CBE 450 Chemical Reactor Fundamentals
Fall, 2011
Sample Codes
David Keffer
Department of Chemical and Biomolecular Engineering
University of Tennessee
dkeffer@utk.edu

Heterogeneous Reactors with Adsorption

There are two input files for modeling heterogeneous batch reactors with adsorption.

Example 1. This example assumes that we know the adsorption and desorption rates and we have an irreversible surface reaction.
Example 2. This example assumes that we know the adsorption and desorption rates and we have an reversible surface reaction.
Example 3. This example assumes that we assume adsorption equilibrium and we have an irreversible surface reaction.

Example 1.
Since we know all of the dynamic information describing adsorption, we just solve sysode.m, as we ordinarily would and treat adsorption and desorption like any other reaction.

There are five reactions in this example. The subscript b represents the bulk phase and the subscript a represents the adsorbed phase. A is the reactant. B is the product. S is a surface site.

adsorption of A:	$A_b + S \rightarrow A_a$	$r_1 = k_1 C_A \theta_S$
desorption of A:	$A_a \rightarrow A_b + S$	$r_2 = k_2 \theta_A$
surface rxn of A:	$A_a \rightarrow B_a$	$r_3 = k_3 \theta_A$
adsorption of B:	$B_b + S \rightarrow B_a$	$r_4 = k_4 C_B \theta_S$
desorption of B:	$B_a \rightarrow B_b + S$	$r_5 = k_5 \theta_B$

Input file provided below.

```
function dydt = sysodeinput(x,y,nvec);
%
% batch reactor
%
% reversible adsorption of A
% reaction on surface A --> B
% reversible adsorption of B
%
% sample usage:
```

```
% [y,x]=sysode(2,1000,0,20,[10,0,0,0]);
%
CA = y(1); % mol/liter
CB = y(2); % mol/liter
thetaA = y(3); % dimensionless
thetaB = y(4); % dimensionless
%
% stoichiometry
%
% adsorption of A
%
nuAb1 = -1;
nuBb1 = 0;
nuAa1 = 1;
nuBa1 = 0;
%
% desorption of A
%
nuAb2 = 1;
nuBb2 = 0;
nuAa2 = -1;
nuBa2 = 0;
%
% irreversible surface reaction
%
nuAb3 = 0;
nuBb3 = 0;
nuAa3 = -1;
nuBa3 = 1;
%
% adsorption of B
%
nuAb4 = 0;
nuBb4 = -1;
nuAa4 = 0;
nuBa4 = 1;
%
% desorption of B
%
nuAb5 = 0;
nuBb5 = 1;
nuAa5 = 0;
nuBa5 = -1;
%
% rate laws
%
thetaS = 1.0 - thetaA - thetaB;
k1 = 1.0;
r1 = k1*CA*thetaS; % mole/liter/sec
k2 = 1.0;
r2 = k2*thetaA; % mole/liter/sec
k3 = 1.0;
r3 = k3*thetaA; % mole/liter/sec
k4 = 1.0;
r4 = k4*CB*thetaS; % mole/liter/sec
k5 = 1.0;
r5 = k5*thetaB; % mole/liter/sec
```

```
%
% parameters
%
epsilon = 0.5;
rhor = 2.0; % mole/liter
%
% mole and energy balances
%
dydt(1) = (nuAb1*r1 + nuAb2*r2 + nuAb3*r3 + nuAb4*r4 + nuAb5*r5)/epsilon;
dydt(2) = (nuBb1*r1 + nuBb2*r2 + nuBb3*r3 + nuBb4*r4 + nuBb5*r5)/epsilon;
dydt(3) = (nuAa1*r1 + nuAa2*r2 + nuAa3*r3 + nuAa4*r4 + nuAa5*r5)/rhor;
dydt(4) = (nuBa1*r1 + nuBa2*r2 + nuBa3*r3 + nuBa4*r4 + nuBa5*r5)/rhor;
%
% eqlbm for initial condition
%
%KA = k1/k2;
%thetaA = KA*CA/(1+KA*CA)
%stop;
```

Example 2.
Again, since we know all of the dynamic information describing adsorption, we just solve sysode.m, as we ordinarily would and treat adsorption and desorption like any other reaction.

There are six reactions in this example. The subscript b represents the bulk phase and the subscript a represents the adsorbed phase. A is the reactant. B is the product. S is a surface site.

adsorption of A:	$A_b + S \rightarrow A_a$	$r_1 = k_1 C_A \theta_S$
desorption of A:	$A_a \rightarrow A_b + S$	$r_2 = k_2 \theta_A$
surface rxn of A:	$A_a \rightarrow B_a$	$r_3 = k_3 \theta_A$
surface rxn of B:	$B_a \rightarrow A_a$	$r_4 = k_4 \theta_B$
adsorption of B:	$B_b + S \rightarrow B_a$	$r_5 = k_5 C_B \theta_S$
desorption of B:	$B_a \rightarrow B_b + S$	$r_6 = k_6 \theta_B$

Input file provided below.

```
function dydt = sysodeinput(x,y,nvec);
%
% batch reactor
%
% reversible adsorption of A
% reversible reaction on surface A <--> B
% reversible adsorption of B
```

```
%
% sample usage:
% [y,x]=sysode(2,1000,0,20,[10,0,0.90909090909091,0]);
%
CA = y(1); % mol/liter
CB = y(2); % mol/liter
thetaA = y(3); % dimensionless
thetaB = y(4); % dimensionless
%
% stoichiometry
%
% adsorption of A
%
nuAb1 = -1;
nuBb1 = 0;
nuAa1 = 1;
nuBa1 = 0;
%
% desorption of A
%
nuAb2 = 1;
nuBb2 = 0;
nuAa2 = -1;
nuBa2 = 0;
%
% surface reaction of A to B
%
nuAb3 = 0;
nuBb3 = 0;
nuAa3 = -1;
nuBa3 = 1;
%
% surface reaction of B to A
%
nuAb4 = 0;
nuBb4 = 0;
nuAa4 = 1;
nuBa4 = -1;
%
% adsorption of B
%
nuAb5 = 0;
nuBb5 = -1;
nuAa5 = 0;
nuBa5 = 1;
%
% desorption of B
%
nuAb6 = 0;
nuBb6 = 1;
nuAa6 = 0;
nuBa6 = -1;
%
% rate laws
%
thetaS = 1.0 - thetaA - thetaB;
k1 = 1.0;
```

```
r1 = k1*CA*thetaS; % mole/liter/sec
k2 = 1.0;
r2 = k2*thetaA; % mole/liter/sec
k3 = 1.0;
r3 = k3*thetaA; % mole/liter/sec
k4 = 1.0;
r4 = k4*thetaB; % mole/liter/sec
k5 = 1.0;
r5 = k5*CB*thetaS; % mole/liter/sec
k6 = 1.0;
r6 = k6*thetaB; % mole/liter/sec
%
% parameters
%
epsilon = 0.5;
rhor = 2.0; % mole/liter
%
% mole and energy balances
%
dydt(1) = (nuAb1*r1 + nuAb2*r2 + nuAb3*r3 + nuAb4*r4 + nuAb5*r5 +
nuAb6*r6)/epsilon;
dydt(2) = (nuBb1*r1 + nuBb2*r2 + nuBb3*r3 + nuBb4*r4 + nuBb5*r5 +
nuBb6*r6)/epsilon;
dydt(3) = (nuAa1*r1 + nuAa2*r2 + nuAa3*r3 + nuAa4*r4 + nuAa5*r5 +
nuAa6*r6)/rhor;
dydt(4) = (nuBa1*r1 + nuBa2*r2 + nuBa3*r3 + nuBa4*r4 + nuBa5*r5 +
nuBa6*r6)/rhor;
%
% eqlbm for initial condition
%
%KA = k1/k2;
%thetaA = KA*CA/(1+KA*CA)
%stop;
```

Example 3.a.
In this case, the bulk and adsorption properties are not governed by known kinetics but rather by known equilibrium distributions. In this case, we can determine the behavior of the total amount of A and B in the system (from both bulk and adsorbed phases) from an ODE mass balance. The respective bulk density and fractional occupancy in the adsorbed phase comes from thermodynamic constraints. The thermodynamic constraints are a set of nonlinear algebraic equations. We use the Multivariate Newton Raphson Method with Numerical approximations to the Derivatives (NRNDN.m) to solve the algebraic constraints at each time step.

This version of solving the problem will return only the plots of the total amount of A and B in the system as a function of time. It has calculated the bulk concentrations of A and B and the fractional occupancies of A and B as a function of time, but it didn't save them.

Before we run sysode.m, we initialize our Newton-Raphson initial guesses by setting the variable ifirst to 1 and making it global so that it can be found in the appropriate routine (sysodeinput.m).

```
global x0 ifirst
ifirst = 1;
[y,x]=sysode(2,1000,0,10,[9.9,0.1]);
```

The sysodeinput.m file with the ODEs is given immediately below. The syseqninput.m file with the algebraic equations follows.

sysodeinput.m

```
function dydt = sysodeinput(x,y,nvec);
%
%  batch reactor
%  equilibrium adsorption of A
%  reaction on surface A --> B
%  equilibrium adsorption of B
%
%  sample usage:
%  [y,x]=sysode(2,1000,0,40,[9.9,0.1]);
%
global KA KB rhoAt rhoBt mwA mwB rhor eps
rhoAt = y(1); % mol/liter
rhoBt = y(2); % mol/liter
%
%  stoichiometry
%
%
% irreversible surface reaction
%
nuAt1 = -1;
nuBt1 = 1;
%
%  parameters
%
eps = 0.5; % void fraction
rhor = 2.0;  % mole/liter
KA = 1.0e-1; % liter/mole, adsorption equilibrium coefficient for A
```

```
KB = 1.0e-2; % liter/mole, adsorption equilibrium coefficient for A
mwA = 0.1060; % kg/mole
mwB = mwA; % kg/mole

%
% determine bulk densities and surface occupancies
% this is a set of 4 nonlinear algebraic equations
% 4 unknowns - rhoAb, rhoBb, thetaA, thetaB
%
global x0 ifirst
if (ifirst == 1)
   x0 = [3,3,0.3,0.3];
   ifirst = 0;
end
tol = 1.0e-6;
iprint = 0;
[f,x] = NRNDN(x0,tol,iprint);
x0 = x;
%
rhoAb = x(1);
rhoBb = x(2);
thetaA = x(3);
thetaB = x(4);
%
%  rate laws
%
thetaS = 1.0 - thetaA -thetaB;
k1 = 1.0;
r1 = k1*thetaA; % mole/liter/sec
%
%  mass balance on total amount of A and B in reactor
%
dydt(1) = nuAt1*r1;
dydt(2) = nuBt1*r1;
```

syseqninput.m

```
function f = syseqninput(y);
%
% binary adsorption equilibrium
%
global KA KB rhoAt rhoBt mwA mwB rhor eps
rhoAb = y(1);
rhoBb = y(2);
thetaA = y(3);
thetaB = y(4);
%
%  langmuir isotherm on A
denom = 1.0 + KA*rhoAb + KB*rhoBb;
f(1) = thetaA - KA*rhoAb/denom;
%  langmuir isotherm on B
f(2) = thetaB - KB*rhoBb/denom;
%  total mass balance on A
f(3) = rhoAt - rhoAb*eps - thetaA*rhor*mwA;
%  total mass balance on B
f(4) = rhoBt - rhoBb*eps - thetaB*rhor*mwB;
```

Example 3.b.
This is the same problem as 3.a., but now we add a "driver.m" code at the front end so that we can plot and save all of the information, including the bulk concentrations of A and B and the fractional occupancies of A and B as a function of time.

We do not make changes to sysode.m or to NRNDN.m or to syseqninput.m. We make a minor change to the end of sysodeinput.m adding lines to save data.

These last three lines are added to the sysodeinput.m provided in Example 3.a.

```
%
%  save data
%
global icount ysave
icount = icount + 1;
ysave(icount,1:6) = [rhoAt rhoBt rhoAb rhoBb thetaA thetaB];
```

We then run a "driver code", as provided below. Driver sets up a couple extra global variables for saving data. It calls sysode.m. When the simulation is complete, it takes the data (every point for the Euler method or every fourth point for the Fourth order Runge-Kutta method) and plots it in a second figure and saves it in a second output file, sysode_v02.out. (The last bit of the code that writes the result is shrunk so that when you copy and paste, there are no line cut-off errors.)

driver.m

```
%
clear all;
close all;
global x0 ifirst
global icount ysave
ifirst = 1;
icount = 0;
nc = 2; % number of components
m = 2;
n = 1000;
[y,x]=sysode(m,n,0,40,[9.9,0.1]);
%
%  for Euler take all points
%
xplot(1:n) = x(1:n);
if (m == 1)
   yplot(:,:) = ysave(:,:);
elseif (m==2)
%
%  for Runge-Kutta fourth order take only every fourth point
%
   for i = 1:1:n
      yplot(i,:) = ysave(4*i,:);
   end
end
%
%  STEP FOUR.  PLOT THE RESULT
```

```
%
figure(2);
inorm = 1;
if (inorm == 1)
%   scale total and bulk densities, don't scale fractional occupancies
   for i = 2*nc:-1:1
       yplot(:,i) = yplot(:,i)/yplot(1,1);
   end
end
o = min(size(yplot));
for i = 1:o
  if (i==1)
     plot (xplot,yplot(:,i),'k-'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1');
  elseif (i==2)
     plot (xplot,yplot(:,i),'r-'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2');
  elseif (i==3)
     plot (xplot,yplot(:,i),'b-'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3');
  elseif (i==4)
     plot (xplot,yplot(:,i),'g-'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3','4');
  elseif (i==5)
     plot (xplot,yplot(:,i),'m-'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3','4','5');
  elseif (i==6)
     plot (xplot,yplot(:,i),'k:'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3','4','5','6');
  elseif (i==7)
     plot (xplot,yplot(:,i),'r:'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3','4','5','6','7');
  elseif (i==8)
     plot (xplot,yplot(:,i),'b:'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3','4','5','6','7','8');
  elseif (i==9)
     plot (xplot,yplot(:,i),'g:'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3','4','5','6','7','8','9');
  elseif (i==10)
     plot (xplot,yplot(:,i),'m:'), xlabel( 'x' ), ylabel ( 'y' )
     legend('1','2','3','4','5','6','7','8','9','10');
 else
     plot (xplot,yplot(:,i),'k-'), xlabel( 'x' ), ylabel ( 'y' )
  end
  hold on
end
hold off
```

```
%
%  STEP FIVE.  WRITE THE RESULT TO sysode.out
%
op1=o+1;
fid = fopen('sysode_v02.out','w');
if (op1 == 2)
  fprintf(fid,'x             y(1) \n');
  fprintf(fid,'%13.7e %13.7e   \n', [xplot;yplot']);
elseif (op1 == 3)
  fprintf(fid,'x             y(1)          y(2) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e  \n', [xplot;yplot']);
elseif (op1 == 4)
  fprintf(fid,'x             y(1)          y(2)          y(3) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e %13.7e \n', [xplot;yplot']);
elseif (op1 == 5)
  fprintf(fid,'x             y(1)          y(2)          y(3)          y(4) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e %13.7e %13.7e \n', [xplot;yplot']);
elseif (op1 == 6)
  fprintf(fid,'x             y(1)          y(2)          y(3)          y(4)          y(5) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e  %13.7e %13.7e %13.7e \n', [xplot;yplot']);
elseif (op1 == 7)
  fprintf(fid,'x             y(1)          y(2)          y(3)          y(4)          y(5)          y(6) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e  \n', [xplot;yplot']);
elseif (op1 == 8)
  fprintf(fid,'x             y(1)          y(2)          y(3)          y(4)          y(5)          y(6)          y(7) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e  \n', [xplot;yplot']);
elseif (op1 == 9)
  fprintf(fid,'x             y(1)          y(2)          y(3)          y(4)          y(5)          y(6)          y(7)          y(8) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e \n', [xplot;yplot']);
elseif (op1 == 10)
  fprintf(fid,'x             y(1)          y(2)          y(3)          y(4)          y(5)          y(6)          y(7)          y(8)          y(9) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e  \n', [xplot;yplot']);
elseif (op1 == 11)
  fprintf(fid,'x             y(1)          y(2)          y(3)          y(4)          y(5)          y(6)          y(7)          y(8)          y(9)          y(10) \n');
  fprintf(fid,'%13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e %13.7e  %13.7e %13.7e \n', [xplot;yplot']);
end
fclose(fid);
```