
A Tour of Ascend Thermodynamics

Release 1.0

Pedro Vale Lima

July 9, 1999

Department of Chemical Engineering
University of Coimbra
Coimbra, Portugal
E-mail: eq3pvl@eq.uc.pt

Abstract

Thermodynamics have never been easy. At least to me. And every single model in Ascend as a thermodynamics state inside, so I soon decided I had to master this model if I wanted to understand other models in the Ascend library.

As I remember it took me some time to understand the thermodynamics library I decided to write a small guide on the subject. Hopefully it will make your learning easier than mine.

I'm very interested in your feedback. I sure this text can be improved and I'm interested in doing it. Thanks to everybody in the Ascend Team. Ascend Rules!

1 Thermodynamics Survival Kit

This guide is a practical thermodynamics document and not an extensive thermodynamics manual, the theory and equations presented are just for explaining the main concepts. The most of this chapter are the examples of using the thermodynamics library to perform simple calculations. I expect the reader to already have some experience working with the Ascend user interface. If not, I recommend reading the Ascend HOWTO book first.

2 Physical Property Data

There can be no thermodynamics calculations without property data. In Ascend these values are added in the library file `components.a41`. To add a new component to the library you just have to do the following

- 1) Add the name to the supported components list

```
supported_components ::= [  
    'hydrogen',  
    'carbon_dioxide',  
    [...]
```

- 2) Add a CASE block to model

```

CASE 'hydrogen':
rpp_index := 75;
[...]
lden := 0.071 {g/cm^3};

```

Some explanation on the meaning of the constants is presented in the following table.

Constant	Meaning
rpp_index	Just a index number (optional)
formula	Chemical formula (optional)
subgroups	Components subgroups for the UNIFAC model
wilson_set	Wilson groups for the Wilson liquid model
mw	Molecular weight
Tb	Boiling point temperature
Tc	Critical temperature
Pc	Critical pressure
Vc	Critical volume
Zc	Critical compressibility factor
omega	Pitzer acentric factor
cpvapa..d	Constants for vapor phase Cp calculation $Cp^V = Cp_a^V + Cp_b^V T + Cp_c^V T^2 + Cp_d^V T^3$
Hf	Entalpy of formation
Gf	Gibbs free energy of formation
vp..d	Constants for vapor pressure calculation (three equations can be used)
vp_correlation	number that represents the vapor pressure equation choosen
Hv	Entalpy of vaporization
lden	Liquid density
Tliq	Temperature of the liquid density

3) Add the name in the check list of the components model

```

FOR i IN components CHECK
(i IN ['hydrogen', 'carbon_dioxide', 'water', 'chloroform',
'methane', 'methanol', 'ethylene', 'ethane',
[...]
'nitrogen', 'aniline', 'nitrobenzene' ]) == TRUE;
END FOR;

```

All the properties presented can be found in the book *Properties of Gases and Liquids* from Reid, Prausnitz & Poling.

3 Single Component

The simplest state to model is the one that represents a single component in a single phase. To elaborate this model one needs an equation of state (EOS). As there is no universal equation of state, the thermodynamic libraries often contain several models and let the choosing decision to the user.

The Ascend Thermodynamics library has two models for vapor components (`ideal_vapor_component` and `Pitzer_vapor_component`) and one model for liquid components (`Rackett_liquid_component`). All these models are similar in concept, only the EOS changes. I'll use the ideal vapor model to illustrate the main concepts.

The model defines the equations needed to calculate the molar volume, V , enthalpy, H , and gibbs free energy, G , based on the temperature, T , pressure, P , and component properties.

The first equation needed by the model is the well known equation of state

$$PV = RT \quad (1)$$

where R the perfect gas constant. To define H one can use the constant pressure heat capacity, C_p . The heat capacity is defined implicitly as polynomial function of temperature.

$$C_p = C_p^a + C_p^b T + C_p^c T^2 + C_p^d T^3 \quad (2)$$

with C_p^a , C_p^b , C_p^c , C_p^d , defined in the property data library. Remembering that for an ideal gas H is defined as

$$H - H_0 = \int_{T_0}^T C_p dT \quad (3)$$

results in the following explicit equation for H . Note that if it was not an ideal gas H would also depend on P .

$$H = H_0 + C_p^a(T - T_0) + \frac{C_p^b(T^2 - T_0^2)}{2} + \frac{C_p^c(T^3 - T_0^3)}{3} + \frac{C_p^d(T^4 - T_0^4)}{4} \quad (4)$$

Looking back again to the old (but rather unused :-)) manual of thermodynamics, it can be proved that

$$d\left(\frac{G}{RT}\right) = \frac{V}{RT}dP - \frac{H}{RT^2}dT \quad (5)$$

Substitution of equations 1 and 4 on equation 5 followed by integration originates a equation for G with the form

$$G = f(T, P) \quad (6)$$

Obtaining explicit formula remains as an exercise for the reader (please don't do that, just look at the library code).

As a resume, this model has three equations 1, 4 and 6, and five variables, P , T , G , H and V .

Ideal vapor component (IVC)		
Equations	Variables	Fix
3	5 : P, T, G, H, V	2 : P, T

Normaly you don't need to use this model directly because you can use the thermodynamics generic interface, but, for illustrative purposes, here is an example of how to use the model.

First you need to open the `thermodynamics.a41` file and look at the header of the model. It shows the following

```
MODEL ideal_vapor_component(
  P WILL_BE pressure;
  T WILL_BE temperature;
  data WILL_BE td_component_constants;
) REFINES pure_component;
```

Now that you know what objects you need to create to run the model you can build a model that calls the `ideal_vapor_component`.

```
MODEL howto_thermo_ex1 REFINES cmumodel;
  cd IS_A components_data(['water'], 'water');
  P IS_A pressure;
  T IS_A temperature;
  ivc IS_A ideal_vapor_component(P, T, cd.data['water']);

[...]

METHOD values;
  P := 1 {atm};
  T := 400 {K};
END values;

END howto_thermo_ex1;
```

By default P and T are fixed which makes the problem square. You need to have `default_self`, `scale_self`, `specify` and `values` methods and then you can then run the model. If you need some help on doing it check the source of example files that come with the howto. For the sake of clarity only the important parts of the models are presented in the text.

So, what kind of problem can this model solve? This one, for example.

Problem 1 : Calculate the change of enthalpy for water being heated from 400 K to 500 K at constant atmospheric pressure.

Solution: As the initial value for temperature had already been set to 400 K just run the model with the following standard script (or use the graphical user interface if you prefer)

```
DELETE TYPES;
READ FILE "howto_thermo.a4c";
COMPILE ex OF howto_thermo_ex1;
BROWSE {ex};
RUN {ex.values};
RUN {ex.reset};
SOLVE {ex} WITH QRS1v;
```

In the browser window you can check the value for H in the model path `ex.ivc.h` (-238.532 kJ/mole for lazy guys that don't want to run the simulations). Now change the value of T and solve the model again

```
ASSIGN {ex.T} 500 {K};
SOLVE {ex} WITH QRS1v;
```

And voilà, the new H is -235.040 kJ/mole and ΔH is 3.492 kJ/mole.

The model for liquid components (`Rackett_liquid_component`) has one slight difference from the vapor model presented. As the enthalpy and free energy depend on knowledge of the vapor pressure there is an additional equation with the correlation to calculate this value. You still have two degrees of freedom, but now you have six variables and four equations.

Rackett liquid component (RLC)		
Equations	Variables	Fix
4	6 : P, T, G, H, V, VP	2 : P, T

Problem 2 : Calculate the temperature at which the vapor pressure for water is 1 atm.

Solution: The model to be used is very similar to the previous one (remember you can check the complete model in the `howto_thermo.a4c` file). You need to change the model to Rackett

```
ivc IS_A Rackett_liquid_component(P, T, cd.data['water']);
```

then you must make a method to pose the new problem structure. You must free the temperature, fix and give a value to the vapor pressure

```
METHOD reset_VP_problem;
  RUN ivc.specify;
  ivc.T.fixed := FALSE;
  ivc.VP.fixed := TRUE;
  ivc.VP := 1 {atm};
END reset_VP_problem;
```

now just call the this method and solve the problem. You'll see that the temperature is ... 373.15 K (what pleasure of making such redundant stuff!)

4 Mixtures

As with the model for single components, the model for mixtures pretends to represent the mixture molar volume, enthalpy and Gibbs free energy.

To understand this model let me present an example. Suppose you have a mixture of two components and want to calculate the molar volume. For each component you first calculate the molar volume for each pure specie, V_i^p (where i is the index for the component and p represents pure specie) with the model from the previous section. Then you calculate the molar volume for the specie *in* the mixture, V_i , assuming some model for the mixing. At last, the molar volume is the sum of all species contributions weighted by the molar fraction of the specie in the mixture. As an example, for a ideal vapor mixture the equations are the following

$$PV_i^p = RT \quad (7)$$

$$V_i = V_i^p \quad (8)$$

$$V = \sum_i y_i V_i \quad (9)$$

The first equation comes from the `ideal_vapor_component` model (one for each component), the second equation comes from the `ideal_vapor_mixture` model (again one for each component) and the last one is generic for all mixture models (is part of `phase_partials` which all mixture models inherit from). The same concepts apply to enthalpy and free energy. The equations for ideal vapor mixing and mixture calculation are the following

$$H_i = H_i^p \quad (10)$$

$$H = \sum_i y_i H_i \quad (11)$$

$$G_i = G_i^p + RT \ln(y_i) \quad (12)$$

$$G = \sum_i y_i G_i \quad (13)$$

This model has one extra special equation whose meaning will be easier to understand after talking about phase equilibria.

$$\sum_i y_i + s = 1 \quad (14)$$

This equation introduces a new variable `s` called `slack_PhaseDisappearance`. With this formulation of the molar fraction sum, when `s` is fixed at 0 normal calculations are performed, when `s` is free, if the phase ceases to exist `s` will be different from 0 but simulation can continue. It's a clever and important trick for multi-phase equilibria. You can forget this for a while, I'll get back to the issue later.

Now it's time to do some counting.

Ideal vapor mixture (IVM)		
Equations	Variables	Fix
n <i>IVC</i> ($3n$)	n (<i>IVC</i> - 2) ($3n$)	P, T, y_{n-1}, s
mixture ($3n + 4$)	$4n + 6$: $P, T, H, V, G, s, y_n, H_n, V_n, G_n$	
$6n + 4$	$7n + 6$	$n + 2$
n - number of components		

Just a minor warning, when using models where $n-1$ components are fixed, usually, the unfixed one is the reference component, but be sure to check this and save yourself some aspirin. OK, now it's time to see some action.

Problem 3 : Calculate the change of enthalpy for a mixture of 60% water and 40% ethanol going from 400 K to 500 K.

Solution: Another easy one. First look at the `ideal_vapor_mixture` header. Then make the model

```
MODEL howto_thermo_ex3 REFINES cmumodel;
  cd IS_A components_data(['water', 'ethanol'], 'water');
  ivm IS_A ideal_vapor_mixture(cd);
[...]
```

```
METHOD values;
  ivm.P := 1 {atm};
  ivm.T := 400 {K};
  ivm.y['ethanol'] := 0.4;
END values;
END howto_thermo_ex3;
```

And then solve the model with a standard script. After a second solve with $T = 500$ K you'll get $\Delta H = 3.177$ kJ/mole. Now let's see what value is obtained with the `Pitzer_vapor_mixture`.

```
MODEL howto_thermo_ex4 REFINES cmumodel;
  cd IS_A components_data(['water', 'ethanol'], 'water');
  ivm IS_A Pitzer_vapor_mixture(cd);
[...]
```

```
END howto_thermo_ex4;
```

The result is $\Delta H = 5.632$ kJ/mole. So now, if you had to solve this problem for real you would have to take a decision ...

For liquid mixtures the library has a UNIFAC model to account for the mixing of components represented with the Rackett model. The number of equations and variables depends on the total number of UNIFAC groups of the species in the mixture. This number is represented by l .

As the UNIFAC equations are complex I'll only present the number of equations and variables. The degrees of freedom are the same as for a vapor mixture.

UNIFAC liquid mixture (ULM)		
Equations	Variables	Fix
n <i>RLC</i> ($4n$) mixture ($5n + 2l + 4$)	n (<i>RLC</i> - 2) ($4n$) $4n + 6 : P, T, H, V, G, s, y_n, H_n, V_n, G_n$	P, T, y_{n-1}, s
		$2n + 2l : Jv_n, Ls_n, \theta_l, \eta_l$
$9n + 2l + 4$	$10n + 2l + 6$	$n + 2$
n - number of components, l - number of UNIFAC groups		

5 Phase Equilibria

Phase equilibria is the ultimate model of the library. And you can be sure it is the base model of most chemical process models, from simple streams to complex reaction and separation systems.

To model phase equilibria one uses the `thermodynamics` model in the library. This model is the generic model for thermodynamics calculations and can be used for every calculation, from single component to multi-phase mixtures. Of course that a model for a single component using this interface has a lot of redundant information but that's the price to pay for flexibility. The model has a generic interface and equilibrium relations that are only used for systems with at least two phases.

5.1 Generic Interface

As mixtures are models that group simple components, phase equilibria groups the same mixture at different phases. By defining γ_j to be the molar fraction of phase j we have usual fraction relation

$$\sum_j \gamma^j = 1 \quad (15)$$

You still remember the three main properties, V , H and G , don't you? Now you must have another three equations to account for the phases

$$\sum_j \gamma^j V^j \quad (16)$$

$$\sum_j \gamma^j H^j \quad (17)$$

$$\sum_j \gamma^j G^j \quad (18)$$

where V^p , H^p , and G^p are the mixture properties defined by equations 8, 11 and 13 for each phase.

Another group of equations (one for each component) are need to define a global fraction for a component.

$$y_i = \sum_p \gamma^p y_i^p \quad (19)$$

5.2 Relative Volatility Model

This model has extra equations for multi-phase systems in equilibrium. The first group just equates pressure and temperature on all phases to the ones in the reference phase.

$$P^j = P^{ref} \quad (20)$$

$$T^j = T^{ref} \quad (21)$$

This accounts for $m-1$ equations where m is the number of phases (the reference phase is excluded). OK, now it's time to go to the fridge and get a snack. We're almost over.

Are you back? Great, let's show one clever trick from the library. Remember s , the `slack_PhaseDisappearance`? There is one of these for each mixture. One problem with multi-phase models is that models are written thinking of the existence of a phase, and if by some thermodynamics change the phase ceases to exist the equations become invalid. By using the following equation for each phase

$$s^j \gamma^j = 0 \quad (22)$$

it's possible for a phase to *disappear*. If γ^j is not null, which means the phase exists, then s^j must be zero and by the already presented relation $\sum_i y_i + s = 1$ the mixture properties must be properly calculated. On the other hand, if γ^j becomes zero then s^j can be relaxed and the model continues to be solved without regard for that mixture.

The library can be used to perform just T and P equilibrium and fractions are computed by specification of relative volatilities. This kind of equilibrium is easier to compute and can be used as a first guess to solve the full equilibrium.

The following set of equations is a special representation of relative volatilities

$$\bar{\alpha}^j y_i^j = \alpha_i^j y_i^{ref} \quad (23)$$

where $j = 1, \dots, phases - 1, i = 1, \dots, components$. To make it easier to understand the goal of this representation let's see what these equations represent for a two phase system (V,L) of two components (A,B).

$$\bar{\alpha} y_A^V = \alpha_A^V y_A^L \quad (24)$$

$$\bar{\alpha} y_B^V = \alpha_B^V y_B^L \quad (25)$$

in the relative volatility model you fix the α 's and compute $\bar{\alpha}$. Note that $\alpha_A^V / \bar{\alpha}$ will be a K-value. Before seeing the model in action let's do some accounting (to simplify, I'll suppose a two phase system since Ascend is not yet able to compute Liquid-Liquid models).

VL-Equilibrium α -model		
Equations	Variables	Fix
ULM ($9n + 2l + 4$)	ULM ($10n + 2l + 6$)	P, T, y_{n-1}
IVM ($6n + 4$)	IVM ($7n + 6$)	α_n, γ^V
general interface ($n + 4$)	$n + 5 : y_n, \gamma^V, \gamma^L, V, H, G$	
equilibrium ($n + 4$)	$n + 1 : \bar{\alpha}, \alpha_n^V$	
$17n + 2l + 16$	$17n + 2l + 16$	$2n + 2$
n - number of components, l - number of UNIFAC groups		

5.3 Chemical Potencial Equilibrium

In the full equilibrium model the Gibbs free energy for each phase is equated. To choose full equilibrium you should set the variable `equilibrated` to `TRUE`.

$$G_i^j = G_i^{ref} \quad (26)$$

for each partial component and each phase except the reference one.

When using this model $\bar{\alpha}$ is fixed and equal to one and the α 's are released and computed by equation 23. This equation becomes a computation of K-values. Now let's see the infamous accounting table for this model.

VL-Equilibrium G-model		
Equations	Variables	Fix
ULM ($9n + 2l + 4$)	ULM ($10n + 2l + 6$)	$P, T, y_{n-1}, \bar{\alpha}$
IVM ($6n + 4$)	IVM ($7n + 6$)	
general interface ($n + 4$)	$n + 5 : y_n, \gamma^V, \gamma^L, V, H, G$	
equilibrium ($2n + 4$)	$n + 1 : \bar{\alpha}, \alpha_n^V$	
$18n + 2l + 16$	$17n + 2l + 16$	$n + 2$
n - number of components, l - number of UNIFAC groups		

Let's solve some problems.

Problem 4 : Calculate the change of enthalpy for a 50% water, 50% ethanol being heated from 300 K to 400 K at constant atmospheric pressure.

Solution: This is a simple problem but it's useful to introduce the way to use the `thermodynamics` model.

```

MODEL howto_thermo_ex8 REFINES cmumodel;
  cd IS_A components_data(['water','ethanol'], 'water');
  pd IS_A phases_data('VL', 'ideal_vapor_mixture', 'UNIFAC_liquid_mixture',
                    'none');
  equilibrated IS_A boolean;
  phases ALIASES pd.phases;

  FOR j IN phases CREATE
    smt[j] IS_A select_mixture_type(cd, pd.phase_type[j]);
  END FOR;

  FOR j IN phases CREATE
    phase[j] ALIASES smt[j].phase;
  END FOR;

  state IS_A thermodynamics(cd, pd, phase, equilibrated);
[... ]
  METHOD values;
    state.P := 1 {atm};
    state.T := 300 {K};
    state.y['ethanol'] := 0.5;
    equilibrated := TRUE;
  END values;

END howto_thermo_ex8;

```

Use of the thermodynamics model is slightly more complex. First you should define the components data, then you choose the models for the phases with the `phases_data` model. This model's first parameter is a phase definition (it can be V,L or VL), then the name for the vapor model (or none if there is no vapor phase) and then the liquid model name. The last parameter is for a second liquid phase model when LL equilibrium is implemented, for now use none.

The `select_mixture_type` model builds the choosen models so that they can be sent to the main thermodynamic model as a parameter. This main seem complicated at first but as you probably will never need to change that piece of code you don't need to worry about it. Just peek at the library code when you fell brave enough.

The value definitions are very simple, just notice the newly introduced `equilibrated` variable.

When solving multi-phase models you should always check `state.slack_PhaseDisappearance`. Only if it's zero the phase exists. Sometimes you could be getting strange results because you're looking at properties of a non-existing phase.

If you run the model you'll get only a liquid phase and $H = -282.351$ kJ/mole. With $T = 400$ K you'll get $H = -233.022$ kJ/mole and so $\Delta H = 49.329$ kJ/mole.

Problem 5 : Build the VL equilibrium chart for water-ethanol at atmospheric pressure.

Solution: The model presented in the last problem specifies P and T. To build a equilibrium chart it's better to specify P and a mole fraction and to free T. That way we can be sure to have an equilibrium. The following method, added to the last problem model does that trick. We'll also use the Pitzer vapor model in the hope of improving the accuracy.

```

METHOD reset_Px;
  equilibrated := TRUE;
  RUN state.specify;
  state.T.fixed := FALSE;
  state.phase['liquid1'].y['water'].fixed := TRUE;
  state.P := 1 {atm};
  state.y['ethanol'] := 0.5;
  state.phase['liquid1'].y['water'] := 0.6;
END reset_Px;

```

Now, instead of calling `reset` and values just run `reset_Px`. We'll run the model several times and build a table like the following.

y_w^L	y_w^V	T (K)
0.99	0.8786	369.764
0.95	0.652	362.471
0.9	0.5517	358.80
⋮	⋮	⋮

When building the table you must take some care for a phase not to disappear. You can do that by changing the variable `state.phase['liquid1'].y['water']`.