Reactor Design I: Tutorial 1. Reactor Systems

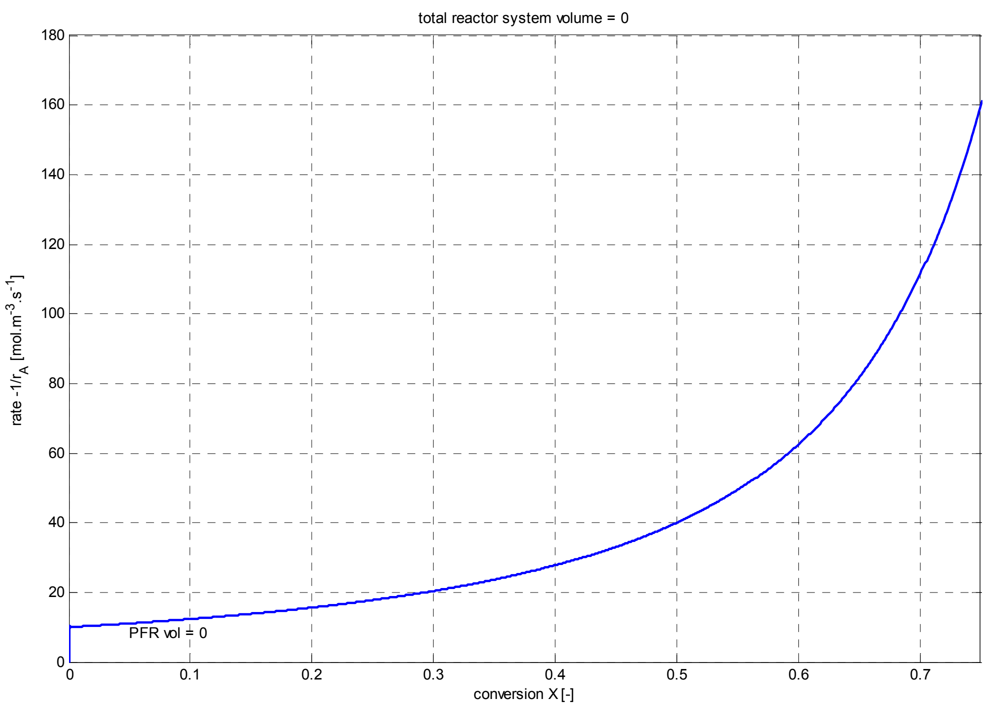
Department of Chemical Engineering, University of Cape Town
R Rawatlal, February 2006

Question 1

a) Develop the relationship between X, the overall conversion from a series of reactors and ΔX_i', the conversion in reactor *i* in the series (based on the flowrate to that reactor)
b) Develop the reactor design equation from first principles for the CSTR and PFR. Discuss how the reactor volume can be obtained from the $1/r_A$ vs X curve in each case.

Question 2

Consider the $1/r_A$ vs X curve shown in the figure given below:



What is the total reactor system volume required to achieve a final conversion of 70% for the following sets of reactors-in-series configurations:

a) one big CSTR
b) one big PFR
c) a CSTR followed by a PFR (intermediate conversion = 0.35)
d) a PFR followed by a CSTR (intermediate conversion = 0.35)
e) 4 CSTRs in series
f) 4 PFRs in series

Prepare your results in a table and discuss these results. If the reactors were in parallel, how would the results change?

Question 3

Develop an algorithm for determining the reaction rate constant and order of reaction for reactant A which disappears according to: $dC_A/dt = -kC_A^n$

Write matlab code that performs this search for a given data set (batch data… A is the reactant):

T = [0 555 1111 1666 2222 2777 3333 3888 4444 5000] s

CA = [1.000 0.642 0.473 0.375 0.310 0.264 0.230 0.204 0.183 0.166] mol.m^{-3}

Solutions
Question 1.a

KDI : TUT 01 (R. Rawatlal)

Question 1 a

let's say we have a series of reactors like so:

we define the o/all conversion $X_i = \dfrac{F_{A0} - F_{Ai}}{F_{A0}}$

whereas the conversion in the i^{th} reactor is

$$\Delta X_i' = \frac{F_{Ai} - F_{Ai-1}}{F_{Ai}}$$

whats the general relationship between X_i and $\Delta X_i'$?

for 1 reactor, obviously, $X_1 = \Delta X_1'$

for 2 reactors: $X_2 = \dfrac{F_{A0} - F_{A2}}{F_{A0}} = 1 - \dfrac{F_{A2}}{F_{A0}}$ or $\dfrac{F_{A2}}{F_{A0}} = 1 - X_2$

$$\Delta X_1' = \frac{F_{A0} - F_{A1}}{F_{A0}} \rightarrow F_{A1} = F_{A0}(1 - \Delta X_1')$$

$$\Delta X_2' = \frac{F_{A1} - F_{A2}}{F_{A1}} = \frac{F_{A0}(1 - \Delta X_1') - F_{A2}}{F_{A0}(1 - \Delta X_1')}$$

$$= 1 - \frac{F_{A2}}{F_{A0}} \frac{1}{1 - \Delta X_1'}$$

$$= 1 - \frac{1 - X_2}{1 - \Delta X_1'}$$

or $(1 - X_2) = (1 - \Delta X_1')(1 - \Delta X_2')$

generally, can show that

$$(1 - X_i) = \prod_{j=1}^{i} (1 - \Delta X_j') \longrightarrow$$

Well, to tell the truth, this is not at all rigous. We can only say that we now have an indication, or a suspicion, that

$$(1-X_N)=\prod_{i=1}^{N}(1-\Delta X_i^{'})$$

We still need to prove that this holds for all choices of N.

Let's prove it by induction.

Proof by Induction works as follows:
1. Prove your hypothesis is true for N = 1,
2. Then prove that IF it holds for any N, then necessarily it also holds for N'=N+1.
(Think about numerically solving ODEs from some initial point to understand why this works.)

Anyway, for N = 1, statement gives:

$$(1-X_1)=(1-\Delta X_1^{'})$$

… which is clearly true because

$$X_1=\frac{F_{A,0}-F_{A,1}}{F_{A,0}}=\Delta X_1^{'}$$

So part 1 is satisfied.

Part 2: Let's assume $(1-X_N)=\prod_{i=1}^{N}(1-\Delta X_i^{'})$

For the case N+1:

$$(1-X_{N+1})=F_{A,N+1}/F_{A,0}$$

And

$$\prod_{i=1}^{N+1}(1-\Delta X_i^{'})=(1-\Delta X_{N+1}^{'})\left[\prod_{i=1}^{N}(1-\Delta X_i^{'})\right]$$

$$=\frac{F_{A,N+1}}{F_{A,N}}\left[\frac{F_{A,N}}{F_{A,0}}\right]$$

(nb: we are using the assumption $(1-X_N)=\prod_{i=1}^{N}(1-\Delta X_i^{'})$)

$$=\frac{F_{A,N+1}}{F_{A,0}}$$

$$=(1-X_{A,N+1})$$

… which completes the proof rigorously!

Question 1.b

See class notes (Prof Harrison's section)

Question 2

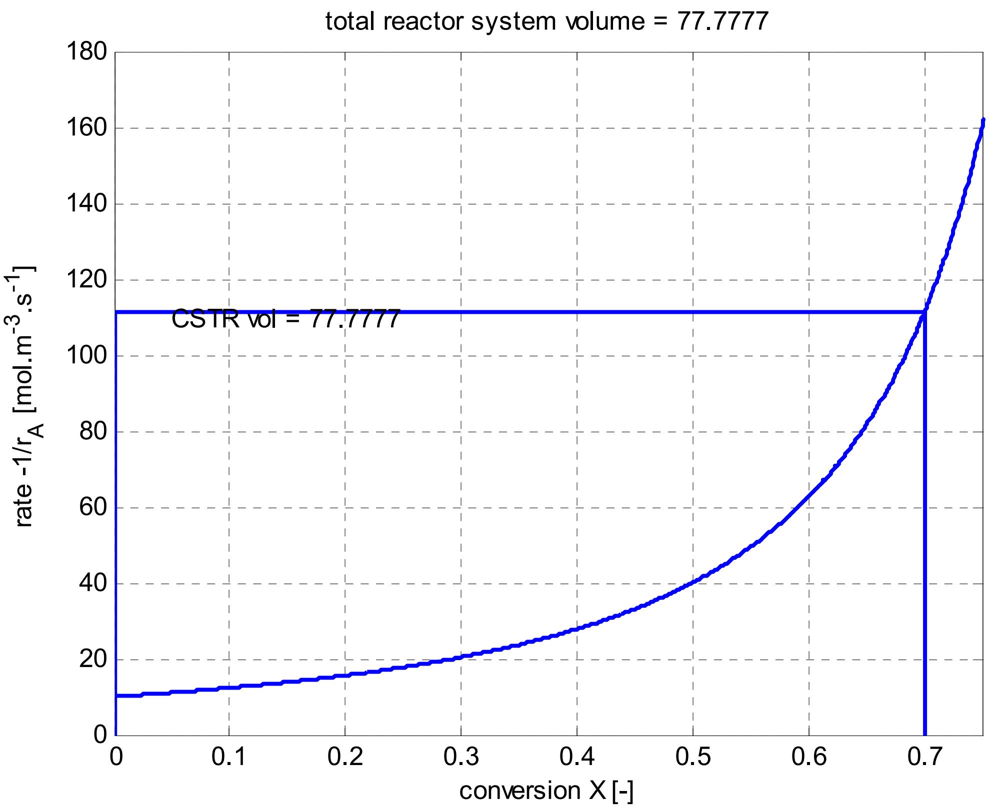
Generally speaking, we are going to use the two design equations:

PFR: $V = F_{A0} \int_0^{X_1} \frac{dX}{-r_A}$

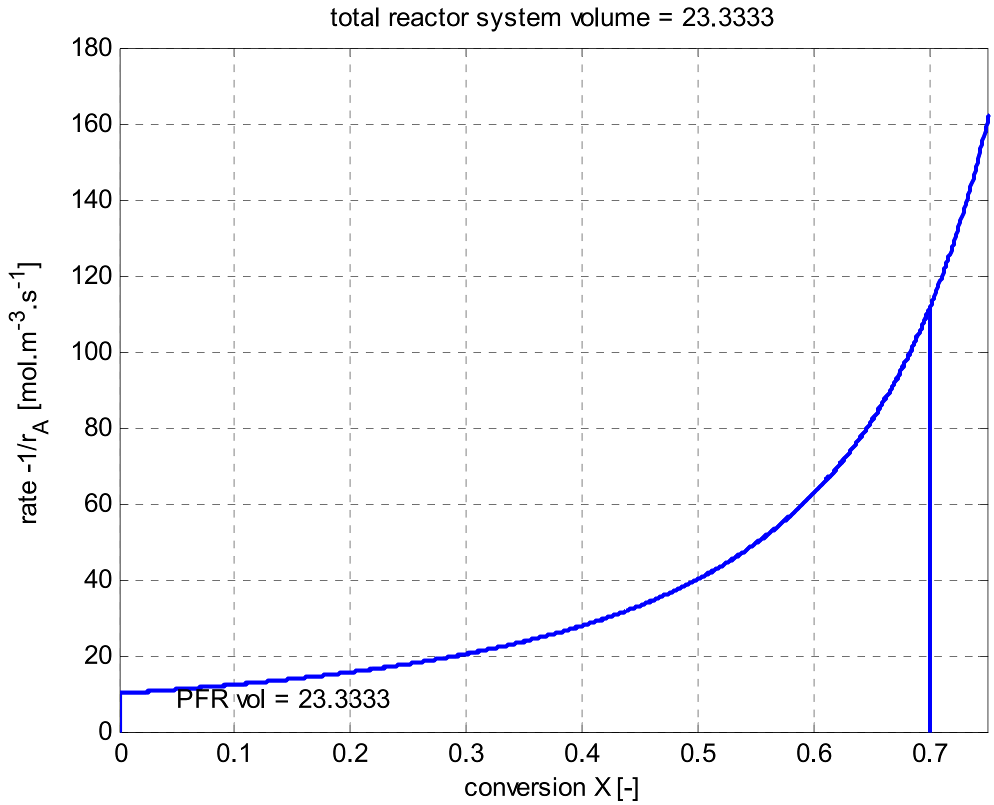
CSTR: $V = \frac{F_{A0} X}{-r_{A2}}$

Clearly then, the volume of the PFR with be the area under the curve (over the required conversion range) of $1/r_A$ vs X whereas the volume of the CSTR will be the volume of the 'block' at the end point given at X.

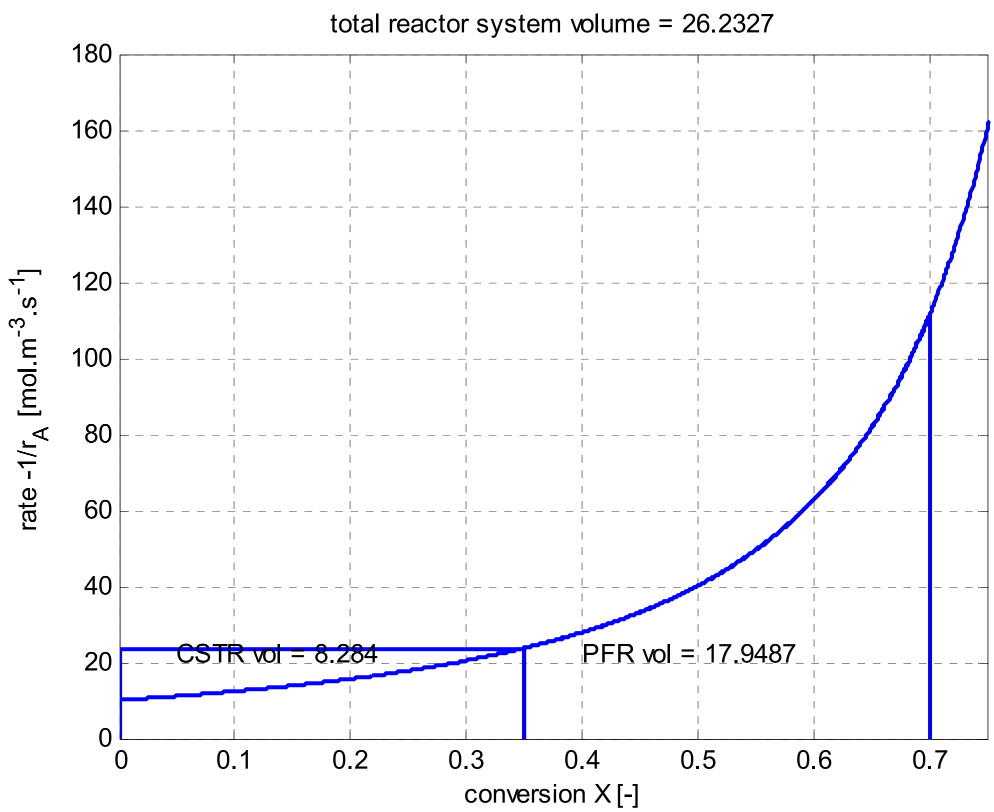
a) One big CSTR:



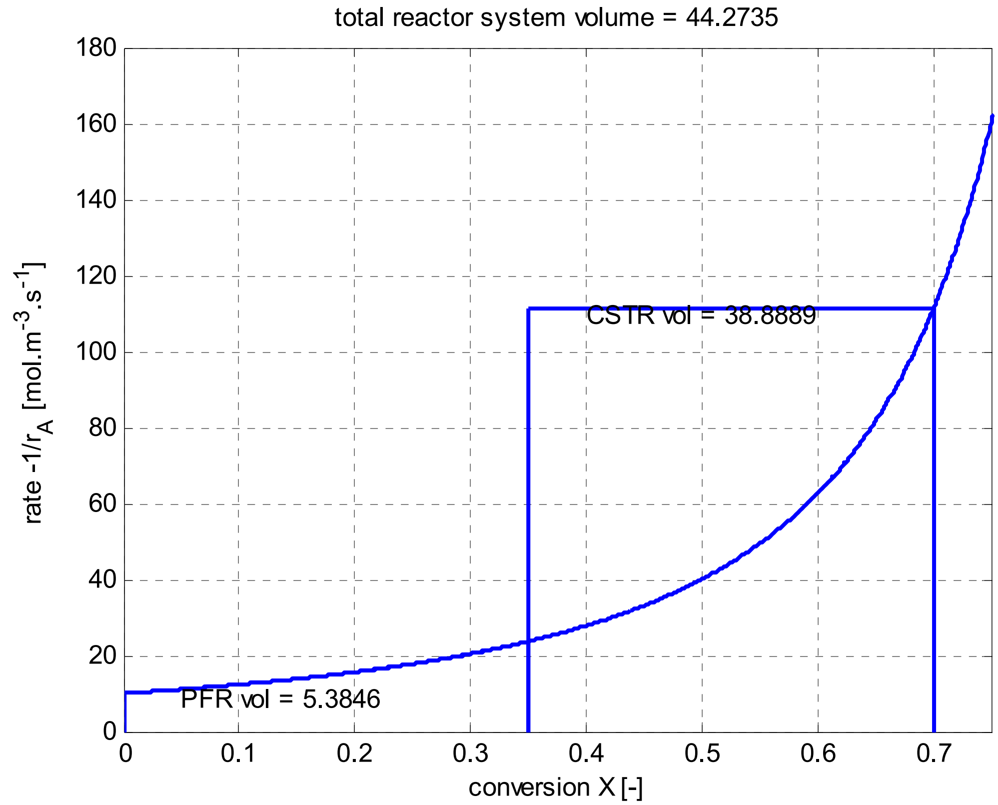
b) one big PFR:



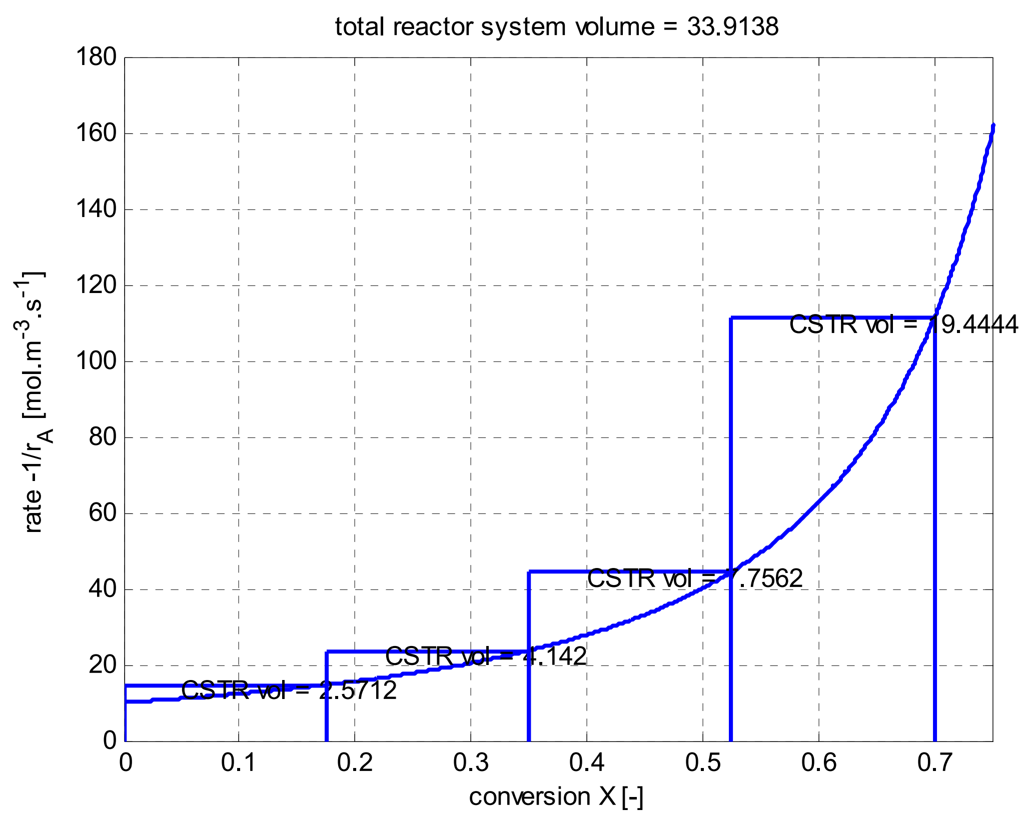
c) a CSTR followed by a PFR (intermediate conversion = 0.35)



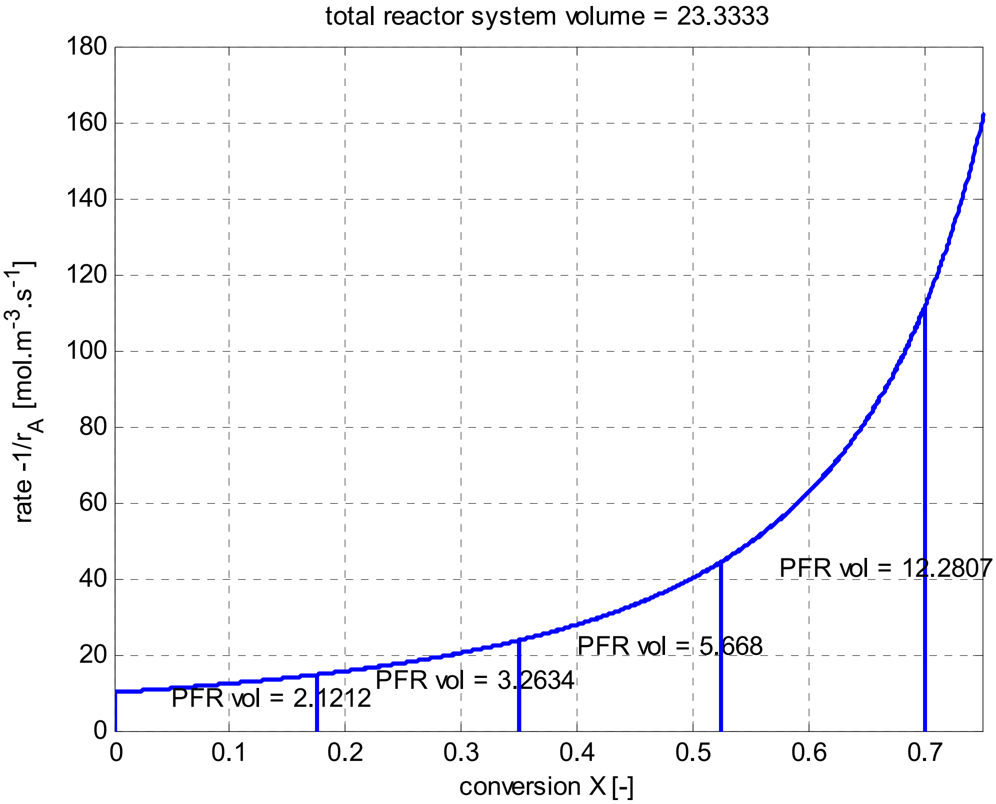
d) a PFR followed by a CSTR (intermediate conversion = 0.35)



e) 4 CSTRs in series



f) 4 PFRs in series



Clearly, the more like a PFR the system looks, the smaller the required volume.

Question 3
Develop an algorithm for determining the reaction rate constant and order of reaction for reactant A which disappears according to: $dC_A/dt = -kC_A^n$
Write matlab code that performs this search for a given data set (batch data… A is the reactant):
T = [0 555 1111 1666 2222 2777 3333 3888 4444 5000] s
CA = [1.000 0.642 0.473 0.375 0.310 0.264 0.230 0.204 0.183 0.166] mol.m^{-3}

Method 1: Assume rate orders and try to fit.
Fitting method: Define least squares index $I = \sum_i (C_{A,i}^{ext} - C_{A,i}^{mdl})$
Assume a first order model: $C_A^{mdl}(t) = C_{A0} \exp(-kt)$
And use a minimizer to find the k-value which corresponds to the smallest possible I.

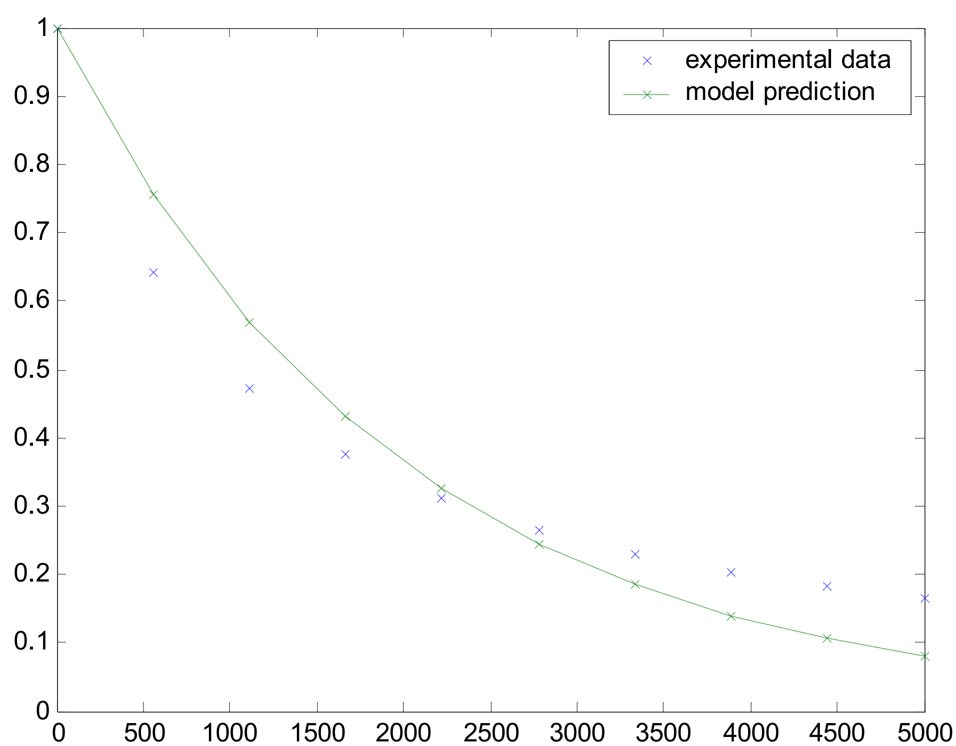
The code looks like:

```
function I = LSI1(k)
Caexp = [  1.000   0.642   0.473   0.375   0.310   0.264   0.230   0.204   0.183   0.166];
texp = [0   555   1111   1666  2222   2777   3333   3888   4444   5000];

Cmod = Caexp(1)*exp(-k*texp);

I = sum((Caexp-Cmod).^2);
```

Runfile:

```
close all;clear all;clc

p=fminsearch('LSI1',1e-3);
```

The result looks like:



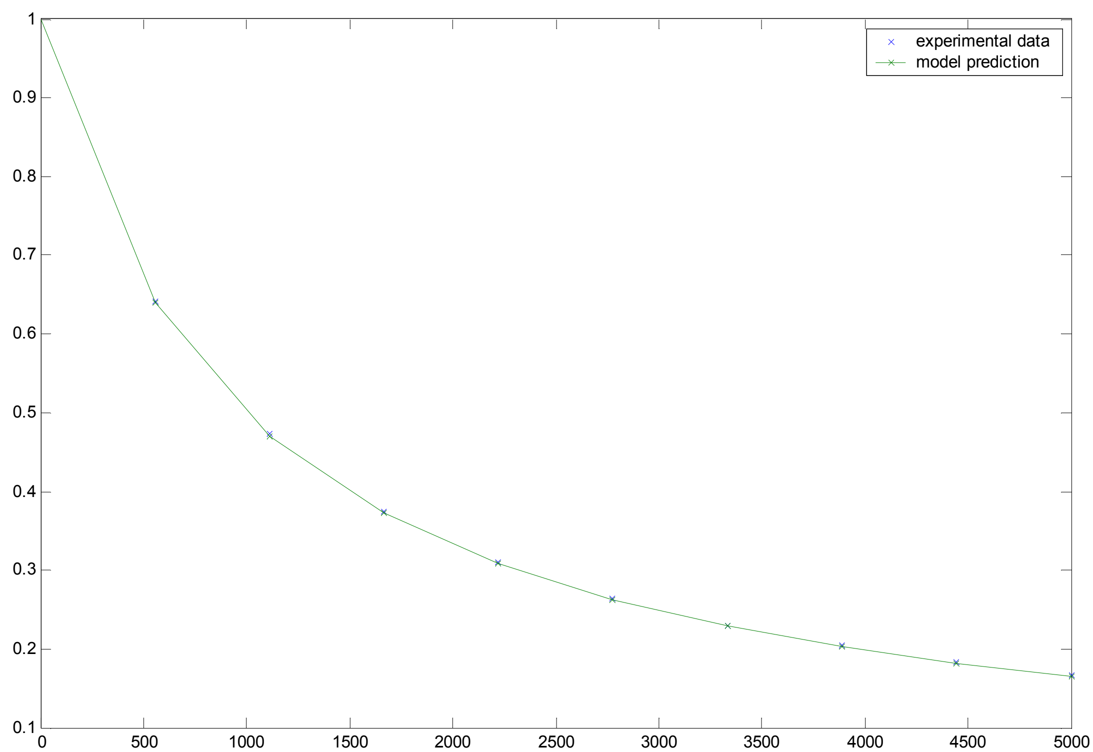
… which is rather poor.

Let's now try second order reaction:

$$C_A(t) = \frac{1}{kt + 1/C_{A,0}}$$

The code looks like:
The only change to the code is that we replace the old model:
Cmod = Caexp(1)*exp(-k*texp);
with the new
Cmod = 1./(k*texp+1/Caexp(1));

The result looks like:



This looks much better; the k-value is 1×10^{-3}.

The problem with this method is that it is not exhaustive; we have to assume something (the order) and then search. It is more robust to do away with such assumptions. Let's try to linearize the rate instead:

$$\frac{dC_A}{dt} = -kC_A^n$$

We need to do some data processing to get the derivative on the LHS. Let's use the approximation:

$$\frac{dC_A}{dt}(t = t_i) = \frac{1}{2}\left(\frac{C_{A,i+1} - C_{A,i}}{t_{i+1} - t_i} + \frac{C_{A,i} - C_{A,i-1}}{t_i - t_{i-1}} \right)$$

…and apply it to just the internal points (ignore the endpoints, since for these we can calculate a derivative on just one side, which might be inaccurate).

So: $\ln\left(-\frac{dC_A}{dt} \right) = \ln(k) + n\ln(C_A)$

Let's, instead of using the least squares index, define I to be the squared difference between the left and right hand sides of the equation.

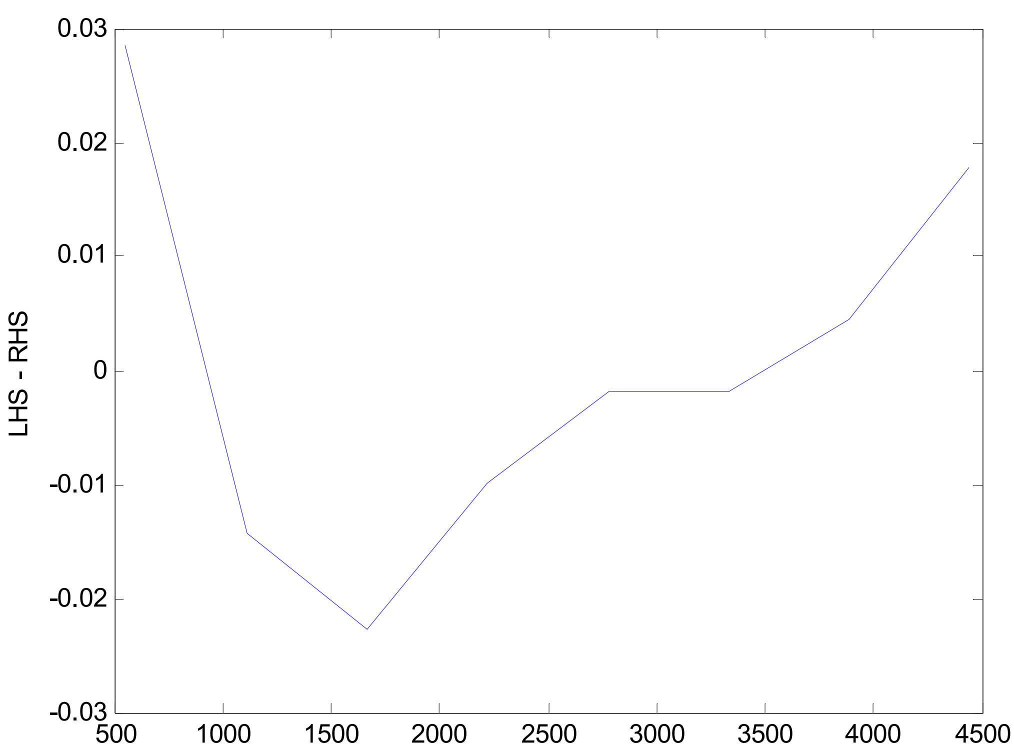
The code looks like:

```
function I = LSI2(p)
p
Caexp = [   1.000   0.642   0.473   0.375   0.310   0.264   0.230   0.204   0.183   0.166];
texp = [0   555    1111    1666  2222    2777    3333    3888    4444    5000];

k = p(1);
n = p(2);

for i =2:(length(texp)-1)
    dcdt(i-1) = 0.5*( (Caexp(i)-Caexp(i+1))/(texp(i)-texp(i+1)) + (Caexp(i)-Caexp(i-1))/(texp(i)-texp(i-1)) );
end;

t2 = texp(2:(length(texp)-1));
C2 = Caexp(2:(length(texp)-1));

LHS = log(-dcdt);
RHS = log(k)+n*log(C2);

figure(1)
plot(t2,LHS-RHS,'-')
drawnow
I = sum((LHS-RHS).^2);
ylabel('LHS - RHS')
```

main program:

```
close all;clear all;clc
p=fminsearch('LSI3',[1e-2 3]);
```

The result looks like:



Clearly, the fit is not so good. The parameters are OK: k = 0.00116 and n=2.088. However, the result is not as good as when we assumed 2nd order. The method, however, is more robust or more general. This is one problem we experience in parameter fitting: the more general the model, the greater the number of parameters and the harder it is to fit; the more specific the model, the smaller the space we can search for parameters.

So there isn’t any perfect answer; in regression, you have to find some middle ground between these two extremes if you want to make any real progress.