A First Course on Kinetics and Reaction Engineering
Example 24.1

Problem Purpose

This example illustrates the situation where a CSTR can display multiple steady states.

Problem Statement

Irreversible reaction (1) is going to take place in an adiabatic CSTR with a volume of 500 cm^3. A solution flowing at 1.0 cm^3 s^{-1} and containing equal amounts of A and B (0.015 mol cm^{-3}) at 50 °C will be used. The heat capacity of the fluid is essentially equal to that of the solvent, 0.35 cal g^{-1} K^{-1} and can be considered to be constant. The (constant) density of the fluid is 0.93 gm cm^{-3}. The rate expression for reaction (1) is given in equation (2), and the heat of reaction (1) may be assumed to be constant and equal to -20 kJ mol^{-1}. At these conditions three different steady states are possible; determine the conversion and outlet temperature for each of them.

$$\text{A} + \text{B} \rightarrow \text{Y} + \text{Z} \tag{1}$$

$$r_1 = 3.24\times 10^{12}\left(\frac{\text{cm}^3}{\text{mol s}}\right)\exp\left\{\frac{-25\text{ kcal mol}^{-1}}{RT}\right\}C_AC_B \tag{2}$$

Problem Analysis

This problem involves a reaction taking place in a CSTR. The rate expression is known and we are asked to calculate reactor variables, hence it is a reaction engineering problem. To solve it, the mole and energy balance design equations will be written and solved, and the results will then be used to calculate any other quantities of interest. In this case, nothing has been changed and nothing is varying with time, so the reactor operates at steady state and the steady state form of the design equations can be used.

Problem Solution

The system can be represented schematically as shown in Figure 1. Some of the quantities provided in the problem statement will need to be converted to different units so that all quantities are in a consistent set of units. One way to do this is to convert the inlet temperature from °C to K and to convert the fluid heat capacity and the activation energy from cal or kcal to J. By doing so all volumes will use cm^3, all times will use s, all molar quantities will use mol and all energies will use J. The fluid density and the fluid heat capacity also contain mass units, but it will be seen that these two quantities only appear once, and when they do, the mass units will cancel out.

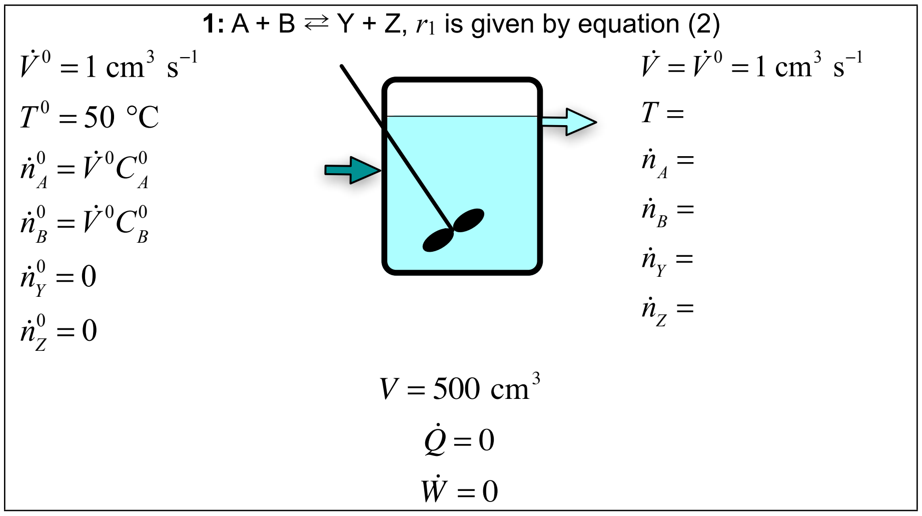
1: A + B ⇄ Y + Z, r_1 is given by equation (2)
$\dot{V}^0 = 1\ \text{cm}^3\ \text{s}^{-1}$
$T^0 = 50\ °\text{C}$
$\dot{n}_A^0 = \dot{V}^0 C_A^0$
$\dot{n}_B^0 = \dot{V}^0 C_B^0$
$\dot{n}_Y^0 = 0$
$\dot{n}_Z^0 = 0$
$\dot{V} = \dot{V}^0 = 1\ \text{cm}^3\ \text{s}^{-1}$
$T =$
$\dot{n}_A =$
$\dot{n}_B =$
$\dot{n}_Y =$
$\dot{n}_Z =$
$V = 500\ \text{cm}^3$
$\dot{Q} = 0$
$\dot{W} = 0$

Figure 1. Schematic representation of the reactor.

Mole balances can be written for each species present in the system. The generalized mole balance equation is given below. In this case, the reactor operates at steady state, so the derivative with respect to time is equal to zero. Furthermore, there is only one reaction taking place, so the summation reduces to a single term with j = 1.

$$\frac{d}{dt}\left(\frac{\dot{n}_i V}{\dot{V}}\right) = \dot{n}_i^0 - \dot{n}_i + V \sum_{\substack{j=\text{all}\\ \text{reactions}}} \nu_{i,j} r_j$$

$$0 = \dot{n}_i^0 - \dot{n}_i + V\nu_{i,1} r_1$$

Writing the mole balance for each species then lead to equations (3) through (6).

$$0 = \dot{n}_A^0 - \dot{n}_A + V\nu_{A,1} r_1 = \dot{n}_A^0 - \dot{n}_A - Vr_1 = f_1\left(\dot{n}_A, \dot{n}_B, \dot{n}_Y, \dot{n}_Z, T\right) \tag{3}$$

$$0 = \dot{n}_B^0 - \dot{n}_B + V\nu_{B,1} r_1 = \dot{n}_B^0 - \dot{n}_B - Vr_1 = f_2\left(\dot{n}_A, \dot{n}_B, \dot{n}_Y, \dot{n}_Z, T\right) \tag{4}$$

$$0 = \dot{n}_Y^0 - \dot{n}_Y + V\nu_{Y,1} r_1 = \dot{n}_Y^0 - \dot{n}_Y + Vr_1 = f_3\left(\dot{n}_A, \dot{n}_B, \dot{n}_Y, \dot{n}_Z, T\right) \tag{5}$$

$$0 = \dot{n}_Z^0 - \dot{n}_Z + V\nu_{Z,1} r_1 = \dot{n}_Z^0 - \dot{n}_Z + Vr_1 = f_4\left(\dot{n}_A, \dot{n}_B, \dot{n}_Y, \dot{n}_Z, T\right) \tag{6}$$

The general energy balance equation is given below. In this case the reactor operates at steady state, so each of the time derivatives is equal to zero. In addition, the heat term equals zero because the reactor is adiabatic and the work term is negligible. Instead of summing the individual heat capacities of the reagents, we can use the heat capacity of the fluid as a whole in this problem. However, that is provided as a mass heat capacity, so it will need to be multiplied by the total mass flow rate of the fluid, which, in turn, is equal to the volumetric flow rate times the fluid density. Also, the heat capacity is a constant allowing evaluation of the integral. This results in the simplified energy balance given in equation (7).

$$V\left(\sum_{\substack{i=\text{all}\\ \text{species}}}\frac{\dot{n}_i\hat{C}_{p-i}}{\dot{V}}\right)\frac{dT}{dt}-P\frac{dV}{dt}-V\frac{dP}{dt}$$

$$=\dot{Q}-\dot{W}-\sum_{\substack{i=\text{all}\\ \text{species}}}\left(\dot{n}_i^0\int_{T^0}^{T}\hat{C}_{p,i}\,dT\right)-V\sum_{\substack{j=\text{all}\\ \text{reactions}}}\left(r_j\Delta H_j\left(T\right)\right)$$

$$0=\dot{V}\rho_{fluid}\tilde{C}_{p,fluid}\left(T-T^0\right)+Vr_1\Delta H_1\left(T\right)=f_5\left(\dot{n}_A,\dot{n}_B,\dot{n}_Y,\dot{n}_Z,T\right) \tag{7}$$

We now need to solve equations (3) through (7), but in order to do so, we need to identify the five unknown quantities we will solve for. In this case it is clear from the schematic diagram, Figure 1, that the values of $\dot{n}_A$, $\dot{n}_B$, $\dot{n}_Y$, $\dot{n}_Z$ and T are unknown. Since these quantities all appear in equations (3) through (7), they will be used as the unknowns.

Equations (3) through (7) do not contain integrals or derivatives. As such, one could attempt to solve them manually, using symbolic algebra software or using numerical methods software. A variety of software packages are available for doing so, and you should use the one you feel most comfortable with. Supplemental Unit S2 presents a brief introduction to the numerical solution of sets of non-differential equations like these using numerical methods software. No matter what software package you use, you typically will need to provide two things as input in order to solve the equations:

- Code that evaluates the functions, f_1 through f_5 in equations (3) through (7), given values of the unknown variables, $\dot{n}_A$, $\dot{n}_B$, $\dot{n}_Y$, $\dot{n}_Z$ and T
- A guess for the solution, that is, a guess for the values of $\dot{n}_A$, $\dot{n}_B$, $\dot{n}_Y$, $\dot{n}_Z$ and T that cause all of the functions, f_1 through f_5, to equal zero

The functions to be evaluated contain quantities other than the five unknown quantities. In order to evaluate the functions, values will be needed for each of those other quantities. The inlet volumetric flow rate is constant and its value is specified, as are the concentrations of the reagents in the inlet stream. That information is sufficient to calculate the inlet molar flow rates ($\dot{n}_A^0$, $\dot{n}_B^0$, $\dot{n}_y^0$ and $\dot{n}_Z^0$) as indicated in the schematic diagram, Figure 1. The reaction volume (V), heat capacity ($\hat{C}_{p,fluid}$), density (ρ_{fluid}), inlet temperature (T^0) and the heat of reaction ($\Delta H_1(T)$) are also constants whose values are given in the problem statement.

The rate, however, is not a constant, but its value can be computed when needed using the rate expression, equation (2). In order to do so, values are needed for the concentrations of A and B. These are easily calculated using equations (8) and (9). In equations (8) and (9), the outlet volumetric flow rate may be taken to equal the inlet volumetric flow rate (a known constant) since the liquid density is constant. The values of the molar flow rates of A and B will be given at the time the design equations are being evaluated.

$$C_A = \frac{\dot{n}_A}{\dot{V}} \tag{8}$$

$$C_B = \frac{\dot{n}_B}{\dot{V}} \tag{9}$$

That is everything that is needed in order to evaluate the functions, f_1 through f_5. The only other thing that will be needed in order to solve the equations numerically is a guess for the values of the unknowns. Since all that is needed is a guess, one possibility is to use the inlet molar flow rates and temperature as the guess. With that, equations (3) through (7) can be solved numerically to find $\dot{n}_A$, $\dot{n}_B$, $\dot{n}_Y$, $\dot{n}_Z$ and T. The problem asked for the conversion and the outlet temperature. The latter is found directly by solving the design equations. The fractional conversion can be computed using equation (10) once the design equations have been solved to obtain $\dot{n}_A$ ($\dot{n}_A^0$ is a known constant).

$$f_A = \frac{\dot{n}_A^0 - \dot{n}_A}{\dot{n}_A^0} \tag{10}$$

Upon performing these calculations one finds that there are three solutions to the steady state design equations. One steady state corresponds to an outlet temperature of 323 K and a conversion of 0.03%, the second corresponds to a temperature of 409 K and a conversion of 38.8% and the third corresponds to a temperature of 538 K and a conversion of 97.6%.

Calculation Details Using MATLAB

Supplemental Unit S2 describes how to solve sets of non-differential equations numerically using MATLAB, and it provides a template file named SolvNonDif.m for doing so. Before it can be used to solve a problem, that template file must be modified in four places, each indicated by a comment that begins "% EDIT HERE". In addition to those required modifications I made a few additional modifications that will be described here along with the required modifications.

I recommend that you work with a copy of the file that has been given a more meaningful name; I used Example_24_1.m. Since the function name must match the filename, I changed the name of the function to Example_24_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. However I know I will want to run it using different guesses for the solution, so I additionally changed the function statement so that the guesses for the outlet molar flow rates and the outlet temperature are passed in as arguments nAg, nBg, nYg, nZg and Tg. 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. As a result of making them, the beginning of the file looks as shown in Listing 1.

```
% Modified version of the MATLAB template file SolvNonDif.m used in the
% solution of Example 24.1 of "A First Course on Kinetics and Reaction
% Engineering."
%
function Example_24_1(nAg,nBg,nYg,nZg,Tg)
```

Listing 1. Non-required modifications made at the beginning of the template file.

The first *required* modification is to enter the values of all universal and problem specific constants at the point indicated. At the same time these are entered, they should be converted to a consistent set of units. Listing 2 shows the next part of Example_24_1.m where these modifications were made.

```
    % Known quantities and constants (in consistent units)
    V = 500; % cm3
    VFR = 1; % cm3/s
    CA0 = 0.015; % mol/cm3
    CB0 = CA0; % mol/cm3
    nA0 = VFR*CA0; % mol/s
    nB0 = VFR*CB0; % mol/s
    nY0 = 0; % mol/s
    nZ0 = 0; % mol/s
    T0 = 50 + 273.15; % K
    Cp = 0.35*4.184; % J/g/K
    rho = 0.93; % g/cm3
    dH = -20000; % J/mol
    k0 = 3.24e12; % cm3/mol/s (pre-exponential factor in equation (2))
    E = 25000*4.184; % J/mol (activation energy in equation (2))
    R = 8.31446; % J/mol/K
```

Listing 2. Portion of the modified template file SolvNonDif.m where problem-specific and universal constants were entered in consistent units.

The second *required* modification involves entering the code to evaluate functions f_1 through f_5, equations (3) through (7). In the code, this occurs within an internal function named evalEqns; within evalEqns, both the unknowns and the equations are provided as vector quantities named `z` and `f`, respectively. Thus, it is necessary to map the variables used in the problem solution to represent the unknowns ($\dot{n}_A$, $\dot{n}_B$, $\dot{n}_Y$, $\dot{n}_Z$ and T) to a vector `z`, and to return the values of the functions in the vector `f`. I find it useful at the start of the internal function that will evaluate the functions, 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`.

Recall from the solution that the functions contained variable quantities (r_1, C_A and C_B) that depend upon the unknowns. In the code being written here to evaluate the functions, we are given values of the unknowns as just described. Therefore, these variable quantities can be evaluated here using the equations given in the problem statement, specifically equations (8), (9) and (2). The constants from the problem statement have already been entered and are available at this point in the MATLAB file, so the functions f_1 through f_5 can next be evaluated. The code containing all these modifications is shown in Listing 3.

```
    % Function that evaluates the equations
    function f = evalEqns(z)
        % mapping of unknowns into vector z
        nA = z(1);
        nB = z(2);
        nY = z(3);
        nZ = z(4);
        T = z(5);
        % calculate variables quantities
        CA = nA/VFR; % equation (8)
        CB = nB/VFR; % equation (9)
        r1 = k0*exp(-E/R/T)*CA*CB; % equation (2)
        f = [
            nA0 - nA - V*r1
            nB0 - nB - V*r1
            nY0 - nY + V*r1
            nZ0 - nZ + V*r1
            VFR*rho*Cp*(T-T0) + V*r1*dH
        ];
    end % of internal function evalEqns
```

Listing 3. Portion of the modified template file SolvNonDif.m showing the entry of the code to evaluate the functions being solved.

The third *required* modification is where guesses for the unknowns are provided. The guesses are entered in the array named z_guess. They must be entered using the same mapping of the unknowns to the vector z as was used above. The previous modification, where variables with more meaningful names were defined, serves as a key to remind you which variable is z_1, which is z_2, and so on. Since I will need to find different solutions to the equations, I will need to provide different guesses. For this reason, the function statement was modified earlier to accept the guesses as arguments. Thus, all that needs to be done is to set the values of `z_guess` equal to the guesses that were passed in as arguments, as shown in Listing 4.

```
    % guesses for the solution
    z_guess = [
        nAg
        nBg
        nYg
        nZg
        Tg
    ];
```

Listing 4. Portion of the modified template file SolvNonDif.m where guesses are provided for the unknowns.

The final *required* modification only applies if you need to use the results from solving the set of equations to calculate other quantities. In this case, the problem asked for the conversion of A, which can be calculated using equation (10). Listing 5 shows the code that was used to do this; it also prints out the value of the temperature separately from the other unknowns, since the problem also asked for its value.

```
    % Report the results
    T = z(5)
    pct_conv = 100*(nA0-z(1))/nA0  % equation (10)
```

Listing 5. Portion of the modified template file SolvNonDif.m where the results of solving the set of non-differential equations are used to calculate additional quantities.

At this point, the modified template file can be used to solve the equations. To do so, the function name is typed at the MATLAB command prompt, with guesses for the outlet molar flow rates and the final temperature (in K) as arguments within parentheses following the function name. One option for guessing the outlet molar flows and temperature is to guess the inlet values. (The inlet volumetric flow rate is 1.0 cm^3 s^{-1} and the inlet concentrations of A and B are 0.015 mol cm^{-3}, so the inlet molar flows of A and B are 0.015 mol s^{-1}. The inlet flows of Y and Z are zero, and the inlet temperature is 25 ºC or 323 K.) Listing 6 shows that in this case this works and a solution is found.

```
>> Example_24_1(0.015,0.015,0,0,323)

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 =
    0.0150
    0.0150
    0.0000
    0.0000
  323.2167

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

f =
   1.0e-13 *

   -0.0001
   -0.0001
    0.0001
    0.0001
   -0.3893

T =
  323.2167

pct_conv =
    0.0303
```

Listing 6. Execution of 24_Example_1 using the inlet molar flows and temperature as guesses for the outlet molar flows and temperature.

In order to find another different steady state solution, I'll need to provide a different guess for the solution. This steady state solution corresponds to a situation where essentially no reaction took place. Therefore one possibility is to guess essentially complete conversion of the reactants, and a higher final temperature (since the reaction is exothermic). Listing 7 shows that when I tried this, the equation solver stopped before it converged to a solution. Therefore, I used the unconverged solution it found as the guess, after which a second steady state solution was found as also shown in Listing 7.

At this point, I have no way of knowing whether this second steady state solution corresponds to the highest conversion steady state or the intermediate conversion steady state. Suspecting that there might be a steady state with a higher conversion, my third guess again used molar flows corresponding to essentially complete conversion, but I guessed a temperature (500 K) greater than the final temperature of the second solution I just found (409 K). Once again, the equation solver stopped before it converged to a solution, so I re-started as before, and at this point it converged to the third steady state solution, as shown in Listing 8.

```
>> Example_24_1(0,0,0.015,0.015,400)

Solver stopped prematurely.

fsolve stopped because it exceeded the function evaluation limit,
options.MaxFunEvals = 500 (the default value).

The solver found the following values for the unknowns:

z =
    0.0093
    0.0093
    0.0061
    0.0061
  408.2840

       ⋮

>> Example_24_1(0.0093,0.0093,0.0061,0.0061,408)

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 =
    0.0092
    0.0092
    0.0058
    0.0058
  408.6522

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

f =
   1.0e-12 *

    0.0000
    0.0000
   -0.0000
   -0.0000
    0.2558

T =
  408.6522

pct_conv =
   38.8149
```

Listing 7. Execution of 24_Example_1 using a guess with near complete conversion.

```
>> Example_24_1(0.008,0.008,0.015,0.015,508)

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 =
    0.0004
    0.0004
    0.0146
    0.0146
  538.1711

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

f =
   1.0e-12 *

   -0.0000
   -0.0000
    0.0000
    0.0000
   -0.1705

T =
  538.1711

pct_conv =
   97.6118
```

Listing 8. Execution of 24_Example_1 using the unconverged solution as a guess.