Maple Solutions to the Chemical Engineering Problem Set

Ross Taylor

Department of Chemical Engineering

Clarkson University

Potsdam, New York 13699-5705

taylor@sun.soe.clarkson.edu

Maple is a computer algebra system or CAS. Gonnet and Gruntz (Algebraic Manipulation Systems, in Encylcopedia of Computer Science and Engineering, 3rd Edition, Van Nostrand Reinhold, 1991) define computer algebra as follows:

"Computer algebra (sometimes called algebraic manipulation, or symbolic computation) can be defined to be computation with variables and constants according to the rules of algebra, analysis and other branches of mathematics, or formula manipulation involving symbols, unknowns, and formal operations rather than with conventional computer data of numbers and character strings."

What this means is that Maple is capable of far more than the simple (or not so simple) numerical solution of the equations that arise during the modelling of chemical engineering operations. It can be used in all facets of problem solving from helping us to derive and analyse the model equations in the first place, computing numerical solutions when needed, and graphical visualization of the results. In the worksheets written to solve the selection of chemical engineering problems we have tended to gloss over the derivation and analysis part, since such was not considered part of the assignment.

In creating the Maple solutions I considered whether or not to stick to standard Maple (i.e. Maple out of the box). It is important to recognise that Maple is a very powerful programming language in its own right and it is possible to teach Maple new tricks. I decided not to pass on the opportunity to use some of the tools that I have developed to make working with Maple somewhat simpler than it might otherwise have been. *Some* of the worksheets that accompany these notes therefore incorporate some commands that are not standard Maple. Readers can tell which worksheets fall into this category by the inclusion of a read statement followed by a file name. There are three such files used in these worksheets: utils.mpl, newton.mpl, and besirk. The first of these contains various utility routines designed in part to make it easier to work with subscripted and unsubscripted variables with the same root name (i.e. x and x_1). The other two files are described in the following paragraphs.

Maple contains a large number of routines for solving equations. One of these routines is fsolve, for

obtaining numerical solutions to systems of equations. We use fsolve in several of the examples. However, in this authors opinion fsolve is not very effective at solving systems of nonlinear equations (i.e. more than one), largely because the procedure allows the user no control over the starting point, or over the iteration history. To deal with this weakness in Maple I have implemented Newton's method for solving systems of equations in Maple. The procedure is moderately sophisticated and has numerous options that allow the user considerable control over how the calculations are to be done. The code has been part of the Maple Share Library (contributions by users) for some time. A more up to date version is included with these examples.

Maple out of the box is not capable of solving problems involving mixed systems of differential and algebraic equations (DAEs). However, in view of their importance in what we do with Maple, we have implemented a numerical method for solving DAE systems. Full details of the method are available in the worksheet BSIRKPAP.MWS that has been included with this collection of problem solutions. The code that implements the method is contained in the file BESIRK and is loaded into the relevant worksheets using the read command. This method is also used in an alternative worksheet to one of the other examples where Maple's own dsolve/numeric is another choice. Possible reasons for using BESIRK over dsolve/numeric include the fact that BESIRK is many times faster. It can also be useful in solving systems of PDEs obtained when systems of PDEs are approximated using the method of lines.

The code packages used here as well as the example files are available by anonymous ftp from ftp.clarkson.edu in the \pub\maple directory. Utilities are in the \utils subdirectory, Newton's method in \numerics, BESIRK in \besirk. Other exmaples are available in the \chemeng subdirectory. The worksheets that follow are in the \polymath subdirectory.

The Van der Waals Equation of State

In 1873 Van der Waals proposed an equation of state that serves as an inspiration to thermodynamicists more than 100 years after it was proposed. While the equation has several shortcomings from a theoretical and practical standpoint it remains useful as a starting point for studying cubic equations of state.

The Van der Waals equation of state has the following form

```
> VDWConsts := [R,a,b]:
> EOS[VDW]:=P=R*T/(v-b)-a/v^2: EOS[VDW];
```

$$P = \frac{R\,T}{v-b} - \frac{a}{v^2}$$

where P is the pressure, v is the specific volume and T is the temperature. a and b are parameters in the model (yet to be determined) and R is the gas constant. The first term on the right hand side is the attractive term, the second term accounts for repulsion forces that attempt to keep the individual molecules apart.

```
> VDWparams:={b = 1/8*R*T[c]/P[c], a = 27/64/P[c]*R^2*T[c]^2}: VDWparams;
```

$$\left\{ b = \frac{1}{8}\frac{R\,T_c}{P_c},\ a = \frac{27}{64}\frac{R^2\,T_c^{\,2}}{P_c} \right\}$$

Polynomial Forms of the VDW Equation

We can write the VDW EOS as a polynomial in volume as follows:

```
> numer(lhs(EOS[VDW])-rhs(EOS[VDW]))=0:
> polyv:=expand("/coeff(lhs("),v,3)): polyv; #Divide by leading coefficient
```

$$v^3 - v^2\,b - \frac{R\,T\,v^2}{P} + \frac{a\,v}{P} - \frac{a\,b}{P} = 0$$

We may also express the VDW equation in terms of the compressibility as follows

```
> subs(v=z*R*T/P,polyv):
> polyz:=collect(expand("/coeff(lhs("),z,3)),z): polyz;
```

$$z^3 + \left(-\frac{P\,b}{R\,T} - 1 \right) z^2 + \frac{P\,a\,z}{R^2\,T^2} - \frac{P^2\,a\,b}{R^3\,T^3} = 0$$

Both polynomial forms of the VDW equation are useful but the compressibility form is, perhaps, the most useful. We solve the compressibility polynomial to give z in terms of the parameters:

```
> zroots[VDW]:=solve(polyz,z): zroots[VDW];
```

$$\frac{\frac{1}{6}\%1^{1/3} - 6\,\%2 + \frac{1}{3}P\,b + \frac{1}{3}R\,T}{T\,R},\ \frac{-\frac{1}{12}\%1^{1/3} + 3\,\%2 + \frac{1}{3}P\,b + \frac{1}{3}R\,T + \frac{1}{2}I\sqrt{3}\left(\frac{1}{6}\%1^{1/3} + 6\,\%2\right)}{T\,R},$$

$$\frac{-\frac{1}{12}\%1^{1/3} + 3\,\%2 + \frac{1}{3}P\,b + \frac{1}{3}R\,T - \frac{1}{2}I\sqrt{3}\left(\frac{1}{6}\%1^{1/3} + 6\,\%2\right)}{T\,R}$$

$$\%1 := 72 P^2 a b - 36 P a R T + 8 P^3 b^3 + 24 P^2 b^2 R T + 24 R^2 T^2 P b + 8 R^3 T^3 + 12 (12 P^5 a b^4 + 24 P^4 a^2 b^2 + 12 P^3 a^3 - 60 P^3 a^2 R T b - 3 P^2 a^2 R^2 T^2 + 36 P^4 a b^3 R T + 36 P^3 a b^2 R^2 T^2 + 12 P^2 a b R^3 T^3)^{1/2}$$

$$\%2 := \frac{\frac{1}{3} P a - \frac{1}{9} P^2 b^2 - \frac{2}{9} P b R T - \frac{1}{9} R^2 T^2}{\%1^{1/3}}$$

Note that there are three solutions (as we should have anticipated since the polynomial is a cubic). The other two solutions include the term *I* (the square root of -1). However, this does not mean that the other two roots are complex. Under certain conditions all three roots are real and under other conditions only one root is real as we shall demonstrate below.

It is worth noting here that even though explicit analytical expressions for the roots may be obtained, these formulae are rarely used in engineering computations and iterative methods are employed instead. The reason is that thermodynamic computations often involve the repeated evaluation of the compressibility under slightly varying conditions and under these circumstances it proves to be more efficient (from a computational perspective) to use an iterative method. We shall illustrate the use of iterative methods in later examples.

Dimensionless (Reduced) form of the Equation of State

It is sometimes useful to represent thermodynamic functions in terms of so-called reduced properties. The reduced pressure, volume and temperature are defined as follows:

```
> Prdef := P[r]=P/P[c]: Vrdef := v[r]=v/v[c]: Trdef := T[r]=T/T[c]: Prdef, Vrdef, Trdef;
```

$$P_r = \frac{P}{P_c}, v_r = \frac{v}{v_c}, T_r = \frac{T}{T_c}$$

The critcial compressibility is, therefore, defined by

```
> zcrit := Subs(State(c),z=z[c],zdef): zcrit;
```

$$z_c = \frac{P_c v_c}{R T_c}$$

For the VDW EOS this has a special value

```
> Subs(EOSParams[VDW],zcrit);
```

$$z_c = \frac{3}{8}$$

suggesting that all fluids have the same critical compressibility with a value of 0.375. In fact, the critical compressibilities of a great many fluids have almost the same value. However, that value is closer to 0.25. This means that the VDW EOS will not be all that succesful at predicting densities in the critical region even though it does provide a qualitatively accurate model of the PVT behavior of many real fluids.

In what follows we use the above definitions of the reduced properties to rewrite the VDW EOS in

dimensionless form (in terms of the reduced properties).

```
> Solve([Prdef,Vrdef,Trdef],[P,v,T]):
> Subs(",EOSParams[VDW],EOSp[VDW]):
> "/P[c]:
> EOSr[VDW]:=collect(",[P[c],T[r],T[c],R]): EOSr[VDW];
```

$$P_r = \frac{T_r}{\frac{3}{8} v_r - \frac{1}{8}} - \frac{3}{v_r^2}$$

Example 1

Calculate the molar volume and compressibility of ammonia at a pressure of 56 atm and a temperature of 450 K.

First we need to know the critical properties of ammonia. They are as follows

```
> CriticalProps[NH3]:={T[c] = 405.649994, P[c] = 11280000.0}: CriticalProps[NH3];
```

$$\{T_c = 405.649994,\ P_c = .112800000\ 10^8\}$$

where the temperature is in kelvin and the pressure in pascals.
The temperature and pressure of interest (in the same units) are

```
> Specs:={P=56*101325,T=450};
```

$$Specs := \{P = 5674200,\ T = 450\}$$

We use the volume polynomial form of the VDW EOS.

```
> Veqn:=subs(VDWparams,CriticalProps[NH3],Specs,R=8314.3,polyv): Veqn;
```

$$v^3 - .6967513834\ v^2 + .07497636489\ v - .002802221485 = 0$$

We may now invoke fsolve to obtain a numerical answer.

```
> result :=fsolve(Veqn,v);
```

$$result := .5747922281$$

Note that Maple found only one root. To find the others we need to use fsolve as follows

```
> result :=fsolve(Veqn,v,complex);
```

$$result := .06097957764 - .03401001901\ I,\ .06097957764 + .03401001901\ I,\ .5747922281$$

which shows that the one root found earlier was the only real root (under these conditions).

Before we leave this example let us examine the compressibility polynomial.

```
> zeqn:=subs(VDWparams,CriticalProps[NH3],Specs,polyz): zeqn;
```

$$z^3 - 1.056681915\ z^2 + .1724476380\ z - .009774662321 = 0$$

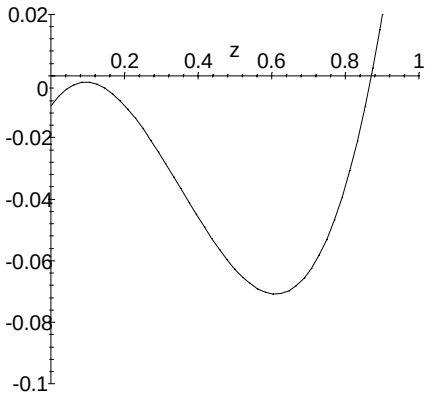
We can solve this polynomial using the `fsolve` command.

```
> result:=fsolve(zeqn,z,complex): result;
```

$$.09248064432 - .05157904649\ I,\ .09248064432 + .05157904649\ I,\ .8717206264$$

where the three compressibilities correspond to the three volumes computed before. A plot of the compressibility polynomial shows the real root.

```
> plot(lhs(zeqn),z=0..1,-0.1..0.02);
```



This illustration clearly shows the only real root; the two complex roots are near the maximum in the curve. Under slightly different conditions the curve will cross the z axis in three places and we would have three real roots.

>

>

Example 2 and 3

Calculate the compressibility of ammonia at a reduced pressures of 1, 2, 4, 10, and 20 at a temperature of 450 K.

The critical properties of ammonia are

```
> CriticalProps[NH3]:={T[c] = 405.649994, P[c] = 11280000.0}: CriticalProps[NH3];
```

$$\{T_c = 405.649994,\ P_c = .112800000\ 10^8\}$$

where the temperature is in kelvin and the pressure in pascals.

The reduced pressure is related to the actual pressure by

```
> prdef:= P[c]=P[r]/P: prdef;
```

$$P_c = \frac{P_r}{P}$$

We will use this relation in the substitution to follow

The temperature and pressure of interest (in the same units) are

```
> Tspec:=T=450: Tspec;
```

$$T = 450$$

The desired dimensionless pressures are given in a list as

```
> prlist := [1,2,4,10,20]: prlist;
```

$$[1, 2, 4, 10, 20]$$

We compute the compressibility at all reduced pressures in one loop

>

```
> for Pr in prlist do
```

```
    lprint(`Reduced pressure is `, Pr);
    zeqn:=subs(VDWparams,CriticalProps[NH3],P=P[r]*P[c],CriticalProps[NH3],Tspec,P[r]=Pr,polyz):
    result:=fsolve(zeqn,z,complex);
    print(result);
  od:
Reduced pressure is    1
                .2046555415 – .1141739223 I, .2046555415 + .1141739223 I, .7033694710
Reduced pressure is    2
                .3799117065 – .4330977317 I, .3799117065 + .4330977317 I, .4655376950
Reduced pressure is    4
                .3597111035 – .8460262618 I, .3597111035 + .8460262618 I, .7313000090
Reduced pressure is    10
                .2965246614 – 1.559052594 I, .2965246614 + 1.559052594 I, 1.533756216
Reduced pressure is    20
                .2346691122 – 2.344033838 I, .2346691122 + 2.344033838 I, 2.784272854
```

We see that there is only one real root for all conditions considered above.

The compressibility of ammonia is the following function of the reduced pressure.

```
> zeqn:=subs(VDWparams,CriticalProps[NH3],P=P[r]*P[c],CriticalProps[NH3],Tspec,polyz): zeqn;
```

$$z^3 + (-.1126805539\,P_r - 1)\,z^2 + .3428164951\,P_r\,z - .03862875255\,P_r^{\,2} = 0$$

We can plot the roots of this equation as a function of reduced pressure as follows. First we solve the polynomial

```
> zroots := solve(zeqn,z);
```

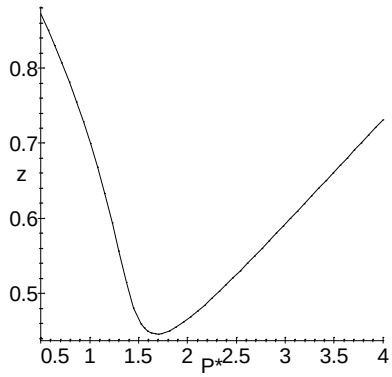
$$zroots := \frac{1}{30000000000}\,\%1^{1/3} - 30000000000\,\%2 + \frac{1126805539}{30000000000}P_r + \frac{1}{3},\ -\frac{1}{60000000000}\,\%1^{1/3} + 15000000000\,\%2 + \frac{1126805539}{30000000000}P_r + \frac{1}{3} + \frac{1}{2}I\sqrt{3}\left(\frac{1}{30000000000}\,\%1^{1/3} + 30000000000\,\%2\right),$$
$$-\frac{1}{60000000000}\,\%1^{1/3} + 15000000000\,\%2 + \frac{1126805539}{30000000000}P_r + \frac{1}{3} - \frac{1}{2}I\sqrt{3}\left(\frac{1}{30000000000}\,\%1^{1/3} + 30000000000\,\%2\right)$$

$$\%1 := 385749494613971554125000000000\,P_r^{\,2} - 120463256625000000000000000000\,P_r + 14306945391790266831278058 19\,P_r^{\,3} + 10000000000000000000000000000 + 15000000000\left(-154390391462129611370152286622000000000\,P_r^{\,3} + 53469097304051310302494518807392771723 7\,P_r^{\,4} + 11097558266205502797000000000000000000\,P_r^{\,2} + 66319134394299482270869525207921306 14\,P_r^{\,5}\right)^{1/2}$$

$$\%2 := \frac{\frac{321235351}{3600000000} P_r - \frac{1269690722721080521}{900000000000000000000} P_r^2 - \frac{1}{9}}{\%1^{1/3}}$$

and plot the first root as a function of reduced pressure.

```
> plot(zroots[1],P[r]=0.5..4,labels=[`P*`,`z`]);
```



where we use the label *P** to denote the reduced pressure.

We can obtain a standard compressibility plot by repeating these calculations for other temperatures and displaying the results together. We leave this as an exercise for the reader.

Steady State Material Balances on a Separation Train

We begin by creating a list of component identities. We could use numbers, chemical formulae, the component names; here we use the first letter of the component names.

```
> components:=[X,S,T,B];
```

$$components := [X, S, T, B]$$

Next, we create a list of units. This problem has three distillation columns which shall be identified as follows:

```
> Units:=[Col1,Col2, Col3];
```

$$Units := [Col1, Col2, Col3]$$

The streams will also have to be identified. Again, we could use any indexing method that is convenient. Here we number the streams

```
> Streams:=[seq(i,i=1..7)];
```

$$Streams := [1, 2, 3, 4, 5, 6, 7]$$

The number of streams is

```
> NumStreams := nops(Streams);
```

$$NumStreams := 7$$

The next step is to set up the material balances. To help us do this in a systematic way we create lists of the input streams to and output streams leaving each process unit. The streams entering and leaving the first distillation column are

```
> Inputs[Col1]:=[1]; Outputs[Col1]:=[2,3];
```

$$Inputs_{Col1} := [1]$$

$$Outputs_{Col1} := [2, 3]$$

The streams entering and leaving the second column are

```
> Inputs[Col2]:=[2]; Outputs[Col2]:=[4,5];
```

$$Inputs_{Col2} := [2]$$

$$Outputs_{Col2} := [4, 5]$$

and the streams entering and leaving the third column are

```
> Inputs[Col3]:=[3]; Outputs[Col3]:=[6,7];
```

$$Inputs_{Col3} := [3]$$

$$Outputs_{Col3} := [6, 7]$$

The component material balances for all units can now be created automatically as follows:

```
> j:='j': i:='i':
  for u in Units do
   lprint(`Material Balances for `,u);
   for i in components do
    CMB[i,u] := add( F[j] * x[i,j], j = Inputs[u]) = add( F[j] * x[i,j], j = Outputs[u]);
    print(CMB[i,u]);
  od; od;
```

```
Material Balances for    Col1
```

$$F_1 x_{X,1} = F_2 x_{X,2} + F_3 x_{X,3}$$

$$F_1 x_{S,1} = F_2 x_{S,2} + F_3 x_{S,3}$$

$$F_1 x_{T,1} = F_2 x_{T,2} + F_3 x_{T,3}$$

$$F_1 x_{B,1} = F_2 x_{B,2} + F_3 x_{B,3}$$

```
Material Balances for    Col2
```

$$F_2 x_{X,2} = F_4 x_{X,4} + F_5 x_{X,5}$$

$$F_2 x_{S,2} = F_4 x_{S,4} + F_5 x_{S,5}$$

$$F_2 x_{T,2} = F_4 x_{T,4} + F_5 x_{T,5}$$

$$F_2 x_{B,2} = F_4 x_{B,4} + F_5 x_{B,5}$$

```
Material Balances for    Col3
```

$$F_3 x_{X,3} = F_6 x_{X,6} + F_7 x_{X,7}$$

$$F_3 x_{S,3} = F_6 x_{S,6} + F_7 x_{S,7}$$

$$F_3 x_{T,3} = F_6 x_{T,6} + F_7 x_{T,7}$$

$$F_3 x_{B,3} = F_6 x_{B,6} + F_7 x_{B,7}$$

In the above Maple construction we have created the component material balances for each component (the inner loop) and for each process unit (the outer loop).

The total molar flows are given the symbol *F* and *x* refers to the mole fraction of some component. The first index of the component mole fraction identifies the component in question, the second associates that quantity with a particular process stream. This double loop will work with all simple material balances regardless of complexity provided we have identified the components, the units and the inputand output streams associated with each unit.

The mole fraction summation equations can be created as follows:

```
> SumEqn:='SumEqn': i:='i':
  for j in Streams do
    SumEqn[j]:=add(x[i,j],i=components)=1;
    print(SumEqn[j]);
  od:
```

$$x_{X,1} + x_{S,1} + x_{T,1} + x_{B,1} = 1$$

$$x_{X,2} + x_{S,2} + x_{T,2} + x_{B,2} = 1$$

$$x_{X,3} + x_{S,3} + x_{T,3} + x_{B,3} = 1$$

$$x_{X,4} + x_{S,4} + x_{T,4} + x_{B,4} = 1$$

$$x_{X,5} + x_{S,5} + x_{T,5} + x_{B,5} = 1$$

$$x_{X,6} + x_{S,6} + x_{T,6} + x_{B,6} = 1$$

$$x_{X,7} + x_{S,7} + x_{T,7} + x_{B,7} = 1$$

The component material balances and the mole fraction summation equations comprise the complete

set of independent equations for this kind of problem. Any other balance equation can be created from simple combinations of these equations.

We now create a set independent equations to describe the flowsheet

```
> Eqns:={seq(seq(CMB[i,u],i=components),u=Units),seq(SumEqn[j],j=Streams)};
```

$$Eqns := \{x_{X,6} + x_{S,6} + x_{T,6} + x_{B,6} = 1,\ F_3 x_{T,3} = F_6 x_{T,6} + F_7 x_{T,7},\ x_{X,7} + x_{S,7} + x_{T,7} + x_{B,7} = 1,$$
$$F_3 x_{B,3} = F_6 x_{B,6} + F_7 x_{B,7},\ F_1 x_{X,1} = F_2 x_{X,2} + F_3 x_{X,3},\ F_1 x_{S,1} = F_2 x_{S,2} + F_3 x_{S,3},$$
$$F_1 x_{B,1} = F_2 x_{B,2} + F_3 x_{B,3},\ F_1 x_{T,1} = F_2 x_{T,2} + F_3 x_{T,3},\ F_2 x_{X,2} = F_4 x_{X,4} + F_5 x_{X,5},$$
$$F_2 x_{S,2} = F_4 x_{S,4} + F_5 x_{S,5},\ x_{X,1} + x_{S,1} + x_{T,1} + x_{B,1} = 1,\ F_2 x_{T,2} = F_4 x_{T,4} + F_5 x_{T,5},$$
$$x_{X,2} + x_{S,2} + x_{T,2} + x_{B,2} = 1,\ x_{X,3} + x_{S,3} + x_{T,3} + x_{B,3} = 1,\ F_2 x_{B,2} = F_4 x_{B,4} + F_5 x_{B,5},$$
$$x_{X,4} + x_{S,4} + x_{T,4} + x_{B,4} = 1,\ F_3 x_{X,3} = F_6 x_{X,6} + F_7 x_{X,7},\ F_3 x_{S,3} = F_6 x_{S,6} + F_7 x_{S,7},$$
$$x_{X,5} + x_{S,5} + x_{T,5} + x_{B,5} = 1\}$$

The list of variables appearing in these equations is easily found

```
> Vars := indets(Eqns,name);
```

$$Vars := \{x_{S,6}, x_{T,7}, x_{T,6}, x_{S,7}, x_{X,7}, x_{X,6}, x_{B,5}, x_{B,4}, x_{T,5}, x_{T,4}, x_{B,6}, x_{B,7}, x_{X,3}, x_{X,2}, x_{X,1}, x_{S,1}, F_2,$$
$$F_1, F_3, x_{S,3}, x_{S,2}, x_{B,3}, x_{B,2}, x_{B,1}, x_{T,3}, x_{T,2}, x_{T,1}, F_4, F_5, x_{X,5}, x_{X,4}, x_{S,4}, F_7, F_6, x_{S,5}\}$$

and we can count them using the nops function:

```
> Nvars := nops(");
```

$$Nvars := 35$$

The number of equations is computed in a similar way

```
> Neqns:=nops(Eqns);
```

$$Neqns := 19$$

The number of degrees of freedom is the difference between these two numbers

```
> DegFree:=Nvars-Neqns;
```

$$DegFree := 16$$

This is the number of variables that we must specify before we have a consistent set of equations that can be solved (numerically, at least).

We know the flow and composition of the feed stream (note that only three of the feed mole fractions are specified).

```
> Specs[1]:=F[1]=70;
```

$$Specs_1 := F_1 = 70$$

```
> Specs[2]:=x[X,1]=0.15; Specs[3]:=x[S,1]=0.25; Specs[4]:=x[T,1]=0.4;
```

$$Specs_2 := x_{X,1} = .15$$

$$Specs_3 := x_{S,1} = .25$$

$$Specs_4 := x_{T,1} = .4$$

We need to specify another 12 variables and must look to the problem statement to find out what other

specifications we may make.

In the problem as posed the composition of all final product streams is given. However, in view of the mole fraction summation equations we may only specify 3 from each stream; the choice of which to omit is arbitrary - we leave out the lowest mole fraction in most cases.

> Specs[5]:=x[X,4]=0.07; Specs[6]:=x[B,4]=0.35; Specs[7] := x[T,4]=0.54;

$$Specs_5 := x_{X,\,4} = .07$$

$$Specs_6 := x_{B,\,4} = .35$$

$$Specs_7 := x_{T,\,4} = .54$$

> Specs[8]:=x[X,5]=0.18; Specs[9]:=x[B,5]=0.16; Specs[10] := x[T,5]=0.42;

$$Specs_8 := x_{X,\,5} = .18$$

$$Specs_9 := x_{B,\,5} = .16$$

$$Specs_{10} := x_{T,\,5} = .42$$

> Specs[11]:=x[X,6]=0.15; Specs[12]:=x[B,6]=0.21; Specs[13] := x[T,6]=0.54;

$$Specs_{11} := x_{X,\,6} = .15$$

$$Specs_{12} := x_{B,\,6} = .21$$

$$Specs_{13} := x_{T,\,6} = .54$$

> Specs[14]:=x[X,7]=0.24; Specs[15]:=x[S,7]=0.65; Specs[16] := x[T,7]=0.10;

$$Specs_{14} := x_{X,\,7} = .24$$

$$Specs_{15} := x_{S,\,7} = .65$$

$$Specs_{16} := x_{T,\,7} = .10$$

We now have the right number of specification equations which we combine in a set

> SpecEqns := {seq(Specs[l],l=1..DegFree)};

$$SpecEqns := \{x_{B,\,6} = .21,\ x_{X,\,6} = .15,\ x_{B,\,5} = .16,\ x_{T,\,5} = .42,\ x_{T,\,6} = .54,\ x_{S,\,7} = .65,\ x_{X,\,7} = .24,\ F_1 = 70,$$
$$x_{T,\,7} = .10,\ x_{X,\,1} = .15,\ x_{S,\,1} = .25,\ x_{T,\,1} = .4,\ x_{X,\,4} = .07,\ x_{T,\,4} = .54,\ x_{B,\,4} = .35,\ x_{X,\,5} = .18\}$$

> nops(SpecEqns);

16

We augment the set of equations with the specification equations

> AllEqns := Eqns union SpecEqns;

$$AllEqns := \{x_{B,\,6} = .21,\ x_{X,\,6} + x_{S,\,6} + x_{T,\,6} + x_{B,\,6} = 1,\ x_{X,\,6} = .15,\ x_{B,\,5} = .16,\ x_{T,\,5} = .42,$$
$$F_3\,x_{T,\,3} = F_6\,x_{T,\,6} + F_7\,x_{T,\,7},\ x_{T,\,6} = .54,\ x_{S,\,7} = .65,\ x_{X,\,7} + x_{S,\,7} + x_{T,\,7} + x_{B,\,7} = 1,\ x_{X,\,7} = .24,\ F_1 = 70,$$
$$F_3\,x_{B,\,3} = F_6\,x_{B,\,6} + F_7\,x_{B,\,7},\ x_{T,\,7} = .10,\ x_{X,\,1} = .15,\ F_1\,x_{X,\,1} = F_2\,x_{X,\,2} + F_3\,x_{X,\,3},\ x_{S,\,1} = .25,$$
$$F_1\,x_{S,\,1} = F_2\,x_{S,\,2} + F_3\,x_{S,\,3},\ F_1\,x_{B,\,1} = F_2\,x_{B,\,2} + F_3\,x_{B,\,3},\ F_1\,x_{T,\,1} = F_2\,x_{T,\,2} + F_3\,x_{T,\,3},\ x_{T,\,1} = .4,$$
$$F_2\,x_{X,\,2} = F_4\,x_{X,\,4} + F_5\,x_{X,\,5},\ F_2\,x_{S,\,2} = F_4\,x_{S,\,4} + F_5\,x_{S,\,5},\ x_{X,\,4} = .07,\ x_{X,\,1} + x_{S,\,1} + x_{T,\,1} + x_{B,\,1} = 1,$$

$$x_{T,4} = .54,\ x_{B,4} = .35,\ F_2 x_{T,2} = F_4 x_{T,4} + F_5 x_{T,5},\ x_{X,2} + x_{S,2} + x_{T,2} + x_{B,2} = 1,$$

$$x_{X,3} + x_{S,3} + x_{T,3} + x_{B,3} = 1,\ F_2 x_{B,2} = F_4 x_{B,4} + F_5 x_{B,5},\ x_{X,4} + x_{S,4} + x_{T,4} + x_{B,4} = 1,$$

$$F_3 x_{X,3} = F_6 x_{X,6} + F_7 x_{X,7},\ F_3 x_{S,3} = F_6 x_{S,6} + F_7 x_{S,7},\ x_{X,5} + x_{S,5} + x_{T,5} + x_{B,5} = 1,\ x_{X,5} = .18\ \}$$

We check to see if the numbers of equations and variables are the same.

```
> nops(AllEqns); nops(Vars);
```

35

35

Now we can ask Maple to solve the equations.

```
> result:=solve(AllEqns,Vars);
```

$$result := \{x_{S,4} = .04000000000,\ x_{X,3} = .2100000000,\ F_6 = 8.750000000,\ F_5 = 17.50000000,$$

$$x_{B,7} = .01000000000,\ x_{B,1} = .2000000000,\ x_{X,2} = .1140000000,\ F_7 = 17.50000000,\ x_{S,6} = .1000000000,$$

$$F_4 = 26.25000000,\ x_{S,5} = .2400000000,\ x_{B,6} = .2100000000,\ x_{T,5} = .4200000000,\ x_{B,5} = .1600000000,$$

$$x_{X,6} = .1500000000,\ x_{T,6} = .5400000000,\ x_{S,7} = .6500000000,\ x_{X,5} = .1800000000,\ x_{T,1} = .4000000000,$$

$$x_{X,4} = .07000000000,\ x_{T,4} = .5400000000,\ x_{B,4} = .3500000000,\ x_{X,7} = .2400000000,\ x_{T,7} = .1000000000,$$

$$x_{S,1} = .2500000000,\ x_{X,1} = .1500000000,\ x_{B,2} = .2740000000,\ x_{S,2} = .1200000000,\ x_{S,3} = .4666666667,$$

$$x_{T,3} = .2466666667,\ x_{B,3} = .07666666667,\ x_{T,2} = .4920000000,\ F_3 = 26.25000000,\ F_2 = 43.75000000,\ F_1 = 70.$$

}

We check the solution by substituting the result into the equations.

```
> subs(result,Eqns);
```

$$\{1.000000000 = 1,\ 6.475000001 = 6.475000000,\ 2.012500000 = 2.012500000,\ 10.50000000 = 10.50000000,$$

$$17.50000000 = 17.50000000,\ 14.00000000 = 14.00000000,\ 28.00000000 = 28.00000000,$$

$$4.987500000 = 4.987500000,\ 5.250000000 = 5.250000000,\ 21.52500000 = 21.52500000,$$

$$11.98750000 = 11.98750000,\ 5.512500000 = 5.512500000,\ 12.25000000 = 12.25000000\ \}$$

and we see that the result that Maple computed does indeed satisfy the equations.

Problem 3: Data Regression

The problem at hand is the fitting of experimental data to equations used to represent the vapor pressure. We begin with a list of the temperatures (in Celcius) at which the vapor pressure has been measured.

```
> T(C) := [-36.7,-19.6,-11.5,-2.6,7.6,15.4,26.1,42.2,60.6,80.1];
```

$$T(C) := [-36.7, -19.6, -11.5, -2.6, 7.6, 15.4, 26.1, 42.2, 60.6, 80.1]$$

Note the perhaps unconventional symbol used to denote the list of temperatures. As far as Maple is concerned $T(C)$ here is a name just like T would be, and can be assigned to any valid Maple object; in this case a list of temperature values. The corresponding values of the vapor pressure (in mm Hg) is next

```
> P(mmHg) := [1,5,10,20,40,60,100,200,400,760];
```

$$P(mmHg) := [1, 5, 10, 20, 40, 60, 100, 200, 400, 760]$$

A quick check to see that the number of values of temperature and pressure are equal.

```
> nops(P(mmHg)), nops(T(C));
```

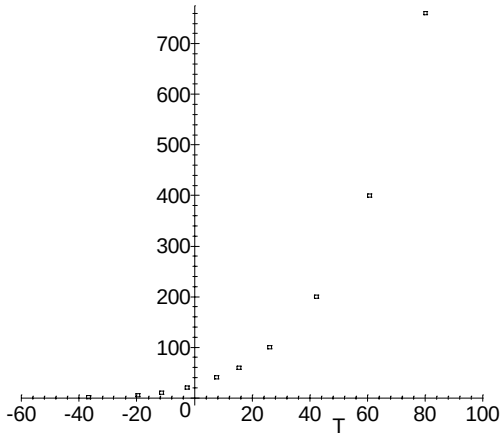
$$10, 10$$

We plot the data to see its general form

```
> datapoints := [seq([T(C)[k],P(mmHg)[k]],k=1..nops(T(C)))];
```

$$datapoints := [[-36.7, 1], [-19.6, 5], [-11.5, 10], [-2.6, 20], [7.6, 40], [15.4, 60], [26.1, 100], [42.2, 200], [60.6, 400], [80.1, 760]]$$

```
> dataplot:=plot(datapoints,T=-60..100,style=point,symbol=box): dataplot;
```




```
> p2:=logplot([seq([Tlist[k],Plist[k]],k=1..nops(Tlist))],style=point,symbol=box):
```

We continue by attempting to fit simple polynomials to the data. We define a function to give us a polynomial in T of any order.

```
> p := n->sum(a[k]*T^k,k=0..n);
```

$$p := n \rightarrow \sum_{k=0}^{n} a_k T^k$$

A quadratic, for example, is given by

```
> p(2);
```

$$a_0 + a_1\,T + a_2\,T^2$$

and a cubic is

```
> p(3);
```

$$a_0 + a_1\,T + a_2\,T^2 + a_3\,T^3$$

Maple includes a least squares fitting procedure in its statistical package. The package is loaded as follows

```
> with(stats):
```

We fit a quadratic to the data as follows

```
> n:=2; expr := P=p(n);
```

$$n := 2$$

$$expr := P = a_0 + a_1\,T + a_2\,T^2$$

```
> eqn[n]:=fit[leastsquare[[T,P], expr, {seq(a[k],k=0..n)}]]([T(C),P(mmHg)]);
> n:=2; expr := P=p(n);
```

$$n := 2$$

$$expr := P = a_0 + a_1\,T + a_2\,T^2$$

```
> eqn[n]:=fit[leastsquare[[T,P], expr, {seq(a[k],k=0..n)}]]([T(C),P(mmHg)]);
```

$$eqn_2 := P = -.5820645411 + 2.067147952\,T + .08615257053\,T^2$$

We can fit higher order polynomials just as easily

```
> n:=3; expr := P=p(n);
```

$$n := 3$$

$$expr := P = a_0 + a_1\,T + a_2\,T^2 + a_3\,T^3$$

```
> eqn[n]:=fit[leastsquare[[T,P], expr, {seq(a[k],k=0..n)}]]([T(C),P(mmHg)]);
```

$$eqn_3 := P = 24.45935897 + 1.198099375\,T + .03944807267\,T^2 + .0007449112375\,T^3$$

```
> n:=4; expr := P=p(n);
```

$$n := 4$$

$$expr := P = a_0 + a_1\,T + a_2\,T^2 + a_3\,T^3 + a_4\,T^4$$

```
> eqn[n]:=fit[leastsquare[[T,P], expr, {seq(a[k],k=0..n)}]]([T(C),P(mmHg)]);
```

$$eqn_4 := P = 24.67875735 + 1.606195814\,T + .03604429596\,T^2 + .0004131215973\,T^3 + .3963147729\,10^{-5}\,T^4$$

```
> n:=5; expr := P=p(n);
```

$$n := 5$$

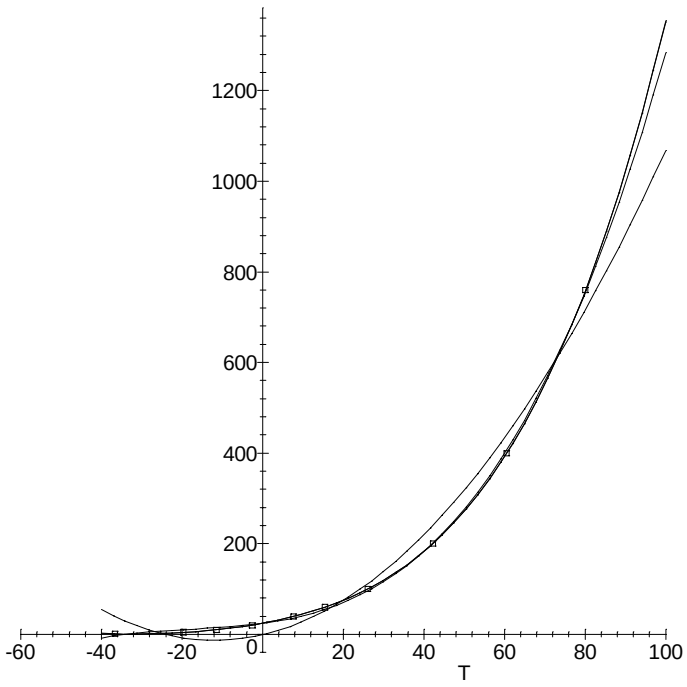
$$expr := P = a_0 + a_1\,T + a_2\,T^2 + a_3\,T^3 + a_4\,T^4 + a_5\,T^5$$

```
> eqn[n]:=fit[leastsquare[[T,P], expr, {seq(a[k],k=0..n)}]]([T(C),P(mmHg)]);
```

$$eqn_5 := P = 24.75426215 + 1.609016444\,T + .03560532992\,T^2 + .0004129782380\,T^3 + .4226058650\,10^{-5}\,T^4 - .2505026818\,10^{-8}\,T^5$$

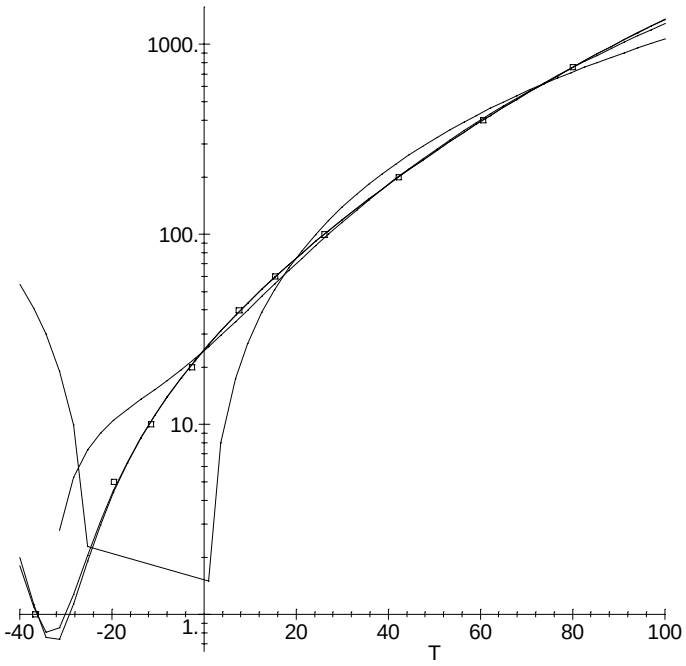
We can compare these expressions graphically

```
> n:='n':
  polyplot:=plot([seq(rhs(eqn[n]),n=2..5)],T=-40..100,color=[red,black,blue,green]):
> plots[display]({polyplot,dataplot});
```



From which we observe that the red line (the quadratic) is clearly poorer than the others; all of which seem more or less equally good. However, if we plot the equations on a semilog plot we see that the polynomials suffer from a rather serious problem.

```
> lplot1:=plots[logplot]([seq(rhs(eqn[n]),n=2..5)],T=-40..100,color=[red,black,blue,green]):
> lplot2:=plots[logplot](datapoints,style=point,symbol=box):
> plots[display]({lplot1,lplot2});
```



From this diagram (if viewed in color) we would notice that the fourth and fifth order polyomials are best (as might be expected).

We now attempt to fit the Clausius Clapeyron (CC) equation to the data.

```
> CCeqn:= log10(P) = A-B/(T+273.15): CCeqn;
```

$$\log10(P) = A - \frac{B}{T + 273.15}$$

The bottom line of the second term on the right is the temperature in Kelvin. We define two new variables

```
> v1:= Tau =1/(T+273.15): v1;
```

$$\mathrm{T} = \frac{1}{T + 273.15}$$

```
> v2:=rho=log10(P): v2;
```

$$\rho = \log10(P)$$

and substitute into the CC equation.

```
> CC2:=subs(rhs(v2)=lhs(v2),rhs(v1)=lhs(v1),CCeqn): CC2;
```

$$\rho = A - B\,\mathrm{T}$$

Thus, this equation is linear in $\frac{1}{\mathrm{T}(K)}$ and $\log10(P)$.

The next step is to transform the data, beginning with the vapor pressures

```
> lnpdata := evalf(map(log10,P(mmHg)));
```

lnpdata := [0, .6989700043, 1.000000000, 1.301029996, 1.602059991, 1.778151250, 2.000000000, 2.301029996, 2.602059991, 2.880813592]

and now for the temperatures.

```
> ToKelvin:=x->x+273.15;
  T(K) := map(ToKelvin,T(C));
```

$$ToKelvin := x \to x + 273.15$$

$$\mathrm{T}(K) := [\,236.45,\ 253.55,\ 261.65,\ 270.55,\ 280.75,\ 288.55,\ 299.25,\ 315.35,\ 333.75,\ 353.25\,]$$

We must take the reciprocal of these values

```
> r := x-> 1/x;
  Tover := map(r,T(K));
```

$$r := x \to \frac{1}{x}$$

$$Tover := [\,.004229223937,\ .003943995267,\ .003821899484,\ .003696174459,\ .003561887801,\ .003465603881,\ .003341687552,\ .003171079753,\ .002996254682,\ .002830856334\,]$$

We are now ready to fit the transformed data

```
> CCfit:=fit[leastsquare[[Tau,rho], CC2, {A,B}]]([Tover,lnpdata]): CCfit;
```

$$\rho = 8.752009662 - 2035.330939\ \mathrm{T}$$

This result can be expressed in terms of our orginal variables

```
> CC3:=subs(v1,v2,CCfit): CC3;
```

$$\log10(P) = 8.752009662 - \frac{2035.330939}{T + 273.15}$$

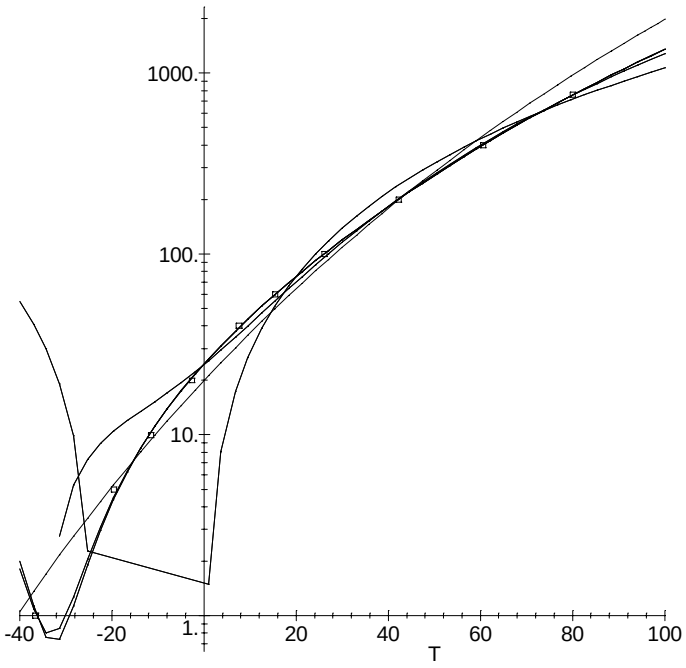
We make a function of the right hand side for plotting purposes

```
> CCF:=unapply(10^rhs(CC3),T);
```

$$CCF := T \to 10^{\left(8.752009662 - \frac{2035.330939}{T + 273.15}\right)}$$

and display this new result along with the polynomial fits and the data.

```
> n:='n':
  p1:=plots[logplot]([seq(rhs(eqn[n]),n=2..5),CCF(T)],T=-40..100,color=[red,yellow,green,black,blue,purple]):
> plots[display]({p1,lplot2,lplot1});
```



From this we see immediately that the Clausius Clapeyron form is far superior to the simple polynomial fits.

Maple cannot do non-linear fitting (unless you program a method yourselves) so we have not attempted to fit the Antoine equation.

>

Reaction Equilibrium for Multiple Gas Phase Reactions

We begin by creating a list of component identities and to calculate the number of components.

```
> components:=[A,B,C,D,X,Y,Z]; nc:=nops(components);
```

$$components := [A, B, C, D, X, Y, Z]$$

$$nc := 7$$

Next, we create a list of reactions:

```
> Reactions:=[R1,R2,R3];
```

$$Reactions := [R1, R2, R3]$$

Each reaction may be expressed as a Maple equation:

```
> Reaction[R1] :=A+B=C+D;
```

$$Reaction_{R1} := A + B = C + D$$

```
> Reaction[R2] := B+C=X+Y;
```

$$Reaction_{R2} := B + C = X + Y$$

```
> Reaction[R3] :=A+X=Z;
```

$$Reaction_{R3} := A + X = Z$$

The stoichiometric cofficients can be deduced from the set of reactions as follows:

```
> for r in Reactions do
    for i in components do
     nu[i,r]:=coeff(lhs(Reaction[r])-rhs(Reaction[r]),i);
  od: od:
  print(nu);
```

table([

$(Z, R3) = -1$

$(Y, R2) = -1$

$(X, R1) = 0$

$(A, R2) = 0$

$(B, R3) = 0$

$(X, R3) = 1$

$(D, R2) = 0$

$(C, R1) = -1$

$(Z, R2) = 0$

$(Y, R1) = 0$

$(B, R2) = 1$

$(C, R3) = 0$

$(Y, R3) = 0$

$(X, R2) = -1$

$(D, R1) = -1$

$(A, R1) = 1$

$(A, R3) = 1$

$(Z, R1) = 0$

$(D, R3) = 0$

$(C, R2) = 1$

$(B, R1) = 1$

])

We need mole balances for each component:

```
> r:='r': for i in components do
    CMB[i] := C[i,0] = C[i] + add(nu[i,r]*e[r],r=Reactions);
    print(CMB[i]);
  od:
```

$$C_{A,0} = C_A + e_{R1} + e_{R3}$$

$$C_{B,0} = C_B + e_{R1} + e_{R2}$$

$$C_{C,0} = C_C - e_{R1} + e_{R2}$$

$$C_{D,0} = C_D - e_{R1}$$

$$C_{X,0} = C_X - e_{R2} + e_{R3}$$

$$C_{Y,0} = C_Y - e_{R2}$$

$$C_{Z,0} = C_Z - e_{R3}$$

where we use *e* to denote the extent of reaction. The initial concentration is denoted with a second subscript of zero.

The reaction equilibrium constants may be expressed as

```
> i:='i': for r in Reactions do
    Kdef[r]:=K[r] = mul ((C[i])^(-nu[i,r]),i=components);
    print(Kdef[r]);
  od:
```

$$K_{R1} = \frac{C_C C_D}{C_A C_B}$$

$$K_{R2} = \frac{C_X C_Y}{C_B C_C}$$

$$K_{R3} = \frac{C_Z}{C_A C_X}$$

We create a set of equations to be solved

```
> Eqns := {seq(CMB[i],i=components),seq(Kdef[r],r=Reactions)};
```

$$Eqns := \left\{ C_{C,0} = C_C - e_{R1} + e_{R2},\ C_{D,0} = C_D - e_{R1},\ C_{X,0} = C_X - e_{R2} + e_{R3},\ C_{B,0} = C_B + e_{R1} + e_{R2},\right.$$

$$C_{A,0} = C_A + e_{R1} + e_{R3},\ C_{Y,0} = C_Y - e_{R2},\ C_{Z,0} = C_Z - e_{R3},\ K_{R1} = \frac{C_C C_D}{C_A C_B},\ K_{R2} = \frac{C_X C_Y}{C_B C_C},\ K_{R3} = \frac{C_Z}{C_A C_X}\Bigg\}$$

and a set of unknown variables:

```
> Vars := indets(Eqns,name);
```

$$Vars := \{C_{C,0}, C_C, C_{D,0}, C_D, C_{X,0}, C_X, C_B, C_{B,0}, C_{A,0}, C_A, e_{R1}, e_{R2}, e_{R3}, C_{Y,0}, C_Y, C_{Z,0}, C_Z, K_{R1}, K_{R2}, K_{R3}\}$$

The number of degrees of freedom follows directly

```
> DegFree := nops(Vars)-nops(Eqns);
```

$$DegFree := 10$$

Case 1

```
> Specs[1] := C[A,0]=1.5;
```

$$Specs_1 := C_{A,0} = 1.5$$

```
> Specs[2] := C[B,0]=1.5;
```

$$Specs_2 := C_{B,0} = 1.5$$

```
> Specs[3] := seq(C[i,0]=0,i=[C,D,X,Y,Z]);
```

$$Specs_3 := C_{C,0} = 0,\ C_{D,0} = 0,\ C_{X,0} = 0,\ C_{Y,0} = 0,\ C_{Z,0} = 0$$

At the conditions of interest the reaction equilibrium coefficient has the following values

```
> Kvals := {K[R1] = 1, K[R2]=2.63, K[R3]=5};
```

$$Kvals := \{K_{R1} = 1,\ K_{R2} = 2.63,\ K_{R3} = 5\}$$

We now augment the set of equations with the set of specifications

```
> AllEqns:=Eqns union {seq(Specs[k],k=1..3),op(Kvals)};
```

$$AllEqns := \Bigg\{C_{B,0} = 1.5,\ C_{C,0} = C_C - e_{R1} + e_{R2},\ C_{D,0} = C_D - e_{R1},\ C_{X,0} = C_X - e_{R2} + e_{R3},$$

$$C_{B,0} = C_B + e_{R1} + e_{R2},\ C_{A,0} = C_A + e_{R1} + e_{R3},\ C_{Y,0} = C_Y - e_{R2},\ C_{Z,0} = C_Z - e_{R3},\ K_{R1} = \frac{C_C C_D}{C_A C_B},\ K_{R1} = 1,$$

$$K_{R2} = 2.63,\ K_{R3} = 5,\ K_{R2} = \frac{C_X C_Y}{C_B C_C},\ K_{R3} = \frac{C_Z}{C_A C_X},\ C_{A,0} = 1.5,\ C_{C,0} = 0,\ C_{D,0} = 0,\ C_{X,0} = 0,\ C_{Y,0} = 0,$$

$$C_{Z,0} = 0\Bigg\}$$

and ask Maple to solve the problem

```
> result:=solve(AllEqns,Vars);
```

$$result := \{C_{C,0} = 0,\ C_{D,0} = 0,\ C_{X,0} = 0,\ C_{Y,0} = 0,\ C_{Z,0} = 0,\ C_B = -.3879574470,\ C_X = -.3207318471,$$

$$C_D = 1.072193162,\ C_Z = 1.136496132,\ e_{R1} = 1.072193162,\ \%3,\ C_C = .2564288777,\ K_{R3} = 5.,\ C_Y = .8157642847,$$

$$e_{R3} = 1.136496132,\ e_{R2} = .8157642847,\ C_A = -.7086892941,\ \%2,\ \%1,\ K_{R1} = 1.\},\ \{$$

$$e_{R2} = .9256703707 + .1402769617\,I,\ C_{C,0} = 0,\ C_{D,0} = 0,\ C_{X,0} = 0,\ C_{Y,0} = 0,\ C_{Z,0} = 0,\ \%3,\ K_{R3} = 5.,$$

$C_Y = .9256703707 + .1402769617\ I$, $e_{R3} = 1.727932631 + .1181659648\ I$, $C_Z = 1.727932631 + .1181659648\ I$,

$e_{R1} = .2016941637 - .07686691417\ I$, $C_C = -.7239762070 - .2171438759\ I$,

$C_X = -.8022622599 + .02211099693\ I$, $C_A = -.4296267943 - .04129905063\ I$,

$C_B = .3726354656 - .06341004756\ I$, $C_D = .2016941637 - .07686691417\ I$, %2, %1, $K_{R1} = 1.$ }, { $C_{C,\,0} = 0$,

$C_{D,\,0} = 0$, $C_{X,\,0} = 0$, $C_{Y,\,0} = 0$, $C_{Z,\,0} = 0$, %3, $K_{R3} = 5.$, $e_{R3} = 1.727932631 - .1181659648\ I$,

$C_B = .3726354656 + .06341004756\ I$, $C_C = -.7239762070 + .2171438759\ I$, $C_X = -.8022622599 - .02211099693\ I$,

$e_{R2} = .9256703707 - .1402769617\ I$, $e_{R1} = .2016941637 + .07686691417\ I$, $C_Y = .9256703707 - .1402769617\ I$,

$C_A = -.4296267943 + .04129905063\ I$, $C_D = .2016941637 + .07686691417\ I$, $C_Z = 1.727932631 - .1181659648\ I$,

%2, %1, $K_{R1} = 1.$ }, { $C_{C,\,0} = 0$, $C_{D,\,0} = 0$, $C_{X,\,0} = 0$, $C_{Y,\,0} = 0$, $C_{Z,\,0} = 0$, %3, $K_{R3} = 5.$, $e_{R1} = -.8625720556$,

$C_B = 3.783668561$, $C_Z = 2.489900223$, $C_D = -.8625720556$, $C_Y = -1.421096505$, $e_{R3} = 2.489900223$,

$e_{R2} = -1.421096505$, $C_X = -3.910996728$, $C_C = .5585244494$, $C_A = -.1273281670$, %2, %1, $K_{R1} = 1.$ }, {

$C_{C,\,0} = 0$, $C_{D,\,0} = 0$, $C_{X,\,0} = 0$, $C_{Y,\,0} = 0$, $C_{Z,\,0} = 0$, %3, $K_{R3} = 5.$, $e_{R3} = .6779864947 - .06771720101\ I$,

$C_B = -.1082051188 - .6351221223\ I$, $C_C = -.2577874160 + .03181297494\ I$, $e_{R1} = .6752088514 + .3334675486\ I$,

$C_Y = .9329962674 + .3016545737\ I$, $e_{R2} = .9329962674 + .3016545737\ I$, $C_A = .1468046539 - .2657503476\ I$,

$C_D = .6752088514 + .3334675486\ I$, $C_X = .2550097727 + .3693717747\ I$, $C_Z = .6779864947 - .06771720101\ I$,

%2, %1, $K_{R1} = 1.$ }, { $C_{C,\,0} = 0$, $C_{D,\,0} = 0$, $C_{X,\,0} = 0$, $C_{Y,\,0} = 0$, $C_{Z,\,0} = 0$, %3, $K_{R3} = 5.$,

$e_{R3} = .6779864947 + .06771720101\ I$, $e_{R1} = .6752088514 - .3334675486\ I$, $C_Z = .6779864947 + .06771720101\ I$,

$C_D = .6752088514 - .3334675486\ I$, $C_Y = .9329962674 - .3016545737\ I$, $C_C = -.2577874160 - .03181297494\ I$,

$C_B = -.1082051188 + .6351221223\ I$, $C_A = .1468046539 + .2657503476\ I$, $e_{R2} = .9329962674 - .3016545737\ I$,

%2, %1, $C_X = .2550097727 - .3693717747\ I$, $K_{R1} = 1.$ }, { $C_{C,\,0} = 0$, $C_{D,\,0} = 0$, $C_{X,\,0} = 0$, $C_{Y,\,0} = 0$, $C_{Z,\,0} = 0$,

%3, $K_{R3} = 5.$, %2, %1, $K_{R1} = 1.$, $C_A = .3630593987$, $C_X = .5975408420$, $e_{R1} = .05222650742$,

$C_C = -1.630028428$, $e_{R2} = 1.682254936$, $C_Z = 1.084714094$, $C_D = .05222650742$, $C_Y = 1.682254936$,

$C_B = -.2344814433$, $e_{R3} = 1.084714094$ }, { $C_{C,\,0} = 0$, $C_{D,\,0} = 0$, $C_{X,\,0} = 0$, $C_{Y,\,0} = 0$, $C_{Z,\,0} = 0$, %3, $K_{R3} = 5.$,

%2, %1, $K_{R1} = 1.$, $C_B = .2474609670$, $e_{R1} = .7011220209$, $C_X = .1766927072$, $C_A = .4241536742$,

$C_D = .7011220209$, $C_Z = .3747243049$, $C_Y = .5514170121$, $e_{R3} = .3747243049$, $e_{R2} = .5514170121$,

$C_C = .1497050089$ }

%1 := $C_{A,\,0} = 1.500000000$

%2 := $C_{B,\,0} = 1.500000000$

%3 := $K_{R2} = 2.630000000$

Maple found no less than 8 solutions without much effort. It is easy to see, however, that not all of the solutions are physically meaningful; some of them contain negative concentrations.

```
> result[1];
```

$$\{C_{C,0} = 0, C_{D,0} = 0, C_{X,0} = 0, C_{Y,0} = 0, C_{Z,0} = 0, C_B = -.3879574470, C_X = -.3207318471,$$
$$C_D = 1.072193162, C_Z = 1.136496132, e_{R1} = 1.072193162, K_{R2} = 2.630000000, C_C = .2564288777, K_{R3} = 5.,$$
$$C_Y = .8157642847, e_{R3} = 1.136496132, e_{R2} = .8157642847, C_A = -.7086892941, C_{B,0} = 1.500000000,$$
$$C_{A,0} = 1.500000000, K_{R1} = 1.\}$$

while others are complex.

```
> result[2];
```

$$\{e_{R2} = .9256703707 + .1402769617\,I, C_{C,0} = 0, C_{D,0} = 0, C_{X,0} = 0, C_{Y,0} = 0, C_{Z,0} = 0, K_{R2} = 2.630000000,$$
$$K_{R3} = 5., C_Y = .9256703707 + .1402769617\,I, e_{R3} = 1.727932631 + .1181659648\,I,$$
$$C_Z = 1.727932631 + .1181659648\,I, e_{R1} = .2016941637 - .07686691417\,I, C_C = -.7239762070 - .2171438759\,I,$$
$$C_X = -.8022622599 + .02211099693\,I, C_A = -.4296267943 - .04129905063\,I,$$
$$C_B = .3726354656 - .06341004756\,I, C_D = .2016941637 - .07686691417\,I, C_{B,0} = 1.500000000,$$
$$C_{A,0} = 1.500000000, K_{R1} = 1.\}$$

Only one of the solutions is physically meaningful.

```
> result[8];
```

$$\{C_{C,0} = 0, C_{D,0} = 0, C_{X,0} = 0, C_{Y,0} = 0, C_{Z,0} = 0, K_{R2} = 2.630000000, K_{R3} = 5., C_{B,0} = 1.500000000,$$
$$C_{A,0} = 1.500000000, K_{R1} = 1., C_B = .2474609670, e_{R1} = .7011220209, C_X = .1766927072, C_A = .4241536742,$$
$$C_D = .7011220209, C_Z = .3747243049, C_Y = .5514170121, e_{R3} = .3747243049, e_{R2} = .5514170121,$$
$$C_C = .1497050089\}$$

Note that we did not have to rearrange the equations into a form more suitable for numerical computation as was done in the Polymath solution. We did not have to provide initial guesses of the unknown variables, and we did not have to work out the independent equations since we could do that with Maple as well. Of course, it is possible to compute a purely numerical solution using the Newton's method package in the share library as was done for another example in this collection.

```
>
```

Terminal Velocity of Falling Particles

The force balance on the particle is

```
> forcebal := v[t]=sqrt(4*g*(rho[p]-rho)*D[p]/(3*C[D]*rho)):
  forcebal;
```

$$v_t = \frac{2}{3}\sqrt{3}\sqrt{\frac{g\,(\rho_p - \rho)\,D_p}{C_D\,\rho}}$$

The Reynolds number is defined for this system by

```
> reeqn := re = D[p]*v[t]*rho/mu: reeqn;
```

$$re = \frac{D_p\,v_t\,\rho}{\mu}$$

The parameters in these equations have the following values (in consistent units):

```
> params := {g=9.81,rho[p]=1800, D[p]=0.208e-3,rho=994.6,mu=8.931e-4};
```

$$params := \{\rho = 994.6,\ \mu = .0008931,\ D_p = .000208,\ \rho_p = 1800,\ g = 9.81\}$$

We substitute these parameters into the force balance to get

```
> subs(params,forcebal);
```

$$v_t = .02709920169\sqrt{3}\sqrt{\frac{1}{C_D}}$$

We cannot solve this equation directly since the drag coefficient is a function of the terminal velocity through the Reynolds number. We continue by writing a Maple procedure to compute the drag coefficient as a function of the Reynolds number. We have to take this approach since the drag coefficient is computed from different expressions depending on the value of the Reynolds number.

```
> f := proc(re)
  local CD;
  if not type(re,numeric) then RETURN(C[D](re)) fi;
  if re < 0.1 then
    CD := 24/re
  elif re >= 0.1 and re <= 1000 then
    CD := 24/re*(1+0.14*re^0.7)
  elif re > 1000 and re <= 350000 then
    CD := 0.44
  elif re > 350000 then
    CD := 0.19-8e4/re
  fi;
  end:
```

Note that we use re for the Reynolds number since Re is a reserved phrase in Maple for the real part of a complex variable. Let us test this proc to see what it does

```
> f(10);
```

4.083989105

```
> f(100);
```

1.083993841

```
> f(2000);
```

.44

```
> f(re);
```

$$C_D(re)$$

The next step is to create a Maple procedure to evaluate the force balance for different values of the terminal velocity. There are many ways to do this; our example follows:

```
> eqn := proc(vt)
  local C, re;
  global params;
   re:= subs(params,D[p]*vt*rho/mu);
   C[D]:=f(re);
   subs(params,vt-sqrt(4*g*(rho[p]-rho)*D[p]/(3*C[D]*rho)));
  end:
```

Finally, we invoke Maple's built in floating point solver

```
> vt1:=fsolve('eqn'(vt),vt=0.00001..0.05);
```

$$vt1 := .01578618582$$

This is the terminal velocity of the particle under the given conditions. Note that eqn in the call to fsolve is contained within quote marks. If this were not done then Maple would attempt to call the eqn procedure and pass that result to fsolve. Needless to say, that would be a disaster in this case. the quote marks force Maple to re-evaluate the equation every time it tries a new value of the terminal velocity. The units here are m/s. The Reynolds number at this velocity is

```
> subs(v[t]=vt1,params,reeqn);
```

$$re = 3.656696458$$

and the drag coefficient is

```
> f(rhs(''));
```

$$8.840561062$$

For part (b) we are asked for the terminal velocity in a centifugal separator where the acceleration is $30\,g$. The only changes needed are to revise the set of parameters (in this case only g is changed).

```
> params := {g=9.81*30,rho[p]=1800,  D[p]=0.208e-3,rho=994.6,mu=8.931e-4};
```

$$params := \{g = 294.30,\ \rho = 994.6,\ \mu = .0008931,\ D_p = .000208,\ \rho_p = 1800\}$$

```
> eqn := proc(vt)
  local C, re;
  global params;
   params := {g=9.81*30,rho[p]=1800,  D[p]=0.208e-3,rho=994.6,mu=8.931e-4};
   re:= subs(params,D[p]*vt*rho/mu);
   C[D]:=f(re);
   subs(params,vt-sqrt(4*g*(rho[p]-rho)*D[p]/(3*C[D]*rho)));
  end:
```

Again we call fsolve

```
> vt2:=fsolve('eqn'(vt),vt);
```

$$vt2 := .2060692237$$

The units here are m/s. The Reynolds number at this velocity is

```
> subs(v[t]=vt2,params,reeqn);
```

$$re = 47.73367101$$

and the drag coefficient is

```
> f(rhs(''));
```

$$1.556428713$$

Heat Exchange in a Series of Tanks

Make a list of tank numbers

```
> Units:=[seq(i,i=1..3)];
```

$$Units := [1, 2, 3]$$

Write down the energy balances for each tank.

```
> for i in Units do
    EB[i] := M*C[P]*diff(T[i](t),t) = W* C[P]*(T[i-1](t)-T[i](t)) + U*A*(T[s]-T[i](t));
    print(EB[i]);
  od:
```

$$M\,C_P\left(\frac{\partial}{\partial t}T_1(t)\right) = \mathrm{W}\,C_P\,(T_0(t) - T_1(t)) + U\,A\,(T_s - T_1(t))$$

$$M\,C_P\left(\frac{\partial}{\partial t}T_2(t)\right) = \mathrm{W}\,C_P\,(T_1(t) - T_2(t)) + U\,A\,(T_s - T_2(t))$$

$$M\,C_P\left(\frac{\partial}{\partial t}T_3(t)\right) = \mathrm{W}\,C_P\,(T_2(t) - T_3(t)) + U\,A\,(T_s - T_3(t))$$

Solve for the temperature derivatives

```
> for i in Units do
    t1 := diff(T[i](t),t)=solve(EB[i],diff(T[i](t),t));
    EB2[i]:=collect(t1,[W,U,A,C[P],M]);
    print(EB2[i]);
  od:
```

$$\frac{\partial}{\partial t}T_1(t) = \frac{(T_0(t) - T_1(t))\,\mathrm{W}}{M} + \frac{(T_s - T_1(t))\,A\,U}{M\,C_P}$$

$$\frac{\partial}{\partial t}T_2(t) = \frac{(T_1(t) - T_2(t))\,\mathrm{W}}{M} + \frac{(T_s - T_2(t))\,A\,U}{M\,C_P}$$

$$\frac{\partial}{\partial t}T_3(t) = \frac{(T_2(t) - T_3(t))\,\mathrm{W}}{M} + \frac{(T_s - T_3(t))\,A\,U}{M\,C_P}$$

The model parameters and specifications are provided as a set of equations.

```
> Params := {U=10/A,C[P]=2.0};
```

$$Params := \{U = \frac{10}{A},\ C_P = 2.0\}$$

```
> Specs:={W=100,T[0](t)=20,T[s]=250,M=1000};
```

$$Specs := \{M = 1000,\ T_s = 250,\ \mathrm{W} = 100,\ T_0(t) = 20\}$$

These sets are substituted into the family of differential equations.

```
> Diffeqns := subs(Specs,Params,{seq(EB2[i],i=Units)});
```

$$Diffeqns := \{\frac{\partial}{\partial t}T_3(t) = \frac{1}{10}T_2(t) - .1050000000\,T_3(t) + 1.250000000,$$

$$\frac{\partial}{\partial t}T_2(t) = \frac{1}{10}T_1(t) - .1050000000\,T_2(t) + 1.250000000,\ \frac{\partial}{\partial t}T_1(t) = 3.250000000 - .1050000000\,T_1(t)\}$$

The initial condition is

```
> Initial := {seq(T[i](0)=20,i=Units)};
```

$$Initial := \{T_3(0) = 20,\ T_2(0) = 20,\ T_1(0) = 20\}$$

We combine the differential equations with the initial conditions to give

```
> Alleqns := {op(Diffeqns2),op(Initial)};
```

$$Alleqns := \{T_3(0) = 20,\ T_2(0) = 20,\ \frac{\partial}{\partial t} T_3(t) = \frac{1}{10} T_2(t) - .1050000000\ T_3(t) + 1.250000000,$$

$$\frac{\partial}{\partial t} T_2(t) = \frac{1}{10} T_1(t) - .1050000000\ T_2(t) + 1.250000000,\ \frac{\partial}{\partial t} T_1(t) = 3.250000000 - .1050000000\ T_1(t),$$

$$T_1(0) = 20\}$$

We solve these equations using Maple's own `dsolve` command, invoking options to to use the Runge-Kutta-Fehlberg numerical method. Other numerical methods available in Maple including lsode, Gear's method and 7/8th order RK method.

```
> g:=dsolve(Alleqns,{seq(T[i](t),i=Units)}, type=numeric, method=rkf45, output=procedurelist);
```

$$g := \mathbf{proc}(rkf45\_x)\ \ldots\ \mathbf{end}$$

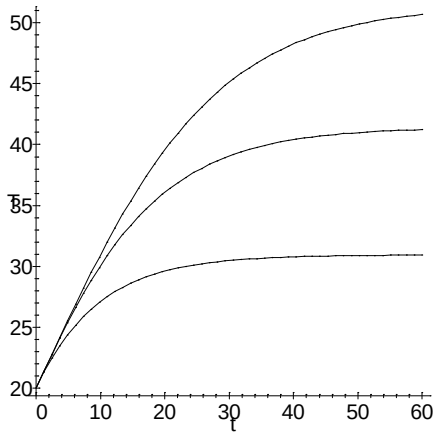
The output is a procedure (function), here asigned to the name g, that can be called to find specific values of the temperatures at particular times. For example, the temperatures after 60 minutes are

```
> g(63);
```

$$[t = 63,\ T_1(t) = 30.93770350756057,\ T_3(t) = 50.80357693563310,\ T_2(t) = 41.26209607145896]$$

We can plot the solution using the odeplot command from the plots package.

```
> plots[odeplot](g,[seq([t,T[i](t)],i=Units)],0..60,color=black,labels=[t,T]);
```



The disadvantage to Maple's built in numerical integration routines is that they are slow, especially for stiff or for larger systems of equations. An alternative method of obtaining numerical solutions to systems of ODEs is discussed in our solution to Problem 8.

Diffusion and Reaction in a One Dimensional Slab

This example involves the concentration in plug flow reactor with axial dispersion. The governing ODE is

```
> ODE := diff(C[A](z),z,z)=k/D[A,B]*C[A](z): ODE;
```

$$\frac{\partial^2}{\partial z^2} C_A(z) = \frac{k\, C_A(z)}{D_{A,B}}$$

where $D_{A,B}$ is the diffusion coefficient and C_A is the concentration. The boundary conditions for this example are written in the following form.

```
> BC0 := C[A](0)=C[A,0]: BC0;
```

$$C_A(0) = C_{A,0}$$

```
> BC1:= D(C[A])(L)=0: BC1;
```

$$\mathrm{D}(C_A)(L) = 0$$

This problem has an analytical solution that is easily obtained with Maple.

```
> dsolve({ODE, BC0,BC1},C[A](z));
```

$$C_A(z) = \frac{C_{A,0}\, \mathbf{e}^{\left(\frac{\sqrt{D_{A,B}\, k}\, z}{D_{A,B}}\right)}}{\mathbf{e}^{\left(2\frac{L\sqrt{D_{A,B}\, k}}{D_{A,B}}\right)}+1} + \frac{C_{A,0}\, \mathbf{e}^{\left(-\frac{\sqrt{D_{A,B}\, k}\, z}{D_{A,B}}\right)}}{\mathbf{e}^{\left(2\frac{L\sqrt{D_{A,B}\, k}}{D_{A,B}}\right)}+1}$$

We can express this in the more familir form involving hyperbolic functions as follows

```
> convert(",trig);
```

$$C_A(z) = \frac{C_{A,0}\left(\cosh\left(\frac{\sqrt{D_{A,B}\, k}\, z}{D_{A,B}}\right)+\sinh\left(\frac{\sqrt{D_{A,B}\, k}\, z}{D_{A,B}}\right)\right)}{\cosh(\%1)+\sinh(\%1)+1} + \frac{(\cosh(\%1)+\sinh(\%1))\, C_{A,0}\left(\cosh\left(\frac{\sqrt{D_{A,B}\, k}\, z}{D_{A,B}}\right)-\sinh\left(\frac{\sqrt{D_{A,B}\, k}\, z}{D_{A,B}}\right)\right)}{\cosh(\%1)+\sinh(\%1)+1}$$

$$\%1 := 2\frac{L\sqrt{D_{A,B}\, k}}{D_{A,B}}$$

We shall also attempt a numerical solution. The problem is not straightforward because the ODE to be solved is second order and we will need an estimate of the first derivative in order to use the shooting method as described in the Polymath solution.

We define $y(z)$ as the first derivative term.

```
> ODE3:=diff(C[A](z),z)=y(z): ODE3;
```

$$\frac{\partial}{\partial z} C_A(z) = y(z)$$

The second derivative is obtained on differentiation.

```
> diff(ODE3,z);
```

$$\frac{\partial^2}{\partial z^2} C_A(z) = \frac{\partial}{\partial z} y(z)$$

Substitution of the original ODE gives the following expression for the ODE

```
> ODE4:=subs(ODE,"): ODE4;
```

$$\frac{k\, C_A(z)}{D_{A,\,B}} = \frac{\partial}{\partial z} y(z)$$

The parameter values are specified

```
> Params :={D[A,B]=1.2e-9,k=1e-3};
```

$$Params := \{k = .001,\ D_{A,\,B} = .12\ 10^{-8}\}$$

and substituted into the set of differential equations.

```
> Deqns := subs(Params,{ODE3,ODE4});
```

$$Deqns := \{\frac{\partial}{\partial z} C_A(z) = y(z),\ 833333.3333\ C_A(z) = \frac{\partial}{\partial z} y(z)\}$$

The initial conditions are the known initial concentration and a guess for the unknown first derivative term.

```
> IC := {C[A](0)=0.2,y(0)=-130};
```

$$IC := \{y(0) = -130,\ C_A(0) = .2\}$$

We combine the sets of ODEs and initial conditions.

```
> Alleqns := Deqns union IC;
```

$$Alleqns := \{\frac{\partial}{\partial z} C_A(z) = y(z),\ y(0) = -130,\ C_A(0) = .2,\ 833333.3333\ C_A(z) = \frac{\partial}{\partial z} y(z)\}$$

dsolve is invoked to make a procedure for numerical computation of the result.

```
> g:=dsolve(Alleqns,{C[A](z),y(z)}, type=numeric, method=rkf45, output=procedurelist);
```

$$g := \mathbf{proc}(rkf45\_x)\ \ldots\ \mathbf{end}$$

We compute the "solution" (subject to the guessed initial value of y) as follows

```
> g(1e-3);
```

$$[z = .001,\ C_A(z) = .1404605642564124,\ y(z) = 2.764382915334039]$$

where the value of 0.001 in the call to g is the final value of z.

We need to find an initial value of y that leads to $y(.001) = 0$ (more or less). We make a procedure to automate the preceding computations:

```
> f := proc(y0)
  local IC, Deqns, Params, Alleqns,g, err;
    Params :={D[A,B]=1.2e-9,k=1e-3, y[f,0]=0};
    Deqns := {833333.3333*C[A](z) = diff(y(z),z), diff(C[A](z),z) = y(z)};
```

```
    IC := {C[A](0)=0.2,y(0)=y0};
    Alleqns := Deqns union IC;
    g:=dsolve(Alleqns,{C[A](z),y(z)}, type=numeric, method=rkf45, output=procedurelist);
    err := subs(Params,rhs(g(1e-3)[2]) - y[f,0]);
  end:
```

and test it for some differing initial values

```
> f(-130);
```

.1404605643

```
> f(-140);
```

.1290126434

We can ask fsolve to compute a numerical solution to this equation as shown below. Note that we must place " marks around the name of the function to force its re-evaluation each time the function must be computed.

```
> result:=fsolve('f'(y0),y0=-140..-130);
```

$$result := -131.9111927$$

The value obtained above is the "correct" value of the first derivative at the origin. We need to compute all the other variables at the end using this value for the first derivative. To this end we repeat our initial computations, first remaking the initial condition.

```
> IC := {C[A](0)=0.2,y(0)=result};
```

$$IC := \{ y(0) = result,\ C_A(0) = .2 \}$$

then reforming the full set of differential equations and initial conditions,

```
> Alleqns := Deqns union IC;
```

$$Alleqns := \{ y(0) = -131.9111927,\ 833333.3333\ C_A(z) = \frac{\partial}{\partial z} y(z),\ C_A(0) = .2,\ \frac{\partial}{\partial z} C_A(z) = y(z) \}$$

asking dsolve to generate a procedure from which we may compute a numerical solution

```
> g:=dsolve(Alleqns,{C[A](z),y(z)}, type=numeric, method=rkf45, output=procedurelist);
```

$$g := \mathbf{proc}(rkf45\_x) \ \ldots \ \mathbf{end}$$

Finally, we invoke the procedure to find the desired concentration.

```
> g(1e-3);
```

$$[z = .001,\ y(z) = .1070730348828874\ 10^{-7},\ C_A(z) = .1382726459711341]$$

```
>
```

Binary Batch Distillation

We will begin by creating a list of component names

```
> components:=[Benzene, Toluene];
```

$$components := [Benzene, Toluene]$$

In order to shorten the equations that we shall derive we will identify the components using numerical subscripts that represent the position of the component in the list of their names:

```
> label := proc(i)
    global components;
    local pos;
    member(i,components,`pos`);
    RETURN(pos);
  end:
```

Other methods of labelling the components are possible (first letters, chemical formulae etc); the method used here happens to be simple and widely used. The list of component identities now becomes

```
> compid := map(label,components);
```

$$compid := [1, 2]$$

The number of components is

```
> nc := nops(components);
```

$$nc := 2$$

Now for the equations that model the process. The amount of liquid in the still is given by

```
> DiffEqns :=Diff(L,x[2])=L/x[2]/(K[2]-1): DiffEqns;
```

$$\frac{\partial}{\partial x_2} L = \frac{L}{x_2 (K_2 - 1)}$$

The composition of the vapor leaving and the liquid remaining in the still is related by the equilibrium equations

```
> i:='i': Eqmeqn := i->y[i]=K[i]*x[i]: Eqmeqn(i);
```

$$y_i = K_i x_i$$

The K's are the so-called K-values or equilibrium ratios, the calculation of which will be discussed below. The K's are, in general, complicated functions of the composition of both phases ($x_j, j = 1 .. c$ and $y_j, j = 1 .. c$), temperature (T) and pressure (P); the latter variables not appearing explicitly in any of the above expressions. We have a binary system so we need two of these equations. The system of equations is completed by forcing the mole fractions to sum to unity

```
> i:='i': AEqns := [seq(Eqmeqn(i),i=compid),add(y[i],i=compid)=1,add(x[i],i=compid)=1]: AEqns;
```

$$[y_1 = K_1 x_1, y_2 = K_2 x_2, y_1 + y_2 = 1, x_1 + x_2 = 1]$$

The K-values for this system are to be computed using Raoult's law

```
> RAOULT := i->K[i] = P[i,sat]/P: RAOULT(i);
```

$$K_i = \frac{P_{i,\,sat}}{P}$$

where P is the system pressure, $P_{i,\,sat}$ is the vapor pressure. We form a list of the K-values for all species to be used later.

```
> Kvalues := [seq(RAOULT(i),i=compid)];
```

$$Kvalues := \left[K_1 = \frac{P_{1,\,sat}}{P}, K_2 = \frac{P_{2,\,sat}}{P} \right]$$

Vapor pressures can be computed using the Antoine equation:

```
> Antoine := (T, A, B, C) ->  10^(A - B / (T + C));
```

$$Antoine := (T, A, B, C) \rightarrow 10^{\left(A - \frac{B}{T + C}\right)}$$

Parameters for this equation are as follows:

```
> params:={AntC[Benzene] = 220.79, AntB[Benzene] = 1211.033, AntA[Benzene] = 6.90565, AntC[Toluene] =
  219.482, AntB[Toluene] = 1344.8, AntA[Toluene] = 6.95464}:
```

which we modify below to use the indexing system employed here.

```
> Antparams:=subs(seq(components[i]=i,i=compid),params);
```

$$Antparams :=$$

$$\{ AntC_1 = 220.79, AntC_2 = 219.482, AntB_2 = 1344.8, AntA_2 = 6.95464, AntB_1 = 1211.033, AntA_1 = 6.90565 \}$$

In order to carry out the numerical compuations we substitute the parameters into the K-value model equations.

```
> Kval2 := subs(seq(P[i,sat]=Antoine(T,AntA[i],AntB[i],AntC[i]),i=compid),Antparams,Kvalues): Kval2;
```

$$\left[K_1 = \frac{10^{\left(6.90565 - \frac{1211.033}{T + 220.79}\right)}}{P}, K_2 = \frac{10^{\left(6.95464 - \frac{1344.8}{T + 219.482}\right)}}{P} \right]$$

and the complete set of (differential-algebraic) equations formed by combining the differential and algrbraic equations.

```
> DAEqns := [DiffEqns,op(AEqns)]: DAEqns;
```

$$\left[\frac{\partial}{\partial x_2} L = \frac{L}{x_2 (K_2 - 1)}, y_1 = K_1 x_1, y_2 = K_2 x_2, y_1 + y_2 = 1, x_1 + x_2 = 1 \right]$$

into which we substitute the K-value model

```
> DAEqns2:=subs(Kval2,DAEqns): DAEqns2;
```

$$\left[\frac{\partial}{\partial x_2} L = \frac{L}{x_2 \left(\frac{10^{\left(6.95464 - \frac{1344.8}{T + 219.482}\right)}}{P} - 1 \right)}, y_1 = \frac{10^{\left(6.90565 - \frac{1211.033}{T + 220.79}\right)} x_1}{P}, y_2 = \frac{10^{\left(6.95464 - \frac{1344.8}{T + 219.482}\right)} x_2}{P}, y_1 + y_2 = 1, x_1 + x_2 = 1 \right]$$

The variables appearing in these equations are

```
> Vars := indets(DAEqns2,name);
```

$$Vars := \{y_1, x_1, y_2, x_2, L, P, T\}$$

Note that this system of DAEs includes both temperature and pressure as implicit variables. The number of degrees of freedom is

```
> nops(Vars)-nops(DAEqns);
```

$$2$$

In fact, there is only one degree of freedom since x_2 must be specified at the start of the integration. We specify the pressure as 1.2 atmosphere (to be converted to mm Hg) and recreate the DAE system.

```
> Pspec := P = 1.2*760: Pspec;
```

$$P = 912.0$$

```
> DAEqns3 := subs(Pspec,DAEqns2);
```

$$DAEqns3 := \left[\frac{\partial}{\partial x_2} L = \frac{L}{x_2 \left(.001096491228\ 10^{\left(6.95464 - \frac{1344.8}{T + 219.482} \right)} - 1 \right)}, \right.$$

$$y_1 = .001096491228\ 10^{\left(6.90565 - \frac{1211.033}{T + 220.79} \right)} x_1, y_2 = .001096491228\ 10^{\left(6.95464 - \frac{1344.8}{T + 219.482} \right)} x_2,$$

$$\left. y_1 + y_2 = 1, x_1 + x_2 = 1 \right]$$

The initial value of all the algebriac variables is determined by solving the only algebraic equations.

```
> AEqns2:=subs(seq(Kval2[i],i=compid),AEqns): AEqns2;
```

$$\left[y_1 = \frac{10^{\left(6.90565 - \frac{1211.033}{T + 220.79} \right)} x_1}{P}, y_2 = \frac{10^{\left(6.95464 - \frac{1344.8}{T + 219.482} \right)} x_2}{P}, y_1 + y_2 = 1, x_1 + x_2 = 1 \right]$$

The pressure and initial mole fractions in the liquid are fixed.

```
> AEqns3:=subs(Pspec,x[2]=0.4,AEqns2): AEqns3;
```

$$\left[y_1 = .001096491228\ 10^{\left(6.90565 - \frac{1211.033}{T + 220.79} \right)} x_1, y_2 = .0004385964912\ 10^{\left(6.95464 - \frac{1344.8}{T + 219.482} \right)}, \right.$$

$$\left. y_1 + y_2 = 1, x_1 + .4 = 1 \right]$$

The unknown variables are

```
> indets(AEqns3,name);
```

$$\{y_1, x_1, y_2, T\}$$

We solve these equations using Newton's method.

```
> read `c:/maple/numerics/newton.mpl`:
> with(linalg):
Warning, new definition for norm
Warning, new definition for trace
> start:=Newton(AEqns3,[x[1]=0.3,y[1]=0.5,y[2]=0.5,T=80]);
```

$$start := [x_1 = .6000000000, y_1 = .7869861477, y_2 = .2130138523, T = 95.58508721]$$

The complete starting point is given by adding the initial values of x_2 and L to this list

```
> start2:=[x[2]=0.4,L=100,op(start)];
```

$$start2 := [x_2 = .4, L = 100, x_1 = .6000000000, y_1 = .7869861477, y_2 = .2130138523, T = 95.58508721]$$

Maple out of the box is not capable of solving DAE problems. However, we have implemented a numerical method for solving DAE systems called BESIRK, the code for which is read into Maple now.

```
> read `c:/maple/numerics/integ/besirk`:
```

The equations are to be integrated until the liquid contains 80% toluene. The integration range is, therefore:

```
> xrange := 0.4..0.8;
```

$$xrange := .4 .. .8$$

We integrate the DAE system using BESIRK in the next command. Note that BESIRK can take a number of optional commands to limit the step size, or to increase the accuracy of the calculations. None of these options are needed here.

```
> result:=BESIRK(DAEqns3,start2,xrange);
result := array(0 .. 15, [
    (0) =
    [.4000000000000000, 100., .6000000000000000, .7869861477000000, .2130138523000000, 95.58508721000000]
    (1) = [.4500000000000000, 77.23598270681931, .5495305996552959, .7494166446101018, .2505833553898983,
    96.98940528339986]
    (2) = [.4950000000000000, 61.96997239390999, .5045775396897669, .7131561538646747, .2868438461353253,
    98.28929623227872]
    (3) = [.5355000000000000, 51.16504481005628, .4641197857207883, .6780289519517009, .3219710480482992,
    99.50124726740842]
    (4) = [.5719500000000000, 43.19826793590705, .4277078071487052, .6441992919330006, .3558007080669994,
    100.6280179654252]
    (5) = [.6047549999999999, 37.13310017365944, .3949370264338384, .6117989326156591, .3882010673843412,
    101.6728118832454]
    (6) = [.6342794999999999, 32.39574367167756, .3654433237904527, .5809277625662439, .4190722374337565,
    102.6391809167882]
    (7) = [.6608515499999998, 28.61733139139860, .3388989914114108, .5516553917979685, .4483446082020320,
    103.5309341417773]
    (8) = [.6847663949999998, 25.55117975621302, .3150090922702689, .5240234792648611, .4759765207351387,
    104.3520534930238]
    (9) = [.7062897554999998, 23.02676511082109, .2935081830432426, .4980485673526396, .5019514326473602,
    105.1066178810242]
```

(10) = [.7256607799499999, 20.92284248420555, .2741573647389155, .4737252130914831, .5262747869085169, 105.7987366618055]

(11) = [.7430947019549999, 19.15108189836650, .2566718387902401, .4509367122315387, .5490632877684611, 106.4350610257799]

(12) = [.7605286239600000, 17.48338052905409, .2392379167852399, .4274136972287800, .5725863027712200, 107.0801020554603]

(13) = [.7797059381655000, 15.75733643261899, .2201140250765259, .4006456517445849, .5993543482554154, 107.8001749521963]

(14) = [.7969655209504500, 14.29140918366140, .2028724459673752, .3756054279901092, .6243945720098907, 108.4609083812793]

(15) = [.8000000000000000, 14.04160195907746, .1999456135875499, .3712663769649157, .6287336230350838, 108.5741715536768]

])

In this example the output is not too extansive (it is an easy problem) and we have included it so we can see the structure. The result is a table with eaach entry a list with numbers for the variables in the same order that they appear in the starting list. The get the last point (which contains the numbers we need) we use another command from the BESIRK package

```
> l1:=result[Lastpoint(result)];
```

$l1 := [.8000000000000000, 14.04160195907746, .1999456135875499, .3712663769649157, .6287336230350838, 108.5741715536768]$

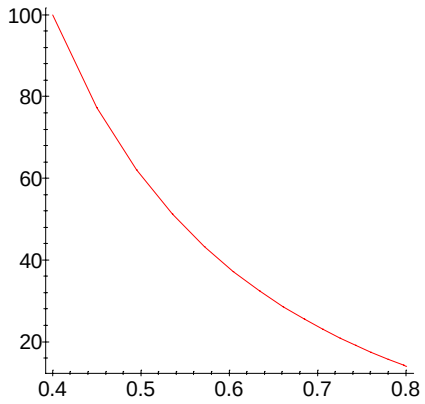
We make a list that identifies the variables with their final values as follows:

```
> [seq(lhs(start2[i])=l1[i],i=1..nops(start2))];
```

$[x_2 = .8000000000000000, L = 14.04160195907746, x_1 = .1999456135875499, y_1 = .3712663769649157, y_2 = .6287336230350838, T = 108.5741715536768]$

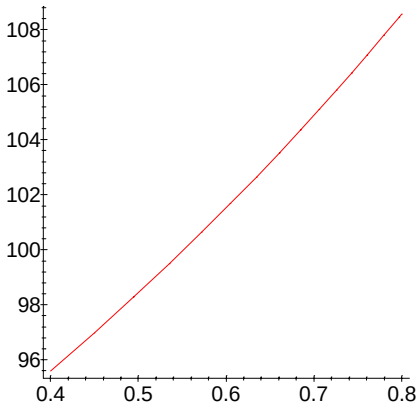
The amount of liquid in the still is plotted as a function of x_2 in the following command

```
> plot({makelist(result,1,2)});
```



and the temperature as a function of x_2 is

```
> plot({makelist(result,1,6)});
```


108
106
104
102
100
98
96
0.4
0.5
0.6
0.7
0.8

>

Reversible, Exothermic, Gas Phase reaction in a Catalytic Reactor

The component material balances for a tubular reactor may be written as follows

```
> restart:
> CMB := Diff(X,W)=-r[A]/F[A,0]: CMB;
```

$$\frac{\partial}{\partial W} X = -\frac{r_A}{F_{A,0}}$$

The reaction rate is given by

```
> rateeqn:=r[A]=-k*(C[A]^2-C[C]/K[C]): rateeqn;
```

$$r_A = -k\left(C_A^{\,2} - \frac{C_C}{K_C}\right)$$

with the reaction rate coefficient calculated from

```
> keqn:=k=k[ref]*exp(E[A]/R*(1/T[ref]-1/T)): keqn;
```

$$k = k_{ref}\,\mathbf{e}^{\left(\frac{E_A\left(\frac{1}{T_{ref}} - \frac{1}{T}\right)}{R}\right)}$$

The reaction equilibrium coefficient is given by

```
> Keqn:=K[C]=K[C,ref]*exp(Delta(H[R])/R*(1/T[ref]-1/T)): Keqn;
```

$$K_C = K_{C,ref}\,\mathbf{e}^{\left(\frac{\Delta(H_R)\left(\frac{1}{T_{ref}} - \frac{1}{T}\right)}{R}\right)}$$

The concentrations of A and C are related to conversion by

```
> CAeqn:=C[A]=C[A,0]*(1-X)/(1+epsilon*X)*y*T[0]/T: CAeqn;
```

$$C_A = \frac{C_{A,0}\,(1-X)\,y\,T_0}{(1+\varepsilon X)\,T}$$

```
> CCeqn:=C[C]=-epsilon*C[A,0]*X/(1+epsilon*X)*y*T[0]/T: CCeqn;
```

$$C_C = -\frac{\varepsilon\,C_{A,0}\,X\,y\,T_0}{(1+\varepsilon X)\,T}$$

The energy balance is

```
> EB := Diff(T,W)=(U[a]*(T[a]-T)+r[A]*Delta(H[R]))/(F[A,0]*C[P,A]): EB;
```

$$\frac{\partial}{\partial W} T = \frac{U_a\,(T_a - T) + r_A\,\Delta(H_R)}{F_{A,0}\,C_{P,A}}$$

The pressure drop is given by

```
> dpeqn:=Diff(y,W)=-alpha*(1+epsilon*X)/2/y*T/T[0]: dpeqn;
```

$$\frac{\partial}{\partial W} y = -\frac{1}{2}\frac{\alpha\,(1+\varepsilon X)\,T}{y\,T_0}$$

where $y = \frac{P}{P_0}$ is the pressure ratio.

The parameters in the model are

```
> params :=
  {C[P,A]=40,C[P,C]=80,Delta(H[R])=-40000,E[A]=41800,k[ref]=0.5,K[C,ref]=25000,R=8.314,U[a]=0.8,T[a]=500,alpha
  =0.015,T[ref]=450,epsilon=-0.5};
```

$$params := \{C_{P,A} = 40,\ C_{P,C} = 80,\ \Delta(H_R) = -40000,\ E_A = 41800,\ k_{ref} = .5,\ K_{C,ref} = 25000,\ R = 8.314,\ U_a = .8,$$
$$T_a = 500,\ \alpha = .015,\ T_{ref} = 450,\ \varepsilon = -.5\}$$

Values at the start of the reactor include

```
> startvalues := {F[A,0]=5.0,P[0]=10,C[A,0]=0.271,T[0]=450,y[0]=1,C[C,0]=0}: startvalues;
```

$$\{F_{A,0} = 5.0,\ P_0 = 10,\ C_{A,0} = .271,\ T_0 = 450,\ y_0 = 1,\ C_{C,0} = 0\}$$

The initial condition is given as a list of equations.

```
> Start := [C[A]=C[A,0],C[C]=0]: Start;
```

$$[W = 0,\ X = 0,\ T = T_0,\ y = y_0,\ C_A = C_{A,0},\ C_C = 0]$$

We are going to use the BESIRK code to solve this particular problem. BESIRK is an implementation of a semi-implicit Runge-Kutta method (see Schwalbe et al, 1996) and is much faster than the methods built in to dsolve/numeric that we used in another example. We read the code.

```
> read `c:/maple/numerics/integ/besirk`:
```

BESIRK needs three arguments. The first is a list of the differential (and algebraic equations):

```
> DEqns := [CMB,EB,dpeqn]: DEqns;
```

$$\left[\frac{\partial}{\partial W} X = -\frac{r_A}{F_{A,0}},\ \frac{\partial}{\partial W} T = \frac{U_a\,(T_a - T) + r_A\,\Delta(H_R)}{F_{A,0}\,C_{P,A}},\ \frac{\partial}{\partial W} y = -\frac{1}{2}\frac{\alpha\,(1+\varepsilon X)\,T}{y\,T_0}\right]$$

The algebraic equations are collected in a list as follows.

```
> AEqns:=[rateeqn,keqn,Keqn,CAeqn,CCeqn]: AEqns;
```

$$\left[r_A = -k\left(C_A^{\,2} - \frac{C_C}{K_C}\right),\ k = k_{ref}\,\mathbf{e}^{\left(\frac{E_A\left(\frac{1}{T_{ref}} - \frac{1}{T}\right)}{R}\right)},\ K_C = K_{C,ref}\,\mathbf{e}^{\left(\frac{\Delta(H_R)\left(\frac{1}{T_{ref}} - \frac{1}{T}\right)}{R}\right)},\right.$$
$$\left.C_A = \frac{C_{A,0}\,(1-X)\,y\,T_0}{(1+\varepsilon X)\,T},\ C_C = -\frac{\varepsilon\,C_{A,0}\,X\,y\,T_0}{(1+\varepsilon X)\,T}\right]$$

```
> read `c:/maple/utils/utils.mpl`:
> read `c:/maple/thermo/td/tdtools.mpl`:
> Subs(params,Start,startvalues,AEqns,right):
  AIC:=Subs(",",right);
```

$$AIC := [r_A = -.0367205,\ k = .5,\ K_C = 25000,\ C_A = .2710000000,\ C_C = 0]$$

```
> DEqns2:=subs(AEqns,AEqns,DEqns):
> DEqns3:=subs(params,startvalues,DEqns2): DEqns3;
```

$$\left[\frac{\partial}{\partial W} X = .1000000000\ \mathbf{e}^{\left(11.17258707 - \frac{5027.664181}{T}\right)}\right.$$

$$\left(14871.80250\frac{(1-X)^2 y^2}{(1-.5X)^2 T^2} - .002439000000\frac{X y}{(1-.5X)\ T\ \mathbf{e}^{\left(-10.69147088 + \frac{4811.161896}{T}\right)}}\right), \frac{\partial}{\partial W} T =$$

$$2.000000000 - .004000000000\ T + 100.0000000\ \mathbf{e}^{\left(11.17258707 - \frac{5027.664181}{T}\right)}$$

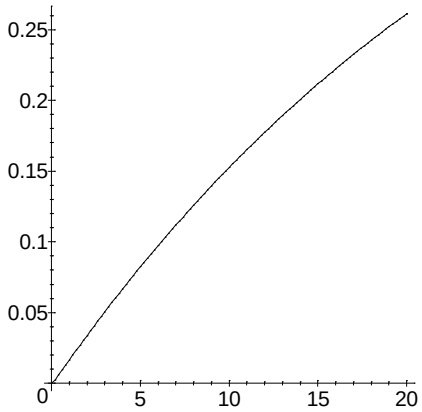
$$\left(14871.80250\frac{(1-X)^2 y^2}{(1-.5X)^2 T^2} - .002439000000\frac{X y}{(1-.5X)\ T\ \mathbf{e}^{\left(-10.69147088 + \frac{4811.161896}{T}\right)}}\right),$$

$$\left.\frac{\partial}{\partial W} y = -.00001666666667\frac{(1-.5X)\ T}{y}\right]$$

```
> 
> 
> IC := [W=0,X=0,T=450,y=1]: IC;
```

$$[W = 0,\ X = 0,\ T = 450,\ y = 1]$$

```
> result := BESIRK(DEqns3,IC,0..20,hmax=0.2):
> plot({seq(makelist(result,1,k),k=[2])});
```




```
> 
> DAEqns := [op(DEqns),op(AEqns)];
```

$$DAEqns := \left[\frac{\partial}{\partial W} X = -\frac{r_A}{F_{A,0}}, \frac{\partial}{\partial W} T = \frac{U_a (T_a - T) + r_A \Delta(H_R)}{F_{A,0} C_{P,A}}, \frac{\partial}{\partial W} y = -\frac{1}{2} \frac{\alpha (1 + \varepsilon X) T}{y T_0}, r_A = -k \left(C_A^2 - \frac{C_C}{K_C} \right), \right.$$
$$k = k_{ref} \mathbf{e}^{\left(\frac{E_A \left(\frac{1}{T_{ref}} - \frac{1}{T} \right)}{R} \right)}, K_C = K_{C,ref} \mathbf{e}^{\left(\frac{\Delta(H_R) \left(\frac{1}{T_{ref}} - \frac{1}{T} \right)}{R} \right)}, C_A = \frac{C_{A,0} (1 - X) y T_0}{(1 + \varepsilon X) T},$$
$$\left. C_C = -\frac{\varepsilon C_{A,0} X y T_0}{(1 + \varepsilon X) T} \right]$$

> DAEqns2:=Subs(params,startvalues,DAEqns);

$$DAEqns2 := \left[\frac{\partial}{\partial W} X = -.2000000000\, r_A, \frac{\partial}{\partial W} T = 2.000000000 - .004000000000\, T - 200.0000000\, r_A, \right.$$
$$\frac{\partial}{\partial W} y = -.00001666666667 \frac{(1 - .5 X) T}{y}, r_A = -k \left(C_A^2 - \frac{C_C}{K_C} \right), k = .5\, \mathbf{e}^{\left(11.17258707 - \frac{5027.664181}{T} \right)},$$
$$\left. K_C = 25000\, \mathbf{e}^{\left(-10.69147088 + \frac{4811.161896}{T} \right)}, C_A = 121.950 \frac{(1 - X) y}{(1 - .5 X) T}, C_C = 60.9750 \frac{X y}{(1 - .5 X) T} \right]$$

> DAEIC:=[op(IC),op(AIC)];

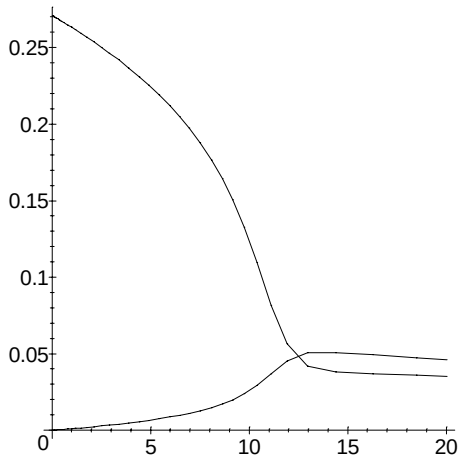
$$DAEIC := [W = 0, X = 0, T = 450, y = 1, r_A = -.0367205, k = .5, K_C = 25000, C_A = .2710000000, C_C = 0]$$

> DAEresult:=BESIRK(DAEqns2,DAEIC,0..20):

polation error
polation error
polation error
polation error
polation error
polation error
polation error
polation error

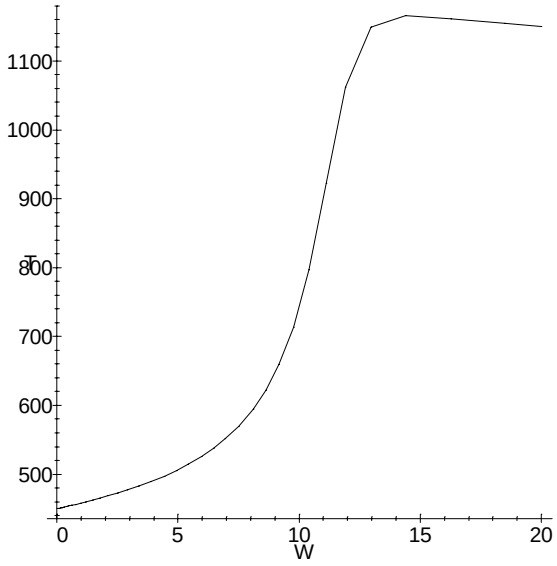
The concentrations are variable numbers 8 and 9 in each line of the output table; we can plot the concentration profiles as follows

> plot({seq(makelist(DAEresult,1,k),k=[8,9])});



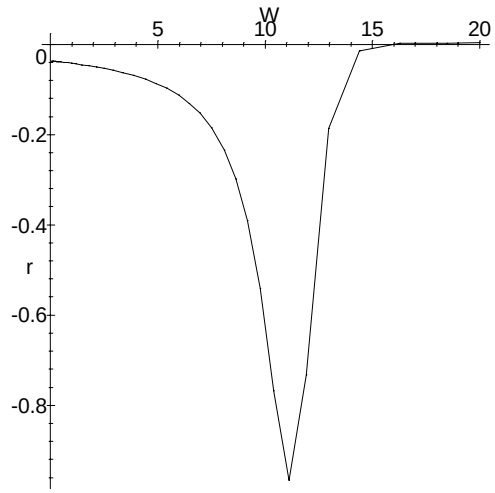
The temperature is variable number 3 in the output list; its profile is

```
> plot({seq(makelist(DAEresult,1,k),k=[3])},labels=[`W`,`T`]);
```



The reaction rate profile is

```
> plot(makelist(DAEresult,1,5),labels=[`W`,`r`]);
```


W
5
10
15
20
0
-0.2
-0.4
r
-0.6
-0.8

Dynamics of a Heated Tank with PI Temperature Control

The energy balance for the stirred tank is

```
> EB := Diff(T[t],t)=(W*C[P]*(T[i]-T[t])+q)/(rho*V*C[P]): EB;
```

$$\frac{\partial}{\partial t} T_t = \frac{\mathrm{W}\, C_P\, (T_i - T_t) + q}{\rho\, V\, C_P}$$

where T_t is the temperature in the tank. The temperature of the exit stream is given by

```
> Teqn:=Diff(T[0],t)=(T[t]-T[0]-tau[d]/2*Diff(T[t],t))*2/tau[d]: Teqn;
```

$$\frac{\partial}{\partial t} T_0 = 2\,\frac{T_t - T_0 - \frac{1}{2}\,\tau_d \left(\frac{\partial}{\partial t} T_t\right)}{\tau_d}$$

The thermocouple shielding and electronics are modelled by

```
> Teqn3:=Diff(T[m],t)=(T[0]-T[m])/tau[m]: Teqn3;
```

$$\frac{\partial}{\partial t} T_m = \frac{T_0 - T_m}{\tau_m}$$

The energy input to the tank is modelled by

```
> qeqn:=q=q[s]+K[c]*(T[r]-T[m])+K[c]/tau[I]*epsilon: qeqn;
```

$$q = q_s + K_c\,(T_r - T_m) + \frac{K_c\,\varepsilon}{\tau_I}$$

where

```
> eeqn:=Diff(epsilon,t)=T[r]-T[m]: eeqn;
```

$$\frac{\partial}{\partial t}\varepsilon = T_r - T_m$$

and

```
> qseqn:= q[s]=W*C[P]*(T[r]-T[i,s]): qseqn;
```

$$q_s = \mathrm{W}\, C_P\, (T_r - T_{i,\,s})$$

The parameters in the model are

```
> params :={V=4000/rho/C[P],W=500/C[P],T[i,s]=60,T[r]=80,tau[d]=1,tau[m]=5,tau[I]=2,K[c]=0};
```

$$params := \{V = \frac{4000}{\rho\, C_P},\ \mathrm{W} = \frac{500}{C_P},\ T_{i,\,s} = 60,\ T_r = 80,\ \tau_d = 1,\ \tau_m = 5,\ K_c = 0,\ \tau_I = 2\}$$

The initial values of the differential variables are given as a list of equations.

```
> Start := [t=0,T[t]=80,T[0]=80,T[m]=80,epsilon=0]: Start;
```

$$[t = 0,\ T_t = 80,\ T_0 = 80,\ T_m = 80,\ \varepsilon = 0]$$

Several different approaches for solving this system of equations can now be followed. A difficulty is that the step change in the inlet temperature occurs not at $t = 0$, but at $t = 10$. One possible approach is to integrate the equations for the first 10 minutes, then make the step change and integrate the equations with the new parameters for as long as needed. Maple's own dsolve/numeric will accept a procedure for evaluation of the right hand sides of the differential equations. We can program the step change in that procedure. Here is our code for this particular system.

```
> deproc := proc(n,t,y,dy)
  local q,qs,k,epsilon;
  global params,T;
  T[tank]:=y[1];
  T[0]:=y[2];
  T[m]:=y[3];
  epsilon:=y[4];
  if t < 10 then T[i]:=60 else T[i]:=40 fi;
  qs:=W*C[P]*(T[r]-T[i,s]);
  q:=qs+K[c]*(T[r]-T[m])+K[c]/tau[l]*epsilon;
  dy:=vector(n):
  dy[1]:=(W*C[P]*(T[i]-T[tank])+q)/(rho*V*C[P]):
  dy[2]:=(T[tank]-T[0]-tau[d]/2*dy[1])*2/tau[d]:
  dy[3]:=(T[0]-T[m])/tau[m]:
  dy[4]:=T[r]-T[m]:
  for k from 1 to n do dy[k] := subs(params,dy[k]); od;
  RETURN(eval(dy));
  end:
```

The arguments (required by Maple) are the number of equations, the independent variable, the vector of dependent variables and a vector of right hand sides (computed by the procedure). Note where we have programmed the step change in the inlet temperature and that set of model parameters is declared to be a global variable. This facilitates changing the parameters without changing the procdure, as we will have to do for later calculations.

We solve the problem using dsolve/numeric using the following command. Note that by using the optional arguments we can tell dsolve/numeric to use the proceure above for the right hand sides. In this case the first three arguments serve no role except that dsolve/numeric expects them to be there. The last argument tells dsolve/numeric to return an array with the results tabulated as shown

```
> result:=dsolve({diff(x(t),t)=0,diff(y(t),t)=0,diff(w(t),t)=0,diff(z(t),t)=0}, {w(t),x(t),y(t),z(t)},type=numeric,
  method=rkf45, initial=vector([80,80,80,0]),start=0,procedure=deproc,value=array([0,seq(i,i=9..20)]));
```

$$result := \left[[t,\ dsolve/numeric\_F_1(t),\ dsolve/numeric\_F_2(t),\ dsolve/numeric\_F_3(t),\ dsolve/numeric\_F_4(t)] \right.$$

0	80	80	80	0
9	80	80	80	0
10	80.00000516	79.99999484	80.	0
11	77.64681396	79.64269439	80.02650031	-.02217814937
12	75.58158780	77.59747344	79.76982224	.04954268262
13	73.82512871	75.57269420	79.18312375	.5490806715
14	72.40877288	73.82166335	78.35992679	1.762104706
15	71.36231380	72.40641866	77.39937837	3.874889469
16	70.70902677	71.36074478	76.39059205	6.979304493
17	70.46248083	70.70837739	75.41166771	11.08363213
18	70.62437518	70.46288279	74.52992468	16.12345525
19	71.18325249	70.62589984	73.80184548	21.97243756
20	72.11402574	71.18590699	73.27275148	28.45330713

$$\left. \right]$$

To extract the table, perhaps for plotting purposes we use the op command in the following form

```
> op([1,3,2,2],result);
```

0	80	80	80	0
9	80	80	80	0
10	80.00000516	79.99999484	80.	0
11	77.64681396	79.64269439	80.02650031	-.02217814937
12	75.58158780	77.59747344	79.76982224	.04954268262
13	73.82512871	75.57269420	79.18312375	.5490806715
14	72.40877288	73.82166335	78.35992679	1.762104706
15	71.36231380	72.40641866	77.39937837	3.874889469
16	70.70902677	71.36074478	76.39059205	6.979304493
17	70.46248083	70.70837739	75.41166771	11.08363213
18	70.62437518	70.46288279	74.52992468	16.12345525
19	71.18325249	70.62589984	73.80184548	21.97243756
20	72.11402574	71.18590699	73.27275148	28.45330713

The first column is the time, the next three are the tank, thermocouple, and measured temperatures.

A. Open Loop Performance

Open loop performance is simulated by setting $K_c = 0$, which is what was done for the above calculations. We need to integrate for a longer time, however, so we repeat the above command but integrate to 60 minutes.

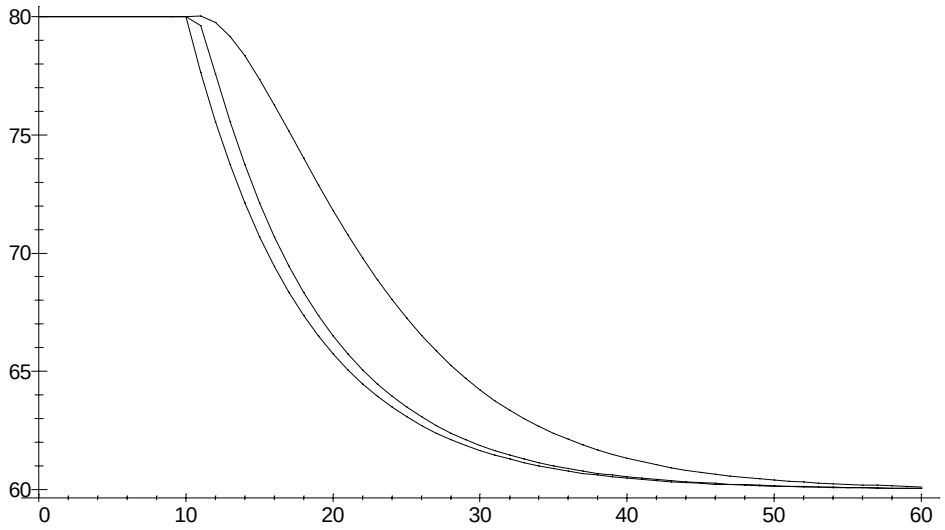
```
> result:=dsolve({diff(x(t),t)=0,diff(y(t),t)=0,diff(w(t),t)=0,diff(z(t),t)=0}, {w(t),x(t),y(t),z(t)},type=numeric,
  method=rkf45, initial=vector([80,80,80,0]),start=0,procedure=deproc,value=array([0,seq(i,i=9..60)]));
```

We extract the results table

```
> Ttable :=op([1,3,2,2],result):
```

and plot the three temperatures.

```
> with(linalg):
> plot({seq([seq([Ttable[i,1],Ttable[i,k]],i=1..rowdim(Ttable))],k=2..4)},color=[red,blue,black]);
```



B. Closed loop performance

This requires a change in K_c to 50 (it was 0 in the first case).

```
> params :={V=4000/rho/C[P],W=500/C[P],T[i,s]=60,T[r]=80,tau[d]=1,tau[m]=5,tau[l]=2,K[c]=50};
```

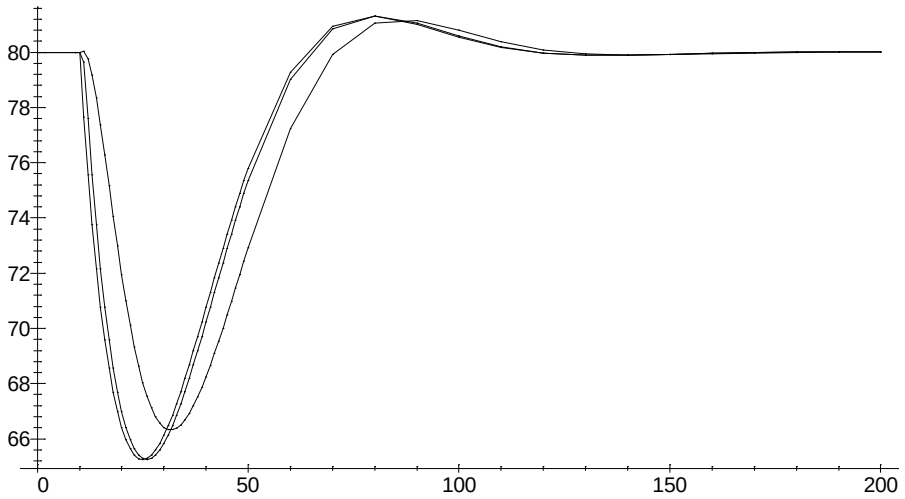
$$params := \{ V = \frac{4000}{\rho\, C_P},\ W = \frac{500}{C_P},\ T_{i,\,s} = 60,\ T_r = 80,\ \tau_d = 1,\ \tau_m = 5,\ \tau_I = 2,\ K_c = 50 \}$$

repeat the integration

```
> result:=dsolve({diff(x(t),t)=0,diff(y(t),t)=0,diff(w(t),t)=0,diff(z(t),t)=0}, {w(t),x(t),y(t),z(t)},type=numeric,
  method=rkf45, initial=vector([80,80,80,0]),start=0,procedure=deproc,
  value=array([0,seq(i,i=9..50),seq(i*10,i=6..20)]));
```

and plot the results.

```
> Ttable :=op([1,3,2,2],result):
> plot({seq([seq([Ttable[i,1],Ttable[i,k]],i=1..rowdim(Ttable))],k=2..4)},color=[red,blue,black]);
```



C. Closed loop performance for $K_c = 500$

We increase K_c to 500

```
> params :={V=4000/rho/C[P],W=500/C[P],T[i,s]=60,T[r]=80,tau[d]=1,tau[m]=5,tau[I]=2,K[c]=500};
```

$$params := \{ V = \frac{4000}{\rho\, C_P},\ W = \frac{500}{C_P},\ T_{i,\,s} = 60,\ T_r = 80,\ \tau_d = 1,\ \tau_m = 5,\ \tau_I = 2,\ K_c = 500 \}$$

and repeat the integration, this time asking for results to be returned for every minute after the 9th so that we can make a nice looking plot.

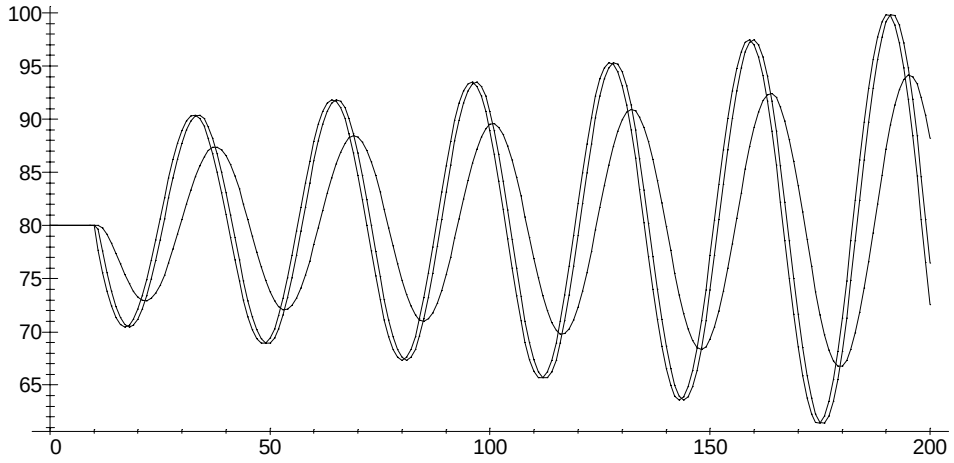
```
> result:=dsolve({diff(x(t),t)=0,diff(y(t),t)=0,diff(w(t),t)=0,diff(z(t),t)=0}, {w(t),x(t),y(t),z(t)},type=numeric,
  method=rkf45, initial=vector([80,80,80,0]),start=0,procedure=deproc,
  value=array([0,seq(i,i=9..200)])):
```

The output table is extracted from the result:

```
> Ttable :=op([1,3,2,2],result):
```

and the three temperatures plotted as a function of time.

```
> plot({seq([seq([Ttable[i,1],Ttable[i,k]],i=1..rowdim(Ttable))],k=2..4)},color=[red,blue,black]);
```



D. Proportional Control

Proportional only control is simulated by setting the term $\frac{K_c}{\tau_I} = 0$. We can accomplish this by giving τ_I a very high value as shown here.

```
> params :={V=4000/rho/C[P],W=500/C[P],T[i,s]=60,T[r]=80,tau[d]=1,tau[m]=5,tau[I]=1e99,K[c]=500};
```

$$params := \{V = \frac{4000}{\rho\, C_P},\ W = \frac{500}{C_P},\ T_{i,\,s} = 60,\ T_r = 80,\ \tau_d = 1,\ \tau_m = 5,\ \tau_I = .1\ 10^{100},\ K_c = 500\}$$

We carry out the integrations for this situation.

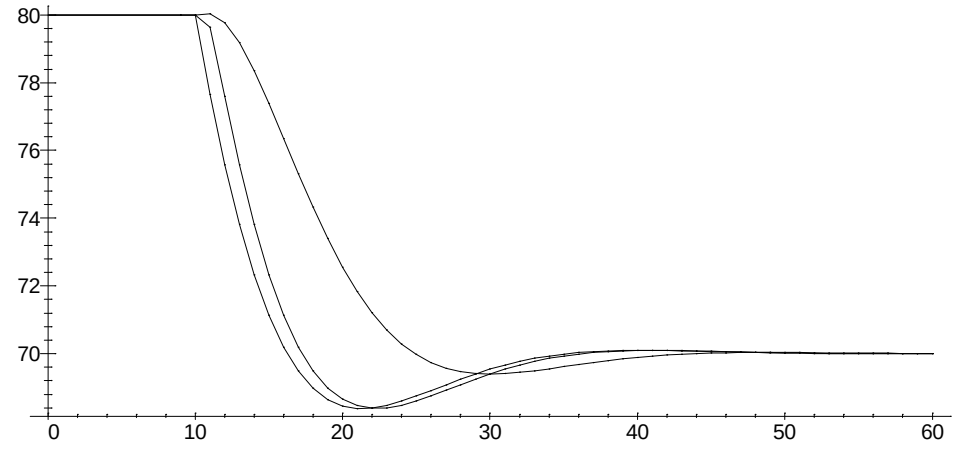
```
> result:=dsolve({diff(x(t),t)=0,diff(y(t),t)=0,diff(w(t),t)=0,diff(z(t),t)=0}, {w(t),x(t),y(t),z(t)},type=numeric,
  method=rkf45, initial=vector([80,80,80,0]),start=0,procedure=deproc,
  value=array([0,seq(i,i=9..60)])):
```

The output table is extracted from the result:

```
> Ttable :=op([1,3,2,2],result):
```

and the three temperatures plotted as a function of time.

```
> plot({seq([seq([Ttable[i,1],Ttable[i,k]],i=1..rowdim(Ttable))],k=2..4)},color=[red,blue,black]);
```




```
>
```