Introduction to Reactor Design, 3K4

Tutorial 5/Assignment 3B

Kevin Dunn, kevin.dunn@mcmaster.ca **Due at class, 28 February; no late hand-ins**

Objectives of this short assignment:

We are going to start using software more frequently in the course, because the equations we deal with are too difficult to integrate analytically. So the main purpose of this tutorial is:

- To install and use a numerical integration software package on your computer. You may use any software, but Python, MATLAB and Polymath are recommended candidates.
- Ensure you can use the software to successfully integrate one or more differential equations.

Question 1

A can of your favourite beverage, initially at 25°C, is placed in the fridge, where the ambient temperature is 5°C. Plot the temperature profile of the can's contents (assume the contents are well mixed) over time. The temperature of the can's contents, T, can be derived and modelled as:

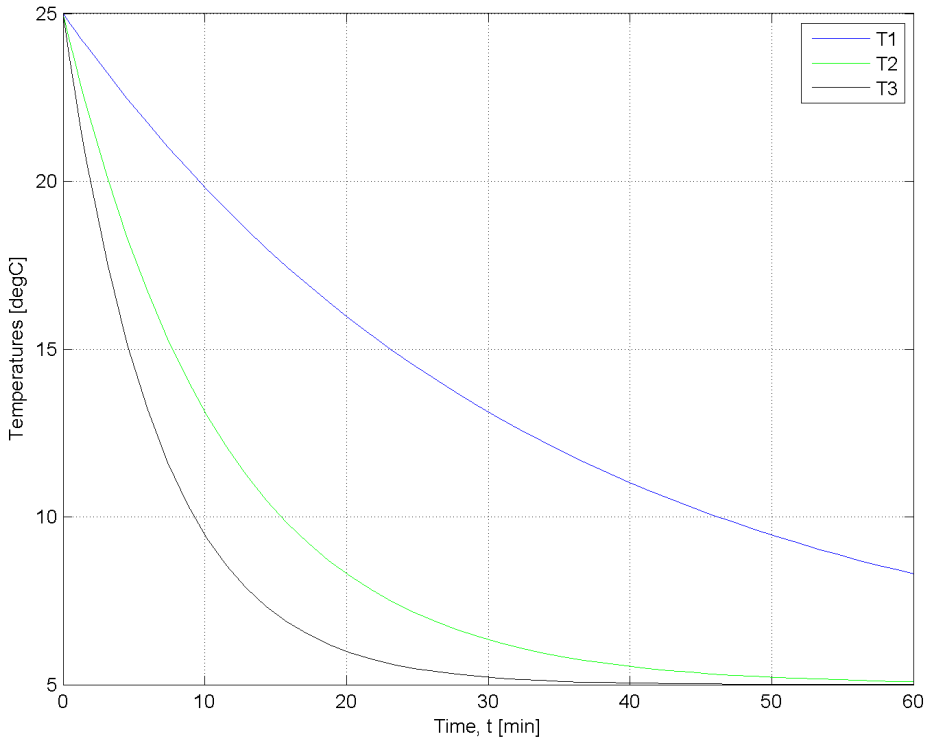
$$\frac{dT}{dt} = -K(T - T_f)$$

where t is the time, in minutes, K is the heat transfer coefficient that lumps the can's properties into a single variable, and T_f is the fridge reference temperature.

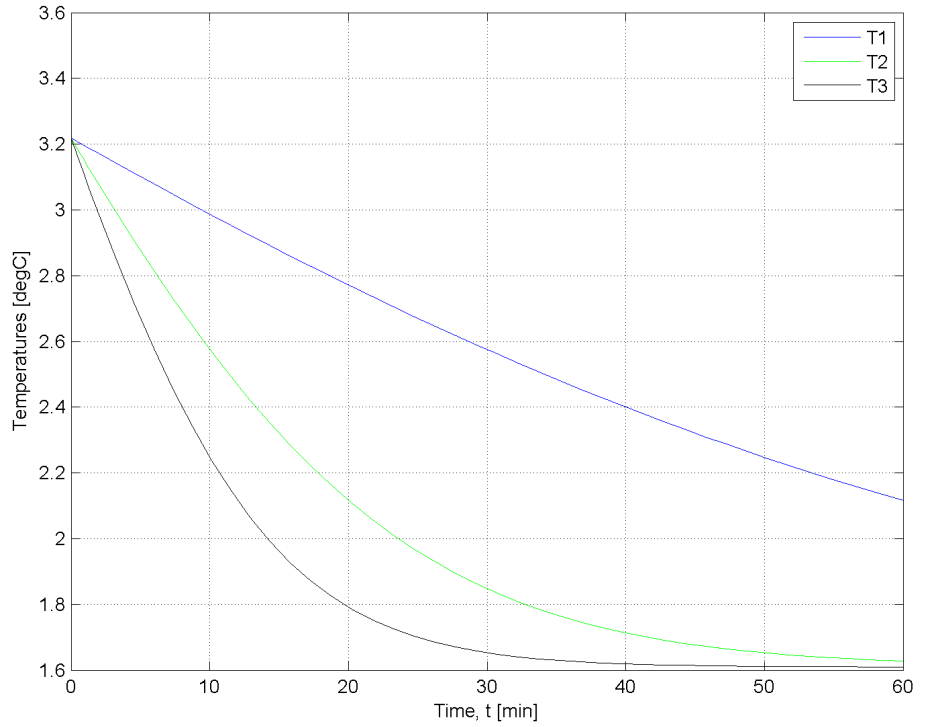
1. Plot the temperature profiles for $K = 0.03\,\text{min}^{-1}$, $K = 0.09\,\text{min}^{-1}$ and $K = 0.15\,\text{min}^{-1}$ on the same plot, on a range to 1 hour.
2. Repeat the same exercise, but use a logarithmic y-axis. Comment on the results from this part, as compared to the previous part.

Solution

1. The temperature profiles are shown below:



2. A plot with logarithmic y-axis should show linear lines, because the integral of the ODE gives rise to an exponential function. However, they are not perfectly linear because the true integral is $T = T_f - \frac{e^{-Kt}}{K}$ and taking logs on both sides does not lead to a linear function. You could also have used the `semilogy` function to plot the graphs.



The MATLAB code for this question is:

```
% fridge.m
%-------------------------
function d_depnt__d_indep = fridge(indep, depnt)

% Dynamic balance for beverages in the fridge
%
%    indep: the independent ODE variable, such as time or length
%    depnt: a vector of dependent variables
%
%    X = depnt(1) = the conversion
%    y = depnt(2) = the pressure ratio = P/P_0 = y
%
%    Returns d(depnt)/d(indep) = a vector of ODEs

% Assign some variables for convenience of notation
T1 = depnt(1);
T2 = depnt(2);
T3 = depnt(3);

% Constant(s)
T_f = 5;     % degC
K1 = 0.03;   % 1/min
K2 = 0.09;   % 1/min
K3 = 0.15;   % 1/min

% Output from this ODE function must be a COLUMN vector, with n rows
n = numel(depnt);
d_depnt__d_indep = zeros(n,1);
d_depnt__d_indep(1) = -K1 * (T1 - T_f);
d_depnt__d_indep(2) = -K2 * (T2 - T_f);
d_depnt__d_indep(3) = -K3 * (T3 - T_f);

% ode_driver_fridge.m
%-------------------------
% The independent variable always requires an initial and final value:
indep_start = 0.0;   % min
indep_final = 60;    % min

% Set initial condition(s): for integrating variables (dependent variables)
T1_depnt_zero = 25;     % i.e. T1(t=0) = 25 degC
T2_depnt_zero = 25;
T3_depnt_zero = 25;

% Integrate the ODE(s):
[indep, depnt] = ode45(@fridge, [indep_start, indep_final], ...
                               [T1_depnt_zero, T2_depnt_zero, T3_depnt_zero]);
```

```
% Plot the results:
clf;
plot(indep, depnt(:,1), 'b')
grid('on')
hold('on')
plot(indep, depnt(:,2), 'g')
plot(indep, depnt(:,3), 'k')
xlabel('Time, t [min]')
ylabel('Temperatures [degC]')
legend('T1', 'T2', 'T3')

% Plot the results on logarithmic y-axis:
figure;
plot(indep, log(depnt(:,1)), 'b')
grid('on')
hold('on')
plot(indep, log(depnt(:,2)), 'g')
plot(indep, log(depnt(:,3)), 'k')
xlabel('Time, t [min]')
ylabel('Temperatures [degC]')
legend('T1', 'T2', 'T3')
```

Question 2

This question is a good check to ensure you are using the software correctly: you must be able to get the same answer with the software that you get analytically.

A batch bioreactor is charged with substrate at time $t = 0$ to start a reaction that will consume the substrate. The reaction mechanism is too complex to model exactly, but previous experience suggests it is roughly a first-order reaction, with apparent rate expression is $-r_S = k_S C_S$, where $k_S = 0.58\,\text{hour}^{-1}$, and C_S is the concentration of the substrate being depleted.

1. Show that the rate of change of the concentration of S is given by:

$$\frac{dC_S}{dt} = -k_S C_S(t)$$

2. The time required to consume the substrate from a starting level of $260\,\text{mol.L}^{-1}$ to $5\,\text{mol.L}^{-1}$ can be calculated by analytical integration of the above expression. What is the total time required?

3. The tank volume is doubled; how long will the duration be to go from the same starting to final substate concentration?

4. Use software tools and integrate between the same limits and calculate the result numerically. Verify that your result matches the analytical answer.

   Show the software source code you wrote to find the solution.

Solution

1. The mole balance on a well-mixed batch reactor is given below. If the reactor volume, V remains constant:

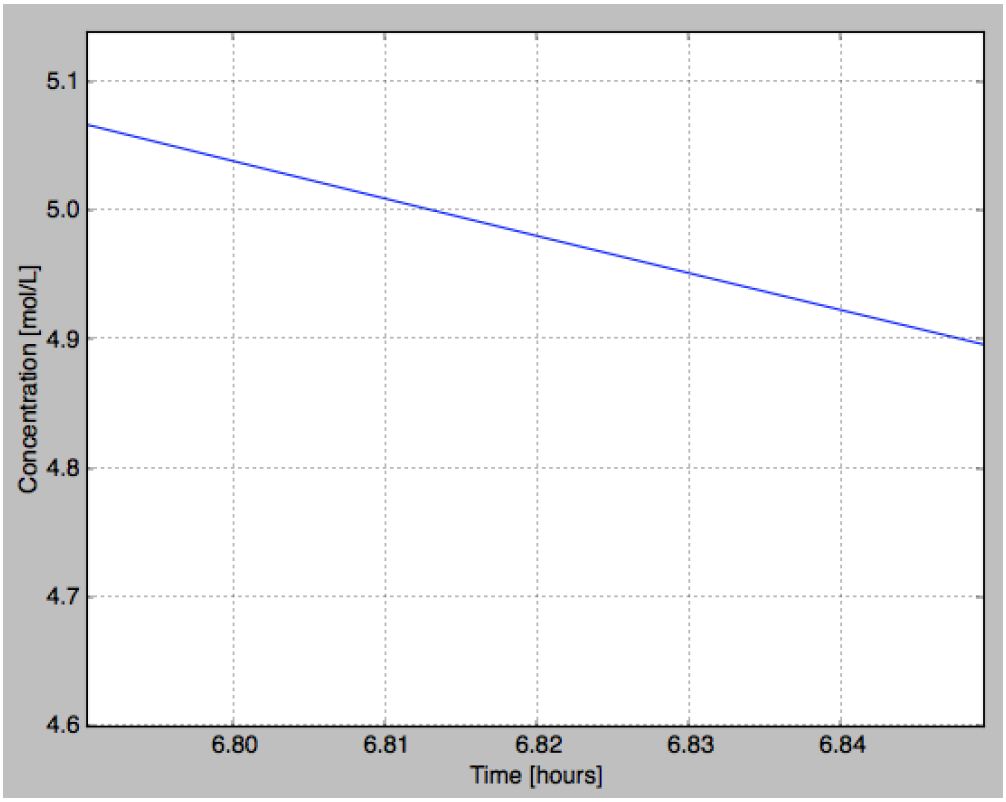
$$\frac{dN_S}{dt} = r_S V$$
$$\frac{dC_S}{dt} = r_S = -k_S C_S$$

2. Integrating the above expression:

$$\int_0^t dt = -\frac{1}{k_S}\int_{260}^{5}\frac{dC_S}{C_S}$$
$$t = -\frac{1}{0.58}\ln\left(\frac{5}{260}\right) = 6.81\,\text{hours} = 409\,\text{minutes}$$

3. In this instance the duration will be the same, because the integral is not a function of the reactor volume.

4. In this example our independent variable, time t, is the solution we are looking for. Integrate for various time durations, from a starting substrate concentration of $C_S = 260\,\text{g.L}^{-1}$ and aim for a final concentration of $5\,\text{g.L}^{-1}$ using different time durations. You will find that around 6.81 hours that the trajectory crosses the $C_S = 5\,\text{g.L}^{-1}$ mark



Question 3

From the midterm, the solution to the last question was to write the ODEs:

$$\frac{dX}{dW} = \frac{-r_A}{F_{A0}} = \frac{(1 \times 10^{-5})(30.07)^2 \dfrac{(1-X)^2}{(1+\varepsilon X)^2}\,(y)^2}{0.005\,\text{mol.s}^{-1}} = 1.808\frac{(1-X)^2(y)^2}{(1+0.5X)^2}$$

$$\frac{dy}{dW} = -\frac{\alpha}{2y}(1+\varepsilon X) = -\frac{5.17 \times 10^{-4}}{2y}(1+0.5X)$$

$$X(W=0) = 0.0$$

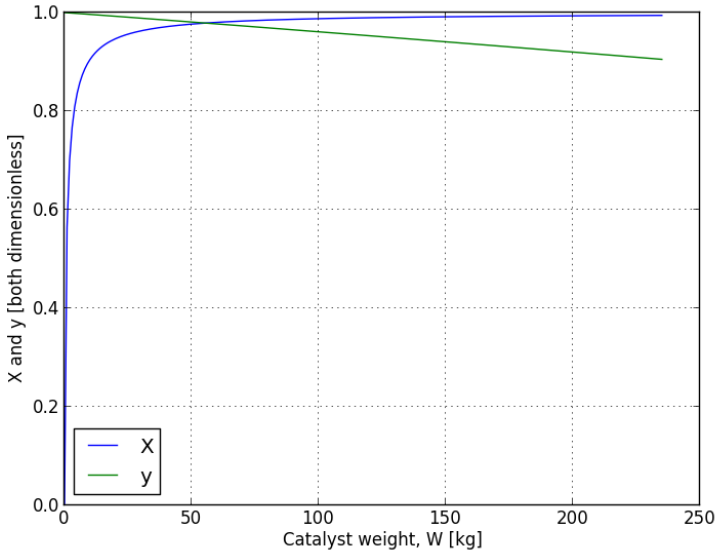
$$y(W=0) = 1.0$$

1. Integrate the solution from $W = 0$ to $W = 235.6\,\text{kg}$ and include the profiles of conversion and y (on the same graph) as your answer.
2. What is the pressure leaving the reactor?
3. What is the conversion leaving the reactor?
4. What is the conversion midway along the reactor's length?

Note: in the course project and future questions, you will NOT simplify the equations. You will write multiple equations for the knowns and unknowns and let the software do all the work for you.

Solution

1. The code below was used to perform the integration. The profiles of conversion, X and pressure drop ratio, y are shown. Notice how most of the reactor does not benefit the incremental conversion of the material leaving; the pressure drop appears to be linear across most of the reactor.



2. The pressure at the outlet of the PFR is $0.9051 \times P_0$, and in the midterm we were given $P_0 = 200\,\text{kPa}$. So the pressure at the outlet is $181\,\text{kPa}$.

3. The conversion leaving the PFR is 99.43%.

4. The conversion, midway along the reactor is found by integrating up to $W = 235.6/2 = 118.25\,\text{kg}$, and is found to be 98.93%. Note that it is not half the conversion from the previous part, because conversion does not change linearly from entry to exit of the reactor. Also note that most of the second half of the reactor is unnecessary, as it does not serve to increase the conversion by a substantial amount. This reactor is clearly over-designed.

The code for this question is:

```
import numpy as np
from scipy import integrate
from matplotlib.pylab import (plot, grid, xlabel, ylabel, show, legend)

def PBR_model(indep, depnt):
    """
    Dynamic balance for the packed bed reactor (PBR); 3K4 midterm 2013.

    indep: the independent ODE variable, such as time or length
    depnt: a vector of dependent variables

    X = depnt[0] = the conversion
    y = depnt[1] = the pressure ratio = P/P_0 = y

    Returns d(depnt)/d(indep) = a vector of ODEs
    """

    # Assign some variables for convenience of notation
    X = depnt[0]
    y = depnt[1]

    # Output from this ODE function must be a COLUMN vector, with n rows
    n = len(depnt)
    d_depnt__d_indep = np.zeros((n,1))
    d_depnt__d_indep[0] = 1.808 * (1-X)**2 * y**2 / (1+0.5*X)**2
    d_depnt__d_indep[1] = -5.17E-4/(2*y) * (1+0.5*X)
    return d_depnt__d_indep

# The "driver" that will integrate the ODE(s):
# -----------
# Start by specifying the integrator:
# use ''vode'' with "backward differentiation formula"
r = integrate.ode(PBR_model).set_integrator('vode', method='bdf')

# Set the independent variable's range
indep_start = 0.0   # kg
indep_final = 235.6 # kg
```

```
delta = 1.0          # the results will be shown only at these ``delta`` points
                     # Number of steps: 1 extra for initial condition
num_steps = np.floor((indep_final - indep_start)/delta) + 1

# Set initial condition(s): for integrating variables and independent variable!
X_indep_zero = 0.0   # i.e. X(W=0) = 0.0
y_indep_zero = 1.0   # i.e. y(W=0) = 1.0
r.set_initial_value([X_indep_zero, y_indep_zero], indep_start)

# Create vectors to store trajectories
W = np.zeros((num_steps, 1))
X = np.zeros((num_steps, 1))
y = np.zeros((num_steps, 1))
W[0] = indep_start
X[0] = X_indep_zero
y[0] = y_indep_zero

# Integrate the ODE(s) across each delta
k = 1
while r.successful() and k < num_steps:
    r.integrate(r.t + delta)

    # Store the results to plot later
    W[k] = r.t
    X[k] = r.y[0]
    y[k] = r.y[1]
    k += 1

# All done!  Plot the trajectories:
plot(W, X)
grid('on')
xlabel('Catalyst weight, W [kg]')
plot(W, y)
ylabel('X and y [both dimensionless]')
legend(['X', 'y'], 'best')
show()
```

END