CE 561 Homework 6: (assigned 10/01/09, due 10/12/09)

(1) Suppose we wanted to model the behavior of undergraduate students on a Saturday night at Molly's Pub in the same way we have been modeling chemical rate processes. Let N_M be the number of male undergraduates in the bar, let N_F be the number of female undergraduates in the bar, let N_C be the number of couples in the bar, let N_X be the the number of individuals (male or female) that have not yet come to the bar or have left alone, and let N_{XC} be the number of couples that have left the bar together. Then the rate processes might be written in terms of the following list of "reactions"

(1) M + F → C, with $r_1 = k_{hu} N_M N_F^2$
(2) C → M + F, with $r_2 = k_{bu} N_C N_F$
(3) M → X, with $r_3 = k_{la} N_M$
(4) F → X, with $r_4 = k_{la} N_F$
(5) X → M, with $r_5 = k_{ma}$
(6) X → F, with $r_6 = k_{fa}$
(7) C → XC, with $r_7 = k_{lt} N_C$

The numerical values of the rate constants for these processes are

k_{hu} = 0.2 events undergraduate^{-3} hr^{-1}
k_{bu} = 0.1 events couple^{-1} undergraduate^{-1} hr^{-1}
k_{la} = 0.3 events undergraduate^{-1} hr^{-1}
k_{ma} = 20 events hr^{-1}
k_{fa} = 10 events hr^{-1}
k_{lt} = 0.5 events couple^{-1} hr^{-1}

The "initial conditions" for this problem are that at time = 0 (when the bar opens), $N_M = N_F = N_C = N_{XC} = 0$. That is, there is nobody in the bar initially. Note that none of the rates depend on N_X, the number of people not in the bar, since there are an effectively infinite and constant number of people that are not in the bar. So, an initial condition for N_X is not needed, and we do not need to solve for N_X.

(a) Write the rate equations for N_M, N_F, N_C, and N_{XC}. Can these differential equations be solved analytically? Comment on the physical significance of these (differential) equations and their solutions at short times (say t less than 1/2 hour).

The rate equations may be written in the usual way, as

$$\frac{dN_M}{dt} = -r_1 + r_2 - r_3 + r_5 = -k_{hu} N_M N_F^2 + k_{bu} N_C N_F - k_{la} N_M + k_{ma}$$

$$\frac{dN_F}{dt} = -r_1 + r_2 - r_4 + r_6 = -k_{hu} N_M N_F^2 + k_{bu} N_C N_F - k_{la} N_F + k_{fa}$$

$$\frac{dN_C}{dt} = r_1 - r_2 - r_7 = k_{hu} N_M N_F^2 - k_{bu} N_C N_F - k_{lt} N_C$$

$$\frac{dN_{XC}}{dt} = r_7 = k_{lt} N_C$$

These can't be solved analytically, or at least I didn't find any simple solution. One can find a simple expression for N_M - N_F, but the coupled, nonlinear nature of the equation makes solving for anything else difficult. At short times, it may not make sense to write continuum equations to represent quantities that we know should be integers. For example, at times very near $t = 0$, we will have small, fractional values for N_M and N_F, which is not representative of the system we are trying to model.

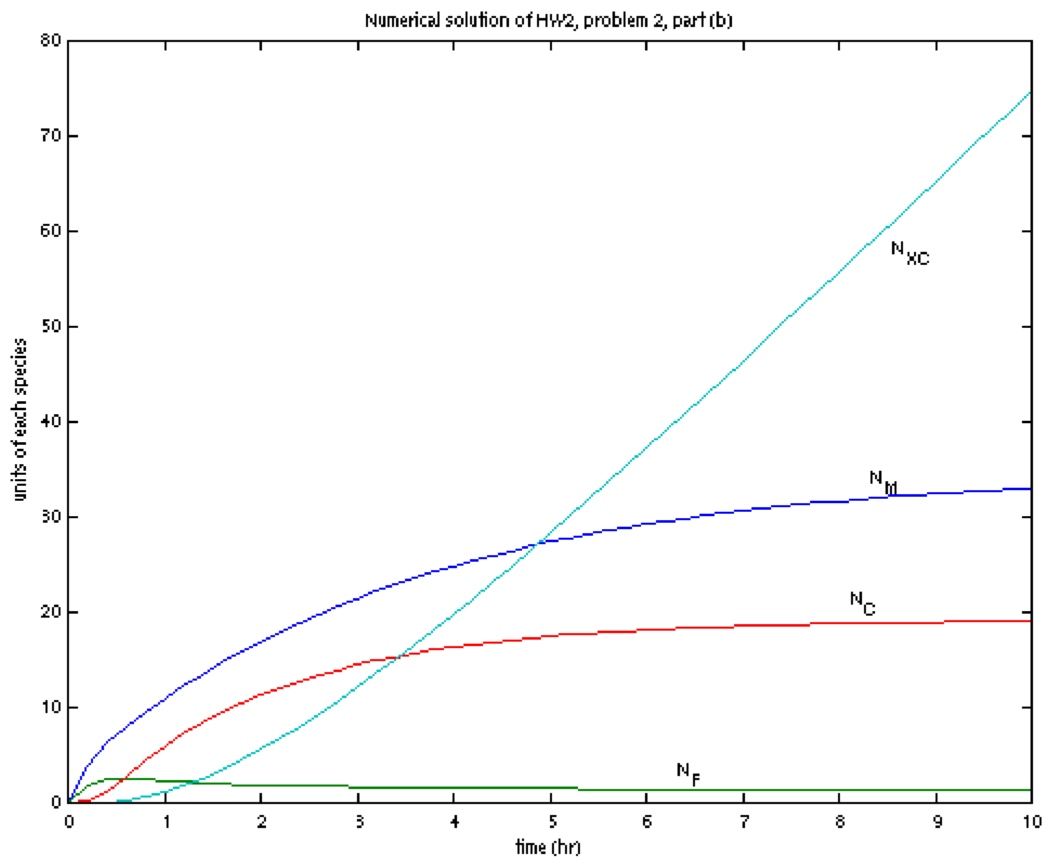
(b) Write a short computer program (in the language of your choice) that uses Euler's method to solve the rate equations numerically (treating N_M, N_F, N_C, and N_{XC} as continuous variables, like we usually do for chemical concentrations). The experiment ends at $t = 10$ hours (when the bar finally closes). Plot the 'concentrations' for $t = 0$ to $t = 10$ hours.

The following Matlab function implements the explicit Euler method for this problem. It is very similar to the example in the Day 5 lecture notes.

```
function Ns=myeuler(ks,Nos,dt,pt,ft)
%  Program to integrate the equations of CE 561, Homework 6, Problem 1 using
%  the explicit Euler method (2009 version).
%
%  Inputs:
%  ks = vector of rate constants, [khu kbu kla kma kfa klt]
%  Nos = vector of initial numbers of each "species", [NM NF NC NXC]
%  dt = time step to be used in Euler's method
%  pt = time interval at which to save/print solution
%  ft = final time, the time to end the simulation
%
%  Output:
%  Ns = matrix containing the solution.
%       the 1st column is time, 2nd through 5th column are NM NF NC NXC
%
t=0;
tp=0;
Ns=[t Nos];
NM=Nos(1);
NF=Nos(2);
NC=Nos(3);
NXC=Nos(4);
khu =ks(1);
kbu =ks(2);
kla =ks(3);
kma =ks(4);
kfa =ks(5);
klt =ks(6);
while (t<=ft)
  t=t+dt;
  deltaNM=(-khu*NM*NF^2+kbu*NC*NF-kla*NM+kma)*dt;
  deltaNF=(-khu*NM*NF^2+kbu*NC*NF-kla*NF+kfa)*dt;
  deltaNC=(khu*NM*NF^2-kbu*NC*NF-klt*NC)*dt;
  deltaNXC=klt*NC*dt;
  NM=NM+deltaNM;
  NF=NF+deltaNF;
  NC=NC+deltaNC;
  NXC=NXC+deltaNXC;
  if ((t-tp)>=(pt-dt/2.))
    Ns=[Ns; t NM NF NC NXC];
    tp=t;
  end
```

```
end
disp([' '])
disp(['       t         NM        NF        NC        NXC'])
disp(Ns)
disp(['  '])
plot(Ns(:,1),Ns(:,2:5))
xlabel('time (hr)')
ylabel('units of each species')
title('Numerical solution of HW2, problem 2, part (b)')
return
```

A plot of the result of running this for the given initial conditions (empty bar) and rate parameters with an integration step size of 0.001 hours are shown in the plot on the following page. The number of males, females, and couples in the bar has nearly reached a steady state by the end of the evening, where the rate of new males and females entering the bar is balanced by the rate of them leaving (either alone or in pairs). Once steady state is achieved, the total number of couples that have left (N_{XC}) will just increase linearly at a rate proportional to the steady state number of couples in the bar. Note that the total number of people who have left the bar alone isn't shown, but in 10 hours, we know that 200 males and 100 females have entered the bar. 75 couples have left the bar, and 33 males, 19 couples, and 1.4 females remain in the bar. That means that (200 - 75 - 33 - 19) = 73 males have left alone, while only (100 - 75 -19 - 1.4) = 4.6 females have left alone.



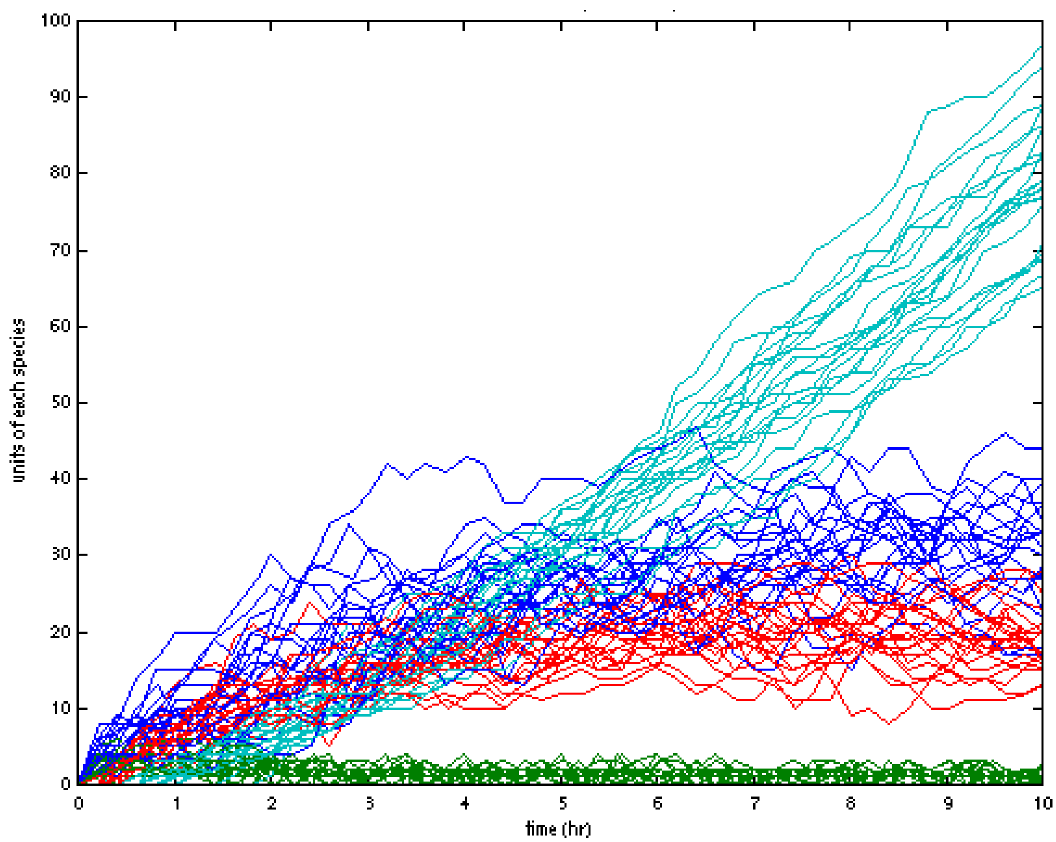
(c) Write a short computer program that uses kinetic Monte-Carlo methods to simulate the bar's population over time, treating N_M, N_F, N_C, and N_{XC} as integer variables. Run the program several times. Plot the 'concentrations' for $t = 0$ to $t = 10$ hours.

The following short Matlab program (much like the example in the notes) implements a kinetic Monte Carlo simulation of this situation:

```
function Ns=montecarlo(ks,Nos,pt,ft)
%  Program to do a kinetic Monte Carlo simulation of the situation
%  described in CE 561, Homework 6, Problem 1 (2009 version).
%
%  Inputs:
%  ks = vector of rate constants, [khu kbu kla kma kfa klt]
%  Nos = vector of initial numbers of each "species", [NM NF NC NXC]
%  pt = time interval at which to save/print solution
%  ft = final time, the time to end the simulation
%
%  Output:
%  Ns = matrix containing the solution.
%       the 1st column is time, 2nd through 5th column are NM NF NC NXC
%
t=0;
tp=0;
Ns=[t Nos];
NM=Nos(1);
NF=Nos(2);
NC=Nos(3);
NXC=Nos(4);
khu =ks(1);
kbu =ks(2);
kla =ks(3);
kma =ks(4);
kfa =ks(5);
klt =ks(6);
while (t<ft)
%  compute the reaction rates
  r(1)=khu*NM*NF^2;
  r(2)=kbu*NC*NF;
  r(3)=kla*NM;
  r(4)=kla*NF;
  r(5)=kma;
  r(6)=kfa;
  r(7)=klt*NC;
  rt=sum(r);
%  generate two random numbers
  rns=rand(1,2);
%  pick which reaction happened
%  based on one of the random numbers
  if (rns(2)<=(r(1)/rt))
%   reaction 1
    NC=NC+1;
    NM=NM-1;
```

```
    NF=NF-1;
  elseif (rns(2)<=sum(r(1:2)/rt))
%   reaction 2
    NC=NC-1;
    NM=NM+1;
    NF=NF+1;
  elseif (rns(2)<=sum(r(1:3)/rt))
%   reaction 3
    NM=NM-1;
  elseif (rns(2)<=sum(r(1:4)/rt))
%   reaction 4
    NF=NF-1;
  elseif (rns(2)<=sum(r(1:5)/rt))
%   reaction 5
    NM=NM+1;
  elseif (rns(2)<=sum(r(1:6)/rt))
%   reaction 6
    NF=NF+1;
  elseif (rns(2)<=sum(r(1:7)/rt))
%   reaction 7
    NC=NC-1;
    NXC=NXC+1;
  else
    disp(['OOPS'])
    return
  end
%  advance the time, based on the other
%  random number
  dt=-1/rt*log(rns(1));
  t=t+dt;
  if ((t-tp)>=pt)
    Ns=[Ns; t NM NF NC NXC];
    tp=tp+pt;
  end
end
disp([' '])
disp(['       t         NM        NF        NC        NXC'])
disp(Ns)
disp(['  '])
plot(Ns(:,1),Ns(:,2:5))
xlabel('time (hr)')
ylabel('units of each species')
title('Numerical solution of HW2, problem 2, part (c)')
return
```

This plot shows the results of repeated runs of this program for the parameter values in the problem statement:



As you can see from this plot, results of different runs vary substantially. The fluctuations in the 'species concentrations' vary widely around the mean values found in part (b).

(d) Compare the results of parts (b) and (c), discussing similarities, differences, and the importance of statistical fluctuations for this problem.

As stated above, in part (c) we observe fluctuations about the mean values that are quite large. The large fluctuations result from the fact that N_F is quite small (usually 0, 1, or 2, occasionally 3 or 4, and rarely higher). So, the discrete nature of N_F becomes very important. This is particularly true in the rate expression for reaction (1), which is second order in F.

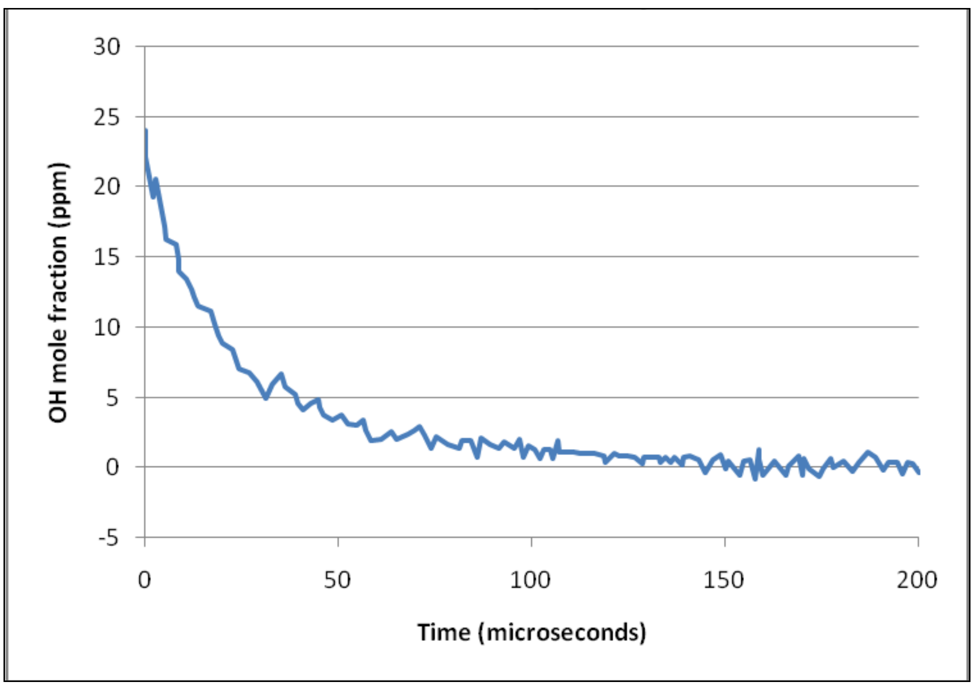
(2) Vasudevan *et al.* (*Int. J. Chem. Kin.*, **40**, 488-495 (2008)) conducted high-temperature gas phase kinetics experiments to determine the rate of reaction of methyl radical (CH_3) with hydroxyl radical (OH). Table I gives key parts of the reaction mechanism and rate parameters that they used to interpret their experiments and to determine rate parameters for the first reaction. I have slightly modified and simplified it.

Table I. Rate Coefficients of Reactions Important in CH_3 + OH Experiments[a]

	Rate Coefficient (cm^3 mol^{-1} s^1)		
Reaction	***A***	***n***	***E* (cal mol^{-1})**
$CH_3 + OH \longrightarrow$ Products	To be determined		
$(CH_3)_3COOH \longrightarrow (CH_3)_3CO + OH$[b]	2.50E+15	0.0	43,017
$(CH_3)_3CO \longrightarrow (CH_3)_2CO + CH_3$[b]	1.30E+14	0.0	15,300
$2CH_3 \longrightarrow C_2H_6$	3.0E+13	0.0	0
$OH + C_2H_6 \longrightarrow C_2H_4 + H + H_2O$	1.614E+06	2.224	741
$C_2H_6N_2 \longrightarrow CH_3 + CH_3 + N_2$[b]	2.220E+39	-7.99	51,505
$(CH_3)_2CO + OH \longrightarrow CH_3COCH_2 + H_2O$	2.951E+13	0.0	4,564
$OH + OH \longrightarrow O + H_2O$	3.57E+04	2.4	-2,110

[a] **Rate constants are given by $k = A\,T^n \exp(-E/(RT))$**
[b] **Rate coefficient units: s^{-1}**

Data on the hydroxyl radical (OH) concentration vs. time, extracted from one of the figures in their paper, is shown in the plot below:



You can download the file containing the data shown in the plot above by clicking here.

For that experiment, the initial concentrations were 21.5 ppm of $(CH_3)_3COOH$ and 75 ppm of $C_2H_6N_2$ in argon. The total pressure was 1.70 bar, and the temperature was 1245 K.

Problem:
Using the reaction mechanism presented above, apply a non-linear least squares analysis to obtain a value of the rate coefficient for the first reaction in the table above (CH_3 + OH → products). Plot the resulting model prediction along with the experimental data for the OH concentration. Compute the sensitivity of the OH concentration profile to each of the rate constants in the mechanism. How do the sensitivities of the model results to the rate coefficient being measured compare to the sensitivity of the model results to the other rate coefficients? Comment on the effect of uncertainties in the other rate constants on the fitted value of the rate constant being measured.

Solution:
First, let's number the compounds as shown, so that we have indices by which we can refer to them. We also note that all of the reactions are irreversible, so we only have to solve for species that appear as reactants.

Species	Number
CH_3	1
OH	2
$(CH_3)_3COOH$	3
$(CH_3)_3CO$	4
$(CH_3)_2CO$	5
C_2H_6	6
$C_2H_6N_2$	7

With this numbering for the species and the reactions from Table I numbered 1 through 8 in the order given, the stoichiometric matrix is:

And the reaction rates are given by

$$\underline{r} = \begin{bmatrix} k_1 C_1 C_2 \\ k_2 C_3 \\ k_3 C_4 \\ k_4 C_1^2 \\ k_5 C_2 C_6 \\ k_6 C_7 \\ k_7 C_2 C_5 \\ k_8 C_2^2 \end{bmatrix}$$

At the specified temperature of 1245 K, the rate constants are (in the same units specified in Table I):

Reaction	*A*	*n*	*E*	*k (1245 K)*

2	2.50E+15	0	43,017	7.01E+07
3	1.30E+14	0	15,300	2.68E+11
4	3.00E+13	0	0	3.00E+13
5	1.61E+06	2.224	741	9.15E+12
6	2.22E+39	-7.99	51,505	3.75E+05
7	2.95E+13	0	4,564	4.66E+12
8	3.57E+04	2.4	-2,110	2.25E+12

We will need to supply an initial guess for the rate constant of reaction 1. We can obtain this from the initial slope of the [OH] decay curve. By eye, it looks to me like [OH] is decreasing by about 1 ppm per microsecond (or 1 s^{-1}).

The total concentration at 1245 K and 1.70 bar is obtained from the ideal gas law as

$$\frac{n}{V} = \frac{p}{RT} = \frac{1.70 \text{ bar}}{83.14 \text{ bar cm}^3 \text{ mol}^{-1} \text{ K}^{-1} \times 1245 \text{ K}} = 1.64 \times 10^{-5} \text{ mol cm}^{-3}$$

Thus, the initial concentration of $(CH_3)_3COOH$ is $2.15\times10^{-5} * 1.64\times10^{-5} = 3.53\times10^{-10}$ mol cm^{-3}, which yields an initial [OH] concentration of 3.53×10^{-10} mol cm^{-3}. Likewise the initial concentration of $C_2H_6N_2$ is 1.23×10^{-9} mol cm^{-3}, which yields 2.46×10^{-9} mol cm^{-3} CH_3 radicals (two from each C_2H_6N). Similarly, we can convert the concentrations in the data file from ppm to mol cm^{-3} by multiplying them by the total concentration. In terms of concentration, the initial slope of the [OH] decay is 1.64×10^{-5} mol cm^{-3} s^{-1}. So, we estimate the rate of reaction 1 as

$$1.64\times10^{-5} \text{ mol cm}^{-3} \text{ s}^{-1} = k_1[OH][CH_3] = k_1 * 3.5\times10^{-10} \text{ mol cm}^{-3} * 2.46\times10^{-9} \text{ mol cm}^{-3}$$

From which $k_1 = 1.9\times10^{13}$ cm^3 mol^{-1} s^{-1}. Of course, this is just a crude estimate, but it should be of the right order of magnitude.

The first thing we need is a function to calculate the rate of change of the concentrations. That is, given the concentrations at a given time, we want to compute

$$\frac{d\underline{C}}{dt} = \underline{f}(\underline{C}) = \underline{\underline{\alpha}}\,\underline{r}(\underline{C})$$

Such a function, in Matlab, is given below. In this case, I have "hard-coded" the rate expressions.

```
function Cp=dCdt(t,Cs,ks)
% function Cp=dCdt(t,Cs,ks)
%
%  This is a function that formulates the ODEs to
%  be integrated for CE561, HW6, problem 2, 2009 version.
%  The input and output arguments are in the order required for
%  use with the Matlab suite of ODE solvers.
%  Inputs:
%     t = time
%     Cs = concentrations at time t
%      Cs is a COLUMN vector
%     ks = rate coefficients (as a COLUMN vector)
%      Each row of alpha corresponds to a reaction
%      Each column of alpha corresponds to a species
%  Output:
%     yp = derivative of concentration with respect to time
%
```

```
alpha=[-1 -1  0  0  0  0  0
        0  1 -1  1  0  0  0
        1  0  0 -1  1  0  0
       -2  0  0  0  0  1  0
        0 -1  0  0  0 -1  0
        2  0  0  0  0  0 -1
        0 -1  0  0 -1  0  0
        0 -2  0  0  0  0  0];
nspec=length(Cs);       % nspec is the number of species
nrxn=length(ks);        % nrxn is the number of reactions
if (size(alpha,2)~=nspec)|(size(alpha,1)~=nrxn)
     disp('Mismatch between Cs, ks, and nu')
     return
end
r=zeros(nrxn,1);
r(1)=ks(1)*Cs(1)*Cs(2);
r(2)=ks(2)*Cs(3);
r(3)=ks(3)*Cs(4);
r(4)=ks(4)*Cs(1)^2;
r(5)=ks(5)*Cs(2)*Cs(6);
r(6)=ks(6)*Cs(7);
r(7)=ks(7)*Cs(2)*Cs(5);
r(8)=ks(8)*Cs(2)^2;
Cp=alpha'*r;
return
```

With the rate constants given by:

```
>> ks

ks =

   1.9000e+13
   7.0100e+07
   2.6800e+11
   3.0000e+13
   9.1500e+12
   3.7500e+05
   4.6600e+12
   2.2500e+12
```

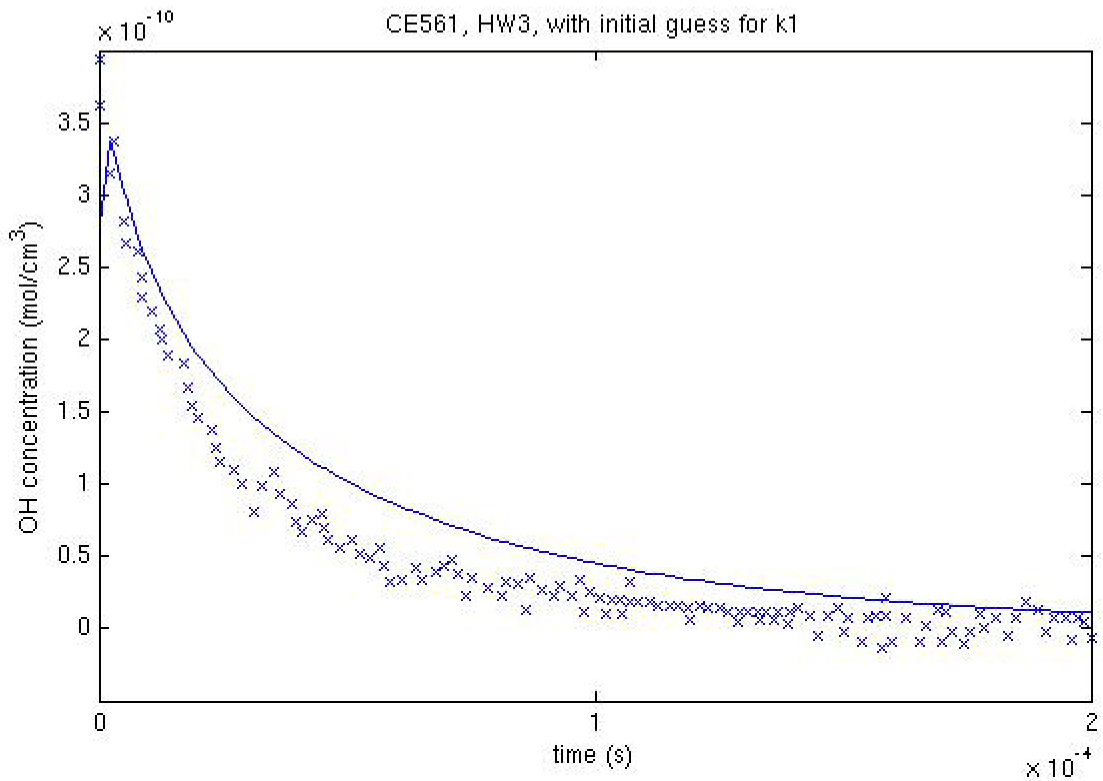
And the initial concentrations given by

```
>> Co

Co =

            0
            0
   3.5300e-10
            0
            0
            0
   1.2300e-09
```

I extracted the times from the experimental data file, into a variable texp, and ran the command:

```
>>  [t Cmodel]=ode15s(@dCdt,texp,Co,options,ks);
```

A plot of the computed OH concentration (column 2 of Cmodel, returned by the above function), along with the experimental data, was as shown below:



Note that in order to get this plot, I had to adjust the default tolerances in Matlab's ODE solver, to an absolute tolerance of 10^{-12} and a relative tolerance of 10^{-5}. This was done with the command:

```
>> options=odeset('AbsTol',1.e-12,'RelTol',1.e-5)
```

Now that we can solve the ODE's to get the concentration of OH as a function of time, the next step is to find the value of k_1 that gives the best fit to the data. To do so, we first need a function that evaluates the difference between the experimental and model values. I used the following function for that purpose:

```
function e=err(lk1,ks,texp,Cexp,Co)
% function e=err(lk1,ks,texp,Cexp,Co)
%
% This function computes the vector of errors, defined as the
% difference between the model prediction and the experimental
% result, for ce 561 HW6, problem 2, 2009 version. It is used with the matlab
% lsqnonlin function to find a best-fit for the rate coefficients
% for reaction 1 (k1).
%
% The fitting variable used is ln(k1) since that will give nicer
% behavior of the minimization function.
%
% Input:
% lk1=current estimate of ln(k1), the rate coefficient being optimized
% ks=vector of rate coefficients
% texp=vector of times for the experimental data points
% Cexp=vector of concentrations for the experimental data points
%
% Output:
% e=vector of errors, defined as the model result minus the
% experimental result at each data point
```

```
%
options=odeset('AbsTol',1.e-16,'RelTol',1.e-6);
% setting solver tolerences as needed to make it work
ks(1)=exp(lk1);
[t Cmodel]=ode15s(@dCdt,texp,Co,options,ks);
e=(Cmodel(:,2)-Cexp);
return
```

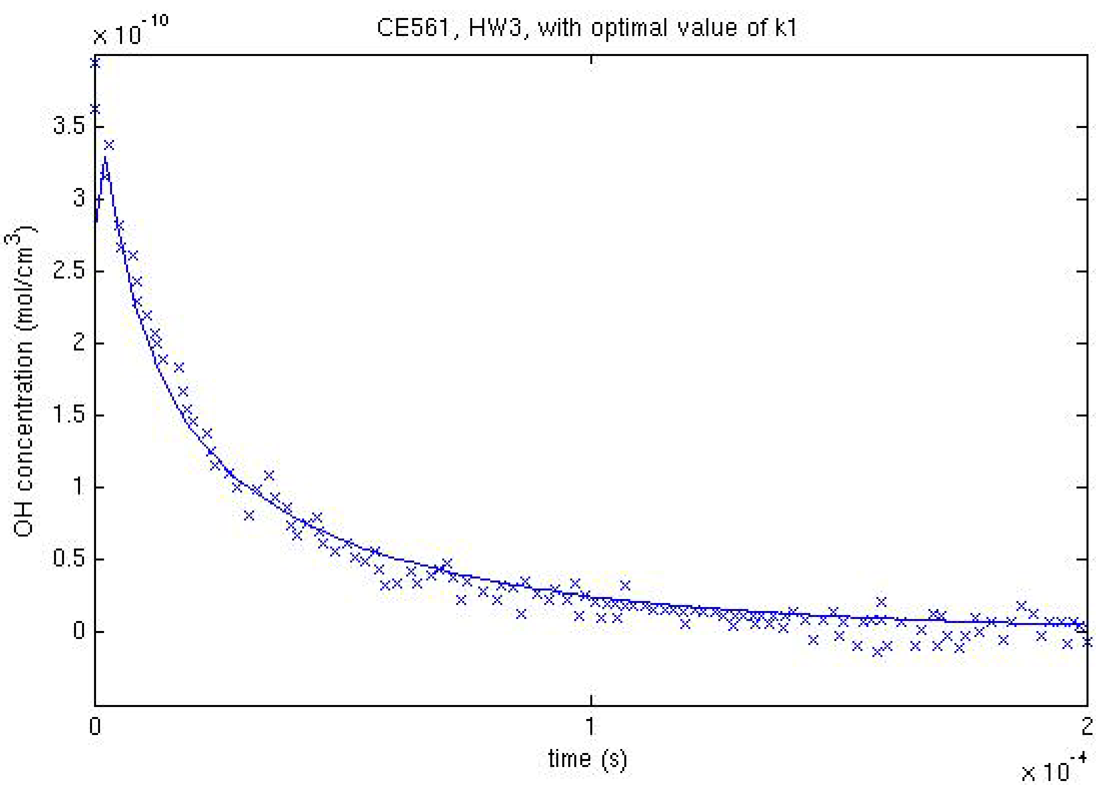
Matlab has a built-in nonlinear least-squares routine that can be used with the above function to find the optimal value for k_1. I did the optimization in terms of $\ln(k_1)$, and again had to change the default tolerances. That was done with the command:

```
>> minoptions=optimset('TolX',1.e-3,'TolFun',1.e-25);
```

And the nonlinear least squares minimize was run with the command:

```
>> lk1_fit=lsqnonlin(@(lk1) err(lk1,ks,texp,Cexp,Co),lk1,[],[],minoptions)

Optimization terminated: relative function value
 changing by less than OPTIONS.TolFun.

lk1_fit =

   3.1098e+01
```

This value of $\ln(k_1)$ corresponds to $k_1 = 3.20\times10^{13}$ cm^3 mol^{-1} s^{-1}. With that value for k_1, a plot of the computed result and data points looks like:



This looks like a reasonable fit to the data, and when I started with different initial guesses for k_1 (from 10^{10} to 10^{18}) the optimization always gave this same value. Thus, we are now ready to proceed to the second part of the problem – sensitivity analysis. My Matlab code for that was as follows:

```
function yp=sens(t,y,ks)
%  function yp=sens(t,y,ks)
%  This is a function that formulates the ODEs to be
%  integrated for the final part of ce561 HW6, prob 2 (2009 version).  The
%  input and output arguments are in the order required
%  for use with the Matlab suite of ODE solvers.  This
%  routine includes equations for all of the sensitivity
%  coefficients as well as the concentrations.
%
%  Inputs:
%     t = time
%     y =  vector of concentrations and sensitivity coefficients
%          1st N components of y are the concentrations,
%          2nd N components of y are the 1st column of the
%                sensitivity matrix, etc.
%          So N*i+1 through N*(i+1) components of y are the ith
%          column of the sensitivity matrix.
%     ks = rate coefficients
%  y and ks are both COLUMN vectors
%
%  Output:
%     yp = derivative of y with respect to time
%
m=length(ks);       %   m is the number of reactions
n=length(y)/(m+1);  %   n is the number of species
Cs=y(1:n);          %   1st part of y is the concs.
for i=1:m
    Z(:,i)=y(n*i+1:n*(i+1));
end
%
% Set up the stoichiometric matrix
  alpha=[-1 -1  0  0  0  0  0
          0  1 -1  1  0  0  0
          1  0  0 -1  1  0  0
         -2  0  0  0  0  1  0
          0 -1  0  0  0 -1  0
          2  0  0  0  0  0 -1
          0 -1  0  0 -1  0  0
          0 -2  0  0  0  0  0];
%
%  Compute the reaction rates
r=zeros(m,1);
r(1)=ks(1)*Cs(1)*Cs(2);
r(2)=ks(2)*Cs(3);
r(3)=ks(3)*Cs(4);
r(4)=ks(4)*Cs(1)^2;
r(5)=ks(5)*Cs(2)*Cs(6);
r(6)=ks(6)*Cs(7);
r(7)=ks(7)*Cs(2)*Cs(5);
r(8)=ks(8)*Cs(2)^2;
%
% Set up the matrix of reaction orders
  nu=zeros(m,n);
  for i=1:m
    for j=1:n
      if (alpha(i,j)<0)
        nu(i,j)=-alpha(i,j);
      end
    end
  end
```

```
%
%  Make diagonal matrices from vectors
  Cm=diag(Cs+1.e-50); % add the 1.e-50 so we don't divide by 0 below
  rm=diag(r);
  km=diag(ks+1.e-50);
%
%  Compute derivatives according to equations
  dCdt=alpha'*r;
  dZdt=(alpha')*(rm*inv(km)+rm*nu*inv(Cm)*Z);
%
  yp=dCdt;
  for i=1:m;
    yp=[yp;dZdt(:,i)];
  end
```

The somewhat automated means of setting up the equations for the sensitivity coefficients is explained in the example shown in the day6n7 lecture notes, excerpted below:

To compute the sensitivity coefficients directly, we integrate the equations that we derived in class (written here in terms of the concentrations ($\underline{C}$ and the rate constants $\underline{k}$):

$$\frac{dZ_{ij}}{dt} = \frac{\partial f_i}{\partial k_j} + \sum_{l=1}^{N} \frac{\partial f_i}{\partial C_l} Z_{lj}$$

where f is the right-hand side of the rate equations when they are written as

$$\frac{d\underline{C}}{dt} = \underline{f}(\underline{C};\underline{k}) .$$

Written in matrix form, this is

$$\frac{d\underline{\underline{Z}}}{dt} = \frac{\partial \underline{f}}{\partial \underline{k}} + \sum_{l=1}^{N} \frac{\partial \underline{f}}{\partial \underline{C}} \underline{\underline{Z}}$$

If the rate equations are written as

$$\frac{d\underline{C}}{dt} = \underline{f} = \underline{\underline{\alpha}}^T \underline{r}$$

and the reaction rates are of the form

$$r_i = k_i \prod_{l=1}^{N} C_{A_l}^{\nu_{il}}$$

then the derivative of the i^{th} reaction rate with respect to the j^{th} rate constant is

$$\frac{\partial r_i}{\partial k_j} = \prod_{l=1}^{N} C_{A_l}^{\nu_{il}} = \frac{r_i}{k_j} \text{ , if } i = j$$

$$\frac{\partial r_i}{\partial k_j} = 0 \text{ , if } i \neq j$$

So, the matrix $\frac{\partial \underline{r}}{\partial \underline{k}}$ has the form

$$\frac{\partial \underline{r}}{\partial \underline{k}} = \begin{bmatrix} \frac{r_1}{k_1} & 0 & 0 & etc. \\ 0 & \frac{r_2}{k_2} & 0 & etc. \\ 0 & 0 & \frac{r_3}{k_3} & etc. \\ etc. & etc. & etc. & etc. \end{bmatrix} = \underline{\underline{r}}\,\underline{\underline{k}}^{-1}$$

where $\underline{\underline{r}}$ is a diagonal matrix of the reaction rates (r_i), and $\underline{\underline{k}}$ is a diagonal matrix of the rate constants (k_j). Likewise, the derivative of the i^{th} reaction rate with respect to the concentration of the j^{th} species is

$$\frac{\partial r_i}{\partial C_{A_j}} = \frac{\nu_{ij} k_i}{C_{A_j}} \prod_{l=1}^{N} C_{A_l}^{\nu_{il}} = \frac{\nu_{ij} r_i}{C_{A_j}}$$

This can be written in matrix form as

$$\frac{\partial \underline{r}}{\partial \underline{C}_A} = \underline{\underline{r}}\,\underline{\underline{\nu}}\,\underline{\underline{C}}_A^{-1}$$

where $\underline{\underline{r}}$ is a diagonal matrix of the reaction rates (r_i), $\underline{\underline{C}}$ is a diagonal matrix of the species concentrations (C_{A_j}), and $\underline{\underline{\nu}}$ is the matrix of reaction orders (ν_{ij}). Substituting these expressions into the equations for the sensitivity coefficients gives:

$$\frac{d\underline{\underline{Z}}}{dt} = \frac{\partial \underline{f}}{\partial \underline{k}} + \frac{\partial \underline{f}}{\partial \underline{C}}\underline{\underline{Z}} = \underline{\underline{\alpha}}^T \frac{\partial \underline{r}}{\partial \underline{k}} + \underline{\underline{\alpha}}^T \frac{\partial \underline{r}}{\partial \underline{C}}\underline{\underline{Z}} = \underline{\underline{\alpha}}^T \left(\frac{\partial \underline{r}}{\partial \underline{k}} + \frac{\partial \underline{r}}{\partial \underline{C}}\underline{\underline{Z}} \right)$$
$$= \underline{\underline{\alpha}}^T \left(\underline{\underline{r}}\,\underline{\underline{k}}^{-1} + \underline{\underline{r}}\,\underline{\underline{\nu}}\,\underline{\underline{C}}_A^{-1}\underline{\underline{Z}} \right)$$

I used the Matlab function listed above with the Matlab ode integrators as follows:

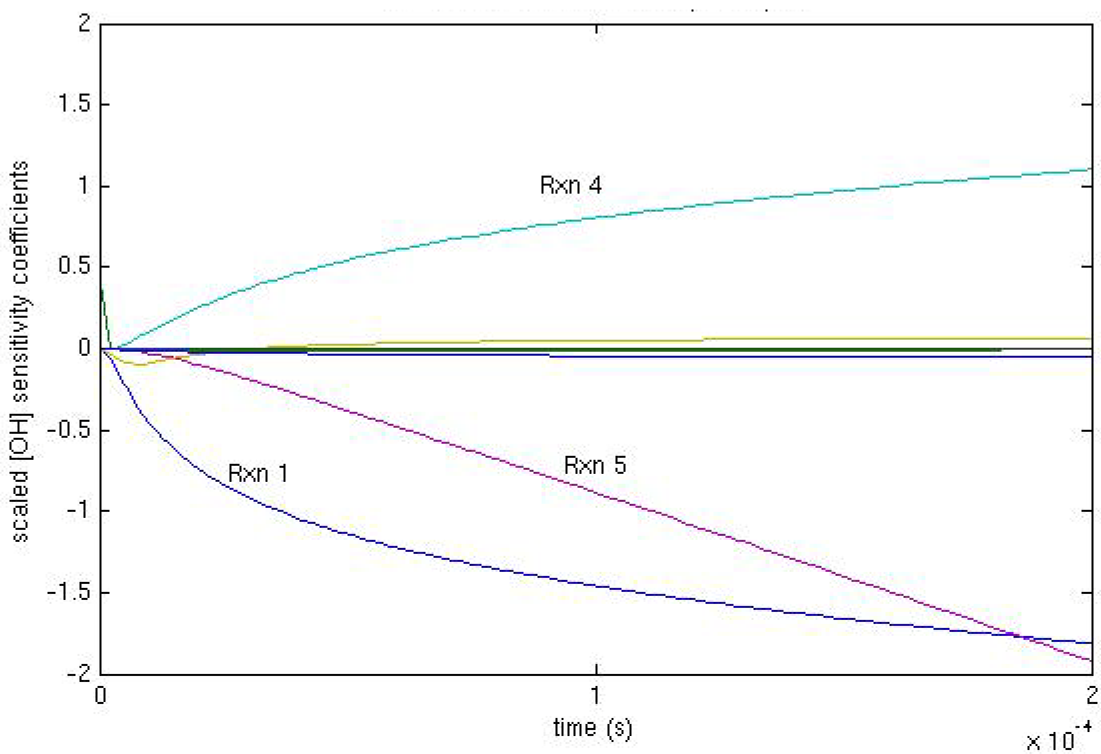
```
>> yo=zeros(63,1);
>> yo(1:7)=Co;
>> options=odeset('AbsTol',1.e-25,'RelTol',1.e-10);
>> [t sol]=ode15s(@sens,texp,yo,options,ks);
```

The solution has 63 elements (8 reactions* 7 species = 56 sensitivity coefficients + 7 species = 63 total solution variables). Thus, the initial condition vector has the initial concentrations as its first 7 elements, followed by 56 zeros for the sensitivity coefficients.

The matrix 'sol' returned from the above command has 63 columns. The first 7 columns have the species concentrations at each time point. The next 7 columns have the sensitivities to k_1, then the next 7 have the sensitivities to k_2, and so on. Here, we care about the sensitivity of the concentration of species 2 (OH) to each of the rate constants. These sensitivity coefficients are in columns 9, 16, 23, 30, 37, 44, 51, and 58 of the solution matrix. To make the *scaled* sensitivity coefficients, we multiply each of them by the corresponding rate constant, and divide each of them by the concentration of species 2 (at the corresponding times). This can be done as follows:

```
>> sigma_OH=[sol(:,9)*ks(1)./sol(:,2) sol(:,16)*ks(2)./sol(:,2)
sol(:,23)*ks(3)./sol(:,2) sol(:,30)*ks(4)./sol(:,2) sol(:,37)*ks(5)./sol(:,2)
sol(:,44)*ks(6)./sol(:,2) sol(:,51)*ks(7)/sol(:,2) sol(:,58)*ks(8)./sol(:,2)];
```

A plot of these scaled sensitivity coefficients, with labels on the most important curves, looks like:



Thus, overall, the OH concentration is most sensitive to reaction 1 (the one that the authors were trying to measure) but also exhibits significant sensitivity to reactions 4 and 5:

Reaction 4: $2CH_3 \rightarrow C_2H_6$
Reaction 5: $OH + C_2H_6 \rightarrow C_2H_4 + H + H_2O$

The sensitivity coefficient for reaction 4 is positive – if its rate increases, the OH concentration also increases (because it removes CH_3, with which OH reacts).

The sensitivity coefficients for reactions 1 and 5 are negative. Increasing the rate of either of those reactions decreases the OH concentration.

If there are substantial uncertainties in the rates of either reaction 4 or reaction 5, those will lead to uncertainties of comparable magnitude in the rate constant for reaction 1.