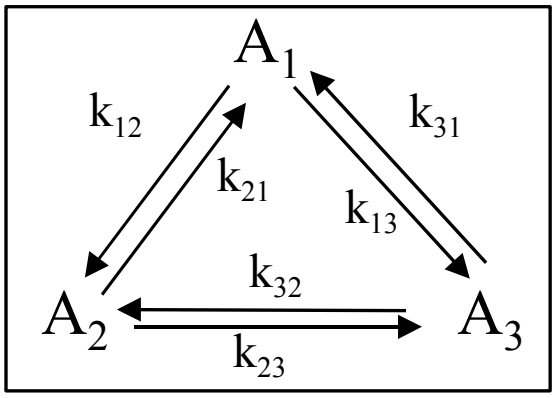
Systems of Linear First Order Ordinary Differential Equations
Example Problems

David Keffer
Department of Chemical Engineering
University of Tennessee
Knoxville, TN 37920

Last Updated: September 24, 2001

Example 1. Transient Behavior of a Batch Reactor

Consider that you have a three-component reactive mixture in a batch reactor, all undergoing reversible reactions, as pictured below:



In this picture, the A's are concentrations of the three species and the k's are rate constants. An example of this system is the kinetic equilibrium between para-, meta-, and ortho-xylene.

Now suppose we want to know what the concentration is as a function of time. We can write the mass balances for each component. There are no in and out terms (the reactor is a batch reactor). There is only the accumulation term and the reaction terms. Also, assume each reaction is first order in concentration.

$$\begin{aligned}
\frac{dA_1}{dt} &= -k_{12}A_1 + k_{21}A_2 - k_{13}A_1 + k_{31}A_3 \\
\frac{dA_2}{dt} &= -k_{21}A_2 + k_{12}A_1 - k_{23}A_2 + k_{32}A_3 \\
\frac{dA_3}{dt} &= -k_{31}A_3 + k_{13}A_1 - k_{32}A_3 + k_{23}A_2
\end{aligned} \tag{2.1}$$

We can gather like terms and rearrange the right hand side:

$$\begin{aligned}
\frac{dA_1}{dt} &= -(k_{12} + k_{13})A_1 + k_{21}A_2 + k_{31}A_3 \\
\frac{dA_2}{dt} &= k_{12}A_1 - (k_{21} + k_{23})A_2 + k_{32}A_3 \\
\frac{dA_3}{dt} &= k_{13}A_1 + k_{23}A_2 - (k_{31} + k_{32})A_3
\end{aligned} \tag{2.2}$$

and we change this system of equations into matrix & vector form:

$$\frac{d\underline{A}}{dt} = \underline{\underline{X}}\,\underline{A} \tag{2.3}$$

where

$$\underline{\underline{X}} = \begin{bmatrix} -(k_{12}+k_{13}) & k_{21} & k_{31} \\ k_{12} & -(k_{21}+k_{23}) & k_{32} \\ k_{13} & k_{23} & -(k_{31}+k_{32}) \end{bmatrix}$$
$$\underline{A} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} \tag{2.4}$$

The matrix $\underline{\underline{X}}$ is singular. It has a determinant of zero and a rank of 2. Therefore, it has one zero value eigenvalue. Nevertheless, the matrix has three distinct real eigenvalues, $\lambda_1, \lambda_2, \lambda_3$. Corresponding to each of these eigenvalues is a real, distinct eigenvector, $\underline{w}_{c,1}, \underline{w}_{c,2}, \underline{w}_{c,3}$. The eigenvectors of $\underline{\underline{X}}^*$ are known as the eigenrows of $\underline{\underline{X}}$: $\underline{w}_{r,1}, \underline{w}_{r,2}, \underline{w}_{r,3}$.

The solution is then:

$$\underline{A}(t) = \sum_{i=1}^{n} \left\{ \frac{(\underline{w}_{r,i} \cdot \underline{A}_o)}{(\underline{w}_{r,i} \cdot \underline{w}_{c,i})} \cdot \underline{w}_{c,i} \cdot \exp[\lambda_i (t - t_o)] \right\} \tag{2.5}$$

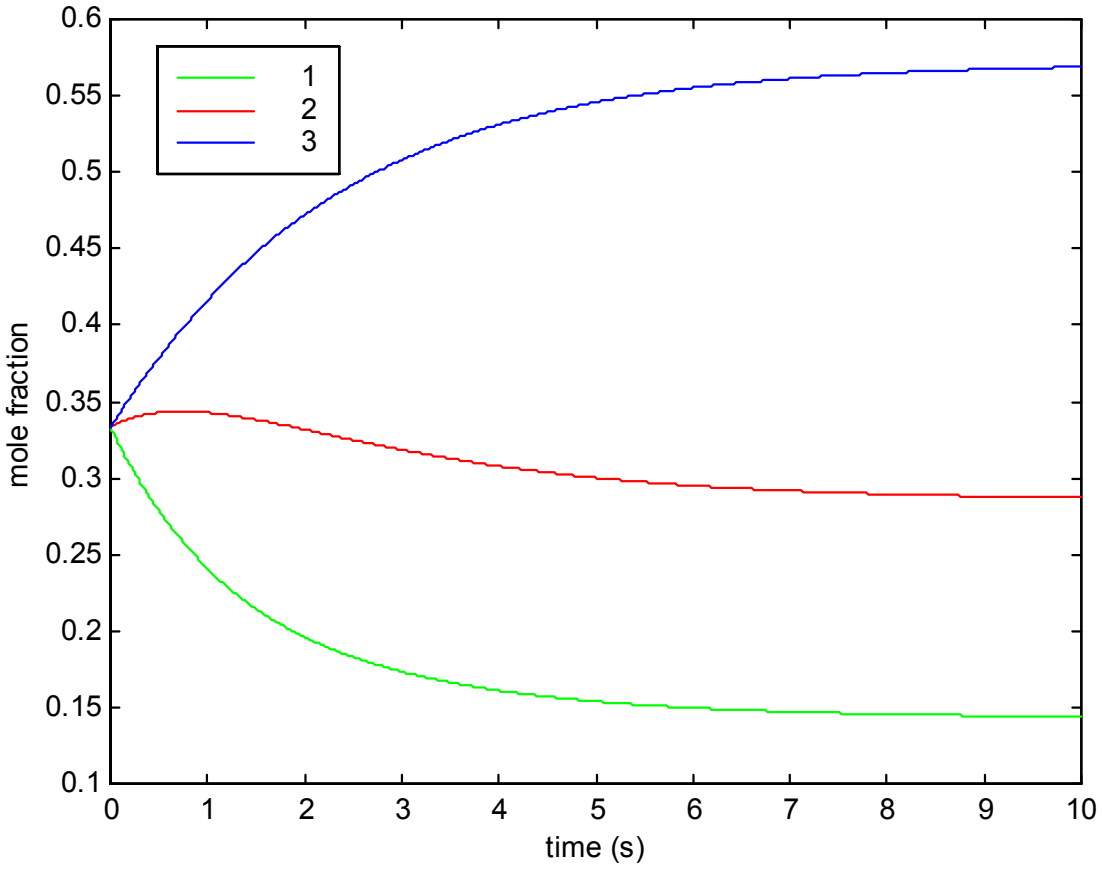
In order to obtain numerical values for this problem, we would have to first obtain the eigenvalues, eigenvectors, and eigenrows. This can be done numerically, using routines discussed in the section on numerical methods for solving systems of linear algebraic equations.

We provide a Matlab code to solve this problem on the following page.

Matlab code to solve Example 1.

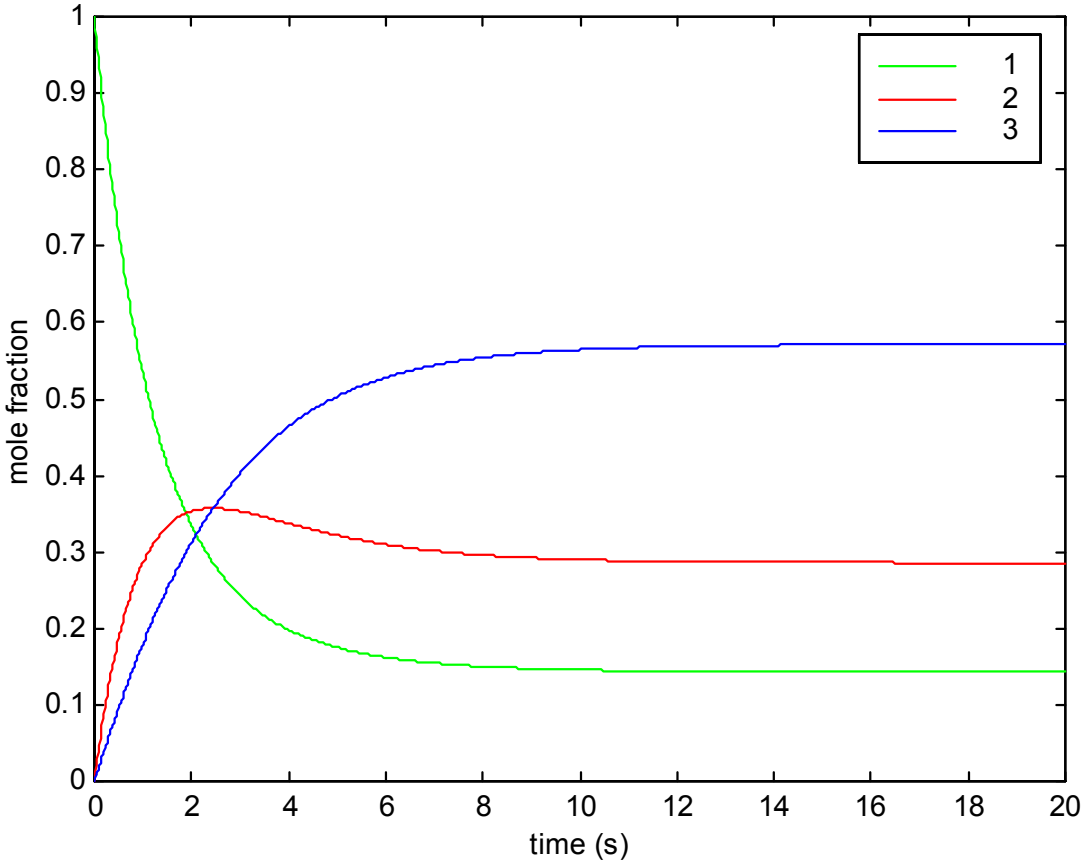
```
%
%  Solution to the Batch Reactor Problem
%
%  David Keffer
%  Department of Chemical Engineering
%  University of Tennessee
%  Knoxville Tennessee
%  September, 2001
%
% input
k12 = 0.50;
k21 = 0.25;
k13 = 0.20;
k31 = 0.05;
k23 = 0.30;
k32 = 0.15;
A = [ (-k13-k12), k21, k31; k12, (-k21-k23), k32; k13, k23, (-k31-k32)];
yo = [1.0/3.0;  1.0/3.0; 1.0/3.0];
to = 0.0;
tf = 10.0;
n = max(size(A));
% eigenvalues and eigenvectors
[wcol,lambdac] = eig(A);
wcolinv = inv(wcol);
% set up discretized solution grid
npoints = 1000;
dt = (tf-to)/npoints;
for i = 1:1:npoints+1
   tp(i) = (i-1)*dt +to;
end
% calculate solution
explambda = zeros(n,n);
yp = zeros(n,npoints);
for i = 1:1:npoints+1
   for j = 1:n
      explambda(j,j) = exp(lambdac(j,j)*(tp(i) - to) );
   end
   yp(:,i) = wcol*explambda*wcolinv*yo;
end
%plot
for j = 1:1:n
  if (j==1)
     plot (tp,yp(j,:),'g-'), xlabel( 't' ), ylabel ( 'y' )
  elseif (j==2)
     plot (tp,yp(j,:),'r-'), xlabel( 't' ), ylabel ( 'y' )
  elseif (j==3)
     plot (tp,yp(j,:),'b-'), xlabel( 't' ), ylabel ( 'y' )
  end
  hold on
end
hold off
```

Plot of Solution to Example 1: Transient Behavior in a Batch Reactor



Comments:

At all times, the sum of the mole fractions is unity. The initial condition is y_o = [1/3 1/3 1/3]. The steady state equilibrium appears to be about y = [0.143 0.286 0.571].



Comments:

At all times, the sum of the mole fractions is unity. The initial condition is y_o = [1 0 0]. The steady state equilibrium appears to be about y = [0.143 0.286 0.571]. This is the same steady state composition as was obtained for the other initial condition, plotted on the previous page. The equilibrium composition is independent of the initial composition, in this case. This is generally true, except where multiple steady states exist.

Example 2. Chemical Reaction Equilibria with addition/deletion of components $\underline{b}(x) \neq \underline{0}$

We can add constant flowrates to our system by adding a constant term to the ODEs.

$$\begin{aligned}
\frac{dA_1}{dt} &= -(k_{12}+k_{13})A_1 + k_{21}A_2 + k_{31}A_3 + F_1 \\
\frac{dA_2}{dt} &= k_{12}A_1 - (k_{21}+k_{23})A_2 + k_{32}A_3 + F_2 \\
\frac{dA_3}{dt} &= k_{13}A_1 + k_{23}A_2 - (k_{31}+k_{32})A_3 + F_3
\end{aligned} \tag{4.1}$$

The flowrates can either be feed or effluent, depending upon the sign. To keep the problem simple, let's enforce conservation of volume by stipulating that

$$\sum_{i=1}^{3} F_i = 0 \tag{4.2}$$

and we change this system of equations into matrix & vector form:

$$\frac{d\underline{A}}{dt} = \underline{\underline{X}}\,\underline{A} + \underline{F} \tag{4.3}$$

We know the solution to the nonhomogeneous equation is

$$\underline{A}_{nh}(t) = \underline{\underline{W}}_c \exp\left[\,\underline{\underline{\Lambda}}(t)\,\right]\left[\exp\left[\underline{\underline{\Lambda}}(t_o)\,\right]^{-1}\underline{\underline{W}}_c{}^{-1}\underline{A}_o - \underline{u}(t_o) + \underline{u}(t)\right] \tag{4.4}$$

The only question is the form of u.

$$\underline{u} = \int \exp\left[-\,\underline{\underline{\Lambda}}(x)\,\right]\underline{\underline{W}}_c{}^{-1}\underline{b}(x)\,dx \tag{3.23}$$

Since, for this problem, b is a constant, all the x functionality lies in $\exp\left[-\,\underline{\underline{\Lambda}}(x)\,\right]$

$$\underline{u}(t) = -\left[\int^{t} \exp\left[-\,\underline{\underline{\Lambda}}(x)\,\right]dx\right]\underline{\underline{W}}_c{}^{-1}\underline{b} \tag{4.5}$$

$$\underline{u}(t) = -\left[\underline{\underline{\Lambda}}^{-1}\exp\left[-\underline{\underline{\Lambda}}(t)\right]\right]\underline{\underline{W}}_c^{-1}\underline{b} \tag{4.6}$$

Plugging back into the solution we find

$$\underline{A}_{nh}(t) = \underline{\underline{W}}_c \exp\left[\underline{\underline{\Lambda}}(t)\right]\begin{bmatrix}\exp\left[\underline{\underline{\Lambda}}(t_o)\right]^{-1}\underline{\underline{W}}_c^{-1}\underline{A}_o + \left[\underline{\underline{\Lambda}}^{-1}\exp\left[-\underline{\underline{\Lambda}}(t_o)\right]\right]\underline{\underline{W}}_c^{-1}\underline{b} \\ -\left[\underline{\underline{\Lambda}}^{-1}\exp\left[-\underline{\underline{\Lambda}}(t)\right]\right]\underline{\underline{W}}_c^{-1}\underline{b}\end{bmatrix} \tag{4.7}$$

A complication arises because one of the eigenvalues is zero. Thus there is no x-dependency in the exponential because there is no exponential. Then we would have a term of the form

$$\underline{u}(t) = -\left[\int^{t}\exp\left[-0\right]dx\right]\underline{\underline{W}}_c^{-1}\underline{b} = -\left[\int^{t}dx\right]\underline{\underline{W}}_c^{-1}\underline{b} = -[t]\underline{\underline{W}}_c^{-1}\underline{b} = -t\underline{\underline{W}}_c^{-1}\underline{b} \tag{4.8}$$

so that we would have

$$\underline{u}(t) = \underline{\underline{U}}\underline{\underline{W}}_c^{-1}\underline{b} \tag{4.9}$$

$$\underline{\underline{U}} = \begin{bmatrix} -\dfrac{\exp\left[-\lambda_1(t)\right]}{\lambda_1} & 0 & 0 \\ 0 & -\dfrac{\exp\left[-\lambda_2(t)\right]}{\lambda_2} & 0 \\ 0 & 0 & t \end{bmatrix} \tag{4.10}$$

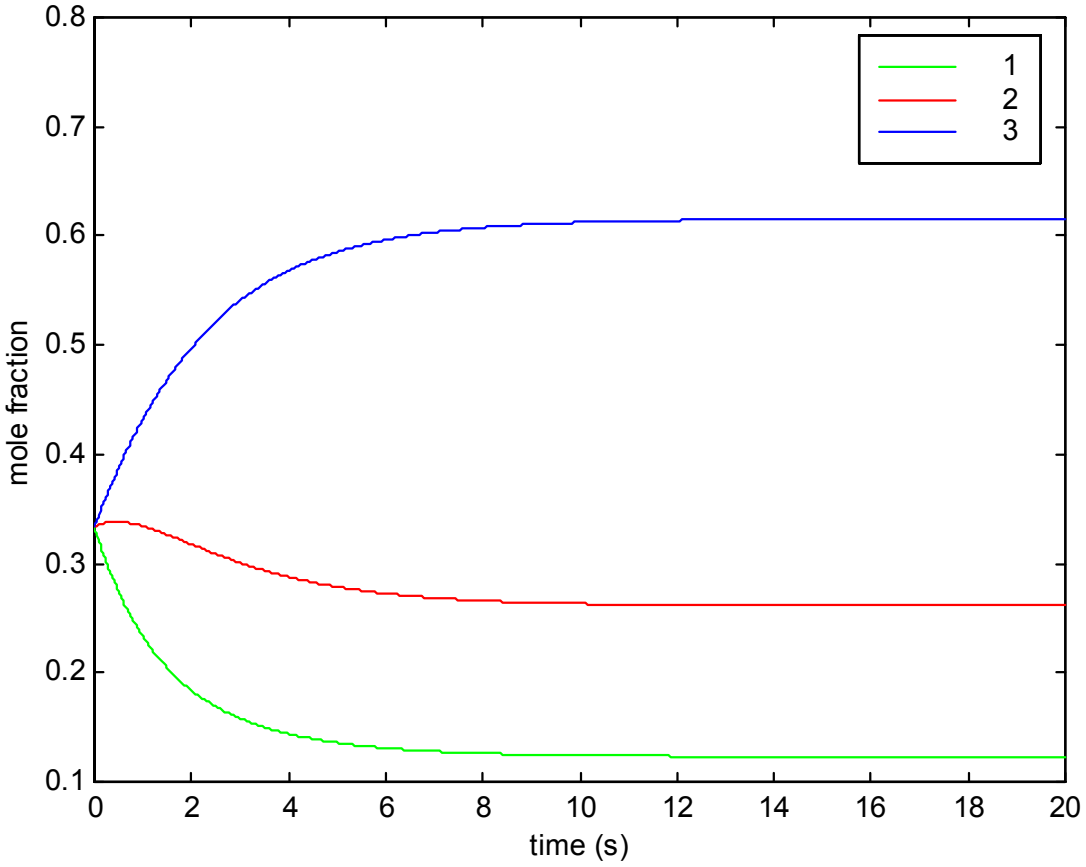
for the case where we had three eigenvalues, the first two of which are non-zero.

We provide a Matlab code to solve this problem on the following page.

Matlab code to solve Example 2.

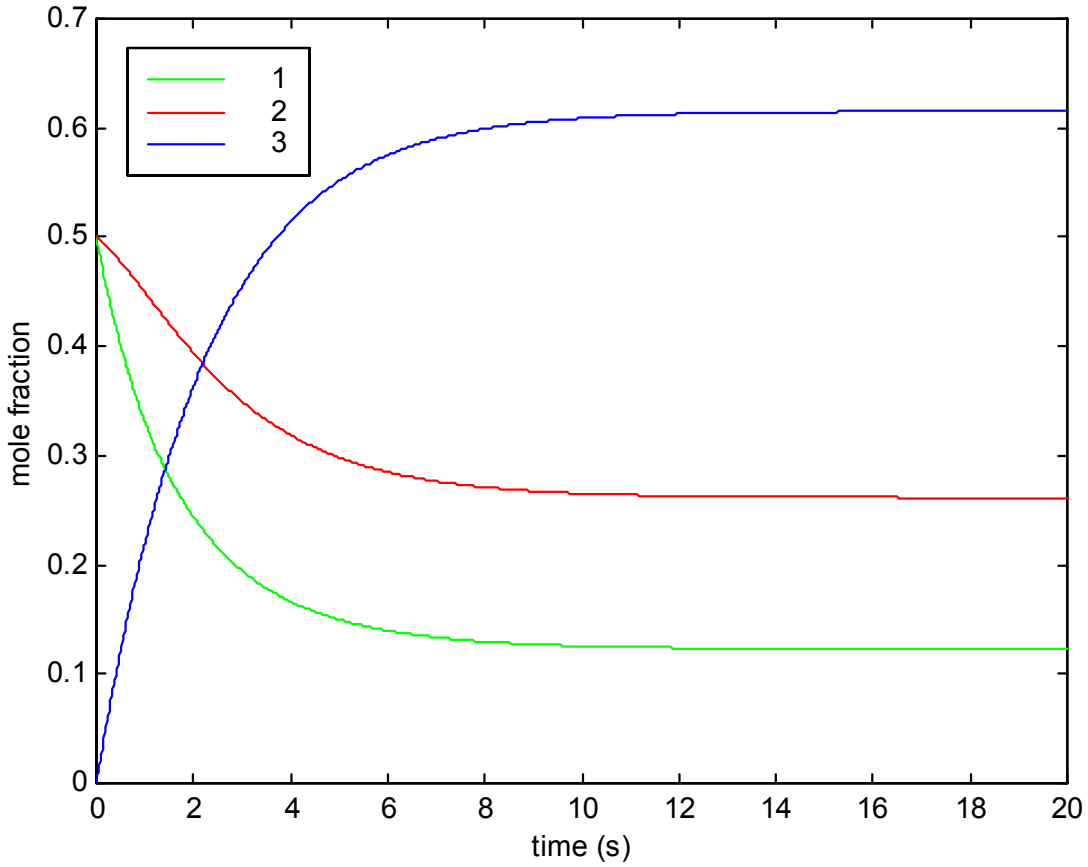
```
%
%  Solution to the Continuous Stirred Tank Reactor Problem
%
%  David Keffer
%  Department of Chemical Engineering
%  University of Tennessee
%  Knoxville Tennessee
%  September, 2001
%
tol = 1.0e-14;
% input
k12 = 0.50;
k21 = 0.25;
k13 = 0.20;
k31 = 0.05;
k23 = 0.30;
k32 = 0.15;
A = [ (-k13-k12), k21, k31; k12, (-k21-k23), k32; k13, k23, (-k31-k32)];
n = max(size(A));
yo = [1.0/3.0;  1.0/3.0; 1.0/3.0];
%yo = [1.0;  0.0; 0.0];
% constant terms in ODE
b = [-0.01;  -0.01; 0.02];
to = 0.0;
tf = 20.0;
% eigenvalues and eigenvectors
[wcol,lambdac] = eig(A);
wcolinv = inv(wcol);
% calculate exp (-lambda*to) matrix
explambdano = zeros(n,n);
for j = 1:1:dim
   explambdano(j,j) = exp(-lambdac(j,j)*to);
end
% calculate u(to)
uo = zeros(n,n);
for j = 1:1:n
   if (abs(lambdac(j,j)) > tol)
      uo(j,j) = -explambdano(j,j)/lambdac(j,j);
   else
      uo(j,j) = to;
   end
end

% set up discretized solution grid
nintervals = 1000;
npoints = nintervals + 1;
dt = (tf-to)/nintervals;
for i = 1:1:npoints
   tp(i) = (i-1)*dt +to;
end

% calculate solution
yp = zeros(n,npoints);
for i = 1:1:npoints
   explambdan = zeros(n,n);
   explambda = zeros(n,n);
   for j = 1:1:dim
      explambdan(j,j) = exp(-lambdac(j,j)*tp(i));
```

```
      explambda(j,j) = exp(lambdac(j,j)*tp(i));
   end
   % calculate u(t)
   u = zeros(n,n);
   for j = 1:1:n
      if (abs(lambdac(j,j)) > tol)
         u(j,j) = -explambdan(j,j)/lambdac(j,j);
      else
         u(j,j) = tp(i);
      end
   end
   term1 = explambdano*wcolinv*yo;
   term2 = (u-uo)*wcolinv*b;
   yp(:,i) = wcol*explambda*(term1 + term2);
end
%plot
for j = 1:1:n
  if (j==1)
     plot (tp,yp(j,:),'g-'), xlabel( 't' ), ylabel ( 'y' )
  elseif (j==2)
     plot (tp,yp(j,:),'r-'), xlabel( 't' ), ylabel ( 'y' )
  elseif (j==3)
     plot (tp,yp(j,:),'b-'), xlabel( 't' ), ylabel ( 'y' )
  end
  hold on
end
hold off
xlabel('time (s)');
ylabel('mole fraction');
legend('1 ','2 ', '3 ');
```



Comments:

At all times, the sum of the mole fractions is unity. The initial condition is y_o = [1/3 1/3 1/3]. The steady state equilibrium appears to be about y = [0.123 0.262 0.615]. The flowrates for each component are b=[-0.01 –0.01 0.02].

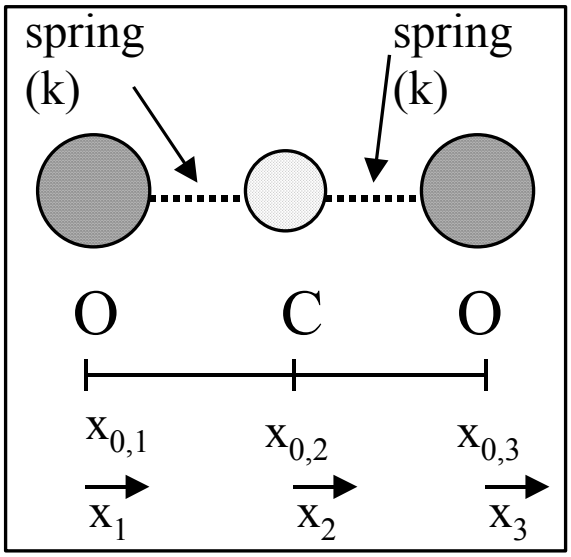


Comments:

At all times, the sum of the mole fractions is unity. The initial condition is y_o = [1/2 1/2 0]. The steady state equilibrium appears to be about y = [0.123 0.262 0.615]. The flowrates for each component are b=[-0.01 –0.01 0.02]. This is the same steady state composition as was obtained for the other initial condition, plotted on the previous page. The equilibrium composition is independent of the initial composition, in this case. This is generally true, except where multiple steady states exist.

Example 3. Transient Vibrational Behavior of Carbon Dioxide

Consider that we want to investigate the vibrational properties of carbon dioxide, CO_2. Our model of the molecule looks like this:



We model the interaction between molecules as Hookian springs. For a Hookian spring, the potential energy, U, is

$$U = \frac{k}{2}(x - x_0)^2 \tag{5.1}$$

and the force, F, is

$$F = -k(x - x_0) \tag{5.2}$$

where k is the spring constant (units of kg/s^2), x_0 is the equilibrium displacement, and x is the actual displacement.

When both ends of the spring are mobile we can write, for the spring that connects mass 1 and 2:

$$U_{12} = \frac{k}{2}((x_2 - x_1) - x_{120})^2 \tag{5.3}$$

and we can write, for the spring that connects mass 2 and 3:

$$U_{32} = \frac{k}{2}((x_3 - x_2) - x_{320})^2 \tag{5.4}$$

The forces are then:

$$F_1 \equiv -\frac{\partial U}{\partial x_1} = k((x_2 - x_1) - x_{120})$$
$$F_2 \equiv -\frac{\partial U}{\partial x_2} = -k((x_2 - x_1) - x_{120}) + k((x_3 - x_2) - x_{320}) \quad (5.5)$$
$$F_3 \equiv -\frac{\partial U}{\partial x_3} = -k((x_3 - x_2) - x_{320})$$

With only a slight sleight of hand, we redefine our variables to be:

$$x_1 = x_1 + x_{120}$$
$$x_2 = x_2 \quad (5.6)$$
$$x_3 = x_3 - x_{320}$$

This eliminates the equilibrium bond distances from the calculation. So, with this definition,

$$x_1 = x_2 = x_3 \quad (5.7)$$

at equilibrium. This does not affect our equations of motion because the derivatives of our variables before and after the transformation of the variables are the same. Our forces become:

$$F_1 \equiv -\frac{\partial U}{\partial x_1} = k(x_2 - x_1)$$
$$F_2 \equiv -\frac{\partial U}{\partial x_2} = -k(x_2 - x_1) + k(x_3 - x_2) \quad (5.8)$$
$$F_3 \equiv -\frac{\partial U}{\partial x_3} = -k(x_3 - x_2)$$

We can write Newton's equations of motion for the three molecules:

$$m_O a_1 = F_1 = k(x_2 - x_1)$$
$$m_C a_2 = F_2 = -k(x_2 - x_1) + k(x_3 - x_2) \quad (5.9)$$
$$m_O a_3 = F_3 = -k(x_3 - x_2)$$

We can generalized this to a non-symmetric linear tri-atomic molecule by writing:

$$\begin{aligned} m_1 a_1 &= F_1 = k_{12}(x_2 - x_1) \\ m_2 a_2 &= F_2 = -k_{12}(x_2 - x_1) + k_{13}(x_3 - x_2) \\ m_3 a_3 &= F_3 = -k_{13}(x_3 - x_2) \end{aligned} \tag{5.10}$$

Knowing that the acceleration is the second derivative of the position, we can rewrite the above equations in matrix form as (first divide both side of all of the equations by the masses)

$$\frac{d^2\underline{x}}{dt^2} = \underline{\underline{A}}\,\underline{x} \tag{5.11}$$

where

$$\underline{\underline{A}} = \begin{bmatrix} -k_{12}/m_1 & k_{12}/m_1 & 0 \\ k_{12}/m_2 & -k_{12}-k_{13}/m_2 & k_{13}/m_2 \\ 0 & k_{13}/m_3 & -k_{13}/m_3 \end{bmatrix} \tag{5.12}$$

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Solving this system of second order linear differential equations yields the integrated equations of motion for carbon dioxide.

We can convert the three second order ODEs into six first order ODEs as shown below. Take atom number one.

$$m_1 a_1 = F_1 = k_{12}(x_2 - x_1) \tag{5.13}$$

which we write as

$$m_1 \frac{d^2 x_1}{dt^2} = k_{12}(x_2 - x_1) \tag{5.14}$$

Now we can rewrite that second order ODE as two first order ODEs.

$$\frac{dx_1}{dt} = x_4 \tag{5.15}$$

$$m_1 \frac{dx_4}{dt} = k_{12}(x_2 - x_1) \tag{5.16}$$

We can write analogous equation for the other two atoms, to come with ODEs for variables 5 and 6. Of course, variables 1, 2, and 3 are still the positions of the atoms. Variables 4, 5, and 6 are now the velocities of the atoms.

All of these equations are homogeneous.

Our 3x3 A matrix becomes a 6x6 matrix:

$$\underline{\underline{A}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -k_{12}/m_1 & k_{12}/m_1 & 0 & 0 & 0 & 0 \\ k_{12}/m_2 & -k_{12}-k_{13}/m_2 & k_{13}/m_2 & 0 & 0 & 0 \\ 0 & k_{13}/m_3 & -k_{13}/m_3 & 0 & 0 & 0 \end{bmatrix} \tag{5.17}$$

where the solution vector, x, is now defined as:

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} \tag{5.18}$$

Now, clearly, we can solve this problem exactly as we solve any system of homogeneous linear first order ODEs. The solution will be given as

$$\underline{x}_h(t) = \underline{\underline{W}}_c \exp\left[\underline{\underline{\Lambda}}(t - t_o)\right] \underline{\underline{W}}_c^{-1} \underline{x}_o \tag{1.14}$$

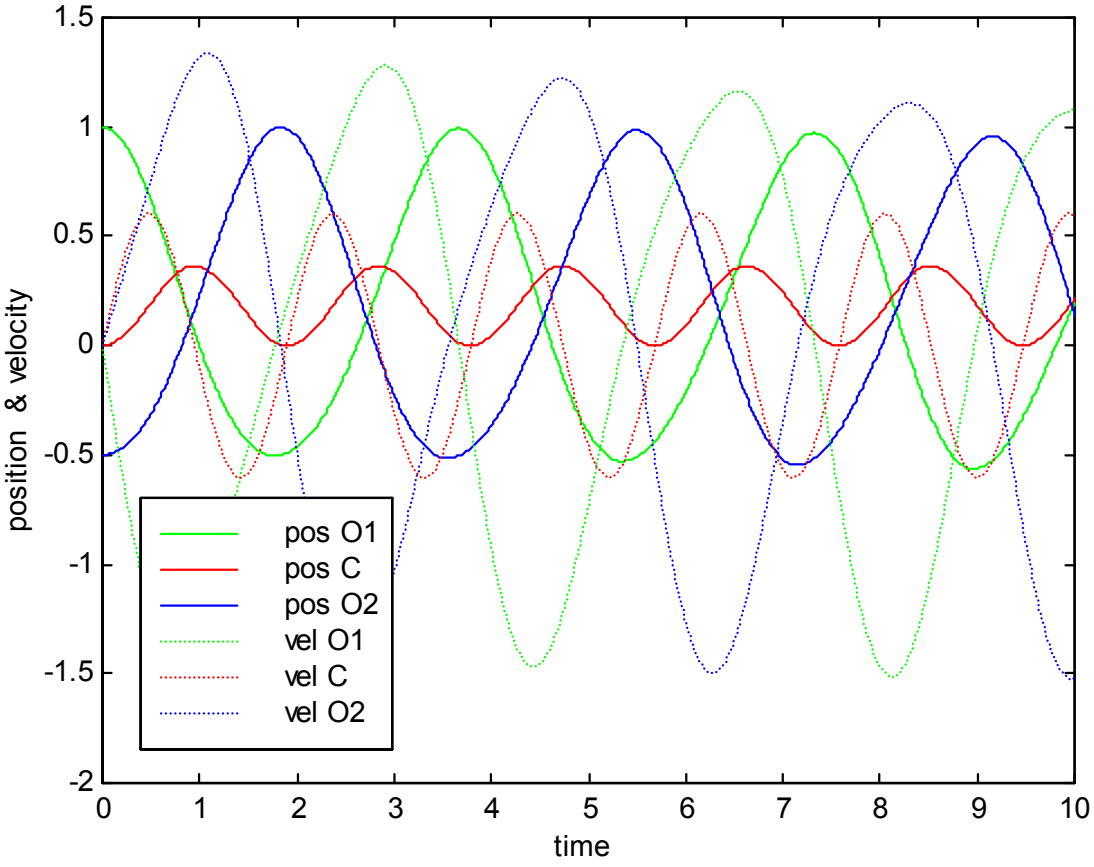
We provide a Matlab code to solve this problem on the following page. The only quantitative difference between this code and the code for example 1, is that this problem has imaginary eigenvalues, which eventually yield real solutions via the Euler Identities. Near the end of the code, we eliminate any residual round-off errors of the imaginary component of the solution.

Matlab code to solve Example 3.

```
%
%  Solution to vibrational modes of CO2 Problem
%
%  David Keffer
%  Department of Chemical Engineering
%  University of Tennessee
%  Knoxville Tennessee
%  September, 2001
%
clear all;
close all;
%
tol = 1.0e-14;
% input
   mass(1) = 16;
   mass(2) = 12;
   mass(3) = 16;
   k12 = 48;
   k13 = 48;
   A = [    0              0                    0              1              0              0
            0              0                    0              0              1              0
            0              0                    0              0              0              1
            -k12/mass(1)   k12/mass(1)          0              0              0              0
            k12/mass(2)    (-k12-k13)/mass(2)   k13/mass(2)    0              0              0
            0              k13/mass(3)          -k13/mass(3)   0              0              0];
n = max(size(A));
yo = [1.0;  0.0; -0.5; 0.0; 0.0; 0.0];
% constant terms in ODE
b = zeros(n,1);
to = 0.0;
tf = 10.0;
% eigenvalues and eigenvectors
[wcol,lambdac] = eig(A);
wcolinv = inv(wcol);
% calculate exp (-lambda*to) matrix
explambdano = zeros(n,n);
for j = 1:1:n
   explambdano(j,j) = exp(-lambdac(j,j)*to);
end
```

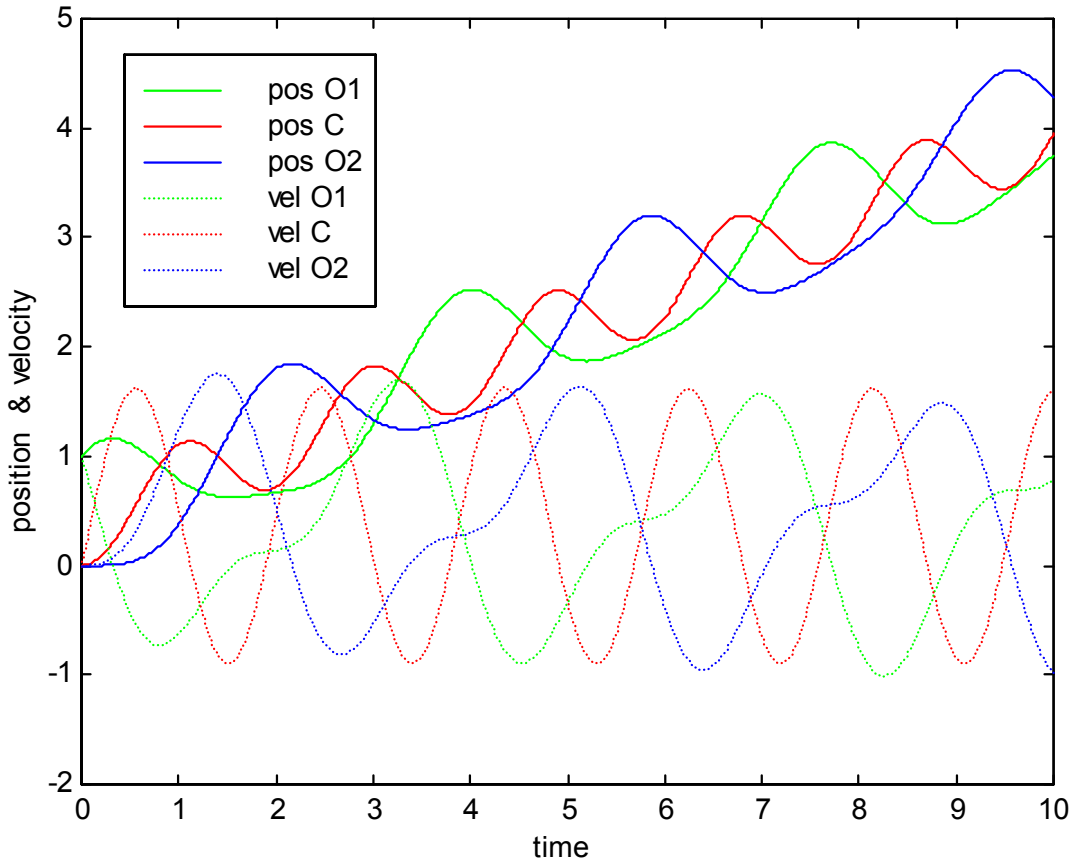
```
% calculate u(to)
uo = zeros(n,n);
for j = 1:1:n
   if (abs(lambdac(j,j)) > tol)
      uo(j,j) = -explambdano(j,j)/lambdac(j,j);
   else
      uo(j,j) = to;
   end
end

% set up discretized solution grid
nintervals = 1000;
npoints = nintervals + 1;
dt = (tf-to)/nintervals;
for i = 1:1:npoints
   tp(i) = (i-1)*dt +to;
end

% calculate solution
yp = zeros(n,npoints);
for i = 1:1:npoints
   explambdan = zeros(n,n);
   explambda = zeros(n,n);
   for j = 1:1:n
      explambdan(j,j) = exp(-lambdac(j,j)*tp(i));
      explambda(j,j) = exp(lambdac(j,j)*tp(i));
   end
   % calculate u(t)
   u = zeros(n,n);
   for j = 1:1:n
      if (abs(lambdac(j,j)) > tol)
         u(j,j) = -explambdan(j,j)/lambdac(j,j);
      else
         u(j,j) = tp(i);
      end
   end
   term1 = explambdano*wcolinv*yo;
   term2 = (u-uo)*wcolinv*b;
   yp(:,i) = wcol*explambda*(term1 + term2);
end
% remove residual imaginary components due to round-off
for i =1:1:npoints
   for j = 1:1:n
```

```
      if (imag(yp(j,i)) < tol)
         yp(j,i) = real( yp(j,i) );
      end
   end
end

%plot
for j = 1:1:n
  if (j==1)
     plot (tp,yp(j,:),'g-')
  elseif (j==2)
     plot (tp,yp(j,:),'r-')
  elseif (j==3)
     plot (tp,yp(j,:),'b-')
  elseif (j==4)
     plot (tp,yp(j,:),'g:')
  elseif (j==5)
     plot (tp,yp(j,:),'r:')
  elseif (j==6)
     plot (tp,yp(j,:),'b:')
  end
  hold on
end
hold off
xlabel('time ');
ylabel('position  & velocity ');
legend('pos O1','pos C', 'pos O2','vel O1','vel C', 'vel O2');
```



Comments:

The initial condition is y_o = [1 0 –0.5 0 0 0]. Clearly periodic behavior is observed. Also observe visually that the velocities do indeed correspond to the time derivatives of the positions. There is no net center of mass motion, since in the initial condition there was no center of mass motion, and conservation of momentum is a consequence of Newton's equations of motion.



Comments:

The initial condition is y_o = [1 0 0 1 0 0]. Clearly periodic behavior is observed. Also observe visually that the velocities do indeed correspond to the time derivatives of the positions. There is a net center of mass motion, since in the initial condition there was a center of mass motion, and conservation of momentum is a consequence of Newton's equations of motion.