Process model formulation and solution, 3E4

Assignment 6

Kevin Dunn, dunnkg@mcmaster.ca **Due: 01 December 2010**

To integrate ODEs - both by hand and with computer software.

Solutions by Ali, Elliot and Kevin.

Question 1 [2]

In assignment 1 you derived the dynamic balance for a waste water treatment system (aerobic heterotrophic growth using Monod kinetics); and in assignment 3 you used Newton's method to find the steady-state solution to the nutrient (N) and biomass (B) balances:

$$\frac{dN}{dt} = N_{\text{in}}(t)\frac{F(t)}{V} - N(t)\frac{F(t)}{V} - \left(\frac{1}{Y_B}\right)\left(\mu_{\max}\frac{N}{K+N(t)}\right)B(t)$$
$$\frac{dB}{dt} = -B(t)\frac{F(t)}{V} + \left(\mu_{\max}\frac{N(t)}{K+N(t)}\right)B(t)$$

- The tank volume is assumed constant at $V = 1600$ m^3
- The conversion efficiency is constant at $Y_B = 0.80$
- The two Monod kinetics parameters are: $\mu_{\max} = 5$ hr^{-1} and $K = 20$ g/m^3
- The inlet flow rate is $F(t) = 5000$ m^3/hr
- The nutrient inflow is $N_{\text{in}}(t) = 100$ g/m^3

These dynamic balances describe how nutrients in the inlet (i.e. the $N_{\text{in}} = 100$ g/m^3 of waste in the wastewater) are converted to biomass, B. The stream leaving the well-mixed CSTR should have lower nutrient levels, $N(t)$.

1. Code these equations in a MATLAB or Python function and use the built-in integrators in either software package to find the steady state solutions when using starting values of $N(t=0) = 100$ g/m^3 and $B(t) = 20$ g/m^3 and show that you obtain the same steady state values as found in assignment 3.
2. Most real processes are never at steady state. In this reactor a (slightly) more realistic input, to account for the diurnal nature of the flow, is:

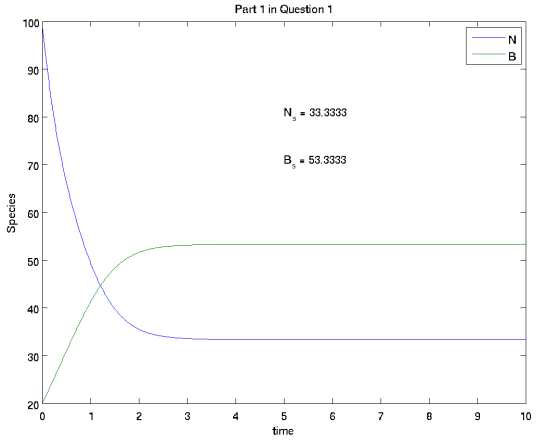
$$F(t) = 5000 + 900\sin\left(\frac{2\pi}{24}t - \frac{\pi}{5}\right)$$

   - Plot 100 hours of process operation at these conditions; use the steady state values from part 1 as your initial conditions.
   - Environmental limits prohibit outlet nutrient concentration from exceeding 60 g/m^3. Does this system meet those standards?
3. Using the simulation from part 2 (with the sinusoidal input) for the biomass and nutrient profiles, see what happens if your plant accidentally dumps nutrient into the inlet stream for 10 hours when an upstream unit fails. Use an increased nutrient influx of $N_{\text{in}}(t) = 150$ g/m^3 between $55 \leq t \leq 65$ hours.
   - Does the plant still manage to treat this accident without violating the 60 g/m^3 outlet limit on $N(t)$?
   - Would it handle this problem had it occurred between $75 \leq t \leq 85$ hours?

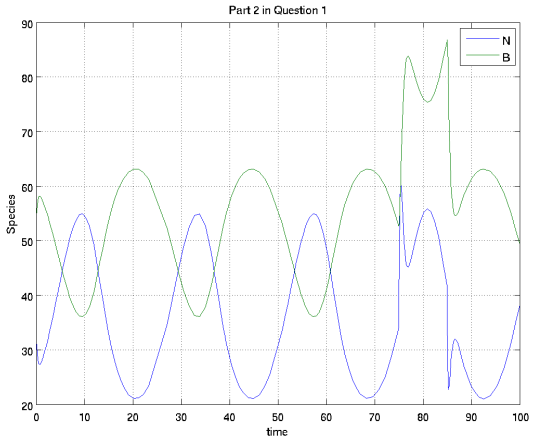
Solution

1. The code simulating the dynamical system is given below. Note that the same code was used to solve all parts of this problem. So, some of the input values or settings may need to change to suit the conditions given in the different parts of this question.

   Solving the dynamical system with the given initial conditions results in the following plots for $N(t)$ and $B(t)$. The steady-state values of $N_s = 33.33$ and $B_s = 53.33$ are seen to match the results obtained in Assignment 3 using nonlinear solution of the steady-state equations.



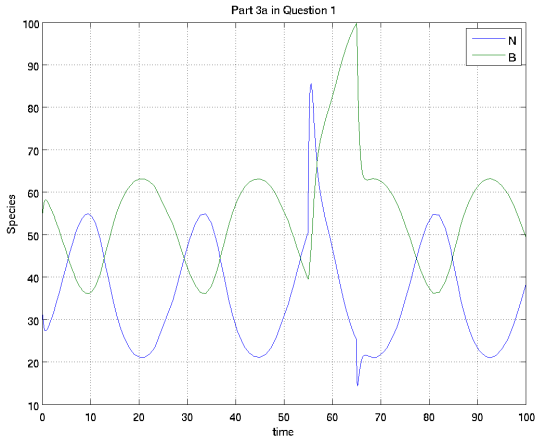
2. The steady-state values from Part 1 are used as initial conditions, i.e. $N(0) = 33.33$ and $B(0) = 53.33$. The input flow rate is now fluctuating rather than being constant. This can be seen in code shown earlier. The simulation results given the new conditions are plotted below.



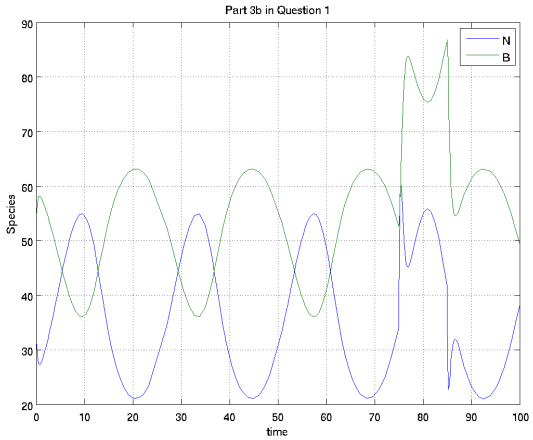
   As seen from the plot, the outlet nutrient concentration $N(t)$ does not exceed the maximum allowable limit of 60 g/m^3 (It hits this point once though.)

3. In case of an upstream failure where the nutrient influx is suddenly increased to $N_{\text{in}}(t) = 150$, we can simulate the system by modelling the nutrient influx as a function of time. It is $N_{\text{in}}(t) = 150$ if $55 \leq t \leq 65$ and $N_{\text{in}}(t) = 100$ otherwise (see the code that shows the implementation). Re-integrating the dynamic equations results the following plots below.

   It is seen that due to the sudden increase in the nutrient influx at $t = 55$, the nutrient out flow also spikes and exceeds the allowable limit of 60 g/m^3. However, this situation does not last as the nutrient outlet decreases as a result of its conversion to B.



If the accident occurred between $75 \leq t \leq 85$, the plant would behave as follows, which shows that the nutrient outlet would remain within its limit.



The following code was used in all 3 parts.

MATLAB code

```
function  dydt = wastemodel(t, y)
%      y1 = dN/dt
%      y2 = dB/dt
%
%      INPUTS:
%          t: time
%          y: the time varying concentrations: N and B

V = 1600.0;    % m^3
Y_B = 0.8;     % efficiency
mu_max = 5.0; % 1/day
K = 20.0;      % g/m^3

% F = 5000.0; % m^3/day
F = 5000 + 900*sin(2*pi/24*t -pi/5); % m^3/day

if (t >= 55 && t <= 65)
    N_in = 150.0;
```

```
else
    N_in = 100.0; % g/m^3
end

N = y(1);
B = y(2);
dydt = zeros(2,1);
dydt(1) = F/V*(N_in - N) - (1/Y_B)*(mu_max*N/(K+N)) * B;
dydt(2) = - B*F/V + (mu_max*N/(K+N)) * B;

% Save the following lines in a different
% file, e.g. wasterwater_driver.m
% =====================================================
t_0 = 0.0;
t_final = 100.0;
% For part 1:
ICs = [100.0, 20.0];
% Part 2: use the steady-state solution
ICs = [33.33333333333,  53.3333333333];
[t, y] = ode15s(@wastemodel, [t_0, t_final], ICs);

plot(t, y(:,1), t, y(:,2))
grid on
xlabel('Time')
ylabel('Species')
title('Part 3b in Question 1')
Nss = ['N_s = ' num2str(y(end,1))];
Bss = ['B_s = ' num2str(y(end,2))];
legend('N','B')
```

Python code

```
import numpy as np
from scipy import integrate
from matplotlib.pylab import *

def dy_dt(t, y):
    """
    y1 = dN/dt
    y2 = dB/dt

    INPUTS:
        t: time
        y: the time varying concentrations: N and B
    """

    V = 1600.0   # m^3
    Y_B = 0.8    # efficiency
    mu_max = 5.0 # 1/day
    K = 20.0     #g/m^3

    F = 5000.0   # m^3/day
    F = 5000 + 900*sin(2*np.pi/24*t -np.pi/5) # m^3/day

    # Change these time values, depending on the question
    if t >= 75 and t <= 85:
        N_in = 150.0
    else:
        N_in = 100.0 # g/m^3
```

```
    N = y[0]
    B = y[1]
    y = np.zeros((2,1))
    y[0] = F/V*(N_in - N) - (1/Y_B)*(mu_max*N/(K+N)) * B
    y[1] = -B*F/V + (mu_max*N/(K+N)) * B
    return y


r = integrate.ode(dy_dt).set_integrator('vode', method='bdf')
# Part 1:
ICs = [100.0, 20.0]
# Part 2: use the steady-state solution
ICs = [33.33333333333,  53.3333333333]
t_0 = 0.0
r.set_initial_value(ICs, t_0)
t_final = 100.0
dt = 0.5

# Create vectors to store the solutions in;
#     add extra space for intial condition
n_steps = np.floor((t_final - t_0)/dt) + 1
time = np.zeros(n_steps)
N = np.zeros(n_steps)
B = np.zeros(n_steps)
N[0], B[0] = ICs
k = 1
while r.successful() and r.t < t_final:
    r.integrate(r.t + dt)
    time[k] = r.t

    N[k] = r.y[0]
    B[k] = r.y[1]
    k += 1

# Clear figure window from previous simulation
clf()
plot(time, N, 'r', label='Nutrient level')
plot(time, B, 'k', label='Biomass level')
legend(loc='best')
grid('on')
```

Question 2 [3]

Consider three CSTR's in series, operating at constant volume of $V = 100$ L. A first order reaction, $\mathrm{A} \rightarrow \mathrm{B} + \mathrm{C}$ takes place in the system, with reaction rate = $r = 0.09C_{\mathrm{A}}$. The inlet conditions to the first reactor are constant with time: inlet concentration = $C_{\mathrm{A},0} = 2.7$ mol/L at a flow rate of F^0_{in} = 15.8 L/min. The outlet from the first reactor, $C_{\mathrm{A},1}$, is the inlet to the second reactor, and the outlet from the second reactor, $C_{\mathrm{A},2}$, is passed to third reactor. The outlet from the third reactor, $C_{\mathrm{A},3}$, can be used to calculate the overall conversion = $\dfrac{C_{\mathrm{A},0} - C_{\mathrm{A},3}}{C_{\mathrm{A},0}}$.

The dynamic balances in each reactor are:

$$\frac{dC_{\mathsf{A},1}(t)}{dt} = \frac{F}{V_1}\Big(C_{\mathsf{A},0} - C_{\mathsf{A},1}\Big) - kC_{\mathsf{A},1}$$
$$\frac{dC_{\mathsf{A},2}(t)}{dt} = \frac{F}{V_2}\Big(C_{\mathsf{A},1} - C_{\mathsf{A},2}\Big) - kC_{\mathsf{A},2}$$
$$\frac{dC_{\mathsf{A},3}(t)}{dt} = \frac{F}{V_3}\Big(C_{\mathsf{A},2} - C_{\mathsf{A},3}\Big) - kC_{\mathsf{A},3}$$

1. Describe how you would ordinarily solve the system of equations $f(\mathbf{x}) = \mathbf{0}$ where $\mathbf{x} = \big[C_{\mathsf{A},1}, C_{\mathsf{A},2}, C_{\mathsf{A},3}\big]$ to find the steady-state values of $\big[C_{\mathsf{A},1}, C_{\mathsf{A},2}, C_{\mathsf{A},3}\big]$. Now use a computer-based ODE solver to integrate the equations to steady state and report the steady-state values of $\mathbf{x} = \big[C_{\mathsf{A},1}, C_{\mathsf{A},2}, C_{\mathsf{A},3}\big]$ and the overall conversion.

You are tasked with increasing the conversion of the overall system. You are investigating two possibilities: adding a recycle stream, and operating at larger tank volumes.

2. What are the new steady state values of $\mathbf{x} = \big[C_{\mathsf{A},1}, C_{\mathsf{A},2}, C_{\mathsf{A},3}\big]$ and the overall conversion when $V_1 = V_2 = V_3 = 150$ L? Why does operating at a larger tank volumes increase conversion? You can answer this question even if you haven't taken the reactor design course yet - *hint*: interpret the above modelling equations.

3. Let's take a look at what happens when a disturbance enters the system. Simulate the case when a step change of +1.2 mol/L in $C_{\mathsf{A},0}$ is made at time $t = 50$ (i.e. the inlet concentration is 2.7 mol/L for 50 minutes, then increased to 3.9 mol/L for the rest of the time). Choose suitable initial conditions so that your system is roughly at steady state before $t = 50$, continue the simulation on until the new steady state is reached.

   What do you notice about the shape of the concentration *vs* time trajectories from each reactor (superimpose the 3 trajectories on the same plot).

Adding a recycle stream from the final reactor to the first or second reactor is an exercise that you can complete on your own. It leads to a counter-intuitive phenomenon: you'd expect an increase in conversion, but ....

Solution

1. Ordinarily one would use Newton's method to solve a system of N non-linear equations in N unknowns. In this case however, the system of equations are linear, so we could use any of the linear algebra methods, such as Gauss reduction, LU decomposition, Jacobi method, *etc*.

   Using the `ode` solvers in computer software and integrating for a *long time* from any starting point will get us the steady state values of $\mathbf{x} = \big[C_{\mathsf{A},1}, C_{\mathsf{A},2}, C_{\mathsf{A},3}\big] = \big[1.720, 1.096, 0.6982\big]$ mol/L and the overall conversion is

$$X_{\text{overall}} = \frac{C_{\mathsf{A},0} - C_{\mathsf{A},3}}{C_{\mathsf{A},0}} = \frac{2.7 - 0.6982}{2.7} = \mathbf{74.1}\%$$

2. Changing the volumes to 150 L in the code and running the integrator to steady state gives $\big[C_{\mathsf{A},1}, C_{\mathsf{A},2}, C_{\mathsf{A},3}\big] = \big[1.456, 0.785, 0.423\big]$ mol/L, and an overall conversion of $\mathbf{84.3}\%$.

   Operating at larger tank volumes can be explained as follows. Consider the first tank at steady state and calculate its conversion. We let $\tau = \frac{F}{V}$ for convenience (but it also has an interpretation, called *residence time*, which you will/have learn about in reactor design).

$$0 = \frac{F}{V}\Big(C_{\mathsf{A},0} - C_{\mathsf{A},1}\Big) - kC_{\mathsf{A},1}$$
$$(k + \tau)\, C_{\mathsf{A},1} = \tau C_{\mathsf{A},0}$$
$$X_1 = \frac{C_{\mathsf{A},0} - C_{\mathsf{A},1}}{C_{\mathsf{A},0}} = 1 - \frac{C_{\mathsf{A},1}}{C_{\mathsf{A},0}} = 1 - \frac{C_{\mathsf{A},0}}{C_{\mathsf{A},0}} \cdot \frac{\tau}{k + \tau}$$
$$X_1 = \frac{\tau}{k + \tau}$$

And you can show for tanks 2 and 3 that the same equation holds:

$$X_2 = \frac{\tau}{k+\tau}$$
$$X_3 = \frac{\tau}{k+\tau}$$

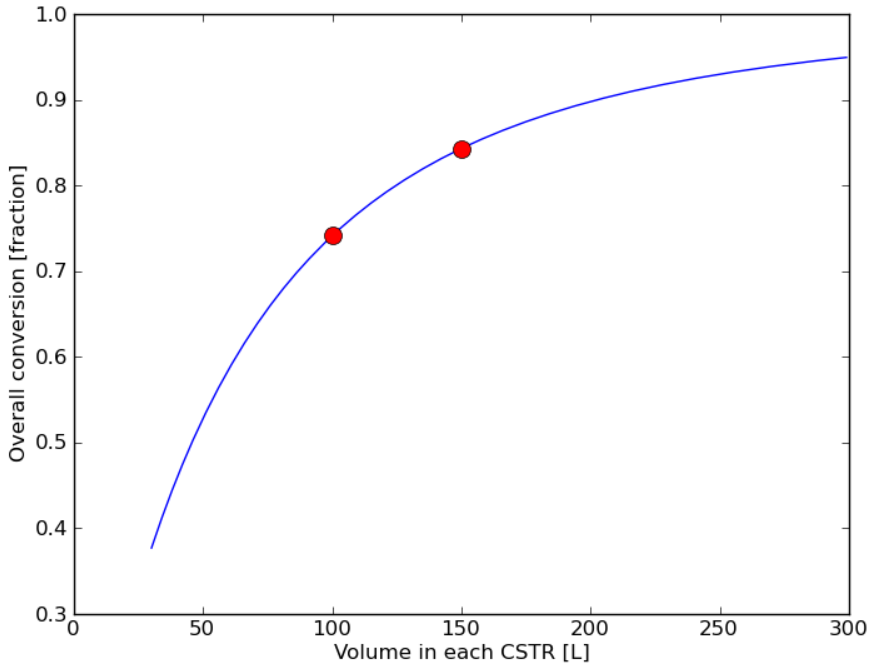
So the outlet concentration is the product of all conversions:

$$C_{\mathrm{A},3} = X_3 C_{\mathrm{A},2} = X_3 X_2 C_{\mathrm{A},1} = X_3 X_2 X_1 C_{\mathrm{A},0} = \left(\frac{\tau}{k+\tau}\right)^3 C_{\mathrm{A},0}$$

Overall conversion can be increased in any way that decreases $C_{\mathrm{A},3}$:

- increasing the rate constant, k (maybe with a catalyst?)
- making the value of τ smaller, which implies using a lower inlet flow rate, F or using increased volumes, V.

However, notice the diminishing returns for the last bullet point above, because the term appears both in the numerator and denominator. This can be seen in the plot here of reactor volume (for each CSTR) against overall conversion:

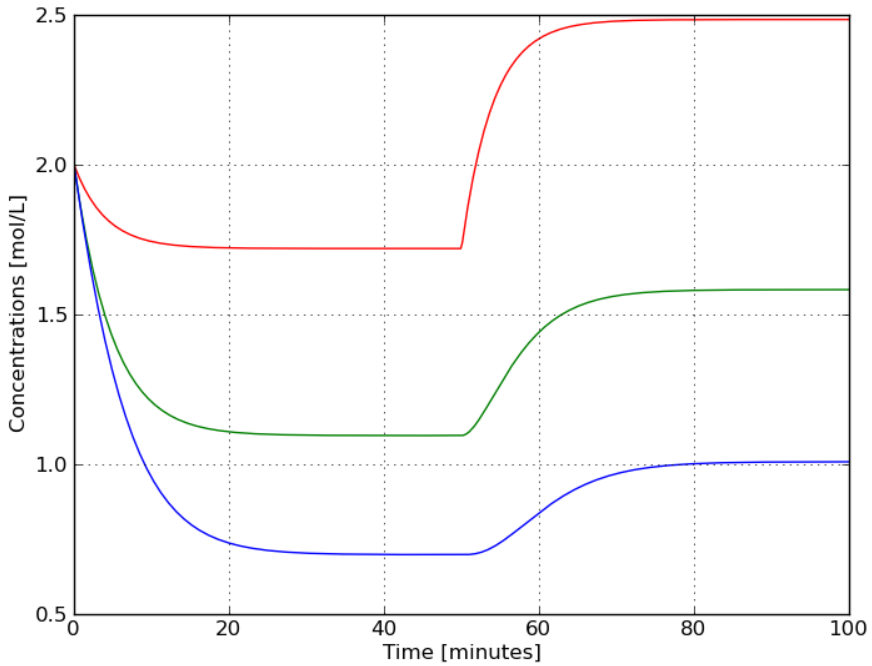


3. The trajectories for this disturbance, which enters the first tank, are shown below.

   The key point is that the first CSTR response has a sharp, rising edge (discontinuity) as the trajectory rises immediately when the disturbance hits. The second reactor has a slightly smoother response, while the third reactor responds even slower still. This is typical behaviour of all series reactors – disturbances in early reactors get "washed out" in later reactors.

The following Python code was used to solve this question; MATLAB code (not available) would be similar to this.

```
import numpy as np
from scipy import integrate
from matplotlib.pylab import *
```




```
def tanks_in_series(t, y):
    """
    Dynamic balance for a series of 3 CSTRs

    C_A1 = y[0] = CA leaving tank 1 [mol/L]
    C_A2 = y[1] = CA leaving tank 2 [mol/L]
    C_A3 = y[2] = CA leaving tank 3 [mol/L]

    Returns dy/dt = [F/V*(C_{A,in} - C_{A,out}) - k*C_{A,out}]
                    repeated for each tank
    """
    F = 15.8                # L/min
    CA_in_1 = 2.7           # mol/L
    V1 = V2 = V3 = 100.0    # L     (changed in part 1 and 2)
    k = 0.09                # 1/min

    # Assign some variables for convenience of notation
    CA1 = y[0]
    CA2 = y[1]
    CA3 = y[2]

    CA_in_2 = CA1
    CA_in_3 = CA2

    # Activate this for part 3
    if t > 50:
        CA_in_1 = 2.7 + 1.2

    # Output from ODE function must be a COLUMN vector, with n rows
    n = len(y)      # 2: implies we have two ODEs
    dydt = np.zeros((n,1))
    dydt[0] = F/V1*(CA_in_1 - CA1) - k*CA1
```

```
    dydt[1] = F/V2*(CA_in_2 - CA2) - k*CA2
    dydt[2] = F/V3*(CA_in_3 - CA3) - k*CA3
    return dydt

# Set the integrator
r = integrate.ode(tanks_in_series).set_integrator('vode', method='bdf')

# Set the time range
t_start = 0.0
t_final = 100.0
delta_t = 0.1
# Number of time steps: 1 extra for initial condition
num_steps = np.floor((t_final - t_start)/delta_t) + 1

# Set initial condition(s): for integrating variable and time!
CA1_t_zero = 2  # mol/L
CA2_t_zero = 2  # mol/L
CA3_t_zero = 2  # mol/L
r.set_initial_value([CA1_t_zero, CA2_t_zero, CA3_t_zero], t_start)

# Additional Python step: create vectors to store trajectories
t = np.zeros((num_steps, 1))
CA1 = np.zeros((num_steps, 1))
CA2 = np.zeros((num_steps, 1))
CA3 = np.zeros((num_steps, 1))

t[0] = t_start
CA1[0] = CA1_t_zero
CA2[0] = CA2_t_zero
CA3[0] = CA3_t_zero

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

    # Store the results to plot later
    t[k] = r.t
    CA1[k] = r.y[0]
    CA2[k] = r.y[1]
    CA3[k] = r.y[2]
    k += 1

# All done!  Plot the trajectories:
fig = figure()
plot(t, CA1, 'r')
plot(t, CA2, 'g')
plot(t, CA3, 'b')
xlabel('Time [minutes]')
ylabel('Concentrations [mol/L]')
grid('on')
fig.savefig('../images/disturbance-propogation.png')

k = 0.09                    # 1/min
F = 15.8                    # L/min
CA_in_1 = 2.7               # mol/L
V = np.arange(30., 300, 1) # L
tau = F/V
outlet_conversion = 1 - np.power(tau/(k+tau),3)
fig = figure()
```

```
plot(V, outlet_conversion)
plot(V[np.where(V==100)],
     outlet_conversion[np.where(V==100)], 'ro', ms=10)
plot(V[np.where(V==150)],
     outlet_conversion[np.where(V==150)], 'ro', ms=10)
xlabel('Volume in each CSTR [L]')
ylabel('Overall conversion [fraction]')
fig.savefig('../images/volume-vs-conversion.png')
```

Question 3 [2]

Note: This question is adapted from the textbook by Bradie, listed under the Suggested readings sections of the course website.

A genetic switch is a biological mechanism to activate whether or not a particular protein product from a cell is synthesized. For example, under certain conditions a pigment protein can be synthesized, which indicates the presence/absence of another chemical (the switch).

This model has been proposed for a genetic switch:

$$\frac{dg}{dt} = 0.01s - 1.51g + 3.03\frac{g^2}{1+g^2} \qquad g(0) = 0.0$$

In this notation, the pigment protein concentration is $g(t)$ and $s(t)$ is the concentration of the chemical (the switch) that activates the gene that produces the pigment.

The $-1.51g$ term indicates the protein's natural decay rate is proportional to its concentration, while the last term describes the positive feedback the protein exerts on its own formation.

A switch has two important characteristics:

- Below a threshold value of s the gene concentration must be close to zero (i.e. little/no pigment produced means the switch is off), while above a threshold, the gene concentration, g, must be high (i.e. the switch has been turned on).
- Once turned on (i.e. high values of g), the switch must remain on, even if s is returned below the threshold value. Similarly, if the switch was never on to begin with, it must still remain "off" if s is below the threshold. This phenomenon is known as the hysteresis effect.

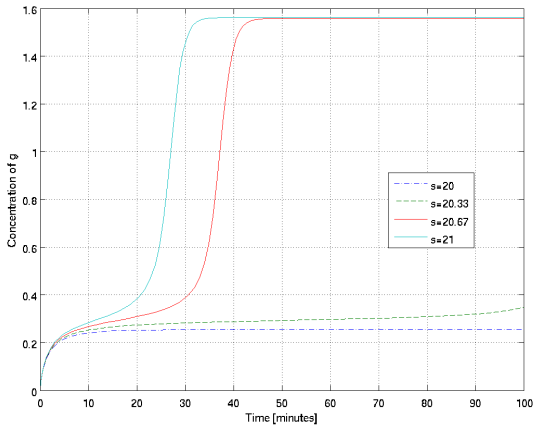
1. First we wish to find the threshold value of s and interpret what s means. Plot and label four trajectories of $g(t)$ using $s = 20.0, 20.33, 20.67, 21.0$ over a time range of 100 minutes, and describe what the plots are showing.
   - Be specific: how would you use these plots to design your genetic switch?
   - What does s mean in your design? (Hint: it has two interpretations)
2. Now we wish to verify if the hysteresis effect is present, because it is important the switch cannot be turned off. For each of the four values of s in part 1, run a simulation where $s(t > 150) = 0$ (i.e. at time $t = 150$ the concentration of s is set from its current value to zero). Show your results over a time horizon of 300 minutes and **comment on the results**.

Solution

1. Trajectories, at different levels of s, are simulated over a time scale of 100 minutes in the plot.

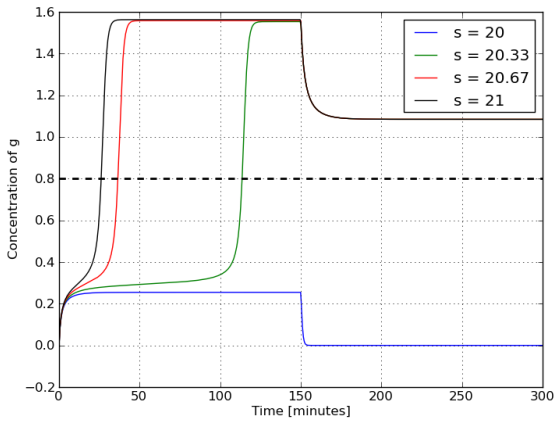
   s must be at least greater than 20.0 for the switch to be activated; values at or below 20.0 never activate the switch, while values above this will activate it, though slowly at first (e.g 20.33 takes a long time to show). So

one interpretation of s is that it is the threshold level above which we can detect it. The other interpretation is that the higher the value of s, the faster it will be detected by the gene. So s affects both the threshold of the switch, and the time taken to activate it.



2. The hysteresis effect is present. The simulations show that for all four levels of s, that if the switch is activated ($s > 20.0$) that the gene will remain present, even if s is removed.

   We can set an arbitrary threshold on the gene concentration, e.g. 0.8, so that above this level we can consider the switch to be on, and below it the switch is off.



The following code was used in both parts of the question.

MATLAB code

```
function dydt = bioswitch(t, g, s)
    % Genetic switch

    if t>150
        s = 0.0;
    end

    n = length(g);
    dydt = zeros(n, 1);
    dydt(1) = 0.01*s - 1.51*g + 3.03*g^2/(1+g^2);
end
```

```
% Save and uncomment the following lines
% in a different file
% ======================================================
t_start  = 0.0;
t_final  = 300.0;
g_t_zero = 0.0;  % mol/L

s = [20.0, 20.33, 20.67 21.0];
lines = {'r-.'; 'c--'; 'b-'; 'k-'};

axis();
hold on
legend_str = {};
for i=1:length(s)
    [t, y] = ode15s(@(t,g) bioswitch(t, g, s(i)),  ...
                    [t_start t_final], g_t_zero);

    plot(t, y, lines{i})
    legend_str{end+1} = ['s=', num2str(s(i))];
end
legend(legend_str, 'Location','Best')

xlabel('Time [minutes]')
ylabel('Concentration of g')
grid('on')
```

Python code

```
import numpy as np
from scipy import integrate
from matplotlib.pylab import *

def switch(t, y, s):
    """ A genetic switch """
    g = y[0]

    # For part 2: hysteresis check
    if t > 150:
        s = 0.0

    # Output from ODE function must be a COLUMN vector, with n rows
    n = len(y)
    dydt = np.zeros((n,1))
    dydt[0] = 0.01*s - 1.51*g + 3.03*g**2/(1+g**2)

    return dydt

def integrate_and_plot(s, colour):
    """ Integrate the ODE system with the given value of ``s``;
        plot the gene concentrationtrajectory with ``colour``.
    """
    r = integrate.ode(switch)
    r.set_integrator('vode', method='bdf')
    r.set_f_params(s)   # Send parameter ``s`` to the ODE

    # Set the time range and number of time steps
    t_start, t_final, delta_t = 0.0, 300.0, 0.1
    num_steps = np.floor((t_final - t_start)/delta_t) + 1

    # Set initial condition(s): for integrating variable and time!
```

```
    g_t_zero = 0.0  # mol/L
    r.set_initial_value([g_t_zero], t_start)

    # Additional Python step: create vectors to store trajectories
    t = np.zeros((num_steps, 1))
    g = np.zeros((num_steps, 1))
    t[0], g[0] = t_start, g_t_zero

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

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

    # All done!  Plot the trajectory with given colour
    plot(t, g, colour, label = 's = %g' % s)

if __name__ == '__main__':

    fig = figure()
    integrate_and_plot(s = 20.0, colour='b')
    integrate_and_plot(s = 20.33, colour='g')
    integrate_and_plot(s = 20.67, colour='r')
    integrate_and_plot(s = 21.0, colour='k')

    # An arbitrarily chosen threshold
    axhline(y=0.8, color='k', linestyle='--', linewidth=2)

    xlabel('Time [minutes]')
    ylabel('Concentration of g')
    grid('on')
    legend(loc="best")
    fig.savefig('../images/genetic-switch.png')
```

Question 4 [2]

Note: This question continues on from question 5, assignment 5. It was adapted from a question in the final exam, 2009, worth 3+1 out of 50 marks.

Consider a well-mixed tank containing species A at concentration $C_\mathsf{A} = 1$ mol/L. There is no inlet or outlet from the tank, i.e. it is a batch system. A fluid of constant density is considered throughout this question. The reactor is filled to 5.2 m^3.

A catalyst is introduced in the tank at time $t = 0$ minutes to initiate the reaction $\mathsf{A} \rightarrow \mathsf{B} + \mathsf{C}$. Due to quick catalyst deactivation, the dynamic equation describing the evolution of concentration C_A in the tank is given by:

$$\frac{\mathrm{d}C_\mathsf{A}(t)}{\mathrm{d}t} = -k\, C_\mathsf{A}(t)\, e^{-\frac{t}{2}}$$

with $k = 1\ \text{min}^{-1}$.

1. Determine the concentration of A in the reactor at time $t = 20$ minutes using the classical Runge-Kutta method with step size $h = 10$ minutes.

2. Compare and discuss your answer with respect to the analytical solution:

$$C_{\mathsf{A}}(t) = C_{\mathsf{A}}(t=0)\ \exp\left[2\,k\left(e^{-\frac{t}{2}}-1\right)\right]$$

Solution

For notation simplicity, let us denote $y = C_{\mathsf{A}}$ and $f(t,y) = \dfrac{\mathrm{d}y(t)}{\mathrm{d}t} = -k\,y(t)\,e^{-\frac{t}{2}}$. $y_0 = y(t=0) = 1$ and the step size $h = 10$.

Classical 4th-order Runge-Kutta formula:

$$\begin{aligned}
y_1 &= y_0 + \frac{1}{6}[K_1 + 2K_2 + 2K_3 + K_4]h \\
K_1 &= f(t_0, y_0) = f(0,1) = -1 \times 1e^0 = -1 \\
K_2 &= f(t_0 + \frac{1}{2}h, y_0 + \frac{1}{2}hK_1) = f(5,-4) = 0.328 \\
K_3 &= f(t_0 + \frac{1}{2}h, y_0 + \frac{1}{2}hK_2) = f(5,2.642) = -0.217 \\
K_4 &= f(t_0 + h, y_0 + hK_3) = f(10,-1.17) = 0.008 \\
y_1 &= 1 + \frac{1}{6}\Big[-1 + 2\times 0.328 + 2\times -0.217 + 0.008\Big] \times 10 = \mathbf{-0.283}
\end{aligned}$$

Does the result make sense? We ended up a negative concentration! Let us double-check with the analytical solution. The actual value of $y(t=10)$ using the analytical solution is 0.1372 [mol/L]. So why did we get a terribly wrong answer using the Runge-Kutta? The answer lies in the stability of ODE methods. Some ODE methods have certain stability restrictions that impose a maximum length on the integration step size. Here we chose $h = 10$ which is too large for our system, thereby leading to an *unstable solution*. A workaround is to choose a very smaller step size and reach $t = 10$ in multi-steps instead. This is not requested in this question though. So, let us continue with $h = 10$ and see what we get as $y(t_2 = 20)$ using the obtained $y(t_1 = 10)$

$$\begin{aligned}
y_2 &= y_1 + \frac{1}{6}[K_1 + 2K_2 + 2K_3 + K_4]h \\
K_1 &= f(t_1, y_1) = f(10,-0.283) = 0.0019 \\
K_2 &= f(t_1 + \frac{1}{2}h, y_1 + \frac{1}{2}hK_1) = f(15,-0.2735) = 1.5127\times 10^{-4} \\
K_3 &= f(t_1 + \frac{1}{2}h, y_1 + \frac{1}{2}hK_2) = f(15,-0.2822) = 1.5608\times 10^{-4} \\
K_4 &= f(t_1 + h, y_1 + hK_3) = f(20,-0.2814) = 1.2776\times 10^{-5} \\
y_2 &= -0.283 + \frac{1}{6}\Big[0.0019 + 2\times 1.5127\times 10^{-4} + 2\times 1.5608\times 10^{-4} + 1.2776\times 10^{-5}\Big] \times 10 = \mathbf{-0.2788}
\end{aligned}$$

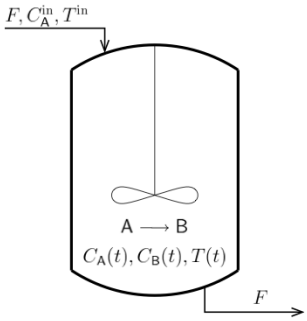
The actual value of $X(t=20)$ using the analytical solution is 0.1353 [mol/L] (compare to the above approximate solution!)

Question 5 [3]

Consider the CSTR shown below, where an irreversible reaction of the form $\mathsf{A} \rightarrow \mathsf{B}$ takes place.

Model parameters

- $C_{\mathsf{A}}^{\mathrm{in}} = 0.79\mathrm{kmol\ m^{-3}}$
- $T^{\mathrm{in}} = 352.6\mathrm{K}$



- $F = 0.2\text{m}^3\ \text{min}^{-1}$
- $V = 0.1\text{m}^3$
- $k_0 = 7.2 \times 10^{10}\text{min}^{-1}$
- $E = 8.314 \times 10^4\text{kJ kmol}{-1}$
- $\Delta H = -4.78 \times 10^4\text{kJ kmol}^{-1}$
- $C_p = 0.239\text{kJ kg}^{-1}\ \text{K}^{-1}$
- $\rho = 1000\text{kg m}^{-3}$
- $R = 8.314\text{kJ kmol}^{-1}\ \text{K}^{-1}$

1. Derive, by hand, the differential equations describing the evolution of the concentration of species A, $C_{\mathsf{A}}(t)$, as well as of the temperature of the outlet stream, $T(t)$, based on the following assumptions:
   - The concentrations of A in the inlet stream is $C_{\mathsf{A}}^{\text{in}}$, the inlet and outlet volumetric flow rates are constant and both equal to F, and the temperature of the inlet stream is T^{in}.
   - The reaction has first-order kinetics, with the dependence of the reaction rate r (moles of A consumed per unit volume and per unit time) on the temperature given by $r = k_0\, e^{-\frac{E}{RT}}\, C_{\mathsf{A}}$.

     where E is the activation energy; R, the universal gas constant; T, the temperature at which the reaction is taking place; and k_0, a constant.
   - The heat of reaction is ΔH and no heat is lost or added to the reactor, other than through the liquid streams.
   - The density and specific heat capacity of the fluid are ρ and C_p respectively.
2. For the values of the process parameters listed in the table and the following initial conditions $C_{\mathsf{A}}(0) = 0.7\ \text{kmol/m}^3$ and $T(0) = 355\ \text{K}$, compute, by hand, the value of $\left[C_{\mathsf{A}}(t),\ T(t)\right]$ at $t = 0.2$, using:
   - Euler's method with $h = 0.1$, and
   - Heun's predictor-corrector method with $h = 0.1$, and
   - the fourth-order Runge-Kutta method with $h = 0.2$
3. Implement these equations in computer software and compare the integration by hand with the computer version at $t = 0.2$.

Solution

1. First note that the rate of consumption of component A is $-r_{\mathsf{A}} = kC_{\mathsf{A}} = k_0 \exp\left(-\frac{E}{RT}\right) C_{\mathsf{A}}$.

A mole balance on A can be done to determine the evolution of the concentration of species A:

$$\frac{dVC_\mathsf{A}}{dt} = FC_\mathsf{A}^{\text{in}} - FC_\mathsf{A} - r_\mathsf{A}V$$
$$V\frac{dC_\mathsf{A}}{dt} = FC_\mathsf{A}^{\text{in}} - FC_\mathsf{A} - k_0 \exp\left(-\frac{E}{RT}\right) C_\mathsf{A}V$$

On the other hand, an energy balance on the reactor can be done to determine the evolution of the temperature in the reactor:

$$\frac{d(\rho V C_p(T - T_{\text{ref}}))}{dt} = \rho C_p F(T_{\text{in}} - T_{\text{ref}}) - \rho C_p F(T - T_{\text{ref}}) + (-\Delta H) r_\mathsf{A} V$$
$$\rho V C_p \frac{dT}{dt} = \rho C_p F(T_{\text{in}} - T) - \Delta H k_0 \exp\left(-\frac{E}{RT}\right) C_\mathsf{A}V$$

2. Substituting the values from the table in the dynamic balances gives:

$$\frac{dy_1}{dt} = = f_1(t, \mathbf{y}) = 1.58 - 2y_1 - 7.2 \times 10^{10}\, e^{-\frac{10^4}{y_2}} y_1$$
$$\frac{dy_2}{dt} = = f_2(t, \mathbf{y}) = 705.2 - 2y_2 + 1.44 \times 10^{13}\, e^{-\frac{10^4}{y_2}} y_1,$$

with $y_1(t) = C_\mathsf{A}(t)$ and $y_2(t) = T(t)$.

- **Explicit Euler method**

  From t = 0 to 0.1

  At $t = 0$, $\boldsymbol{\phi} = \mathbf{f}(0, \mathbf{y}^{(0)}) = \begin{bmatrix} 0.1506 \\ 1.0859 \end{bmatrix}$. The estimated solution at $t = 0.1$ is:

$$\mathbf{y}^{(1)} = \mathbf{y}^{(0)} + h\boldsymbol{\phi} = \begin{bmatrix} 0.7 \\ 355 \end{bmatrix} + 0.1 \begin{bmatrix} 0.1506 \\ 1.0859 \end{bmatrix} = \begin{bmatrix} 0.7151 \\ 355.1086 \end{bmatrix}$$

  From t = 0.1 to 0.2:

  At $t = 0.1$, $\boldsymbol{\phi} = \mathbf{f}(0.1, \mathbf{y}^{(1)}) = \begin{bmatrix} 0.1196 \\ 1.0473 \end{bmatrix}$. The estimated solution at $t = 0.2$ is:

$$\mathbf{y}^{(2)} = \mathbf{y}^{(1)} + h\boldsymbol{\phi}$$
$$= \begin{bmatrix} 0.727 \\ 355.2 \end{bmatrix}$$

  Overall, we thus obtain the following solution:

  $C_A(0.2) = 0.727\ \text{kmol/m}^3$ and $\mathrm{T}(0.2) = 355.2\ \text{K}$

- **Heun's predictor-corrector method**

  From t = 0 to 0.1

  The first step of Heun's method (predictor) uses the Euler slope, calculated above. The corrector step

calculates the slope at $t = 0.1$ and the Heun slope is just the average of the two.

$$\hat{\mathbf{y}}^{(1)} = \begin{bmatrix} 0.7151 \\ 355.1086 \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{\phi}_{\text{Heun}} &= 0.5 \begin{bmatrix} 0.1506 \\ 1.0859 \end{bmatrix} + 0.5\mathbf{f}(0.1, \hat{\mathbf{y}}^{(1)}) \\ &= 0.5 \begin{bmatrix} 0.1506 \\ 1.0859 \end{bmatrix} + 0.5 \begin{bmatrix} 0.1196 \\ 1.0473 \end{bmatrix} \\ &= \begin{bmatrix} 0.1351 \\ 1.0667 \end{bmatrix} \end{aligned}$$

$$\mathbf{y}^{(1)} = \mathbf{y}^{(0)} + h\boldsymbol{\phi}_{\text{Heun}} = \begin{bmatrix} 0.7 \\ 355 \end{bmatrix} + 0.1 \begin{bmatrix} 0.1351 \\ 1.0667 \end{bmatrix} = \begin{bmatrix} 0.7135 \\ 355.107 \end{bmatrix}$$

From t = 0.1 to 0.2:

$$\hat{\mathbf{y}}^{(2)} = \begin{bmatrix} 0.7135 \\ 355.107 \end{bmatrix} + 0.1 \begin{bmatrix} 0.1227 \\ 1.0365 \end{bmatrix}$$

$$\hat{\mathbf{y}}^{(2)} = \begin{bmatrix} 0.72577 \\ 355.210 \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{\phi}_{\text{Heun}} &= 0.5 \begin{bmatrix} 0.1227 \\ 1.0365 \end{bmatrix} + 0.5\mathbf{f}(0.2, \hat{\mathbf{y}}^{(2)}) \\ &= 0.5 \begin{bmatrix} 0.1227 \\ 1.0365 \end{bmatrix} + 0.5 \begin{bmatrix} 0.097435 \\ 0.9851 \end{bmatrix} \\ &= \begin{bmatrix} 0.1101 \\ 1.0108 \end{bmatrix} \end{aligned}$$

$$\mathbf{y}^{(2)} = \mathbf{y}^{(1)} + h\boldsymbol{\phi}_{\text{Heun}} = \begin{bmatrix} 0.7135 \\ 355.107 \end{bmatrix} + 0.1 \begin{bmatrix} 0.1101 \\ 1.0108 \end{bmatrix} = \begin{bmatrix} 0.7245 \\ 355.21 \end{bmatrix}$$

Overall, we thus obtain the following solution:

$C_A(0.2) = 0.7245\ \text{kmol/m}^3 \quad \text{and} \quad \text{T}(0.2) = 355.21\ \text{K}$

- **4th-order Runge-Kutta method (known as the classical Runge-Kutta method)**

  From t = 0 to 0.2:

  At $t = 0$, $\mathbf{y}^{(0)} = \begin{bmatrix} 0.7 \\ 355 \end{bmatrix}$. The intermediate quantities $\mathbf{K}_1, \ldots, \mathbf{K}_4$ are now 2-dimensional vectors:

$$\mathbf{K}_1 = \begin{bmatrix} 0.1506 \\ 1.0859 \end{bmatrix}$$

$$\mathbf{K}_2 = \begin{bmatrix} 0.1196 \\ 1.0473 \end{bmatrix}$$

$$\mathbf{K}_3 = \begin{bmatrix} 0.1259 \\ 1.0269 \end{bmatrix}$$

$$\mathbf{K}_4 = \begin{bmatrix} 0.0986 \\ 0.9870 \end{bmatrix}$$

$$\boldsymbol{\phi} = \frac{1}{6}\mathbf{K}_1 + \frac{1}{3}\mathbf{K}_2 + \frac{1}{3}\mathbf{K}_3 + \frac{1}{6}\mathbf{K}_4 = \begin{bmatrix} 0.1234 \\ 1.0369 \end{bmatrix}$$

  The estimated solution at $t = 0.2$ is:

$$\mathbf{y}^{(1)} = \mathbf{y}^{(0)} + h\boldsymbol{\phi} = \begin{bmatrix} 0.7247 \\ 355.2074 \end{bmatrix}$$

Overall, we thus obtain the following solution:

$C_A(0.2) = 0.725\ \mathrm{kmol/m^3}$ and $\mathrm{T}(0.2) = 355.2\ \mathrm{K}$, which is in good agreement with the solution found earlier with the explicit Euler method and Heun's method.

3. Integrating the ODE equations with MATLAB's `ode45` function, or Python's `integrate` function gave the following results at $t = 0.2$: $\mathbf{y} = [0.72468, 355.2073]$.

   $C_A(0.2) = 0.725\ \mathrm{kmol/m^3}$ and $\mathrm{T}(0.2) = 355.2\ \mathrm{K}$, which agrees very well with the Runge-Kutta solution, and less well with the Heun and Euler methods.

END