is the cell concentration. Each gram of cell mass produced consumes 2.2 g of substrate. A tubular, plug flow fermentor with recycle will be used to grow these cells. Its operation is essentially isothermal, and equation (1) includes the rate coefficients for the temperature at which the reactor will operate. The feed to the process will contain 0.04 g S cm^{-3} and will flow at 60 cm^3 min^{-1}; no cell mass will be present in the feed. If the reactor volume is 4 L and the recycle ratio is equal to 0.2, what will be the outlet cell mass concentration?

$$r = \frac{\left(0.034\ \text{min}^{-1}\right)[S][X]}{\left(0.001\ \text{g cm}^{-1}\right)+[S]} \tag{1}$$

Problem Analysis

Since the rate expression is known and we are asked questions about reactor performance, this is a reaction engineering problem. It involves a PFR with a recycle stream and a cell growth reaction. Since the rate and concentrations are in mass units, mass balances on the substrate and cell mass will be used to model the reactor. Making use of the effective stoichiometry given in the problem statement, the reaction can be represented as in equation (2) where the stoichiometry is mass based. The problem states that the reactor will be isothermal, so an energy balance will not be included in the design equations. In addition, the problem does not provide sufficient detail to account for pressure drop, so a momentum balance will not be used, either. Mole balances on the mixing point likely will need to be solved simultaneously with the PFR design equations.

$$2.2\ \text{S} \rightarrow \text{X} \tag{2}$$

Problem Solution

The system can be represented schematically shown in Figure 1. Reading through the problem statement, one finds that the following quantities are specified: $C_{S,a}$ = 0.04 g cm^{-3}, $\dot{V}_a$ = 60 cm^3 min^{-1}, $C_{X,a}$ = 0 g cm^{-3}, V = 4 L and R_R = 0.2. From these quantities, the feed mass flow rates can be calculated using equations (3) and (4).

$$\dot{m}_{S,a} = \dot{V}_a C_{S,a} \tag{3}$$

$$\dot{m}_{X,a} = \dot{V}_a C_{X,a} \tag{4}$$

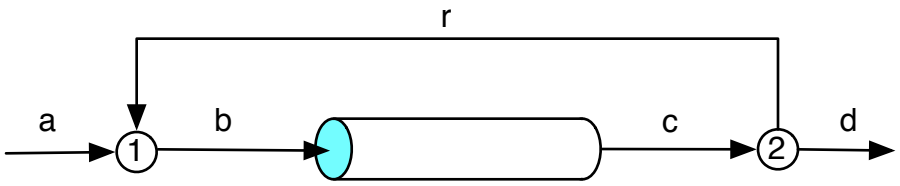

Figure 1. Schematic representation of the recycle PFR system of Example 31.1.

Assuming the fluid density to be constant, the volumetric flow rates can be found using equations (5) through (7).

$$\dot{V}_d = \dot{V}_a \tag{5}$$

$$\dot{V}_r = R_R \dot{V}_d \tag{6}$$

$$\dot{V}_b = \dot{V}_a + \dot{V}_r \tag{7}$$

$$\dot{V}_c = \dot{V}_b \tag{8}$$

Using the cumulative reactor volume as the independent variable, and using the effective mass stoichiometry, the material balance design equations for the PFR take the form shown in equations (9) and (10). As noted in the problem analysis, these two equations are the only design equations for the PFR. In order to solve these equations numerically, no matter what software one uses, it will be necessary to provide (a) initial values, (b) a final value and (c) code that is given values of the independent and dependent variables and uses them to evaluate functions f_1 and f_2 in equations (9) and (10).

$$\frac{d\dot{m}_S}{dV} = f_1\left(V, \dot{m}_S, \dot{m}_X\right) = -2.2r \tag{9}$$

$$\frac{d\dot{m}_X}{dV} = f_2\left(V, \dot{m}_S, \dot{m}_X\right) = r \tag{10}$$

At the inlet to the reactor, corresponding to the initial values, the cumulative reactor volume is equal to zero, but the mass flow rates of S and X are not known. The final value in this case is that the cumulative reactor volume will equal the known PFR volume. The code that must be provided to evaluate functions f_1 and f_2 will be given values for the independent and dependent variables ($\dot{V}$, $\dot{m}_S$ and $\dot{m}_X$). Knowing those values, the mass concentrations of S and X can be computed using equations (11) and

(12). It is important to note that the volumetric flow rate in these equations is the volumetric flow rate in the PFR; it is not the volumetric flow rate entering the process. Once that is done, the rate can be computed using equation (1), and knowing the rate, the functions f_1 and f_2 can be evaluated.

$$[S] = \frac{\dot{m}_S}{\dot{V}_b} \tag{11}$$

$$[X] = \frac{\dot{m}_X}{\dot{V}_b} \tag{12}$$

It is not possible to solve the PFR design equations at this point because the initial values are not known. Instead, the design equations for the mixing point must be formulated. As shown in the informational reading using molar flow rates, equation (31.5), the material balances for the mixing point take the form of equations (13) and (14).

$$\dot{m}_{S,a} + \frac{R_R \dot{m}_{S,c}}{1+R_R} - \dot{m}_{S,b} = f_3\left(\dot{m}_{S,b}, \dot{m}_{X,b}\right) = 0 \tag{13}$$

$$\dot{m}_{X,a} + \frac{R_R \dot{m}_{X,c}}{1+R_R} - \dot{m}_{X,b} = f_4\left(\dot{m}_{S,b}, \dot{m}_{X,b}\right) = 0 \tag{14}$$

The two mixing point design equations, (13) and (14), contain four unknown quantities ($\dot{m}_{S,b}$, $\dot{m}_{X,b}$, $\dot{m}_{S,c}$ and $\dot{m}_{X,c}$). Since there are only two equations, they can only be solved to obtain the value of two of the unknowns. As equations (13) and (14) indicate, the key to solving these equations numerically is to select $\dot{m}_{S,b}$ and $\dot{m}_{X,b}$ as the two unknowns. Then, in order to solve the equations numerically, one must provide guesses for the two unknowns and code that will be given values for $\dot{m}_{S,b}$ and $\dot{m}_{X,b}$ and uses them to evaluate functions f_3 and f_4. Since the code will be given values for $\dot{m}_{S,b}$ and $\dot{m}_{X,b}$, it can use those values to solve the PFR design equations. Doing so will yield values for $\dot{m}_{S,c}$ and $\dot{m}_{X,c}$, at which point functions f_3 and f_4 can be evaluated. Therefore, it is possible to solve the mixing point design equations.

Solving the mixing point design equations as just described yields the values of . Those values can be used to solve the PFR design equations one last time. This will yield the cell mass flow rate in stream c. A mass balance at the stream split, along with the definition of the recycle ratio, then allows calculation of the cell mass flow rate leaving the process, equation (15). The cell mass concentration can then be found to equal 0.0163 g cm^{-3} using equation (16).

$$\dot{m}_{X,c} = \dot{m}_{X,d} + \dot{m}_{X,r} = \dot{m}_{X,d}\left(1+R_R\right) \quad \Rightarrow \quad \dot{m}_{X,d} = \frac{\dot{m}_{X,c}}{\left(1+R_R\right)} \tag{15}$$

$$[X]_d = \frac{\dot{m}_{X,d}}{\dot{V}_d} \tag{16}$$

Calculation Details Using MATLAB

Two MATLAB functions were written to perform the calculations needed to solve this problem. The first, Example_31_1_pfr, takes the inlet mass flow rates of S and X and the inlet volumetric flow rate as arguments and solves the PFR design equations, returning the outlet mass flow rates of S and X. The second MATLAB function, Example_31_1, solves the mixing point material balances, using the first function, then calls the first function using the results to find the mass flow rates of S and X in stream c, and finally uses equations (15) and (16) to compute the cell mass concentration in stream d.

MATLAB function for modeling the PFR. The PFR design equations are initial value ODEs. Supplemental Unit S5 provides template files that can be used to solve them. In this problem, the final value of the independent variable is provided, so the appropriate template file is SolvIVDifI.m. Before that file can be used, you must make four required modifications.

To begin, I made a copy of the template file and saved it as Example_31_1_pfr.m; a copy of that file accompanies this solution. Since the function name must match the filename, I changed the name of the function to Example_31_1_pfr. At the same time, I changed the function declaration so that the inlet mass flow rates of S and X and the inlet volumetric flow rate were passed to the function as arguments and so that the outlet mass flow rates of S and X were returned. The template file begins with a long set of comments describing what it does and how to use it; I replaced these comments with a brief comment stating the purpose of the modified version. None of these modifications were required.

The first *required* modification involves entering all the known quantities from the problem statement along with constants that will be needed (from handbooks or other reference sources). As these are entered, they should be converted to a consistent set of units. In this function, I only entered the constants required for solving the PFR design equations as shown in Listing 1.

```
% Modified version of the AFCoKaRE MATLAB template file SolvIVDifI.m used
% to model the PFR in Example 31.1 of "A First Course on Kinetics and
% Reaction Engineering."
%
function [mSout,mXout] = Example_31_1_pfr(mSin,mXin,VFRin)
    % Known quantities and constants (in consistent units)
    V = 4000.0; % cc
    k1 = 0.034; % /min
    k2 = 0.001; % g/cc
```

Listing 1. Non-required modifications and entry of known constants.

The second *required* modification involves entering the code to evaluate functions f_1 and f_2 in equations (9) and (10). This takes place in the internal function, `odeqns`, where values of the independent variable are passed in as the scalar, `t`, and values of the dependent variables are passed in in the vector, `z`. Thus, it is necessary to map the dependent variables used in the problem statement ($\dot{m}_S$ and $\dot{m}_X$) to the vector `z`; the corresponding derivatives are mapped to a vector `dzdt`. I find it useful at the start of the internal function that will evaluate the derivatives, to define local variables with the names

used in the problem statement. This modification is not required, but in my opinion, it makes the code more readable and easier to debug. In addition, the list of variables here serves as a reminder of the mapping of the problem statement variables to the vector `z`. Given the values of the independent and dependent variables, the mass concentrations of S and X can first be calculated, and then the rate can be calculated using equations (11), (12) and (1), respectively. Following that, the functions f_1 and f_2 in equations (9) and (10) can be evaluated, saving the results in the vector `dzdt`. These modifications are shown in Listing 2.

```
% Function that evaluates the ODEs
function dzdt = odeqns(t,z)
    % t is the value of the independent variable
    % z is a vector containing the values of the dependent variables
    mS = z(1);
    mX = z(2);
    CS = mS/VFRin;
    CX = mX/VFRin;
    rate = k1*CS*CX/(k2 + CS);
    dzdt = [
        -2.2*rate;
        rate;
    ];
end % of internal function odeqns
```

Listing 2. Internal function odeqns after required modifications have been made.

The third ***required*** modification involves providing the initial values of the independent and dependent variables and the final value of the independent variable. The independent variable is the cumulative reactor volume, and its initial and final values are entered as `t0` and `tf`, respectively. The initial values of the dependent variables are entered as a vector named `z0`, and they must use the same mapping as was used previously for `z`. Here the necessary initial values of the mass flow rates are passed into the main function as arguments. The results of performing this modification are shown in Listing 3.

```
% Initial and final values
t0 = 0;
z0 = [
    mSin;
    mXin;
];
tf = V;
```

Listing 3. Specification of the initial values.

The final ***required*** modification is to use the results from solving the ODEs to calculate whatever the problem requested. In this case, all that needs to be done is to set the values of the outlet mass flow rates so that they will be returned by the function. This is shown in Listing 4.

```
    % Set the return values
    mSout = z(1);
    mXout = z(2);

end % of Example_31_1_pfr.m
```

Listing 4. Final modification where the return variables are set to the proper values.

MATLAB function for solving the mixing point balances and evaluating the recycle PFR. The mixing point design equations must be solved in order to analyze the PFR. The mixing point design equations are non-differential, non-linear equations. Supplemental Unit S2 describes how to solve such equations numerically using MATLAB, and it provides a template file named SolvNonDif.m for doing so. That template file was used as the starting point here. Before it can be used, SolvNonDif.m must be modified in four places, each indicated by a comment that begins "% EDIT HERE".

In addition to those required modifications, I recommend that you work with a copy of the file that has been given a more meaningful name. In this case, I made a copy of the file and saved it as Example_31_1.m; a copy of the final version of that file accompanies this solution. Since the function name must match the filename, I changed the name of the function to Example_31_1. At the same time, knowing that I won't need to use the results from these calculations in subsequent calculations, I changed the function so that it does not return any values. The template file begins with a long set of comments describing what it does and how to use it; I replaced these comments with a brief comment stating the purpose of the modified version. None of these modifications were required. The first required modification is to enter the values of all universal and problem specific constants, converting them to a consistent set of units as they are entered. Here I only entered the constants needed for solving the mixing point design equations as can be seen in Listing 5.

The next required modification is to provide code that evaluates the functions f_3 and f_4 in equations (13) and (14). At this point, you need to decide which of the unknowns in the equations being solved is going to be represented as z_1, which as z_2, and so on. I recommend actually defining variables with names similar to those used in your solution and setting them equal to the corresponding z_i. Doing so will make your code a little less efficient, but it may also reduce the chances of coding errors. Listing 6 shows how this was done in Example_31_1.m and how code to evaluate the functions was entered so that the vector named f contains the value of f_3 as its first element and the value of f_4 as its second element. Notice that before the functions could be evaluated, it was necessary to call the Example_31_1_pfr to solve the PFR design equations and obtain the values of $\dot{m}_{S,c}$ and $\dot{m}_{X,c}$.

```
% Modified version of the AFCoKaRE MATLAB template file SolvNonDif.m used
% to model the recycle PFR in Example 31.1 of "A First Course on Kinetics
% and Reaction Engineering."
%
function Example_31_1
    % Known quantities and constants (in consistent units)
    CSa = 0.04; % g/cc
    CXa = 0.; % g/cc
    VFRa = 60.; % cc/min
    RR = 0.2;
    mSa = VFRa*CSa;
    mXa = VFRa*CXa;
    VFRd = VFRa;
    VFRr = RR*VFRd;
    VFRc = VFRr + VFRd;
    VFRb = VFRc;
```

Listing 5. Initial modifications to the template file SolvNonDif.m.

```
    % Function that evaluates the equations
    function f = evalEqns(z)
        mSb = z(1);
        mXb = z(2);
        [mSc,mXc] = Example_31_1_pfr(mSb,mXb,VFRb);
        f = [
            mSa + RR*mSc/(1+RR) - mSb;
            mXa + RR*mXc/(1+RR) - mXb;
        ];
    end % of internal function evalEqns
```

Listing 6. Modifications to evaluate the functions being solved.

The third required modification is where guesses for the unknowns in the mixing point design equations are provided. It can be tricky to provide guesses that lead to a converged solution, and you may need to try several times in order to find guesses that work. The guesses are entered in the array named z_guess. Listing 7 shows the guesses I used. Since I knew the mass flow of S in stream a, I multiplied that by one plus the recycle ratio to get a crude inlet mass flow into the reactor, and then multiplied the result by 0.5 to account for the expected lower concentration of S in the recycle stream. For X, I multiplied the inlet flow of S times the recycle ratio and took 0.2 of the result as my guess.

```
    % guesses for the solution
    z_guess = [
        (1+RR)*mSa*0.5
        RR*mSa*0.2
    ];
```

Listing 7. Modification where guesses are provided for the unknowns.

The final modification is where any additional calculations are performed after the non-linear equations have been solved. In this problem, solving the mixing point design equations yields $\dot{m}_{S,b}$ and $\dot{m}_{X,b}$, the PFR inlet mass flow rates. Therefore, after the mixing point design equations had been solved, the MATLAB function for solving the PFR design equations was called, using those values. Doing so returns the PFR outlet mass flow rates of S and X. At that point, equations (15) and (16) could be used to calculate the requested final mass concentration of cell mass. The code is shown in Listing 8.

```
    % Solve the PFR design equations
    mSb = z(1);
    mXb = z(2);
    [mSc,mXc] = Example_31_1_pfr(mSb,mXb,VFRb);
    % Calculate the outlet cell mass concentration
    mXd = mXc/(1+RR);
    CXd = mXd/VFRd

end % of Example_31_1.m
```

Listing 8. Final modification where the process cell mass concentration is calculated.

At this point, Example_31_1.m can be executed by by typing Example_31_1 at the MATLAB command prompt. The resulting output is shown in Listing 9. It is important to check that the non-linear equation solver converged. In this case it did, and therefore, the results produced by the calculations can be accepted.

```
>> Example_31_1

Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

The solver found the following values for the unknowns:

z =
    2.4504
    0.1953

The corresponding values of the functions being solved are as follows:

f =
   1.0e-07 *

    0.3138
   -0.1426

CXd =
    0.0163
```

Listing 9. Output from the execution of Example_31_1.